COMPUTATIONAL AESTHETICS LEARNING OF TRADITIONAL CHINESE PAINTING

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of School of Computer Science

2022

By
Qianqian Gu
Computer Science

Contents

Word Count: 48586

List of Tables

List of Figures

Acronyms

ML Machine Learning

DL Deep Learning

CNN Convolutional Neural Network

TCP Traditional Chinese Painting

SW Sliding Window

HOG Histogram of Oriented Gradient

SVM Support Vector Machine

RPN Region Proposal Network

A-RPN Assembled Region Proposal Network

NST Neural Style Transfer

Deep CNN Deep Convolutional Neural Network

CBN Complex Backbone Network

LBN Lightweight Backbone Network

ConvNet Convolutional Network

Conv Convolutional (Layers)

BN Batch Normalisation

RoI Regions of Interest

IoU/IOU Intersection over Union

FC Fully-connected (Layers)

FCN Fully Convolutional Network

IOB-NST Image-Optimisation-Based Online Neural Methods

MOB-NST Model-Optimisation-Based Offline Neural Methods

MRFs Markov Random Fields

PSPM Per-Style-Per-Model (Neural Methods)

MSPM Multiple-Style-Per-Model (Neural Methods)

ASPM Arbitrary-Style-Per-Model (Neural Methods)

FCN Fully Convolutional Networks

VGG Very Deep Convolutional Networks

Abstract

This project presents the idea of constructing a Computational Aesthetics Learning System in the Traditional Chinese Painting(TCP) domain. The system tends to imitate the human visual system of processing TCP data and related aesthetic information. To achieve this target, four individual tasks are set. They are feature extraction and analysis, image classification, object detection, and neural style transfer in the TCP domain. Four components were designed to consist of our computational aesthetic system.

First, an image representation descriptor combines both hand-crafted and deep learning representations by employing Hash mapping with Hamming distance and similarity thresholding. This descriptor interprets TCP domain knowledge by applying Sobel's kernel to HOG, and TCP colour palette based on Basic Color Terms [7]. Second, a set of TCP style classifiers, both hand-crafted ML (SVM) and DL (VGG) models, are introduced to distinguish TCP styles and content schools by using the image representations resulting from the stage-one descriptor. The classification process allows us to verify which models and which features are more sensitive to TCP so that we can integrate them to train the backbone network for later use. Next, a two-stage Deep CNN object detector is proposed to improve our aesthetic learning system's ability to recognise objects of TCP artworks. Objects of art are the basis of aesthetic analysis. The Assembled Region Proposal Network (A-RPN), which we promoted, enhanced a general RPN by a CN-Kitten RPN with transposed convolutional feature information. The RoI pooling layer is replaced by a top rank-voting model and position-sensitive score maps. This model is sensitive in detecting small objects and incorporating translation variance. The A-RPN significant outperformed YOLO 2 and original Faster R-CNN with p-values of as $p<0.001$ and $p=0.008$ when their backbone networks were pre-trained on the natural image data. By applying TCP domain knowledge to initialise the backbone CNN, our A-RPN increased around 7% in mAP. The IOU detection rate is also improved by 20%. Generating TCP images is one of our aims for computational aesthetics study. A patch-based neural style transfer (NST) algorithm is proposed for the final stage of our TCP computational aesthetics learning system. The TCP NST algorithm calculates the style loss, both global and local. The model introduces Target Mask Matrix to obtain more precise semantic segmentation so that more accurate content loss is computed. Our model generated competitive TCP artworks.

The data used in this project consists of the online museum collections and high-resolution images licensed by the Chinese Painting and Calligraphy Community. We manually annotated and constructed four datasets based on TCP styles, content schools, landscape with dynasties, and objects.

Declaration

No portion of the work referred to in this thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I would like to show my sincere appreciation to my supervisor Dr. Tim Morris for his generous help, invaluable guidance, moral support, and encouraging comments. Without his supervision, I would never finish my P.h.D. I also gratefully acknowledge the professional advice, sympathetic attitude, strong motivation, and patient communication from my supervisor Prof. Ross D. King. Learning from him is an unforgettable experience.

I would also like to thank my co-supervisor Prof. Xiaojun Zeng, for his precious advice and support in these years. Many thanks to Prof. Gavin Brown for providing resources and advice in completing a P.h.D. I appreciate my undergraduate tutor Dr. Toby Howard, who is also my life mentor.

I am so grateful for everyone in our MLO family, for all of those times you stood by me and encouraged me.

I have shared numerous happy and sad moments with my friends. Some of them might be thousands of miles away, but they all gave me the energy to keep going. Thank you to my idol Lay, who always takes his work seriously and doing it well and rapidly.

My deepest gratitude goes foremost to my beloved parents. I want to dedicate this thesis to them. Their endless support and continual encouragement brought me the confidence to complete my research study successfully, especially when I decided to resign from HSBC and step back to the university. The happiest thing in the world is to be their daughter.

Chapter 1

Introduction

One of the greatest mysteries in cognitive science is the human visual system's architecture and its virtuosity in compiling a set of visual tasks. Completing a visual task like the human brain is complicated, such as depth perception, edge detection, object tracking, etc. It is taken for granted that the human brain scans the environment and positions objects when things are in view. The human brain holds secrets, which underline its impressive ability to recognise a visually-presented target at lightning speed with high accuracy. For a long time, researchers may never have thought of creating a computational system that imitates the human brain's procedures. However, in the past 50 years, they have moved from observing seemingly small breakthroughs in neuroscience to describing scenes in images with "computers".

There are two competing principle's of human perception. One is the the top down approach proposed by Richard Gregory [11]. Another is the bottom-up

principle which introduced by the Nobel prize winners of 1981, Hubel and Wiesel [12]. They discovered the visual system concerning information processing and indicated that the visual cortex is hierarchical. This laid the foundation for many research studies about human vision and computer vision.

The bottom-up approach suggests that human vision starts with the ingestion of raw signals at pixel level. The cells in the cerebral cortex preliminary process information to find features, such as edges and directions. The final step is that the brain determines the object's shape when seen. Finally, further abstraction is specified with the decision.

Extending the work of Hubel and Wiesel, the computational neuroscientists Olshausen and David.J.Field, promoted a sparse grating model to represent visual features using gradient descent to minimise the number of coefficients (figure 1.1). The sparse coding encode and decode how the human brain simple cell simulate sensory respect to information in 1996 and 1997 [13, 1, 14].

When the Convolutional Neural Network (CNN) entered a new era, Olshausen and David.J.Field's assumption made in 1996 became a reality. Inspired by and similar to their sparse neuroscience model, Andrej Karpathy and Li Fei-Fei demonstrated a multi-layer recurrent neural network [2] that provided an image's description (figure 1.2). The recurrent neural network recognises primary image features at lower layers and concatenates features through multiple levels. The convolutional layers perform similar actions as human cortical visual areas from

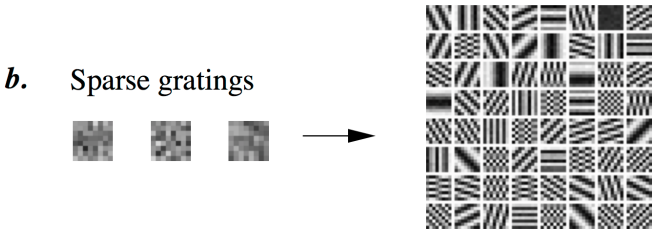

Figure 1.1: *Olshausen and David.J.Field's sparse model in computational neuroscience representation [1]. Sparse grating is a computational strategy that allows brains to encode sensory information using a small number of simultaneously active neurons at a given time. The image on the left shows the optimized sparse representations of natural scenes in the receptive fields that emerge the simple cells of the primary visual cortex. The image on the right shows the principal components calculated on 8×8 monochromatic image patches extracted from natural scenes in higher cortical visual area.*

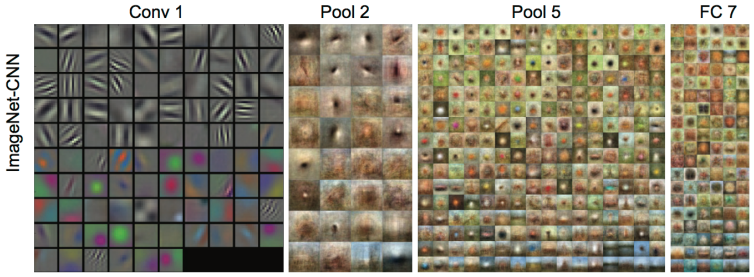

Figure 1.2: *Deep visual-semantic alignments for generating image descriptions [2]*

lower level to higher level. The final combination constructs an image representation of the corresponding image for classification. In another word, Karpathy and Li's CNN structure provides machines with the ability to compute similar visual representations as the human brain (bottom-up model) in mathematics.

This set of milestone papers drove the research direction of "computational thinking" from neuroscience to computer vision in the past fifty years. Computational thinking is a set of problem-solving methods designed to get a machine to explain or interpret a human's complex knowledge. When computational thinking is applied to computer vision, a new subject is produced. Based on

computational thinking, neural networks, more precisely CNN, go beyond image classification. They can be employed in several computer vision tasks, such as object localisation, object detection, semantic segmentation etc. They all represent Computational Vision. When we try to cross-domain between computational vision and art aesthetic, Computational Aesthetic [15] is defined.

Computational Aesthetic [15] is defined as the research of computational methods that can similarly make appropriate aesthetic decisions as humans can. It aims to use machine to learn what features or visual saliency that can develop aesthetic measurements and use them to test aesthetic relevance problems.

The human visual system can detect visual saliency extraordinarily fast and reliable from aesthetic attributes and prior feature knowledge of the objects. But computational methods do not have the basic intelligence to replicate this behaviour. Visual saliency detection remains a challenge in the computer vision domain. However, in paintings, aesthetic attributes and features that affect aesthetic decision making can find their reflections in the computer vision domain [16].

Following the definition of art [17], the research units of computational aesthetics can be ranged from basic features to general representations. Basic features describe low-level visual concepts, such as colour, edges, light and shadow. General representations provide structures that dispatch favourable high-level visual concepts. They are comprehensive compositions that can be divided into plane formation, colour formation and three-dimension composition. Each of

them could be considered various couples of attributes. These aesthetic attributes and features have their interpretations in the computer vision domain as either low-level pixel values or high-level feature representations in computer vision study. For example, the plane formation could be a skeleton feature vector that is formulated from hand-crafting features such as lines, edges, shapes; and the colour formation could be a colour histogram that describes low-level features such as colour and texture.

Aesthetic features play critical roles in machine computational aesthetic analysis [15]. But what features contribute effectively to computational aesthetics learning? And how do these sets of features react in various learning tasks when the machine tries to "understand" paintings? These are the questions we try to answer in the later sections. Moreover, in computer vision, objects can be represented by clusters of features, and images can be segmented into various categories of objects. Therefore, doing computational aesthetic learning can be unified into feature level, object level and image level analysis. In this thesis, we attempt to address this new discipline in the Traditional Chinese Painting domain from these three different levels, respect to chapter 4, chapter 5 and chapter 6.

1.1 Motivation

The challenge in Computational Aesthetic

One major motivation for studying computational aesthetics [15] is to improve tools or algorithms used in CS to have a greater awareness of good aesthetics. To

achieve this target, researchers allocated a strict application plan and a pragmatic view of the measurement of computational aesthetics. There are three main aspects [15]. First, computational aesthetics aim to develop computation systems for aesthetic decision-making. Second, human perception should be taken into account. Third, it is essential to focus on formal aesthetics (style) rather than associations between content and human mental level. However, to satisfy all aspects simultaneously throughout the imitation of the human aesthetic decision-making process is very difficult, especially the second and third. The reason for this situation is because of the definition of aesthetics.

Human beings make aesthetic judgements about art based on knowledge acquired over millions of years of Darwinian evolution. The presentation of visual art dates back to the Paleolithic age. The word "aesthetic" is of Greek origin and means "perceiver". According to Kant's [18] definition, aesthetics is the reinforcing supplement to logical ideas or concepts. The quantification of aesthetics is complex. Humans have diverse opinions about whether an object of higher(additional) value is beautiful, fairly beautiful or not at all. The aesthetic satisfaction level of one particular object can vary among different individuals. This relies on personal experience and cultural background. Also, aesthetics is ergonomics. This might not be obvious in paintings but is in commercial design. For example, clothes are designed to be worn for comfort but they also have a "fancy" appearance. In other words, good aesthetics need to meet satisfaction in terms of both psychological and physical requirements.

Unfortunately, computer science does not have built-in internal logic between aesthetics, ergonomics and psychology. This leads to the fact that computers cannot possess dedicated aesthetic judgement and design in every aspect. In order to obtain better computational aesthetic learning progressively, an entry point was suggested [15]. Objects of art are aesthetically versatile, so they are promoted as an explorative basis for analysis.

There are several reasons for choosing objects of art as the fundamental research unit of computational aesthetics. As introduced in the previous section, objects of art are comprehensive compositions of aesthetic attributes and features. They form general representations that deliver high-level visual concepts which affect aesthetic decision-making. Moreover, aesthetic attributes and features can be interpreted as computer vision features. This offers the ability that every object of art can be described and analysed as a set of computer vision features. Once the object of art has found its representation in a computer vision model, domain knowledge might be able to be applied to the computational aesthetic system to imitate human perception. However, the machine learning process requires a large amount of data for training and testing purposes. Limited resources in art might prevent the model from being expanded. Hence, whether the computational system can "learn" the formal aesthetics (style) of the objects of art remains a question.

The limitation of DL in Computational Aesthetics

With the widespread concept of Deep Learning, people have started thinking about how similar the internal representations of artificial neural networks (ANNs) are to the neural ones measured experimentally. ANNs are designed to imitate the primate brain when dealing with object recognition. Martin Schrimpf et al. [19] introduced a composite of multiple neural and behavioral benchmarks, known as Brain Score, to explain how similar a ANN is to human brain mechanisms for object recognition in 2018. They compared 61 ANNs with a set of integrating experiments, then followed specific steps to obtain the Brain Score for each and the top-1 accuracy of ImageNet performances. Based on their research, the highest a Brain-Like ANN ever scored were Densenet-169 (0.549; 75.9% top-1), CORnet-S (0.544; 74.7% top-1) and ResNet-101 (0.542; 77% top-1), while the lowest is SqueezeNet1-0 (0.454; 57.5% top-1). Most ANNs were able to achieve approximately 70% accuracy regarding top-1 accuracy of ImageNet performances, with the highest at 82.9% (PNASNet-Large scores 0.528) and the lowest at 47.64% (best basenet scores 0.5). Schrimpf further expanded their work with an integrative benchmarking platform to accurately explain domains of human intelligence as executable, neurally mechanistic models [20] in 2020. This platform allows researchers to submit ANNs so that they can obtain the respective Brain Scores. And the current top three models submitted are VOneResNet-50, VOneCORnet-S and ResNet-152.

The DL structure promoted by Karpathy and Li [2] allowed machines to interpret images into similar visual representations to the sparse neuroscience

bottom-up model [1] in mathematics. This model made DL powerful for studying computational aesthetics [15]. To be able to carry out the research, the DL model should have the ability to identify objects of art, i.e. the computational aesthetic basis.

DL has proven its success in natural image object detection [21, 22, 23]. However, is it true that DL recognises an object as a human can? What will happen when these models are applied to paintings rather than to a natural image? Unfortunately, there are few computational aesthetics or object recognition approaches in this area. Elliot J. Crowley [24] provided a response in 2016. He ran an experiment on the western paintings dataset from UK Art with both natural image-trained classifiers and painting-trained classifiers. The results showed that natural image-trained classifiers are inferior to painting-trained classifiers when recognising objects of art in paintings. But the limitation of art forms and styles in his data meant that there was not enough evidence to render this statement true for all types of paintings. DL's transferability between the natural image and art domains remain uncertain.

Limited research in the TCP domain

Traditional Chinese Paintings (TCP) play a significant role in East Asian cultural heritage, have their own unique identification, and most of them have been recognised as world treasures over decades. Different art forms and styles do not have a considerable effect on human visual-base decision making. But the significant difference between Western paintings and Chinese paintings might

result in opposite outcomes in machine object recognition tasks. Inspired by Crowley's research [24], Gu [25] discovered that DL's performance on TCP object detection did not meet the expectation. A natural image-trained Faster R-CNN can achieve at least 63% mAP of Western painting object detection, while the highest mAP in TCP is around 59%.

Since related studies in the TCP domain are rare, it became our primary motivation to devote our research in this domain (check section 3.4 for details).

1.2 Hypothesis and Objectives

In this research, the goal is to deliver a computational aesthetics learning system in the TCP domain. This system is designed to simulate the process of human cognition and learning regarding TCP's formal aesthetics (style) and objects of art (object). The system is divided into four components. Each of them corresponds to one computer vision task: image representation, classification, object detection, and style transfer. The hypothesis of this project is stated as the following:

- The formal aesthetics style of TCP can be learnt.
- The object of art in TCPs can be recognised.
- The machine can use the art style and content knowledge of TCP to generate TCP artworks.

Respectively, the objectives of this project are stated below:

- To investigate effectiveness of features when constructing the precise image representation of TCP.
- To develop TCP Classifiers when distinguishing TCP's style and content.
- To propose a TCP object detector that can recognise the basis (content element) of computational aesthetics.
- To establish a TCP's style sensitive style transfer algorithm.

These four objectives are implemented in steps. Contributions obtained during the research are summarised in the next section.

1.3 Contributions

This project aims to investigate whether Deep Learning (DL) image analysis performs relatively well as human image analysis on Traditional Chinese Paintings(TCPs). The goal is to simulate the process of learning computational aesthetics in the TCP domain. This target is achieved by designing a computational aesthetics learning system with TCP domain knowledge across the model. The contributions obtained are outlined as follows:

- Feature extractor and image representation descriptor are established and improved by applying TCP domain knowledge. Basic Color Terms [7] in culture learning is introduced to create a TCP colour palette encoding process. A mathematical transformation of Hash mapping with Hamming distance

and similarity thresholding is defined. This offers the chance to concatenating hand-crafted representation and DL representation.

- Four self-identified and manual-annotated datasets are constructed. Each of them is employed in end-to-end training and testing with a set of TCP classifiers. Both hand-crafted classifiers and DL classifiers perform well when distinguishing TCP styles and their content schools. They also provide competitive classification results when recognising objects and dynasties.

- A new DL object detection method A-RPN (Assembled Region Proposal Network), is introduced. A-RPN is an enhanced RPN (Region Proposal Network) [26], followed by an R-FCN (Region-based Fully Convolutional Networks) [27]. The A-RPN replaces the general RoI pooling layer with a top rank-voting position-sensitive RoI pooling layer. It achieved lower coarseness and higher sensitivity of small object detection by concatenating low-level visual features and high-level semantic features. It managed to beat some state-of-the-art object detectors (p-value for YOLO 2: $p<0.001$, for Faster R-CNN: $p=0.008$) at both image recognition on the natural image (Pascal VOC2007, mAP (mean Average Precision) for A-RPN: 75.25%, mAP for YOLO 2: 71.37%, mAP for Faster R-CNN: 73.22%) and TCP images (mAP for A-RPN: 61.85%, mAP for YOLO 2: 58.48%, mAP for Faster R-CNN: 59.98%). The A-RPN can achieve 69.18% mAP when employing TCP cross-feature image representation.

- A TCP Neural Style Transfer model is proposed to calculate the combination of local style loss, global style loss, and Target Mask Matrix content loss. This TCP NST model enables more precise content and style representations to generate TCP artworks.

- A TCP computational aesthetics learning system is delivered. The system diagram is shown in figure 1.3. All CNNs in the model share the same TCP image representation. Both A-RPN and TCP NST are initialised with the TCP pre-trained VGG backbone networks in chapter 4.

Our TCP computational aesthetic learning system imitates the human visual system of learning TCP and making related aesthetic decisions, such as generative art. Though it cannot achieve a qualitative aesthetic judgement, it recognises objects of art and re-renders them in TCP style.

These contributions have resulted in the following publications:

- Qianqian Gu and Ross King. Deep learning does not generalize well to recognizing cats and dogs in chinese paintings. In International Conference on Discovery Science, pages 166 - 175. Springer, 2019.[25]

- Qianqian Gu, Tim Morris and Ross King. Computational Aesthetics: On Recognising and Drawing Kittens in Chinese Paintings. In Visual Science of Art Conference 2019 (Poster Presentation)

(Each chapter has an achievement list at the end of the chapter's summary.)

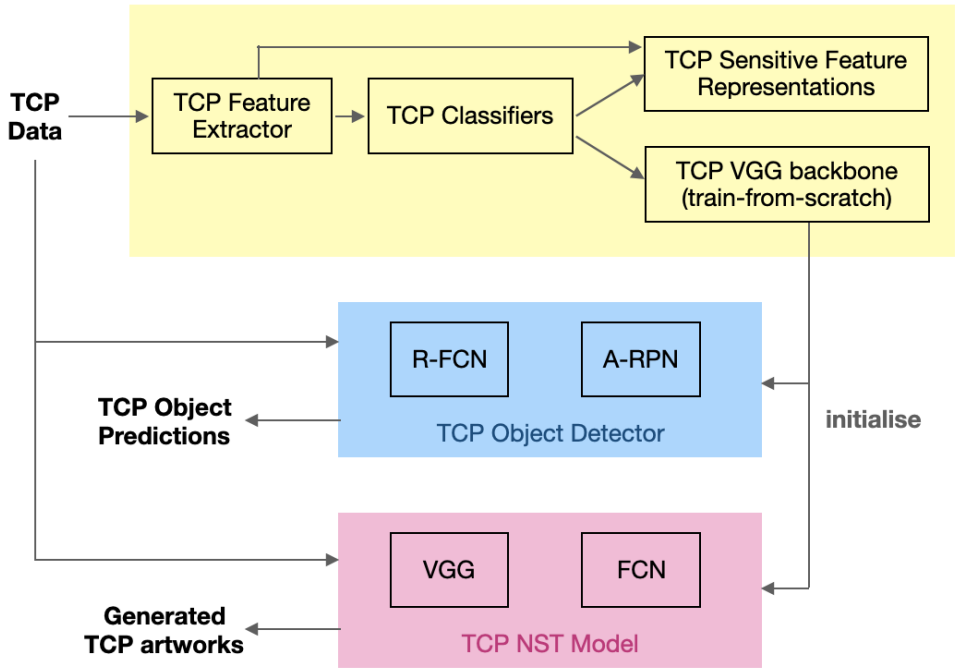

Figure 1.3: *This system diagram is designed for our TCP computational aesthetics learning system. The system can: extract TCP image representations, distinguish TCP styles, recognise objects of TCP art, generate TCP artworks. In the diagram, the most practical components have been listed. The yellow module (chapter 4) shows the procedures of how the system extracts TCP sensitive features. After TCP style classifications, the system obtains a train-from-scratch VGG backbone that can be used for the initialisation of the TCP object detector and NST model. The blue module is the TCP object detector which is constructed by A-RPN and R-FCN (chapter 5). The pink module (chapter 6) is the TCP NST model, whose major components are an FCN that computes the semantic masks, and a VGG used to perform the style transfer.*

1.4 Thesis Overview

Here is a summary of the rest of this thesis:

In Chapter 2, we introduce the background of computational aesthetics and traditional Chinese painting. We first provide the formal definition of computational aesthetics. Next, we explain why making an aesthetic decision is difficult for a machine. Then we describe the Art Composition Rule that is a set of mathematical theories representing objects of arts. Last, we give an overview of traditional Chinese paintings.

In Chapter 3, we provide an overview of the essential background knowledge of our project in the computer vision domain. We first review the definition of low-level features and two popular types of image representations: hand-crafted representation and deep learning representation. Then, we give an overview of visual recognition tasks and explain object detection in detail, i.e. explanations about both hand-crafted machine learning models and DL models. Next, we describe what generative art and style transfer are. Last, we review existing computer vision study in the TCP domain.

In Chapter 4, we introduce TCP cross-feature between hand-crafted image representation and DL representation; simultaneously, we obtain a pre-trained backbone CNN with TCP knowledge applied. We first construct four TCP datasets. Then, we attempt to investigate the difference between various feature extractors and image representation descriptions. Next, we examine all combinations of feature extractors and image representations through the traditional

ML classification model and DL classification model. Last, experiments are conducted to validate the effectiveness of the proposed image representations and classifiers for TCP style recognition.

In Chapter 5, we present a two-stage Deep CNN object detector A-RPN. This novel framework is composed of a general RPN and a CN-Kitten RPN with transposed convolutional feature information. The RoI pooling layer is replaced by a top rank-voting model and position-sensitive RoI pooling layer. We examine the framework's ability to incorporate translation variance and small object sensitivity. Extensive experiments are carried out on the natural image and the TCP data.

In Chapter 6, we present a patch-based TCP neural style transfer (NST) algorithm. We first redefine the NST style loss function as the combined calculation of global style loss and local style loss. Then, we introduce a target mask matrix to calculate the area-controlled content loss. Experiments are conducted and exhibit several generative TCP artworks.

In Chapter 7, we draw conclusions from our research and discuss the limitations and potential improvements for future work.

Chapter 2

Background of Computational Aesthetics and Traditional Chinese Painting

Computational Aesthetics is a set of cross-domain computer vision tasks. Research study related to Computational Aesthetics focuses not only on natural image processing but also on the bridge between art and computer vision.[15] This project aims to develop a computational aesthetics system that can make similar aesthetics decisions on traditional Chinese painting as humans can. This chapter aims to provide a brief introduction to aesthetics concepts in art and TCP background. Also, this chapter presents an overview of the relationship between human aesthetics and computational aesthetics. It provides an introduction to Computational Aesthetics in section 2.1.1 and how it relates various computer

vision tasks under this research category in section 2.1.2 and section **??**. Section 2.2 offers background knowledge of TCP and summarise related research on computational aesthetic in the TCP domain.

2.1 Background of Computational Aesthetics

2.1.1 What is Computational Aesthetics

Computational Aesthetics' history can be traced back to 1933 since George David Birkhoff wrote the first quantitative theory of aesthetics in his book Aesthetic Measure [28]. Birkhoff's formula suggested that the complexity of paintings is considerable so that the content of a painting is analogous to a fine composition of ornamental patterns. These ornamental patterns are objects of art that must be appreciated one by one for comprehension. After then, several researchers tried to develop a solid approach or methodology for computational aesthetics, but they only made apparent aspects of aesthetics, such as order and complexity. The historical summary on this could be checked in Gary Greenfield's notes [29]. With many disciplines showing interest in aesthetics of computer science, the final proper definition was first introduced to the Computer Science research area by the first Eurographics(EG) Workshop on Computational Aesthetics in Graphics, Visualisation and Imaging in 2015 [15].

Computational Aesthetics is defined as the research of computational methods that can similarly make appropriate aesthetic decisions as humans can.

The general approach for computational aesthetic involves investigating a collection of features to developing aesthetic measurements and integrating them to test aesthetic relevance problems. It is also a research task that refines its incompleteness and defines new findings.

2.1.2 Aesthetics in Computational Aesthetics

In the human aspect, aesthetics is the philosophical study of beauty and taste. Humankind can easily distinguish beauty according to experiences. However, the true meaning of aesthetics can be various from individuals' subjectivity. Moreover, proved by philosophy study, the judgment of taste is affected by subjectivity and normativity and recasting normativity [30]. Normativity implies an "ought-type" statement, which is oriented to effecting an action without describing true or false [31]. Measuring one individual's aesthetic preference is easy, but a group of people's would be challenging. To find a universal guiding principle or an absolute standard in matters of artistic beauty and taste is nearly impossible. This statement leads to a consequence that the significant challenge of computational aesthetics is how to evaluate the validity among all and provide logical and reasonable judgements, respectively.

Art grows from conflict of contradictions. Even experienced artist or academics could not describe a full explanation of it. There are at least 62 well-known art movements in the world nowadays. Some of them are representing opposite ideas for example, Constructivism and Deconstructivism. Constructivism uses

Euclidean geometry to reflect the suprematism spirit and focus on the movement within the space [32]. While, Deconstructivism uses symbols to reflect the truth, showing the antipathy of structure, which initially came from Martin Heidegger's Einführung in die Metaphysik (Metaphysics) [33]. Paintings of various art movements vary in their art forms and styles. Simultaneously, each particular form and style have their visual representations in computational aesthetics analysis, respectively.

Human holds various standard on making aesthetic judgement while facing different art movements or styles. Machine learning is designed based on similar pipelines as human learning but has dependencies on the training of art movements or styles. It is challenging to have only one solution to satisfy all types of art movements or styles when designing a computational aesthetics model. Accordingly, when solving computer vision tasks for computational aesthetics study, every approach should be designed and evaluated along with a particular art movement or style.

2.2 Traditional Chinese Painting

As a part of significant East Asian culture heritage, Chinese painting has its unique identification. Chinese painting could be categorised into multiple groups based on different classification standards [34]. The most widely used standard to classify Chinese painting is to divide them according to painting techniques. According to this standard, the paintings could be categorised into two primary

schools of styles: either XieYi or GongBi, which means freehand strokes and skilled brush, by justifying their brushwork styles [35]. For more elaborations, the XieYi school is represented by exaggerated forms and freehand brushwork, which means the painted objects are abstract and sometimes without fully connected edges. Freehand painting generalises shapes and displays rich brushwork and ink techniques. While the GongBi, or meticulous approach, is characterised by close attention to detail and fine brushwork, which means the painted objects are more realistic.

TCP always specialises in tools and materials, consisting of brushes, ink and pigments, xuan paper, silk and various types of ink slabs. There are several principal forms of traditional Chinese paintings. They are the hanging scroll, album of paintings, fan surface and long horizontal scroll. Hanging scrolls are usually mounted and hung on the wall. They can be either horizontal or vertical. Most of the existing TCP art pieces are in this form [34]. The album of paintings is a format that is designed to store conveniently. Artists paint on a specific size of xuan paper and then binds several paintings to create an album. Fan surface is a typical form for folding fans and round fans made of bamboo strips with painted paper or silk pasted on the frame. The long horizontal scroll is also called a handscroll. It is usually less than 50 centimetres high but can be up to 100 meters long. Most of the world-famous TCP art pieces are presented in long horizontal scrolls. All of the above variants of TCP specialisations increase the complexity of solving computer vision tasks in TCP domain.

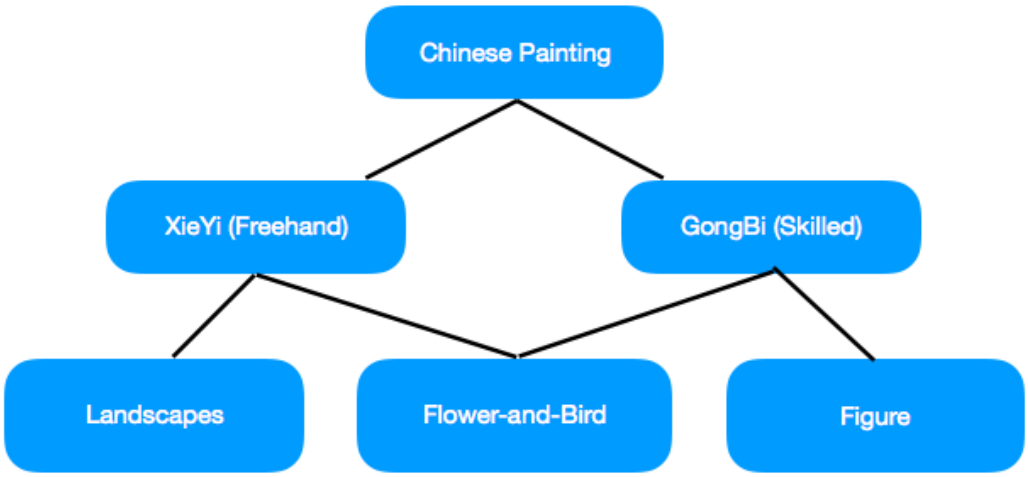

Figure 2.1: *Chinese Painting is divided into either XieYi or GongBi according to painting techniques, and be further classified as Landscapes, Figure and Flower-and-Bird*

TCPs can be further classified as landscapes paintings (figure 2.3), figure paintings (figure 2.4), and flower-and-bird (figure 2.2) paintings based on their contents. Existing work has been done on defining ontologies and classes for TCP [36]. Overview of TCP category is shown in figure 2.1.

Landscapes paintings are mainly depicting mountains and rivers' natural scenery. Representing a major category in TCP, landscape painting had established itself as an independent expression form by the fourth century. In the later centuries, landscape paintings gradually branched out toward two distinct styles: blue-and-green landscapes and ink-and-wash landscapes. The blue-and-green landscape paintings use bright blue, green and red pigments, and it is a fashion in Tang (618 to 906 A.D.) and Song (960 to 1279 A.D.) Dynasties [34]. The ink-and-wash landscape paintings rely on vivid brushwork and inks, and it is popular in Ming (1368 to 1644 A.D.) and Qing (1644 to 1912 A.D.) Dynasties.

The forms of expression in landscape painting changed between dynasties, as the range of subjects did in figure painting. The range of subject matter in figure painting was extended far beyond religious themes during the Song Dynasty between 960 to 1127 A.D.. Flower-and-bird painting originated from within decorative art to form its own independent genre around the ninth century. Both figure paintings, and flower-and-bird paintings, depict objects as their main subject.

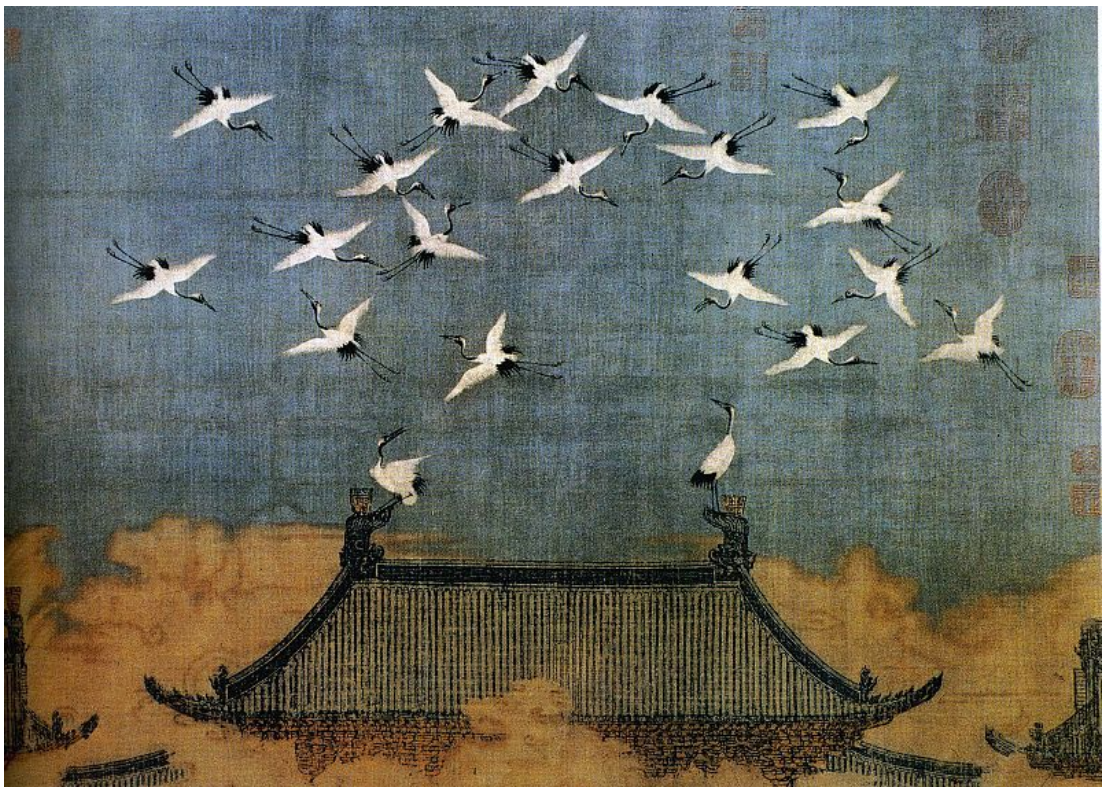

Figure 2.2: *Emperor-Huizong-Song's flower-and-bird painting from Song Dynasty*

Chinese Ancient believed that "Painting in poetry and poetry in the painting". This is the criterion for excellent TCP artworks. A TCP art pieces should consist of all necessary components: traditional Chinese painting, poetry, calligraphy, painting and seal engraving. Inscriptions and seal impressions help explain the

painter's ideas and sentiments, and add beauty to the painting. Every component in TCP is that supplement and enrich one another. However, in this project, we only focus on the painting part of the TCP but not the others.

Figure 2.3: *Landscape paintings. From left to right: ordered by dynasties Tang, Yuan, Ming, Qing. The first and third paintings are ink-and-wash landscape paintings, while the second and fourth are blue-and-green landscape paintings*

Figure 2.4: *Figure painting <Yang Gui Fei Mounting Horse> from Tang Dynasty*

Chapter 3

Literature Review

Computational aesthetics is a research field belonging to machine learning and computer vision. The main focus of this chapter presented here centres on reviewing a number of research hot-spots linked to this topic. Since the core of computational aesthetics is to employ suitable computational methods that allow the machine to imitate the human visual system and how it makes aesthetic decisions. A number of computer vision tasks are employed when studying computational aesthetics. However, it is impossible to cover all of them in our project. Our project concentrates on three sets of computer vision tasks. There are extracting image representation, visual recognition (classification and object recognition) and generative art. This chapter reviews each of them in the coming sections.

3.1 Image Representations

As mentioned in section 2.1, when doing research on computational aesthetics, every single type of paintings should be treated individually from its form and style to its content. And the approaches for selection in features and attributes for computational aesthetic analysis vary in terms of domains and applications. In this thesis, we centred on our applications into the domain of TCP and feature evaluations on how machines learn TCP. These research trends of computational aesthetics can be employed in areas as image representation, visual recognition and generative art.

Image representations are visual concepts denoted by various meaningful features computed from raw data by multiple methods. One of the keys to the computer vision problem is to extract meaningful high-level visual information from the low-level pixel values from graphical data and evaluate how they perform in computer vision algorithms. Image representation is the mathematical representation defined in computer vision that describes aesthetic features knowledge that might affect aesthetic judgements. Low-level features, such as colour and texture, underlying the raw data can be adequately characterised as high-level concepts. High-level concepts are skeleton, shape, object, motion and e.t.c. They form image representation. In computer vision domain, image representations can be generally grouped into two major categories, hand-crafted ones and deep learning ones. Researchers have devoted many efforts to hand-crafting variant features to obtain more precise image representations when analysing computer

vision research topics. In this section, we present reviews on the low-level feature, hand-crafted representation and deep learning representation.

However, with the wide-spread of Deep Learning(DL), deep learning representations have outperformed hand-crafted representations across most areas in the computer vision community. Hand-crafted representation has its limitation. The computational cost of model training and the demand for large-scale data can prevent hand-crafted representations from being routed. In that case, deep learning representations can push it closer to the line and provide a transferable ability to achieve success across computer vision tasks. But is it true that deep learning representations beat hand-crafted representations? The answer is no. There are many reasons. One of them is that deep learning representations require a great demand for training data, opposite to hand-crafted representations. This has been validated in our research in chapter 4. Therefore, will combining DNN with hand-crafted features be useful to improve the learning process? Bogdanova et al. [37] stated that "if you cannot beat them, join them". Hand-crafted representations incorporate domain knowledge during selections. Complementary information provided by hand-crafted representations can enhance the deep learning representations. In some particular model architectures, hand-crafted representations can even guide deep learning. For example, since the optical flow plays influential roles in hand-crafted video representations, the two-stream framework [38] for human action recognition feeds optical flow data into the temporal network for improvement. Researchers employ dense trajectories in local representation

to the pooling process of deep features to obtain better performance [39, 40].

Therefore, we introduce and evaluate both hand-crafted and deep learning representations in our computational aesthetic learning models.

3.1.1 Low-level Feature

The low-level features are relatively stable information which are measured directly from images. They give a similar image capture environment and help to define secondary qualities, which are properties that produce sensations in observers [41]. They can be affected by the camera's setting. They are the fundamental elements to construct an image representation. One important task of computer vision is to rebuild the truth from these secondary qualities to extract more precise image representation. Therefore, colour and texture are briefly introduced in this section.

Colour is a fundamental element in photographs and is an essential low-level feature in image processing. Colour is not only a secondary quality, but also an intrinsic property of a solid object [41]. It provides local information about objects which can be used in object recognition and segmentation. Research papers show that image segmentation can be achieved by utilising colour features [42, 43]. As the original reflection of the ground truth object, an object's colour appearance is mostly stable. However, some attributes can affect colour appearance. Since the visible light with different wavelengths defines the colour, different viewpoint

and illumination can affect the colour appearance [44]. The complexation resulting from illumination conditions and texture can lead to a change in a colour's appearance. Therefore, it is common to construct a colour-texture descriptor to better represent low-level information on the retinal image in computer vision. A colour-texture descriptor is a general approach to determine the pattern's relationships within the retinal image and reveal the hidden information. Review of descriptor is given in section 3.1.2.

Texture represents the reflection over closely composed elements such as similar intensities or patterns [45]. There are numerous methods of texture extractions and measuring in computer vision history, but none of them helps to offer an explicit definition of texture. As one of the "old fashion" terms of texture definitions, texton presents the discriminative information from the local conspicuous features [46]. Texton used to be a kind of image representation that describes shape on the object surface or the primary fractal, such as drawing composed of line segments [47]. However, this definition has no longer been used. The general texture definition researchers adopt now-a-day is that texture is the low-level image representation that depicts a spatial mixture of surface properties by accumulating and modelling local elements.

Similar to colour feature, as a secondary quality, texture can help to refine image classification and segmentation. Feature extraction methods transfer either a single texture or a set of multiple textures into mathematic vectors. These vectors contain the local texture information of target patterns or background for an

individual pixel. This results in improving image classification and segmentation based on distance methods [48]. Moreover, texture feature also plays significant roles in Deep Learning of image challenges [49].

In this project, we apply colour-texture descriptors to TCP data for obtaining image representations that describe low-level information. This information is used to distinguish TCP categories, and the details are given in chapter 4.

3.1.2 Hand-crafted Representation

Following the traditional machine learning flow, hand-crafted image representation is the image representations constructed by a set of manually extracted features. For examples, it includes but not only histogram, edge detection, corner detection, etc. The problem with this approach is that there is no guarantee that the simple features are good descriptors for computer vision tasks.

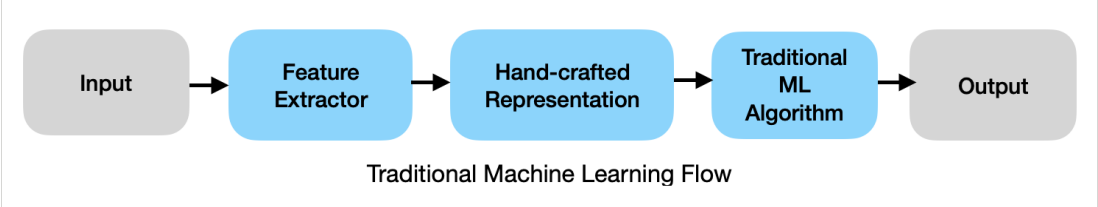

Figure 3.1: *Hand-crafted image representation in traditional machine learning flow*

The hand-crafted image representation for computer vision could be categorized into two types, global and local. The global hand-crafted image representations include but not only colour histogram [50], texture [51, 52] and shape [53]. They directly characterize the raw images or typical regions-of-interest, such as area fragmented by selections and segmented objects as a whole. This type of

representation's weakness is that these applications are only appropriate for images whose intra-class variability is small. However, in real-world applications, the intra-class variability can be huge. It is impossible to have complete control of the intra-class variability in every condition due to the diversities in illumination, perspective, background, scale, etc. On the other hand, local hand-crafted image representations identify the whole images in a bottom-up manner. Local representations denote generic features extracted from salient local regions in images [54, 55], such as regions defined by sliding windows in object detection [56]. They can be drivein from but not only colour, edge detection, threshold decomposition, the distance transform, and thresholding [55]. Local representations are more robust against variations than the global representations in general so that local representations help to achieve better performance in many computer vision tasks.

Good local hand-crafted image representations constitute three crucial components, interest-point detection, local descriptor, and encoding method. Interest-point detectors detect salient image regions, which are then characterised by local descriptors. Last, encoding methods allow information from the previous process to be regenerated into a compact feature vector, which is the local representation for the current salient image region. Every combination of these three components can drive different local representations. The mixture of multiple representations holds the potential to improve computer vision tasks in practice [57].

Interest-point detectors intend to gain salient points or regions in an image

to extract the most valuable information. It helps compress redundancy. Researchers from the computer vision community have invented numerous interest-point detectors. Laplace detector and Harris-Laplace detector [58] are two widespread interest-point detection approaches for image representations. They tend to emphasize the corners and edges by searching salient points or regions in an image. Unlike them, the dense grid [58] approach divides the raw image into grids and computes local descriptors based on each grid, preventing loss in information suffered by other detectors. In this sense, the dense grid approach can obtain image representations from local features and global ones simultaneously. Histograms of Oriented Gradients (HOG) [59] is one of the most effective hand-crafted image representations [60] employed by the dense grid approach. HOG is a feature descriptor that counts occurrences of gradient orientation in a localised area of an image. It is also one of the most commonly used local descriptors in object recognition literature [60].

As the building blocks of image representation, local descriptors derive higher-level visual features from the raw pixels' low-level information. Besides HOG[59], there are other popular local descriptors, for examples, Scale-Invariant Feature Transform (SIFT) [61], Speeded Up Robust Features (SURF) [62], Binarized Statistical Image Features (BSIF) [63], Haar [64]and Local Binary Pattern [65].

An encoding method aims to construct a compact image representation vector that union extractions of local descriptors from an image. Three methods can

succeed in this purpose: concatenation, Fisher Vector and Bag of Words. Concatenation aims to consider independent features together in a feature vector. It is the simplest and the most basic way to combine all the extracted local descriptors. But it is only feasible if and only if the number of visual descriptors and the dimension of local descriptors are moderate. Thus, it is commonly employed with dense grid approaches since the number of grids is moderate. Fisher Vector and Bag of Words are both generic processes that construct lower-dimensional compact representation vectors from many extracted local descriptors. Fisher Vector (FV) [66, 67, 68] defines an image by modelling the extracted local descriptors' distribution. The FV encoding method assumes that local descriptors of images follow Gaussian Mixture Models (GMM) so that the gradients of log-likelihood can be used to compute the single image representation by concerning the parameters of pre-learned GMMs. Meanwhile, the Bag of Words encoding method [69] cluster local descriptors into a collection of descriptors to form a codebook. Each cluster centre is representative of a distinct individual pattern of visual descriptors as a codeword. The image representations, the codebook, are assembled by the histograms of the codewords.

3.1.3 Deep learning Representation

Deep Learning(DL) is an extensively studied topic in recent years. Deep learning image representations driven from DL models have shown their strength in many

computer vision tasks compared to hand-crafted ones, especially computer recognition and segmentation [70, 71, 72, 73]. DL's identical network architectures and the diversity of DL models' tricks have brought innovation to the computer vision community, lifting the performance evolutionarily. While the PReLU-net [74] first introduced, it even outperformed human accuracy with 4.94% against 5.1% for top-5 classification error in the ImageNet large scale object classification challenge. There are many state-of-the-art DL models for image classification include AlexNet [75], OverFeat [76], VGG19 [77], ResNet [78], Google Inception V1-4 (GoogLeNet Family) [79, 80, 81]. They are the pre-trained backbone networks that can be used to draw pre-trained deep learning image representations for other computer vision tasks. As example shown in figure 3.2, the activation (ReLU) layers inference from a pre-trained deep model can serve as high-level visual features which carry meaningful visual concepts such as edges, texture, shapes and object contours etc., from image data.

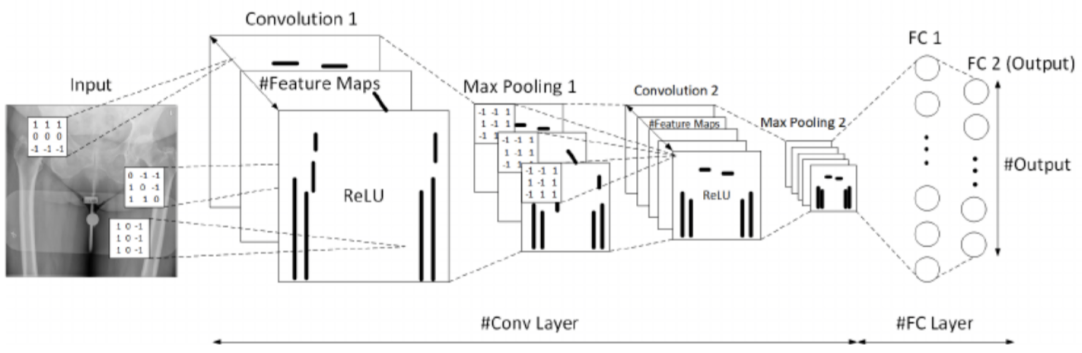

Figure 3.2: *A generic CNN architecture with intuition of low to high level feature learning with ReLU activation function [3]*

DL models are powerful but have limitations. Existing models do not translate

well into all settings. Transfer learning only works when domains similarities, and there is a large body of research investigating domain adaptation, also the demand of volumes of training data. Fine-tuning is one possible effective approach that might improve performance. Fine-tuning is to retraining the generic model as a starting point on a new data set. The fine-tune technique employed on pre-trained DL models on generating image representations might be more effective and better-fit on some datasets like the natural image dataset. However, this might not be the case for transferring current pre-trained deep learning image representation into computational aesthetics study on TCP. Related experiments and results provide an answer to our research.

3.2 Visual Recognition

Visual recognition in computer vision intends to analyze the content of images or videos automatically. It describes a collection of computer vision tasks involving identifying objects from visual data. The series of computer vision tasks are image classification, object localisation, object detection and object segmentation. This thesis mainly concerns image classification, object detection and object segmentation along with it in DL models.

Visual Recognition Task	Input	Output	Locations
Image Classification	An image with one or a set of concepts associated to it to describe its content	One or more integers that are mapped to single or multiple class labels	No
Object Localisation	An image with one or more objects	One or more bounding boxes defined by point, width, and height	Yes
Object Detection	An image with one or more objects	One or more bounding boxes defined by point, width, and height and the associated class label for each bounding box	Yes
Object Segmentation	An image with one or more objects	One or more masks mapping pixels for the same object and their associated class label for each mask	Yes

Table 3.1: *Table summary of visual recognition task that shows input data, target output and localisation ability*

Distinguish computer vision tasks can be confusing, but ImageNet Large Scale Visual Recognition Challenge(ILSVRC) in 2015 [21] provided proper outlines, seen table 3.1. Image classification refers to annotate or categorize a class label to given visual data. Object Localisation involves locating one or more objects in visual data and indicates their location by drawing a bounding box around. Object Detection is more challenging since it combines image classification and

object localisation. Object detection locates objects with a bounding box and assigns them a class label in an image. These three problems are referred to a topic that extensively studied in recent decades as Object Recognition. Object Segmentation is a task driven by object detection. Unlike bounding box location, object segmentation clusters pixels as an object mask to identify both location and class label at the pixel level. This section explains the development and state-of-the-art visual recognition algorithms from two aspects our research focus on, object detection and associated object segmentation with DL.

3.2.1 Object Dectection

Object detection is one of the most widely studied topics in image processing. It differs from the classical image classification problems, where models classify images into a single category corresponding to their most salient object. This traditional setting is problematic, as images are usually complex and contain more than one object. Therefore current object detection models generally identify various objects and locations within one single image. A naive approach to solving this problem is to take different regions of interest from the image and classify the object's presence within that region.

After the rapid development of Deep Learning(DL) in the recent decades, there are two representative pipelines for solving this problem: the traditional hand-crafted Machine Learning(ML) model and the end-to-end DL model; usually, CNNs [75, 82]. Computational aesthetics focus on modelling machine to

make appropriate aesthetic decisions similar to humans. Since DL models perform much better than traditional hand-crafted ML model, the DL overwhelmingly dominate most computational methods in this research area. Therefore, this section introduces the traditional hand-crafted ML model, in brief, and concentrates on the object detection algorithms based on deep convolutional neural networks (CNN).

3.2.2 Hand-crafted Machine Learning Model

As the most popular and successful pipeline for object detection before DL strategy fits prevalent, the traditional hand-crafted ML model is feature-based methods that use feature extraction followed by classifier training. The traditional hand-crafted ML model's basic architecture for object detection can be divided into three components: region selector, feature extractor, and classifier. It aims at the construction of image representation and the design of classifiers. The review of hand-crafted image representation has been given in section 3.1.2. And the widely-used classifiers involved in the hand-crafted ML models include support vector machine (SVM) [57, 83, 84, 85, 86, 87], random forest [88, 89, 90, 91], nearest neighbour [92, 93] and so on.

As the model we have chosen in the chapter 4 for experiments, support vector machines (SVM) [94] is the classifier widely used to produce a state-of-the-art object detection performance in hand-crafted ML models. The basic idea of SVM is to generate a hyperplane to separate binary classes with the maximised

margin. SVM is theoretically clear and convincing, and the effectiveness and efficiency in practice leads to success in traditional object detection models, such as DPM (HOG+Latent SVM) [84], Oxford-MKL (HOG+Cascade SVM [95]) [85], Selective Search (SIFT+SVM) [86], and NLPR-HOGLBP (LBP/HOG+Latent SVM/Boosting) [87], etc. All the listed example models can achieve more than 80% recall-overlap performance rate of the PASCAL dataset. While the DPM obtains AP as 0.869, Selective Search has 0.879, and NLPR-HOGLBP boosted the result by 4% as an improvement compared to the baseline.

The traditional hand-crafted ML models in object detection are relatively mature but have inherent weaknesses. There are two major shortcomings. The first one is the difficulties facing in manually designing robust features to construct effective hand-crafted image representations. As stated in section 3.1, image representations extracted from visual data have diversity and impact on object detection performance. Appearances, illumination, and background are all morphologic factors that increase the difficulty. Another weakness is the high computing complexity in the salient region selection. Methods like sliding-window [56] is computational intensive. And window redundancy in some salients also aggravate this situation.

From computer vision research aspects, the use of ensemble learning can usually boost object detection performance. Researchers keep examining building ensemble systems from different base models that are complementary [96]. Finally, Deep CNN turn researchers' attention to itself [75, 97].

3.2.3 Deep Learning Model

Localizing and classifying objects into the concerned classes is the aim of object recognition. With the increment of computing power, Deep CNN-based object detection develops rapidly [70]. Deep CNNs are robust and can learn image features from low-level to high-level [71, 72, 73]. The variety of image datasets [98, 99, 100, 101] is another key to boost object detection performance. Pascal Visual Object Classes (VOC) Challenges [22, 23] promoted object detection to a large extent by handling 20 concerned classes in one goal with VOC07 [98] 2007 and VOC12 [98] datasets. ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [101, 21] expand the class number to a large-scale with more than 1000 concerned classes.

We provide brief performances of some improved object detection models and will summarize their main ideas in the section 3.2.3.1 and section 3.2.3.2.

As an ILSVRC winner, A. Krizhevsky et al. proposed a Deep CNN model called AlexNet [75] in 2012. The AlexNet minimized the top-5 error rates to 15.3%, while a top-5 accuracy means a model's top 5 highest probability predictions match with the ground truth.. Afterwards, the PReLU activation function [74] released in 2015 boosted this results. Instead of zeroing out the negative input in ReLU, PReLU multiplies the negative input with a small value learned during training. For the first time in object recognition performance, human beings have been surpassed with the top-5 classification error less than 5% in ILSVRC by a

Deep CNN model. Deep CNN models are diverse in various backbone architectures. And this diversity further reduced the top-5 classification error in ILSVRC to 4.49% by ResNet [78] and 4.1% by Inception-v4 [81].

The success of the Deep CNN model triggered a devotion to applying deep CNN models to object detection. Based on the backbone networks of Deep CNNs, researchers divided their research directions into two categories, two-stage object detection strategy and one-stage object detection strategy.

In the following sub-sections, we will introduce two types of backbone networks in section 3.2.3.1 and section 3.2.3.2. Then we will provide a summary of the two-stage object detection strategy and one-stage object detection strategy in section 3.2.4.

3.2.3.1 Complex Backbone Network

Yann LeCun proposed the first CNN called LeNet-5 bases on his CNN study in 1998 [102]. LeNet-5 [103] is a traditional CNN designed to recognise handwritten numeric characters. It achieves an average precision of 98% on the MNIST dataset but did not overwhelm at that time. LeNet-5 determines a Deep CNN's basic architecture by specifying basic components: the convolutional layer, pooling layer, and fully-connected layer. Unfortunately, the first exhibition of CNN does not greatly impact object detection at that period, and little expansion has been made in the next decade due to the limitation of computing power. Things start changing from the time stamp when AlexNet won the championship of the

ILSVRC 2012 competition. Deep CNN attracts researchers' eyes to it.

AlexNet [75] consists of five convolutional layers (Conv), three max-pooling layers, and three fully-connected layers (FC). It expands the depth and breadth of the LeNet-5 architecture and determines 60 million parameters in total. AlexNet has excellent success in ILSVRC 2010 testing data with top-1 and top-5 classification error rates as 37.5% and 17.0%. And they managed to minimize top-5 classification error rates to 15.3% in ILSVRC 2012. AlexNet combines multiple technologies to obtain this progress. It replaces the traditional CNN activation function such as Sigmoid and Tanh by ReLu [104] to overcome gradient dispersion problem in deep networks. Dropout [105] regularization is employed to remove some neurons randomly to avoid inter-adaptive complexity and overfitting better. AlexNet also applies data augmentation methods such as horizontal flipping, random clipping, translational transformation, colour illumination transformation to extend the dataset and reduce overfitting. Furthermore, AlexNet allows Multi-GPUs to run in parallel so that which layers can communicate between each other to speed up training.

ZFNet [106] uses unpooling layers and deconvolution layers [107, 108] to visualize feature maps. The visualization ability for the convolutional layers in ZFNet intuitively describes the feature maps' changes at each layer. To preserve low-level features more accurately, Zeiler reduced the convolution kernel size of the first Conv layer in AlexNet [75] from 11×11 to 7×7 and adjusted the stride size from 4 to 2. A similar approach is employed in DetNet [109] and evidence

has shown that smaller convolutional kernels are conducive to reduce the down-sampling rate, localise large objects and identify small objects

GoogLeNet is a Deep CNN family [79, 80, 81] popular in use of two-stage object detectors. As the two-stage object detection strategy utilises low-level features to locate the object and the Deep CNN to classify the salient region, effective image representation of features can improve both location and classification performance. To achieve this target, GoogLeNet [79] increases the number of Conv layers and the number of neurons of each layer. But simultaneously, the expansion of network size, depth, and width leads to dramatic growth in the number of parameters, and the computation-intensive rises exponentially. In order to overcome this problem, GoogLeNet [79] introduces 1×1 convolution kernel proposed by Min Lin et al. [110]. The Network-in-Network 1×1 convolution kernel trick can decrease the dimension so that the computational complexity can be reduced and cross-channel information can be integrated. Furthermore, GoogLeNet [79] replace the fully connected layer with an average pooling layer and proposed Inception module [79] with dimension reductions to convert serial structure to parallel structure, so that the calculation of parameters further reducted from 7×7×1024 to 1×1×1024.

VGG enhanced AlexNet's architectures by depth to 16 or 19 layers (VGG16 and VGG19) [77] to highlight and extract features precise for generating a DL image representation. Since a smaller convolutional kernel promotes Deep CNN models performance, VGG reduces the convolution kernel size in every Conv layer

to 3×3 with the stride of 1, while AlexNet has 11×11 convolution kernel and stride of 4; and ZFNet has convolution kernel size 7×7 and stride of 2. The smaller kernel and stride of VGG increase the network's depth but keep the receptive field unchanged. It also enhances the network model's feature representation capability, conducive to object localisation by reducing parameter calculations.

CBN model	Top-1 Accuracy	Top-5 Accuracy	Params(M)	Year
AlexNet	57.20%	80.30%	60M	2012
ZFNet	64.00%	85.20%	58M	2013
GoogLeNet	69.80%	93.30%	6.8M	2014
VGG16	71.50%	89.80%	138M	2015
InceptionV2	79.90%	95.20%	12M	2015
InceptionV3	82.70%	96.50%	23.6M	2015
InceptionV4	83.50%	96.90%	41M	2016
ResNet50	79.30%	96.40%	23.4M	2015
ResNet101	80.10%	96.40%	42M	2015

Table 3.2: *CNN Summary of complex backbone networks(CBNs). The Top-1 Accuracy and the Top-5 Accuracy represent the classification accuracy on the ImageNet dataset. Params indicate the number of network parameters.*

ResNet is a deep residual learning approach for image recognition proposed by Kaiming He et al.. The purpose of this model is to ease congestion of gradient explosion and dispersion caused by degradation [78] in Deep CNN training. Degradation is a phenomenon that when a Deep CNN model's accuracy in training reaches saturation, then start to decline dramatically with the increment of

the network's depth during training. Deep CNN holds the ability and ambition on large-scale dataset handling. To gain more validated features, the depth of Deep CNN grows. Before ResNet was proposed, VGG and GoogLeNet [77, 79] employed stochastic gradient descent (SGD) trick [111, 112] to permit networks continuously converging after a certain layer, for example, ten. This SGD trick is effective but does not provide a proper solution to training degradation. ResNet invites an assumption to its model. The model assumes the input value is X, and F(X) is the result taken from one layer or multiple stacked layers, respectively. ResNet creates a shortcut connection between them as essential output H(X)=F(X)+X. This approach prevents degradation from worsening and allows Deep CNN to keep coverage despite many layers added on. The performance continuously improves after ResNet solves degradation, the increasing depth of networks (ResNet50 and ResNet101) enhance the capability of image representations from features. ResNet is one of the most popular backbone networks for object detection now-a-day, and it achieves 3.57% top-5 classification error rates in ILSVRC 2015 [21].

Except for the widely-used backbone Deep CNN architectures listed above, several complex backbone networks are inspired and integrated from them. GoogLeNet family is one example. InceptionV2 inherits the design of GoogLeNet InceptionV1 [79] and split the 5×5 convolution kernel into two 3×3 kernels with Batch Normalisation (BN) [113]. Then, Christian Szegedy et al. concatenates 1×N kernel sequencing by N×1 kernel as N×N kernel to propose GoogLeNet InceptionV3

[80]. Combining the ResNet [78] and Inception model, researchers demonstrates the InceptionResNets [81]. There are other Deep CNN backbone network such as ResNeXt [114], DenseNet (who has thousands of layers) [115], and SE ResNet [116].

A summary of complex backbone networks' performance in ILSVRC is given in table 3.2 with top-1 accuracy, top-5 accuracy and the number of parameters.

3.2.3.2 Lightweight Backbone Network

The complex backbone networks we described above focus on deepening Deep CNN models' depth and employing decomposition methods on convolution kernels to increase image representation capability. However, from AlexNet to ResNet and DenseNet, the number of Conv layers grow from 7 to hundreds, and thousands [75, 78, 115]. With the rapid growth of depth, kernel decomposition methods loss can only guarantee Deep CNN convergence but not the calculations' depreciation. The incremental change of parameters results in intensive computation, shortage of storage space, and lack of efficiency. As the core of Deep CNN-based object detection, the backbone network defines the model's calculation and memory requirements. If the visual data involved is an offline image dataset, the Deep CNN model can digest this problem by consuming more time in practices. But online video datasets do not have a tolerance on these issues. Therefore, researchers invented lightweight backbone networks to replace

the complex ones to ensure performance and limit the number of parameters simultaneously. The computer visions tasks employed with this area cover but not only autonomous driving [117, 118], face recognition [119], embedded systems [120], mobile phone searching [121, 122] and e.t.c.

LBN model	Top-1 Accuracy	Top-5 Accuracy	Params(M)	Year
SqueezeNet	57.20%	80.30%	1.25M	2016
Xception	79.00%	94.50%	22.8M	2017
MobileNetV1	70.70%	89.50%	4.24M	2017
MobileNetV2	72.00%	91.00%	3.4M	2018
ShuffleNet	71.50%	-	3.4M	2017
NASNet-A	74.00%	91.60%	5.3M	2018
PeleeNet	71.30%	90.30%	2.8M	2018
SqueezeNext	67.50%	88.20%	3.2M	2018
MnasNet	70.60%	89.50%	4.2M	2018
MnasNet-92	74.80%	92.10%	4.4M	2018
PNASNet	74.20%	91.90%	5.1M	2018

Table 3.3: *CNN Summary of lightweight backbone networks(LBNs). The Top-1 Accuracy and the Top-5 Accuracy represent the classification accuracy on the ImageNet dataset. Params indicate the number of network parameters.*

Current invented lightweight backbone networks exist in computer vision community include SqueezeNet [123], Xception [124], MobileNet [125], MobileNetV2 [126], ShuffleNet [127], NASNet-A [128], PeleeNet [129], SqueezeNext [130], MnasNet [121], MnasNet-92 [121], PNASNet [131] and e.t.c.

Since our research only focuses on offline image dataset, the project focus on CBNs rather than LBN. This section does not go into an in-depth review of the lightweight backbone network. Still, a performance summary is given in table 3.3 on ILSVRC with top-1 accuracy, top-5 accuracy and the number of parameters for the lightweight backbone network.

3.2.4 Deep CNN Object Detectors

Object detection is a combined visual recognition task from object localisation and object classification. As the most widely studied pipeline of object detection in recent years, the Deep CNN model has proved its strength in feature representation [75, 132, 71, 72, 73], and the outstanding performance overwhelming hand-crafted ML model. The object detection architectures are diverse in two mainstreams: the two-stage object detector and the one-stage object detector.

A two-stage object detector is constructed with a salient region extractor and an object classifier concerning the object localisation task and the object classification task, respectively. The salient region extractor is a model that retrieves a set of regions of interest RoIs. And the object classifier is independent of selected RoIs with concerned class labels. These two sub-models are not necessary shared computationally. Two-stage object detectors are accurate but time-consuming and low in speed. The representatives of two-sage object detectors include R-CNN [97], SPPNet [133], Fast R-CNN [134], Faster R-CNN [26], Mask R-CNN [135] and R-FCN [27] and e.c.t. They are also the state-of-art object detection

models in the computer vision community.

The R-CNN(Regions + CNN)[97] is a milestone toward applying two-stage object detection strategy with Deep CNN models. A series of papers derived from it. The R-CNN method was developed to avoid the use of excessive numbers of regions in object detection. It relies on an external region proposal system based on a selective search algorithm [97]. The Fast R-CNN model [134] was developed to reduce problems with R-CNN. It feeds the input image to the CNN to generate a convolutional feature map. Faster R-CNN eliminates the selective search algorithm to produce a faster object detector called Faster R-CNN [26]. Although the very deep CNN model ResNet can achieve good results in object detection, it is slow [78]. The cost-reduced computational shared model R-FCN [27] attempts to balance translation-invariance in image classification and translation translation-variance in object detection. The Mask R-CNN approach [135] applies FCN (Fully Convolutional Networks) [136] to each Region of Interest (RoI) and then performs classification and bounding box regression in parallel. All these object detection algorithms use regions to localise objects within images: networks only examine regions in the image with an estimated high probability of containing the object. They are therefore termed semi-convolutional in which adopted a two-stage object detection strategy.

Since the two-stage object detection strategy is slow and computationally intensive, the one-stage (single-stage) detector is designed to directly locate and classify objects through Deep CNN without separating them into two individual

components. The one-stage object detectors do not require a salient region extractor and an object classifier. It abandon the region proposal process used in two-stage ones and compute object location coordinates along with class probabilities in the same stage.

The one-stage object detector outperforms the two-stage object detector in speed but has lower detection correctness (mAP) in general. They allow one to use only a single network for prediction with a set of pre-defined boxes, like OverFeat [76], YOLO series [137, 138, 139], SSD (Single Shot MultiBox Detector [140]), DSSD [141] can obtain relatively good results. However, it is no doubt that Deep CNN models lead object detection into a new era. The Deep CNN-based object detection algorithms break traditional hand-crafted ML object detection methods' bottleneck by using DL techniques. In order to provide a more detailed review, comparisons on Deep CNN-based object detection algorithms will be summarised in the following content.

Both the two-stage object detector and the one-stage object detector are introduced below. Some milestone object detectors are summarised into a performance table 3.5 and a characteristic table 3.4.

Table 3.4: *The summary of object detectors with their properties and weakness*

Model	Properties	Weakness
R-CNN	Use selection search to generate region proposals; Use SVM to classify regions; Use bounding box regression to refine regions.	Very slow; Storage space consuming; No end-to-end training.
SPPNet	Use selection search to generate region proposals and map them to feature map; Use spatial pyramid pooling to feed multi-scale input to Deep CNN backbone.	Slow; No end-to-end training.
Fast R-CNN	Use selection search to generate region proposals and map them to feature map; Use ROI Pooling layer for downsampling features to obtain fixed-size feature maps; Use Multi-task loss function.	Slow; No end-to-end training.
Faster R-CNN	Replace selection search with Region Proposal Network (RPN) to generate region proposals and map them to feature map; Share feature maps between RPN and Deep CNN backbone; End-to-end training.	Weak handling of multi-scale objects and small objects; Detection speed is not capable of real-time.

Mask RCNN	Use ROIAlign pooling layer instead of ROI; Object detection and segmentation simultaneously; Conducive to small target detection; Accuracy improved.	Detection speed is not capable of real-time.
FPN	A multi-level feature fusion Feature Pyramid Network. Conducive to multi-scale object and small object detection.	Detection speed is not capable of real-time.
YOLO	A novel one-stage detector; Detection speed is fast and capable of real-time.	Detection accuracy is not high; Weak handling of multi-scale objects and small objects.
YOLO2	New backbone network, DarkNet19; Use k-means clustering to generate anchor boxes; Multi-dataset joint training.	Difficult in training.
YOLO3	New backbone network, DarkNet53; Use multi-level feature fusion to achieve multi-scale detection; Accuracy improved.	IoU increases while performance drops.
YOLO4	New backbone network, CSPDarknet53(Cross-Spatial-Partial connections); Use path aggregation network to improve accuracy. Use bag of freebies and bag of specials	Inference time increased.

SSD	Multi-layer detection; Multi-scale anchors mechanism at different layers.	Not promoting small object detection.
DSSD	Multi-layer feature fusion; Replace with Upsampling using deconvolution; Accuracy improved of small object detection.	Detection speed decreases.

Table 3.4: *YOLO4 was released after submission.*

3.3 Generative Art

The statement "Computer can generate artwork" was proved to be true decades ago, in the early 1960s. Simultaneously, Frieder Nake [4] introduced the term "Generative Art" and created a program to generate computer art at the same time. The idea of "Generative Art" describes a process with various autonomy levels; thus, it is not limited to the digital realm. The art pieces driven by the system are a chemical reaction performed by the collaboration of machines and artists. The artist's role in generative art is to design or influence this process to some degree. As shown in figure 3.3, the result is remarkable, but "as beautiful as humans think" is a topic that is reserved for later study.

In recent years, one of the most outstanding achievements in the area of computational aesthetics study has been the implementation of style transfer algorithms from Werner Reichardt Centre for Integrative Neuroscience, which is

Table 3.5: *A Summary of two-stage object detectors and one-stage object detectors on the PASCAL VOC datasets*

Model	Architecture	Backbone	mAP(%)	Proposals	Speed(fps)	Train set	Batch Size	Year
R-CNN	Two-stage	AlexNet	58.5	2000	0.1	07	1	CVPR14
	Two-stage	ZFNet	59.2	2000	0.07	07	1	
SPPNet	Two-stage	ZFNet	60.9	2000	2.6	07	1	ECCV14
Fast R-CNN	Two-stage	VGG16	70	2000	0.5	07	1	ICCV15
Faster R-CNN	Two-stage	VGG16	73.2	6000	7	07	1	NIPS15
	Two-stage	ResNet101	76.4	300	2.4	07+12	1	
	Two-stage	ResNet101	76.4	300	5	07+12	1	
	Two-stage	ZFNet	62.1	300	18	07+12	1	
R-FCN	Two-stage	ResNet101	80.5	300	9	07+12	1	NIPS16
	Two-stage	ResNet101	79.5	300	5.8	07+12	1	
YOLO	One-stage	VGG16	66.4	98	21	07+12	1	CVPR16
	One-stage	VGG16	63.4	98	45	07+12	1	
YOLOV2	One-stage	DarkNet19	77.8	-	59	07+12	-	CVPR17
	One-stage	DarkNet19	78.5	-	40	07+12	-	
SSD300	One-stage	VGG16	74.3	8732	46	07+12	1	ECCV16
SSD512	One-stage	VGG16	76.8	24564	19	07+12	1	
DSSD321	One-stage	ResNet101	78.6	17080	9.5	07+12	1	
DSSD513	One-stage	ResNet101	81.5	43688	5.5	07+12	1	

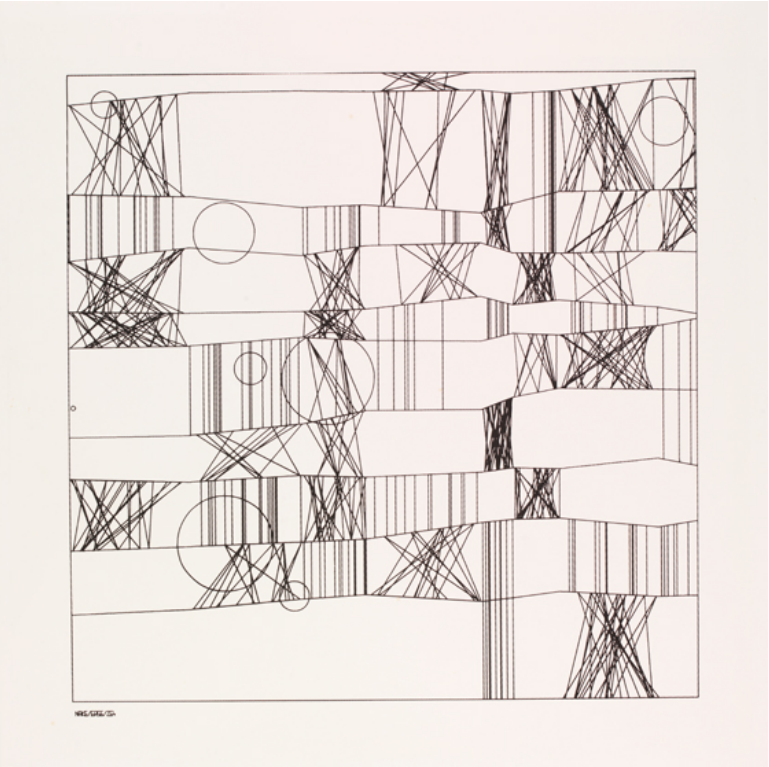

Figure 3.3: *Hommage à Paul Klee, Frieder Nake 1965, Victoria and Albert Museum online collection [4]*

mostly known as Google project DeepDream (figure 3.4) as well. They introduced an artificial system based on a Deep Convolutional Neural Network (Deep CNN) that creates artistic images of high perceptual quality in the summer of 2015 [5]. There is no doubt that this research result shocked not only lots of artists, but also people who have little knowledge in this area.

As mentioned in the previous paragraphs, compared to psychology or philosophy clarification, a lesser abstract form of aesthetic study is the Art Composition Rule [17]. Most of the current pieces of AI-work in computational aesthetics try to learn the pattern from existing artworks, focusing on replicating the style of artists using machine learning and computer vision techniques. Style Transfer is

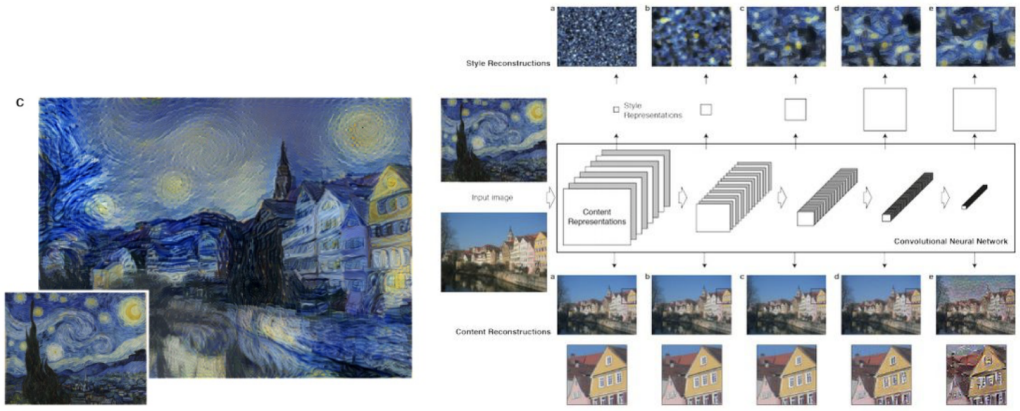

Figure 3.4: *The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images. [5]*

a new research field outcome as the combination of computational interpretation and deep neural networks. These algorithms can be elaborated a bit further; they capture low-level local features from input at lower layers in the Deep CNN. Higher layers of Deep CNN organise high-level visual concepts, for example, an object is identified to generate output.

The explosion of papers in Machine Learning and Art shows people's interests in investigating the area of computational aesthetics. Modern techniques of machine learning and computer vision, such as Deep CNN [75, 79, 80, 81, 77, 78, 136], Kernelized Deep CNN [142], Generative Adversarial Networks (GAN) [5] etc., are all considered as valuable attempts to achieve this target. Some of those methods have produced excellent results under research. But unfortunately, few of them have been applied to art with domain knowledge. In most cases, their results in mathematics or computer science are hard to evaluate and verify [17, 4, 29, 15].

To model AI "thinking" or acting much closer to how humans do while generating paintings, a more feasible approach is to adjust the miming process by applying the Art Composition Rule. For example, both using the building colour quantisation scheme for oil paintings so that it acts like image transformation [143] and comparing aesthetics measurement for evolutionary art [144, 17], aim to pass the "sense of art" to machines. A person who has been formally educated in art would generally follow the composition rule subconsciously. According to a given entry requirement schema, some academic assistants from schools of art would set aside personal opinions and expectations while grading a piece of painting. This strategy offers the possibility in theory that machines could replace humans in this position. However, full replication of human aesthetic actions is currently impossible. A machine does not have the aesthetic consciousness of a human being. One of the biggest problems of generative art in computational aesthetic research is to establish a linkage between aesthetics features and the Deep CNN style transfer method.

Object segmentation and precise feature selections would be reasonable methods to extract essential image representation for building art-sensible machine learning algorithms. Some topics could be specified in detail as: generation of colour palettes to compare colour aesthetic preferences between the types of paintings and use them to improve colour control on style transfer [145]; dynamic sketching simulation of typical brush stock types [146], and art-movement classification to evaluate art generation. These are some of the current research

directions for applying relative domain knowledge to style transfer models.

Our research suggests that art style could be learned and used by a machine to generate paintings. And more precisely, the machine should have the ability to learn the art style of Traditional Chinese Painting(TCP) and generate TCP using the knowledge obtained. Therefore, we review popular state-of-art style transfer models in the next section 3.3.1 and refer to collaborations from researchers across the art painting domain.

3.3.1 Style Transfer

As the most popular form of art, painting requires well-trained techniques to reproduce or redraw appealing artworks. Computer vision researchers strive to develop a mechanism that allows a machine to "paint" like a human being. Generative art is the computational aesthetic research topic in machine learning and computer vision. Meanwhile, the style transfer is a derivative of it but well-known as a computer vision task.

Style transfer is a type of generative art that proposes transforming one image into another artistic style. Before Deep CNN attracted attention, many studies were carried out in the computer vision community. Non-photorealistic rendering (NPR) [147, 148, 149] is one inspired technique structured thoroughly in recent decades. NPR offers an image the ability to be automatically transferred into synthetic artworks. This traditional pipeline is a generalisation approach known as texture synthesis [150, 151, 152, 153]. Texture synthesis in style transfer performs

the extraction and transformation of the texture from the input image to the target. Image analogies is a framework [154] that learns un-stylised and stylised images pairwise in training and generates an analogous transformation, which guides the style transfer. However, these stylisation methods have limitations. Most NPRs only manage to handle one artistic style at a time [149, 155]. They are designed and trained with particular art styles and do not have the compacity to be transferred to others. The analogous transformations of these stylisation methods consist of low-level features without high-level visual concepts, so the traditional style transfer model is not sensitive to image structure either.

There is evidence that CNN can extract image representations from image data [70, 71, 72, 73]. Extending the previous research, Gatys et al. [5] established a Neural Style Transfer (NST) mechanism in 2015 to render an arbitrary content image into different styles. NST obtains content information from the input image and style information from the artwork. It intends to model the input image's content as the feature responses from pre-trained CNN and illustrates the artwork's style as the summary feature statistics. The key to success in the objective of producing a stylised target image is to match CNN feature distributions by iterative image optimisation.

The innovation of NST is that it has no restriction on the types of art style and does not require ground truth for training. This mechanism breaks the constraints of NPR and image analogies stylisation, which only satisfies explicit styles.

NST can be categorised into Image-Optimisation-Based Online Neural Methods (IOB-NST) and Model-Optimisation-Based Offline Neural Methods (MOB-NST) based on image reconstruction algorithms from CNN representations. IOB-NST delivers style by iteratively optimizing images. MOB-NST optimizes the generation model offline, and generates a stylized image through a single forward pass. And it can be further categorised based on different Visual Texture Modelling strategies employed.

Visual texture modelling [156] is the algorithm used in NST to extract style representation. It is influenced by the visual texture algorithms [157, 158] in traditional style transfer mechanisms. It is concerned with modelling visual texture as style information. There are two distinct approaches in the computer vision community: Parametric Texture Modelling with Summary Statistics [159, 160, 161] and Non-parametric Texture Modelling with Markov Random Fields (MRFs) [157, 158]. The parametric texture modelling with summary statistics exploits texture that is captured to summarise the statistics as the statistical property of style. One recent milestone approach is Gram-based representation by Gatys et al. [162, 77] that correlates second-order statistics of the responses of different layers in CNN domain. Berger and Memisevic further improved it by horizontal and vertical feature map translation to address long-range symmetric structures. The non-parametric texture modelling with Markov Random Fields (MRFs) assumes that spatial neighbourhood entirely characterises each pixel [163]. An individual pixel is searched and synthesised by its similar neighbourhood corresponding

pixels in the source texture image.

The taxonomy of NST is shown as figure 3.5.

IOB-NST with parametric texture modelling with summary statistics proposed by Gatys et al. [155, 5, 162] does not require ground truth of explicit styles. They preserved relatively good style representation during stylisation, but Gram-based style representation inevitably loses low-level information in the CNN domain. Li et al. [164, 165, 166] employed Domain Adaption and Maximum Mean Discrepancy to preserve better the stylisation coherence of delicate structures in the CNN domain. The result was impressive but there was a lack of knowledge in semantics, depth, and variations in brush strokes.

Non-parametric IOB-NST with MRFs was first proposed by Li and Wand [167] and achieved success in photorealistic style transfer. To constrain the spatial layout, they introduced a patch-based MRF loss function. However, this model cannot handle the situation when the content image and style image significantly differ in terms of perspective or structure.

Compared to IOS-NST, MOB-NST reduces computational cost and speeds up the stylisation process by optimising a feed-forward network. MOB-NST includes Per-Style-Per-Model Neural Methods (PSPM), Multiple-Style-Per-Model Neural Methods(MSPM), and Arbitrary-Style-Per-Model Neural Methods (ASPM).

Parametric PSPM pre-train a feed-forward style-specific network and generate test result with a single forward pass [168, 169]. This can be improved by adopting batch normalisation (BN) [170] and instance normalisation (IN) [171].

Non-parametric PSPM [172] from Li and Wand was established by applying adversarial Markovian feed-forward network to their patch-based Non-parametric IOB-NST.

MSPM incorporates multiple styles relying on one single style transfer model. It is theoretically reasonable to integrate multiple styles within one model. For example, TCPs have various styles, but styles like XieYi (Freehand) and GongBi (Skilled) share many similarities (or shared features), such as similar colour palette or paint stroke information. Therefore, training the networks on each explicit style is inherently redundant and is a waste of resources. MSPM proposed to keep the same sets of parametrics by holding a set of convolutional parameters constant but introducing conditional instance normalisation (CIN) in IN layers to scale and shift parameters that sufficiently adjust multiple styles [170, 173, 174].

ASPM is known as Zero-shot Fast Style Transfer that is designed to be capable of a new style without training. MRF patch-based ASPM [175] tends to find a style patch that can match the content patch in the CNN feature space, and exchange them to achieve Style Swap. The model uses the fast image reconstruction algorithm to build the exchanged feature map. Huang et al. [171] managed to bring ASPM to a real time approach with Adaptive Instance Normalisation (AdaIN) inspired by CIN, following work by Ghiasi et al [176].

Since our project concentrates on TCP data, MSPM seems to be the most relevant and suitable model. Unfortunately, this is not implemented in the thesis,

but research work for applying NST to the TCP domain is presented in chapter 6.

3.4 Computer Vision and Traditional Chinese Painting

Traditional Chinese Painting is a world-famous heritage. Its unique techniques and relatively abstract styles attract the attention of many artists. It is one of the oldest art styles that the current academy has not fully interpreted. With the rapid development of computing power and the wide-spread of machine learning technologies, computer vision researchers started experimenting visual recognition tasks with the TCP domain in the early twenty-first century [35, 184]. Jiang et al. [185] proposed an SVM based ML classifier for TCP images in 2006, but progress was limited afterwards. The TCP integrated ontology was first determined in 2013 [35]. Most research studies regarding cross-domain TCP-computer-vision approaches are at the classification level. Author classification approaches include Monte Carlo convex Hull Model [186], brushstroke based hybrid CNN [187], and the VGG16 with Dropout [188] which was proposed for TCP style classification etc. Inspired by the object detection model for oil paintings presented by Crowley [189, 24], an assembled Region Proposal Network was introduced as an attempt at detecting objects in TCPs [25]. There are IOB-NST models that work with TCP data [190, 191, 192, 193]. However, few computer vision models,

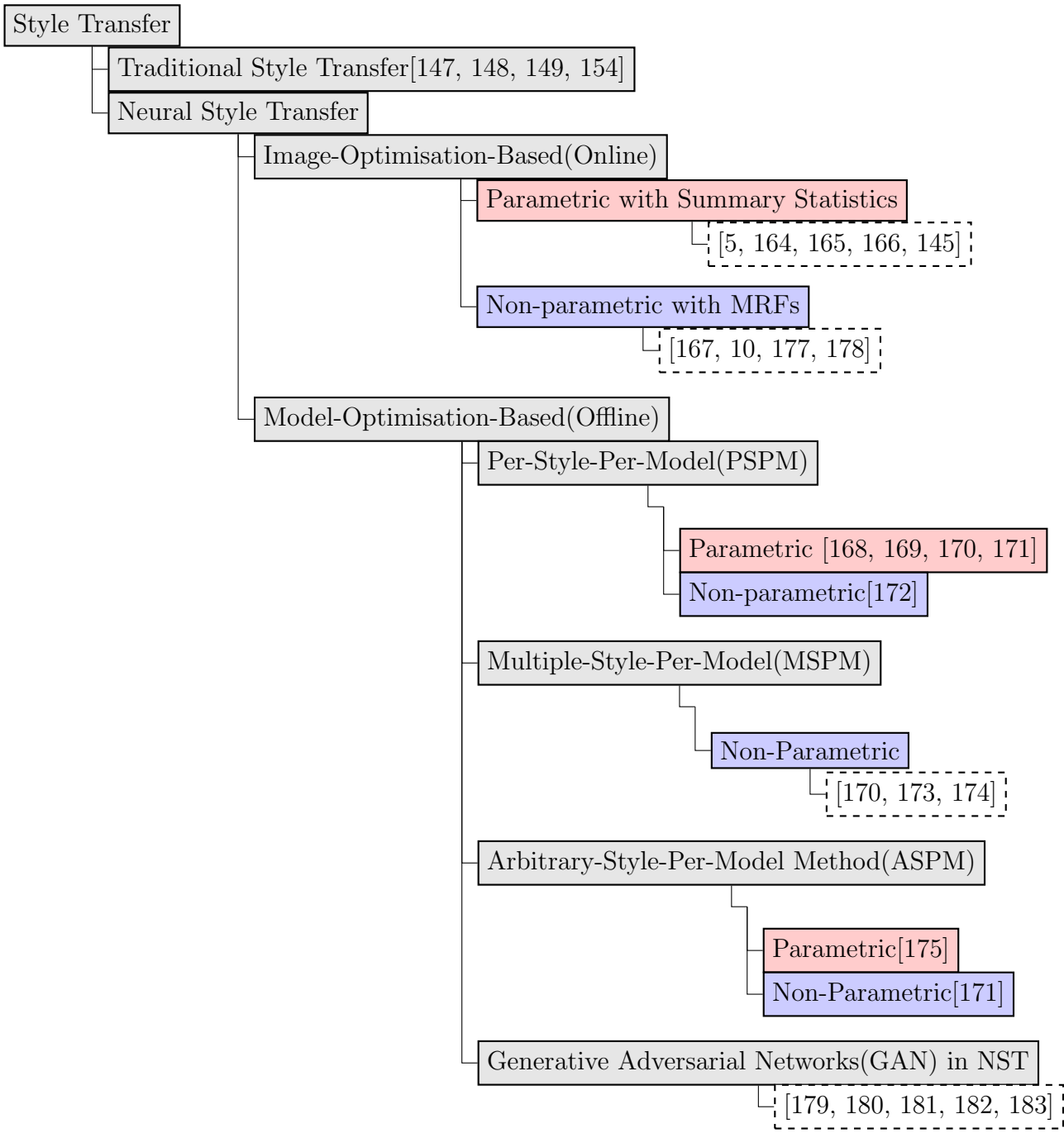

Figure 3.5: *A taxonomy of Neural Style Transfer(NST) techniques. Our NST taxonomy inspired by research from Jing et al. [6]. GAN is a neural network framework that can be used to implement style transfer. GAN-based style transfer is a promising direction, especially in the field of MOB-NST. It can be applied with PSPM, MSPM and ASPM.*

cooperating with the TCP domain, can guarantee to be state-of-the-art.

3.5 Summary

This chapter presents an overview of research topics related to computational aesthetics in the machine learning and computer vision community, including image representation, visual recognition and generative art. Each topic is employed in the context of a research application of computational aesthetics and is studied from a computer vision perspective. The chapter ends with a brief overview of the situation regarding current research studies across co-operations between the TCP and computer vision research fields.

- The first section 3.1 describes image representation and the types of image representation. This section also defines low-level features and explains how they construct the hand-crafted representation and deep learning representation. How image representation holds "reflection" of aesthetics features in computational aesthetics representing domain knowledge is also explained.

- Section 3.2 provides an outline of visual recognition by specifying related computer vision tasks, followed by the backbone CNN performance summaries of hand-crafted machine learning models and deep learning models. Section 3.2.4 also discusses effective Deep CNN detectors on natural image training with pros and cons, then raises the question of whether these set of methods can be transferred into the TCP domain.

- In section 3.3, after the descriptions of the history and knowledge of generative art, the definition and derivation of style transfer are given. Section 3.3.1 suggests a backbone model that might satisfy TCP style transfer by specifying properties and explaining limitation in neural style transfer.

- Section 3.4 summarises computational aesthetics investigations between the computer vision and the TCP domain, but the situation is not optimistic in most trending research tasks as explained above.

Chapter 4

Traditional Chinese Painting Representation and Classification

4.1 Motivation

Unlike western painting, there is no existing official and well structured Chinese Painting Image Database and Ontology System. This is a limitation that avoids computer vision crossing domain with TCP. Nearly all of the current computer research papers on TCP drive their experiment with their self-constructed datasets. This increase the difficulties in cross-comparisons between models that existed with our promoted designs. However, this problem remains unsolved due to the restriction on licenses for using high-resolution TCP data.

As stated in section 3.4, the number of previous research studies of applying

computer vision models to TCP data is limited. But there are many effective natural image classification methods which were established in the past decade, listed as the hand-crafted base (section 3.1.2 and section 3.2.2) and the deep learning base (section 3.1.3 and section 3.2.3). Hand-crafted image classification methods mainly use SIFT or HOG local features to achieve success in natural images classification. Researchers transferred this knowledge to the TCP domain and obtain relatively good performance [185, 186, 187]. With DL's fast development in the computer vision community, TCP classification started taking its steps into the research field, though only a few attempts invited. It is a worth-studied research topic of computational aesthetic learning.

This chapter focuses on modelling machine to distinguish TCP style and investigate what features and image representation designs play significant roles in identifying TCP style and content school. The shallow attempts on TCP classification with dynasty domain knowledge and object classification are also proposed in this chapter.

4.2 Data and Pre-processing

The word "traditional" that emphasised in TCP indicates its lack of image data. As mentioned in the last section, the first problem that our project facing is to solve is to construct a valid and structured TCP dataset.

4.2.1 Dataset Description

Our primary TCP dataset consists of two source datasets. One small dataset is a self-generated dataset resourced from several museums' open resources and online collections. The large dataset is a high-resolution image dataset provided and licensed by the Chinese Painting and Calligraphy Community (CPCC). The sizes of source datasets are 3000 images and 10000 images, respectively. Images licensed by the CPCC are scanned copies with the same image sizes as the original art pieces. They guarantee high resolution with min and max dpi as 300 and 400. In comparison, the TCP images from the small dataset collected online are various in qualities. Their min and max dpi of these images are 72 and 144.

All TCP image data employed in our project are manually categorised with class labels by me. TCP images were annotated with ground truth (bounding boxes and polygonal segmentation) if objects are contained. The application used to annotate objects are LabelImg[194] and LabelMe [195].

In section 2.2, we introduced an overview of the TCP category and some unique forms only presented in TCP (such as fan surface). Our project follows the tree-like TCP category to annotate the TCP data. As explained in figure 4.1, TCP images whose special forms are irregular shapes or incomplete paintings can leads to "noise" blocks. Identifying every single "noise" blocks requires lots of human efforts and time consuming. Therefore, we remove TCP images with special forms to reduce workload in later computer vision learning. There is overlap between the sources datasets, the redundant one with lower resolution

would be removed. All these steps markedly narrow down the size of our TCP dataset. Finally, four TCP datasets are created and shown as:

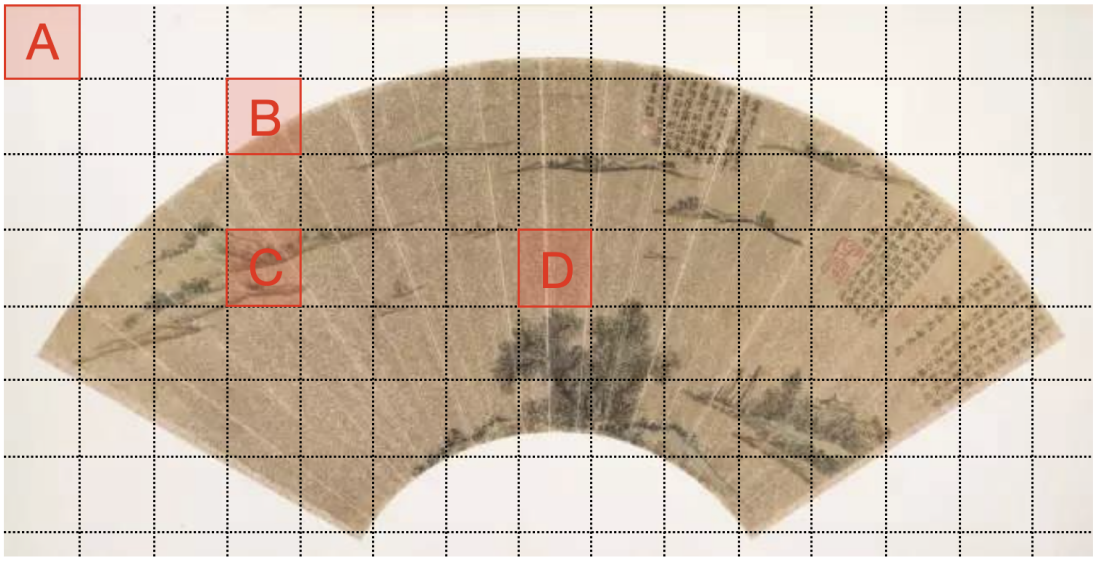

Figure 4.1: *TCP image with irregular shapes, such as fan surface, will results in "noise" blocks when we further process our data as proposing in section 4.2.2. As shown, Block C contains TCP object. Block A, Block B and Block D do not cover any object and can be named as "background". However, Block A represents "background outside of the TCP painting"; Block B is "background partially covers the TCP"; and Block D is the "background in TCP". Since Block A and Block B do not fully cover the TCP itself, their features should not be contributed as TCP's features. They are "noise" blocks.*

4.2.1.1 GongBi and XieYi Dataset

A binary dataset with 3872 in total, while 1723 in GongBi style and 2149 in XieYi style. This dataset is the pure primary TCP dataset.

4.2.1.2 Landscape, Figure and Flower-and-Bird Dataset

A three-class dataset re-generated from the primary TCP, based on the content of the TCP. Since some of the TCP images can be classifying into more than

one categories, for example, some religion paintings can be categorised as either Figure or Flower-and-Bird paintings. We dropped this type of data to avoid being ambiguous. This dataset's size is 3492, with 1278 landscape painting images, 1002 figure painting image and 1212 flower-and-bird painting images.

4.2.1.3 Landscape in Dynasty Dataset

A four-class dataset only with a size of 400 images. There are evenly distributed into four different dynasties: Tang, Yuan, Ming, Qing. This is a small dataset. This dataset's characteristic is that all painting images in this set are handscroll (long horizontal scroll) paintings from CPCC.

4.2.1.4 7-Class TCP Object Dataset

A seven-class dataset consisted of 1400 images in seven classes: human, horse, cow, bird, plant, cat, and dog. Each class has 200 images in either figure or flower-and-bird category. The number of objects presented in one image may be various. All images come with class labels and segmentation knowledge (bounding box) as ground truth.

We always store one-third of the dataset separately as pure and clean testing data to prevent overfitting. The others are randomly divided into 90% and 10% for training and validation, respectively, for experimental purpose.

4.2.2 Sliding Window

Sliding Window (SW) [56] is a method widely used to define salient regions for object detection in many image processing research [56, 27, 109]. It aims to extract local feature of specific salient regions by running a moving window of fixed size to segment the image into grids. Feature extraction and encoding methods will then be applied to salient regions or segments to obtain local representation vectors. These sets of vectors can be determined as the input for image classifiers.

Segmenting image into salient regions improve checking of any objects' presence. In the object detection approach with SW, SW [196] is run over the whole image. Salient regions extracted and interpreted directly connect to the classification model. SW is not only providing satisfaction in object detection but also being used to generate local representation [54, 55] for large-size high-resolution images.

In the TCP, handscroll is a typical TCP form for painting landscape in a long horizontal scroll. The high-resolution data we purchased for the licence is one-to-one in their original size (generally speaking, they can 50 centimetres high and up to 100 meters long). Segmentation and resize of these TCPs are indispensable. Therefore, we defined a set of fixed sizes of SW according to the input data size of image classifiers.

To better represent local features in large-size images, our project will preprocess data to generate SW in the following two trends:

- To run sliding window over the whole image to generate salient regions with

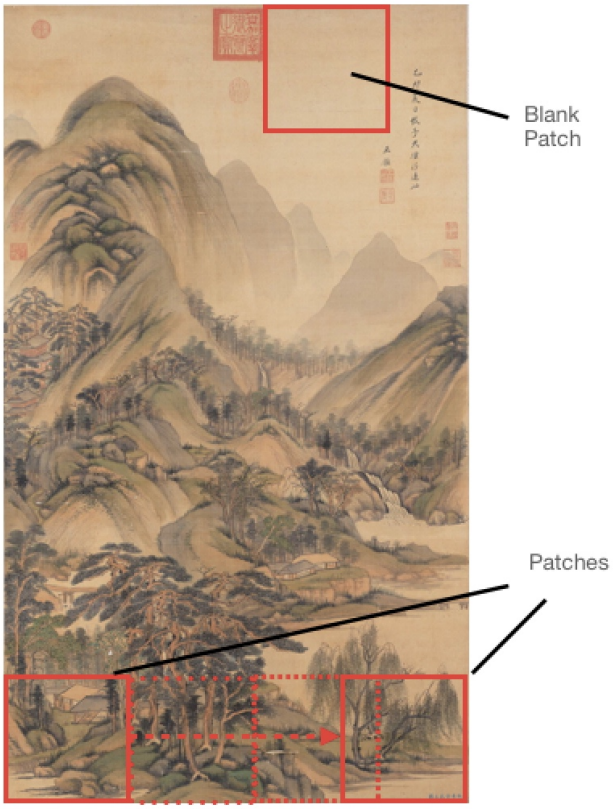

Figure 4.2: *Sliding Window (SW) defining salient regions for generating local image representations. Blank patches generated is ambiguous in classification but can be refined by dynasty domain knowledge in a sense*

three sets of patch size 128×128 pixels, 256×256 pixels, 512×512 pixels, respected stride keep the same as the width of the patch size. For this approach, there are chances that blank patches can be segmented.

- To resize the whole image with height as 128, 256, and 512 pixels by using linear interpolation. Then, to run the SW with the same patch size and stride defined as previous. This approach rarely segments blank patches and can represent the image in a reasonable global manner.

Since this section concentrates on TCP style classification but not object detection, the salient region segmentation methods with SW try to avoid overlapping, except that the original image's width cannot be divided with no remain. In this situation, the SW model will shift to align the patch's edge with the original image's boundary.

4.3 Feature Extraction and Image Representation

A crucial step to deal with image data is to extract features from the images and encode those features into a vector, known as hand-crafted image representation in traditional ML algorithms, and DL image representation in DL algorithms. Our project are interested in two important features in TCP domain, which are Edges and Colours. Due to the water-and-ink diffusion impacts, most TCPs' textures and edges are much smoother than natural images. The decreasing local features knowledge reduces the effectiveness in the TCP classification field. Therefore, enhancement of feature extraction is also an essence.

Section 4.3.1 describes what is HOG[59] and how we use it to derive hand-crafted image representation. Computer vision techniques used to rephrase hand-crafted representation with TCP domain knowledge are explained in section 4.3.2. Then section 4.3.4 promotes our cross-feature concatenation mathematics model that combines hand-crafted representation (HOG) and DL representation.

4.3.1 Histogram of Oriented Gradient (HOG)

Histogram of Oriented Gradient (HOG) is a widely-used feature descriptor which proven by Dalal [59] to be an innovation concerning hand-crafted image representation. It is coherence with the salient region that is the region of interest generated by either detection window or image griding. HOG descriptor has five significant steps in feature extraction.

RGB colour space is chosen for better handling in data with complex structure[59]. Gamma and colour space normalisation are used as the preliminary means of HOG feature extraction. The Gamma Correction equation is:

$$f(I) = I^{\gamma} \tag{4.1}$$

Where, I represents values in each colour channel of the input region; and γ describes the RGB intensity values in this paper. The Gamma correction intends to reduce the influence of brightness and background to improve the robustness of detection.

To compute gradients of pixel $p(x, y)$, the model uses filter kernels, such as Prewitt or Sobel, to filter the input region I. One simple and effective kernel is horizontal and vertical kernel filter, shown as $F_x = \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}$ and $F_y = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}$

The gradient information of the input region I can be driven from the derivatives of $G_x(x, y)$ and $G_y(x, y)$. G_x, G_y are estimates of the partial derivatives of the image in the (x, y) directions, and are estimated as the convolution of F and G. The formulas are as follows:

$$G_x(x,y) = G * F_x = I(x+1,y) - I(x-1,y) \tag{4.2}$$

$$G_y(x,y) = G * F_y = I(x,y+1) - I(x,y-1) \tag{4.3}$$

To accumulate weight votes for gradient and orientation over spatial cells, the magnitude $G(x,y)$ and the gradient orientation θ, for any pixel $p(x,y)$ in image I are driven from:

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \tag{4.4}$$

$$\theta = arctan\frac{G_y(x,y)}{G_x(x,y)} \tag{4.5}$$

For a target region, the HOG descriptor normalises contrast within overlapping blocks of cells. The cells have the same size and then calculate each cell's gradient information, including the gradient size and the gradient direction. The original paper of HOG [59] point out that the gradient direction of the pixel is divided into nine bins evenly in the range of 0-180 degrees. If the number of bins exceeds 9, the calculation becomes significantly intensive with little performance.

Every pixel in each cell performs weighted voting for the histogram of the gradient orientation. The weighted weight can be the amplitude of the gradient and the square or square root for the amplitude. Since the square or square root for the amplitude decreases the performance[197, 59], using the gradient amplitude is the best choice.

There are many normalisation methods introduced to collect HOGs. Overall, the basic idea is to combine several cells into a larger block with their gradient information. Then the image can be regarded as the detected regions. Let v be the non-normalized vector containing all histograms, and ξ is the exact value as a small constant.

$$L_2 - norm : v \leftarrow \frac{v}{\sqrt{||v||_2^2 + \xi^2}} \tag{4.6}$$

$L_2 - Hys$ is a scaled version of $L_2 - norm$. It computes the normalization factor by calculating the $L_2 - norm$ first, then limit the vector v to maximum value 0.2, and then renormalize[198];

$$L_1 - norm : v \leftarrow \frac{v}{||v||_1 + \xi} \tag{4.7}$$

$$L_1 - sqrt : v \leftarrow \sqrt{\frac{v}{||v||_1 + \xi}} \tag{4.8}$$

The block information is normalised separately, and different cell unit sizes and different block sizes will affect the final extraction of HOGs. There are three different sizes used in our research, and further setting can be found in section 4.5.

4.3.2 Enhancing HOG with TCP domain knowledge

With the water-and-ink diffusion impacts and small-range colour palette in TCP, most TCPs' textures and edges are much smoother and weaker than those in

natural images. Enhancing feature sensitivity is necessary. Our project propose some computer vision techniques to improve the hand-crafted image representation extraction for TCP data.

4.3.2.1 Sobel's Kernel

For Edge Orientation, we use Sobel's kernel to enhance HOG detection with five kinds of edge: 0-degree edge (horizontal), 45-degree edge, 90-degree edge (vertical), 135 degrees, and no edge detected. This Sobel's edge detectors' theory is to use particular filters to intensify various kinds of edge and choose the most substantial edge to represent the pixel. The filters we used to detect each kind of edge are shown below.

$$0°Edge = \begin{vmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{vmatrix} \tag{4.9}$$

$$90°Edge = \begin{vmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{vmatrix} \tag{4.10}$$

$$45°Edge = \begin{vmatrix} 2 & 2 & -1 \\ 2 & -1 & -1 \\ -1 & -1 & -1 \end{vmatrix} \tag{4.11}$$

$$135°Edge = \begin{vmatrix} -1 & 2 & 2 \\ -1 & -1 & 2 \\ -1 & -1 & -1 \end{vmatrix} \tag{4.12}$$

$$NoEdge = \begin{vmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{vmatrix} \tag{4.13}$$

Enhancing edge extraction sensitivity and maximise the knowledge gained, two different Gaussian filters blur the image then extract fine-scale and coarse-scale edge simultaneously. This process returns twenty-five kinds of pixels characterised by fine-scale and coarse-scale edge orientations.

4.3.2.2 TCP colour Palette

As one of the aesthetic features, colour aesthetics learning has been proved that plays an essential role in predicting an individual's aesthetics preference [199]. Related papers published in the previous EG Workshops on computational colour formation analysis could be checked online [200].

However, due to the variation of human's body structure and cultural background, decision or judgement made on colour sections can have a huge difference of visual tracking. Linguistic researchers Berlin and Kay performed their study related to this problem in their book Basic Color Terms[7]. The book concludes that language contains eleven basic colour terms: black, blue, brown, grey, green, orange, pink, purple, red, white and yellow. Human can use this set of basic colour terms to represent colours in the world. The colour interpretations are affected by the inter-logic basing on a sort of cultural background. The theory can be shown in figure 4.3

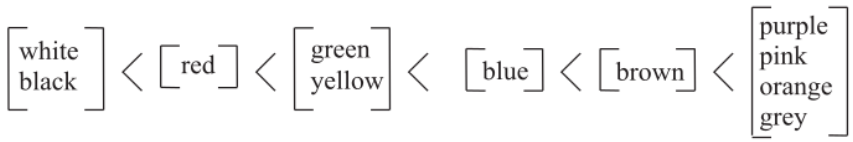

Figure 4.3: *Berlin's primary colour word theory showing the interpretation of cultural logic in colours [7]*

In the area of machine vision, Colour Attributes, or Colour Name [201], is defined as a collection of labelled color chips. It is an artificial intelligence strategy that interprets how people communicated with each other in early society, and the order of colour appearance follows the primary colour word theory. Colour name [201] has been proven to be crucial to many visual tracking problem, such as object recognition, object detection and action recognition[202, 203, 204, 205].

The primary colour palette is a mapping to eleven basic colours in linguistic provided by Weijer and Schmid [201]. It maps the RGB values to a probabilistic eleven dimensional color representation which sums up to 1. It represents a quantisation result of raw images' colours [201, 205].

Inspired by this idea, we quantise the TCP's colours with a TCP colour palette with eleven primary colours. We proposed the inverse colormap algorithm [206] to convert a full colour TCP image into an indexed image with a limited set of colours in RGB. The colours in this limited set are named as representative colours. In our paper, the representative colours are the eleven basic colour terms. Therefore, the specified colormap is a c×3 matrix with values in the range $[0, 1]$ and $c = 11$. Each row of the matrix represents a three-element RGB triplet that specifies the red, green, and blue components of a single colour of the colormap. The square

of the Euclidean distance from a pixel colour (r, g, b) to a representative colour (r_c, g_c, b_c) is

$$d(r, g, b) = (r - r_c)^2 + (g - g_c)^2 + (b - b_c)^2 \tag{4.14}$$

After distance computation, we can obtain an 11 dimensional vector for each pixel. This vector can be used to automatically assign the representative colour with the minimum distance to the respective pixel. The table of the the RGB values for the 11 primary colours could be found in figure 4.4.

The dithering algorithm [207] is also employed in the quantisation process to preserve the smoothness of the image. Colour dithering applies a diffusion of coloured pixel and create illusion of color depth in the defined colour palette [206, 208, 209].

Concerning the colour feature, our model encodes information of each pixel with the edge features. For example, the HOG extractor with Sobel edge detectors will be further encoded into 275 descriptions, 5×5×11. Then transform the description of each pixel to vector by normalising the histogram of 275 descriptions. Normalisation is the general approach to avoid scale invariance and rotation invariance.

Color	Color Name	Hex Code #RRGGBB	Decimal Code R,G,B
	White	#FFFFFF	(255,255,255)
	Black	#000000	(0,0,0)
	Red	#FF0000	(255,0,0)
	Green	#00FF00	(0,128,0)
	Yellow	#FFFF00	(255,255,0)
	Blue	#0000FF	(0,0,255)
	Brown	#993333	(153,51,51)
	Purple	#9900FF	(153,0,255)
	Pink	#FFCCCC	(255,204,204)
	Orange	#FF9900	(255,153,0)
	Grey	#999999	(153,153,153)

Figure 4.4: *A table shows the 11 primary colours used in the experiments with HEX codes and RGB decimal values [7, 8].*

4.3.3 Deep CNN Feature Map

Deep CNN feature maps can be equivalent to the DL image representation. CNN is powerful but has leakages in constructing image representations. Referring to section 3.1.3 and section 4.4.2, CNN extracts deep features from convolutional operation in Conv layers [75]. This kernel matrix defined within the filter slides over the input data to output a feature map, the DL representation [142, 155, 162]. The number of filters establishes the number of feature maps. The different settings of filters will produce different outcomes, such as colour depth and contour (edge and shapes). The feature maps resulting from this approach represent different deep features. The fully connected layer functionally perform as an image representation to image classification [73, 49].

However, the pooling layer, which reduces the data dimension more productively, can lead to another loss of spatial location information [132, 136]. There is a dilemma in Deep CNN models balancing the computational cost and feature preciseness.

4.3.4 Crossing Feature Between Hand-crafted and DL

This section explains a problem-solving approach that we introduce to the Deep CNN model. Our mathematic model provides DL representation with an ability to obtain domain knowledge defined by the hand-crafted representation. We then name the feature consisting of HOG and DL representations and TCP domain knowledge as the TCP cross-feature.

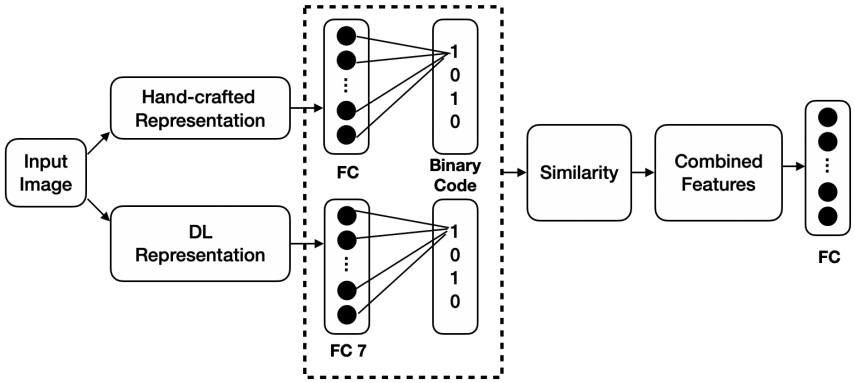

Figure 4.5: *A basic system flow on how to cross-feature between hand-crafted image representation and DL image representation with a similarity approach*

The model we propose to concatenate hand-crafted features and DL features was inspired by Cross-Model Hashing(CMH) algorithm. CMH is widely used for similarity search in multimedia retrieval applications. The basic concept of CMH is to map data points with multi-modalities features from the original space

into a Hamming space with binary code references so that the similarity can be preserved by computing the Hamming distance.

Since the features sampled by Deep CNN have high-dimensional characteristics, we employed a Hash layer to be added to the original CNN architecture to process the DL features. Taking VGG's structure as an example, we keep all the convolution layers, pooling layers, and the first and second fully connected layers (FC 6 and FC 7), but remove the last fully connected layer. Then, we add the Hash layer as a replacement. The feature map of the last convolutional layer is selected as the n-frame feature vector G_{DL} to be extracted. In our CNN model with HOG, the hashing algorithm encodes the high-dimensional data into a set of binary codes and maintains the meta-similarity of dimensional in image data simultaneously [210]. As shown in figure 4.5, we also perform the hand-crafted feature vector as an input to a fully connected layer(FC7) for configuration. The Hash layers represent the latent concepts with a set of binary values which dominate the class labels. Activation function, such as sigmoid, is applied here to keep the feature value in the range of $[0, 1]$ and constructs the feature Hash code. The Hash code then computes the image's feature vectors with Hamming distance. The smaller the Hamming distance, the higher the similarity of the feature vector.

Assuming each data point(patch) has two modalities of features, these two feature vectors are α and β, respectively, and the Hamming distance D. The Hash codes mapping and Hamming distance are defined as shown in equations:

$$\begin{cases} \alpha = \alpha_1, \alpha_2, ...\alpha_n \\ \beta = \beta_1, \beta_2, ...\beta_n \\ \mathbf{I}(\alpha_i, \beta_i) = \begin{cases} 1, & if \quad \alpha_i \neq \beta_i \\ 0, & if \quad \alpha_i = \beta_i \end{cases} \\ \mathbf{D} = \sum_i^n \mathbf{I}(\alpha_i, \beta_i) \end{cases} \tag{4.15}$$

Similarity crosses between hand-crafted representation and DL representation is given by the below equation, with weight of hand-crafted HOG representation w_1 and weight of DL representation w_2:

$$S = w_1 \cdot S_{HOG} + w_2 \cdot S_{DL} \tag{4.16}$$

During the training process, the Hash codes should be regulated by the succeeding fully connected layer that encode semantics and implement classification. The similarity threshold is defined as:

$$\epsilon = \frac{1}{N} \sum_{i=1}^{N} S(f_{i+1}, f_i) + \tau \tag{4.17}$$

With f represents the feature vector for the current image; N is the number of images, and τ is the adaptive adjustment factor.

4.4 Classifying TCP

There are two classification methods applied to our TCP data to identify them into different categories. This section aims to verify which models and image representations are more reliable and effective for later computational aesthetics research applications, object detection and NST. Support Vector Machine and

Convolutional Neural Network are under experimentations.

4.4.1 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a traditional statistical ML algorithm for classification proposed by Vapnik [94]. It promotes the idea of creating a hyperplane that is then used to separate binary classes with the maximised margin. SVM claims its great success in various aspects since its establishment. It achieves outstanding performance in recent year object detection, such as the stage-wise pioneering approaches based on the Deep CNNs [97, 133]. In this section, we will give a brief intro to this milestone traditional ML model (Figure 4.6).

4.4.1.1 Linear separation of a feature space in SVM

In binary classification, the a hyper plane in an n-Dimensional feature space can be defined by the following sets of equation:

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{w} + b = \sum_{i=1}^{n} x_i w_i + b = 0 \tag{4.18}$$

Dividing by $||\mathbf{w}||$, we have

$$\frac{\mathbf{x}^T\mathbf{w}}{||\mathbf{w}||} = -\frac{b}{||\mathbf{w}||} \tag{4.19}$$

This equation indicates the projection of any point $\mathbf{x}$ on the plane onto the vector $\mathbf{w}$ is always $-b/||\mathbf{w}||$, while $\mathbf{w}$ is the normal direction of the plane, and $|b|/||\mathbf{w}||$ is the distance from the origin to the plane. Since the equation of the

hyper plane can be various, $c\,f(\mathbf{x}) = 0$ represents the same plane for any c.

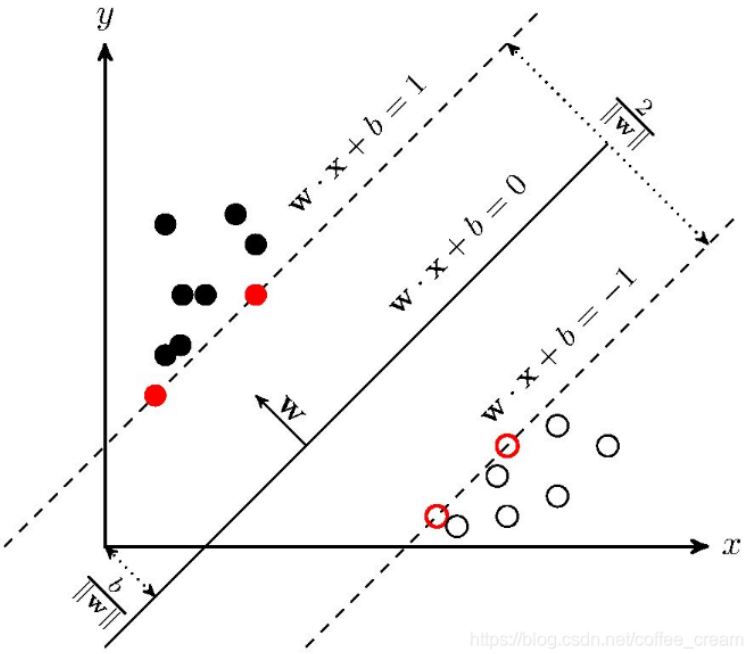

Figure 4.6: *Linear separation of a feature space in SVM [9]*

The n-Dimensional feature space can be partitioned into regions, two in binary classification, by the mapping function $y = sign(f(\mathbf{x})) \in \{1, -1\}$, given as:

$$f(\mathbf{x}) = \mathbf{x}^T\mathbf{w} + b = \begin{cases} > 0, & y = sign(f(\mathbf{x})) = 1,\ \mathbf{x} \in P \\ < 0, & y = sign(f(\mathbf{x})) = -1,\ \mathbf{x} \in N \end{cases} \tag{4.20}$$

Points $\mathbf{x} \in P$ in the positive side of the plane are mapped to 1,while points $\mathbf{x} \in N$ from the opposing side assigned value -1. During testing, point $\mathbf{x}$ will be classified as P if $f(\mathbf{x}) > 0$, vice versa, N if $f(\mathbf{x}) < 0$.

In binary SVM classification, the aim is to find the optimal plane H_0 in the middle of the two classes and separates them with the largest distance between the decision plane and the closest samples [94]. Since for both $\mathbf{x}_i \in P = \{(\mathbf{x}_i, 1)\}$ and $\mathbf{x}_i \in N = \{(\mathbf{x}_i, -1)\}$ should satisfied if and only if the data is linearly separable,

the hyperplane equation can be re-written as:

$$y_i(\mathbf{x}_i^T\mathbf{w} + b) = y_i \tag{4.21}$$

For all points on the positive side $\mathbf{x}_i \in P$:

$$\mathbf{x}_i^T\mathbf{w} + b \geq 1, \quad y_i = 1 \tag{4.22}$$

and all points on the negative side $\mathbf{x}_i \in N$:

$$\mathbf{x}_i^T\mathbf{w} + b \leq -1, \quad y_i = -1 \tag{4.23}$$

By combining these into one inequality $U = P \cup N$, we have:

$$y_i(\mathbf{x}_i^T\mathbf{w} + b) \geq 1, \quad (i = 1, \cdots, m; \mathbf{x}_i \in U) \tag{4.24}$$

H_0 is obtained when $y_i = 0$. In the meanwhile, $y_i = 1$ for the positive plane H_+ and $y_i = -1$ for the negative one H_-. Points on plane, either H_+ or H_-, are called Support Vectors, which is

$$\mathbf{x}_i^T\mathbf{w} + b = y_i \tag{4.25}$$

and the equation that holds for all support vectors is shown as follows, while $\alpha_i \geq 0$ $(i = 1, \cdots, \alpha_m)$ is the Lagrange coefficients.

$$b = y_i - \mathbf{x}_i^T\mathbf{w} = y_i - \sum_{j=1}^{m} \alpha_j y_j (\mathbf{x}_i^T\mathbf{x}_j) \tag{4.26}$$

Minimising the norm $\mathbf{w}$ with b can obtain the maximal margin with the following objective function:

$$\frac{1}{2}\mathbf{w}^T\mathbf{w} = \frac{1}{2}||\mathbf{w}||^2 \tag{4.27}$$

Under the constraint (Linear) of:

$$L_p(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 + \sum_{i=1}^{m} \alpha_i(1 - y_i(\mathbf{x}_i^T\mathbf{w} + b)) \tag{4.28}$$

4.4.1.2 Soft Margin

SVM has ability in solving not linearly separable classification problem. In this condition, extra term should be added to the original function of the optimal hyper-plane:

$$y_i(\mathbf{x}_i^T\mathbf{w} + b) \geq 1 - \xi_i, \quad (i = 1, \cdots, m) \tag{4.29}$$

The objective function on minimizing $||\mathbf{w}||$ with minimum error, $\xi_i \geq 0$ becomes:

$$\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{m} \xi_i^k \tag{4.30}$$

subject to $(i = 1, \cdots, m)$:

$$y_i(\mathbf{x}_i^T\mathbf{w} + b) \geq 1 - \xi_i \tag{4.31}$$

With C as a regularisation parameter that controls the trade-off between maximising the margin and minimising the training error. The model is shown as overfitting if a large C returned. When $k = 2$, the problem is transferred into a 2-norm soft margin problem.

4.4.1.3 Kernel Mapping

For non-linearly separable data, we can map the samples $\mathbf{x}$ into a higher dimension feature space:

$$\mathbf{x} \longrightarrow \phi(\mathbf{x}) \tag{4.32}$$

where the classes can be linearly separated. And the decision function becomes:

$$f(\mathbf{x}) = \phi(\mathbf{x})^T\mathbf{w} + b = \sum_{j=1}^{m} \alpha_j y_j (\phi(\mathbf{x})^T \phi(\mathbf{x}_j)) + b \tag{4.33}$$

$$\mathbf{w} = \sum_{j=1}^{m} \alpha_j y_j \phi(\mathbf{x}_j) \tag{4.34}$$

and b are the parameters of the decision plane in the new space. A kernel is defined as a function that takes two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ as arguments and returns the value of the inner product of their image $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ [211]:

$$K(\mathbf{x}_1, \mathbf{x}_2) = \phi(\mathbf{x}_1)^T \phi(\mathbf{x}_2) \tag{4.35}$$

SVM algorithms can be converged with kernel function to satisfy non-linearly separable data in higher-dimensional spaces by replacing all inner products in the learning algorithm in the original space with the kernels. Some popularly used kernel are listed below.

Linear Kernel with assumption that $\mathbf{x} = [x_1, \cdots, x_n]^T$, $\mathbf{z} = [z_1, \cdots, z_n]^T$,

$$K(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{z} = \sum_{i=1}^{n} x_i z_i \tag{4.36}$$

Polynomial Kernels that map data from a 2-D space to a 3-D space, assuming $\mathbf{x} = [x_1, x_2]^T$, $\mathbf{z} = [z_1, z_2]^T$,

$$\begin{aligned} K(\mathbf{x}, \mathbf{z}) &= (\mathbf{x}^T \mathbf{z})^2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 z_1 x_2 z_2 \\ &=< (x_1^2, x_2^2, \sqrt{2} x_1 x_2), (z_1^2, z_2^2, \sqrt{2} z_1 z_2) > \\ &= \phi(\mathbf{x})^T \phi(\mathbf{z}) \end{aligned} \tag{4.37}$$

Gaussian kernels to radial basis function [212]

$$K(\mathbf{x}, \mathbf{z}) = e^{-||\mathbf{x}-\mathbf{z}||^2/2\sigma^2} \tag{4.38}$$

4.4.2 Convolutional Neural Networks

In this section, we provide a general overview of the methodology of CNN.

A neural network consists of a large number of neurons that connected to each other. After each neuron receives the linear combination input, it is only a simple linear weighting at the beginning. Then a non-linear activation function is added to each neuron to perform a non-linear transformation and output. Each connection between two neurons represents by a weighted value, know as weight w. Choices in different weights and activation functions will lead to varying outputs of the neural network. The mathematical representation can be written as:

$$z = XW^T + b \tag{4.39}$$

Where x is the input vector, w is the weight, b is the bias and z as the output.

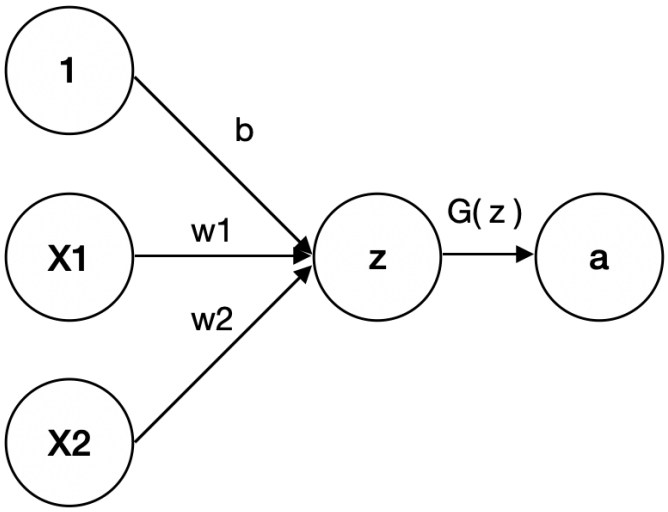

Figure 4.7: *A representative example of a perceptron with three inputs associated with either bias or weight. The perceptron has an activation function G(z) (or σ) and the output is a scalar of value a.*

When the neural network first established, researchers defined the activation function as a linear function so that the output is a linear transformation of the input as shown in figure 4.7. For example, a simple linear activation function is G(z) = z. However, the linear activation function was too limited in practical applications. Non-linear activation functions are then promoted.

Commonly used non-linear activation functions include Sigmoid, Tanh, ReLUs [104], etc. The former two activation functions, sigmoid and Tanh, are popular in the fully connected layer, while the ReLUs is generally applied to the convolutional layer. The activation functions are formulated as:

$$\sigma_{Sigmoid}(z) = \frac{1}{1+e^{-z}} \tag{4.40}$$

$$\sigma_{Tanh}(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \tag{4.41}$$

$$\sigma_{ReLUs}(z) = \begin{cases} 0, & z \leq 0 \\ z, & z \geq 0 \end{cases} \tag{4.42}$$

The most basic Sigmoid function is equivalent to compressing a real number to between 0 and 1. When z is a huge positive number, G(z) will approach 1, vice versa, G(z) tends to 0 when z is a small negative number.

Set of neurons forms an artificial neural network (ANN), and an artificial neural network consists of three types of layers:

- **Input layer:** Neurons accept a large number of non-linear input, called the input vector.
- **Output layer:** the information is transmitted, analyzed, and weighed in the neurons' connection to form the output vector as a result.
- **Hidden layer:** this layer is referred to as "hidden", composing many neurons and links between the input layer and the output layer. If there are multiple hidden layers, multiple activation functions are employed.

Meanwhile, each layer may consist of single or multiple neurons, and each layer's output will be used as the input data of the next layer.

With the rapid development of computing power, Convolutional Neural Networks (CNN) voted the best in the computer vision community [213, 77, 21]. General ANN processes data at the input layer with techniques, such as normalization, PCA/whitening and de-average, etc. The de-average approach is designed to remove the mean of input data so that all input data dimensions are centred on being 0, which avoids excessive data deviation and affects training. CNN only employs the "de-average" step in training.

CNN consists of the convolution calculation layer (Conv layer), activation (excitation) layer, and pooling layer at the middle level and ends with a fully connected or fully convolutional layer [82, 103]. Conv layer is the core of CNN, which calculate linear product summation of the input data. The activation layer is employing the idea of activation function, such as ReLU. The pooling layer is where the average or maximum of the area are computed. The fully connected network outputs of the classification results representing one neuron per class.

The Conv layer's operations require defining in an input matrix and a kernel filter and applying convolution operation on them. The filter is with a set of neurons with fixed weights, which performs convolution calculations on local input data. After each calculation of the local data with a fixed size matrix (3×3 or 5×5), the data window keeps shifting and sliding until all data is computed. In this process, there are several parameters: The number of neurons determines the depth of the output. It also represents the number of filters; Stride determines how many steps to slide to the edge. Conv layer has zero-padding employed so

that the total length is divisible by the step length.

Pooling layer downsample feature map output from the Conv layer. Max-pooling is a widely-used approach to transfer the local maximum value to the next layer. Nagi et al. [214] suggested max-pooling with 2×2 in size can decrease the computational cost but keep features' sophistication by halving the feature map's size. It can not only significantly reduce the amount of calculation but also effectively avoid overfitting.

The data processed by the Conv layers and the pooling layers is input to the fully connected layer to obtain the final desired result.

4.5 Experiment and Evaluation

This section describes various experiments set up to evaluate TCP classification performance with two different types of image presentations. Our experimental works aim to find out the answers to the questions listed below:

- **Question 1:** Which classification algorithms achieve better performance in the TCP domain, hand-crafted ML classifier or Deep CNN classifier?
- **Question 2:** Which features better perform in the TCP classification, HOG (hand-crafted representation), DL feature map (DL representation), or combined TCP cross-feature?
- **Question 3:** By implementing TCP domain knowledge to image representation, is it true that it significantly improves classification results?

In this section, our project uses four datasets, two types of classifiers, four hand-crafted image representations, one DL image representation and four TCP cross-feature with DL image representations to provide answers to these three questions.

4.5.1 Experiment Setting

All four dataset promoted in section 4.2.1 are involved in TCP classification experiments: GongBi and XieYi Dataset, Landscape Figure and Flower-and-Bird Dataset, Landscape in Dynasty Dataset, and 7-Class TCP Object Dataset. We always keep one-third of our data as the test set and the rest of the data is divided into 90% for training and 10% for validation.

Since the 7-Class TCP Object Dataset contains images with bounding boxes and respective class labels as ground truth, which is suitable for object detection, we slightly re-construct it as an object-recognition-satisfied dataset. Object recognition is an object classification task that does not require location detection. Therefore, we run a random selection on the ground truth bounding box we annotated in the 7-Class TCP Object Dataset and generate another set of 1400 TCP images that only one object each. In other words, the ground truth bounding box is no longer RoI but input image data. This refreshed 7-Class TCP Object Dataset keeps evenly distributed in classes.

Feature extractions processes come with sliding windows and HOG. Sliding windows are automatically applied to large-scale painting images. The logic of

sliding window presented as pseudo code algorithm 1.

Algorithm 1 Sliding Window Algorithm

```
1: procedure SlidingWindow(image)                    ▷ Logic of Sliding Window
2:     System Initialisation, define size
3:     Read the image with height and width
4:     current grid centre is [x, y]                 ▷ Start from bottom left
5:     while (height ≥ size)&(width ≥ size) do
6:         Run sliding window over image horizontally
7:         if width − x ≥ size/2 then
8:             x += size/2
9:         else
10:            x = width - size/2
11:        Extract salient region
12:        if x = width − x then
13:            y += size/2
14:            x = size/2
15:    Resize image to fit the region and compute one extra feature vector
```

HOG feature extractor is then applied to sets of salient regions. There are three types of cells defined in our project, 4×4, 8×8 and 16×16. Hand-crafted image representation from HOG features extraction is given examples in figure 4.8 and figure 4.9.

By default, HOG computes image representation with the horizontal and vertical kernel. Then, Sobel's Kernel and TCP colour palette are employed to the original HOG descriptor. Finally, we have three types of HOG hand-crafted image representation: HOG, HOG with Sobel's Kernel, HOG with TCP colour, and HOG with Sobel's Kernel and TCP colour.

DL representation is the feature map extracted during the Deep CNN learning process.

To evaluate whether the hand-crafted image representation can improve DL

classification, TCP cross-feature algorithms concatenate DL image representation with HOG, HOG with Sobel's Kernel, HOG with TCP colour, and HOG with Sobel's Kernel and TCP colour.

Both linear kernel SVM and VGG-16 are employed to represent the hand-crafted ML model and DL model for TCP classification, respectively. The SVM which is applied here is a linear kernel SVM with no slack variable used. For the VGG-16, we initialise the model with the ImageNet pre-trained model, then fine-tuned and local trained it with our datasets. These two models are coherent with the corresponding image representations described above. Then, our project has ten classifiers.

In final, these ten classifiers train end-to-end and test on four TCP datasets with three different cell size of HOG and three separate sliding window's size. (Examples of image representation of HOG feature vectors are given in figure 4.8 and figure 4.9.). The mean average precision (mAP) are used for evaluation measures.

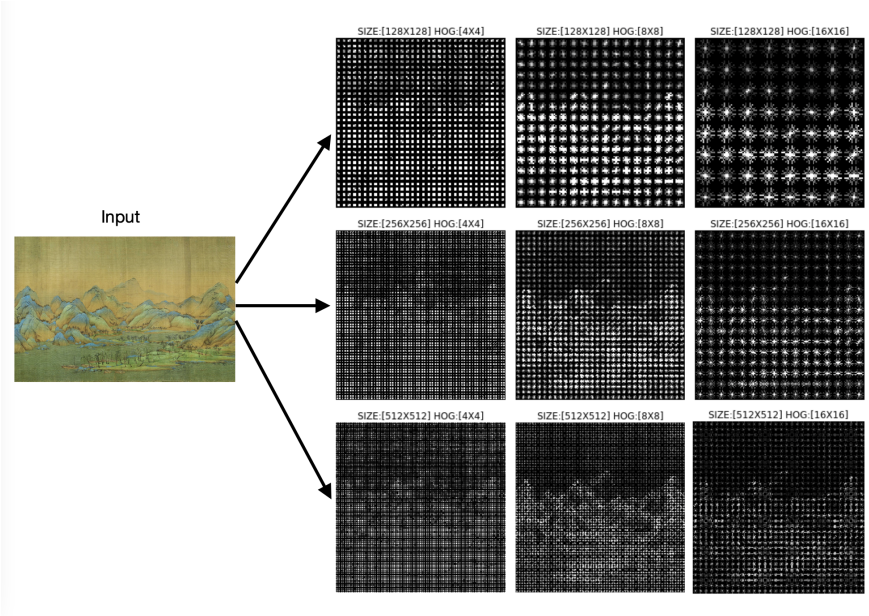

Figure 4.8: *Image representations of HOG feature vector. Variations in relations between the input image size and the size of selected cell for skilled brush landscape painting*

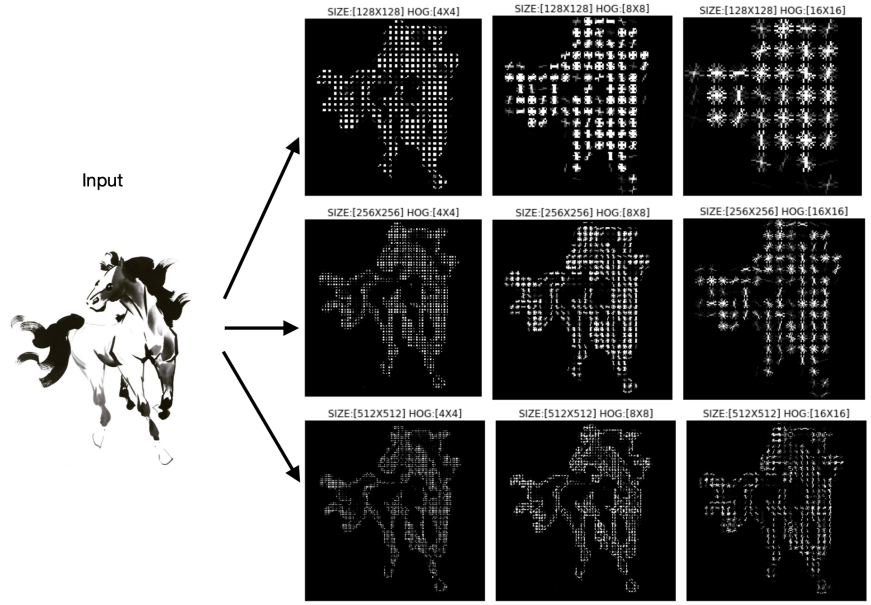

Figure 4.9: *Image representations of HOG feature vector. Variations in relations between the input image size and the size of selected cell for freehand brush flower-and-bird painting*

4.5.2 Experiment Result

This section presents TCP classification testing performance evaluations, and summarises them in three tables (Table 4.1, Table 4.2, Table 4.3). The TCP classification performances are explained in the order of datasets, which interprets TCP art style recognitions. The best performances of each dataset are highlighted in bold.

4.5.2.1 GongBi and XieYi Dataset

This section highlights the best and worst classification performance when identifying GongBi and XieYi TCP art style.

CNN model with the whole image as an input can achieve 91.4% mAP in recognising GongBi and XieYi styles.

With a region size of 128, SVM with colour knowledge and cell size as 4 achieves the best mAP 90.3%, while CNN, with Sobel's detector and cell size as 8, performs the best 90.9%, but with a p-value of 0.21 compared to best SVM model, a p-value equals to 0.13 against with a general CNN. The worst performances from SVM is 77.3% with Sobel's in cell size 16, simultaneously, CNN with Sobel's and cell size 4 has 87.9%.

In region size of 256, SVM with colour info success in cell size 8 as 91.2%. The worst performance SVM has are those with Sobel's detector, 78.9% in cell size 4. CNN can achieve 90.6% mAP with cross-feature in cell size 8, but disappointing in cell size 16 with colour enhancement as 87.7%.

When the region size is 512, SVM can obtain 89.3% in cell size 16 with colour enhancement. The worst performance with SVM is 74.5% in cell size 4 with Sobel's kernel. CNN with colour enhancement only obtains 87.6% in cell size 16. But applying cross-features on CNN brings it a success with 92.9% in cell size 8.

In GongBi and XieYi style classification, classifiers with the same model structure (SVM or VGG) do not have significant difference in when they are defined with same region size and cell size. But increasing the size of the salient region did not boost the performance in classification, the same as scaling up the cell size.

4.5.2.2 Landscape, Figure and Flower-and-Bird Dataset

This section indicates the best and worst classification performance when classifying TCP's landscape, figure and flower-and-bird.

CNN model with the whole image as an input has 90.7% mAP in recognising Landscape, Figure and Flower-and-Bird paintings.

In region size of 128, SVM with colour enhancement successes in cell size as 4 with 90.1% mAP, while failing in cell size 16 with Sobel's kernel. The worst performances from CNN is with colour info in cell size 16, 87.2%. But CNN with Sobel's kernel can achieve 91.% in cell size 4.

In the region size of 256, TCP colour knowledge helps SVM achieve the best performance in cell size 8 with 91% mAP. In the meanwhile, SVM with Sobel's approach only obtains 78.7% in cell size 4. CNN with HOG has 88.2% mAP in

cell size 16, but 90.6% with cross-features in cell size 8.

While the region size is 512, CNN with cross-features can achieve 90.8% in cell size 8, while show negative in cell size 16 with HOG but no domain knowledge enhancement as 83.7%. SVM with colour info have 86% in cell size 16. The lowest mAP of SVM has 78.4% with no enhancement in cell size 4.

Both CNN and CNN with hand-crafted cross-features in cell size being 8 are good at classifying TCP data with their contents. They show little difference in the McNemar test while the z-test score is 1.369 and the p-value is 0.86.

We can observe that SVM performed a similar progression when TCP colour is applied to HOG in this dataset, compared with the previous dataset. Simultaneously, SVM with HOG and TCP colour weakened their prediction when Sobel's kernel was integrated. This is also the same as the situation in GongBi and XieYi classification. Also, there is still no investigation on whether region size and cell size can have an impact on the classification.

4.5.2.3 Landscape in Dynasty Dataset

This section shows the best and worst classification performance when classifying TCP's landscape paintings into four dynasties.

Recognising TCP landscape style by dynasty is a critical challenge. CNN model can only obtain 30.7% mAP. And the highest mAP across all methods is around 61% as maximum.

Within region size of 128, SVM with colour enhancement has the best 59.9%

in cell size as 4. The lowest mAP is 47% in cell size 16 with no enhancement. CNN with cross-features shows its sensitivity to HOG cell size when failing in cell size 16 as 27.9%; it has 51.3% mAP in cell size 4.

While the region size is 256, SVM with colour enhancement outperforms others with 61.6% mAP in cell size 8. The worst performance of SVM is with Sobel's approach as 48.7% in cell size 4. CNN with cross-features only has 27.9% mAP in cell size 16, but 54.9% with cross-features in cell size 8.

If the region size is 512, CNN with cross-features can produce 60.4% mAP in cell size 8. But, CNN with colour enhancement only has 41.9% in cell size 16. The lowest mAP from SVM is 40% in cell size 4 with Sobel's kernel. The highest mAP of SVM is 52.9% with colour enhancement in cell size 8.

SVM, a hand-crafted ML model, shows its strength in a small-size dataset. It achieves success with colour domain knowledge enhancement with descriptor size as 8, with 61.6% mAP.

The Landscape in Dynasty Dataset is a small dataset with four classes identified. The simple CNN shows its weakness in data requirements when handling a small dataset. The alternatives with crossing features present their strength and perform better than simple CNN in most scenarios of classification. But their results with the cell size of 16 demonstrate again that the size of the cell is not proportionally to the performance. Also, this set of experiments suggested that SVM's reaction to a small dataset is more stable than CNN.

4.5.2.4 7-Class TCP Object Dataset

The refreshed dataset proposed here can provide an end-to-end fine-trained feature map on TCP object recognition. This feature map is an essential DL representation for our A-RPN TCP object detection model in chapter 5.

This section gives the best and worst classification performance when recognising objects in TCPs.

CNN obtains reasonable result as 57.9%.

In region size of 128, SVM with colour enhancement has the best 57.1% in cell size as 4. The lowest mAP is 36.7% in cell size 16 with Sobel's kernel associating with HOG. CNN with hand-crafted features act worse, with the highest mAP 41.9% in cell size 4, and the lowest 28.4% in size 16.

If the region size is 256, the lowest mAP of SVM is 40.5% with Sobel's kernel in cell size 4. The highest is 53.4% with colour info in cell size 8. CNN with cross-features beat the original CNN model in cell size 8 with a 2.447 difference in the McNemar test, which shows its significant improvement. But the performance dramatically drops when cell size increased to 16. The worst mAP of CNN is 37.2% with colour enhancement.

While region size is defined as 512, no SVM model can achieve mAP higher than 50%. The greatest is 47.4% in cell size 8 with colour domain knowledge, and the smallest mAP is 40.6% with Sobel's approach in size 16. Similar results from CNN returned. The worst is one with colour info in cell size 16, 30.4% in number. The best is 46.6% with cross-features in cell size 8.

As stated, the CNN model with cross-features outperforms the others with nearly 60% mAP when hand-crafted feature combined with HOG, Sobel's kernel and TCP Colour when cell size is 8. The p-value between it and the simple CNN is 0.455. Therefore, this CNN alternative can be promoted as the best CNN model for classifying our 7-Class TCP Object Dataset.

4.5.3 Discussion

This section discusses the results obtained from our TCP classification models. There are 97 pairs of end-to-end training and testing involved in our research. Each of them has a different setting in the classifier model, feature representation, HOG descriptor's cell size, sliding window's size. Meanwhile, referred to table 4.1, table 4.2, and table 4.3, we obtained the answers to the questions defined at the beginning of section 4.5.

Since the number of experiments is relatively large, the model which achieved best performance for each dataset has been presented in table 4.4. (The 2-Style is the GongBi and XieYi dataset. LFF stands for the Landscape, Figure, and Flower-and-Bird dataset. L_Dynasty is the Landscape with Dynasty dataset. TCP Object is the dataset with 7 classes of TCP objects.)

Question 1: Which classification algorithms achieve better performance in the TCP domain, hand-crafted ML (SVM) classifier or Deep CNN classifier (VGG-16)?

Table 4.1: *A summary of TCP classifiers' performance on salient region in size of 128×128 with HOG feature extraction. HOG's cell sizes are defined as 4×4, 8×8 and 16×16. The bolded items indicate the best performance for each individual scenario. If there is no significant difference between the various models in one scenario, we bolded all the results that suitable.*

Image Size 128 x 128	mAP (%)											
	GongBi and XieYi			Landscape (Tang, Yuan, Ming, Qing)			Landscape, Figure, and Flower-and-Bird			7-Class Object Figure and Flower-and-Bird		
Cell Size	4x4	8x8	16x16	4x4	8x8	16x16	4x4	8x8	16x16	4x4	8x8	16x16
SVM+HOG	87.6	85.4	79.7	57.3	56.4	47.0	87.6	85.4	79.7	55.3	51.2	38.4
SVM+HOG+ TCP Colour	90.3	86.1	79.8	59.9	56.8	49.1	90.1	87.3	79.5	**57.1**	53.2	38.3
SVM+ HOG+Sobel	86.1	82.9	77.3	55.4	55.1	47.2	87.7	85.0	75.6	53.3	50.1	36.7
SVM+HOG+ Sobel+TCP Colour	89.2	87	78.5	57.6	55.0	48.9	89.0	87.4	77.7	55.5	50.8	37.2
CNN	91.4			30.8			90.7			57.9		
CNN+HOG	88.7	89.9	89.2	51.0	33.4	28.7	89.2	89.4	87.3	40.5	37.0	29.1
CNN+HOG+ TCP Colour	89.0	90.0	89.1	50.8	34.1	28.7	88.9	89.1	87.2	39.6	37.2	28.4
CNN+ HOG+Sobel	87.9	90.9	88.9	50.4	34.2	29.3	91.0	90.8	88.8	41.1	37.1	28.9
CNN+HOG+ Sobel+TCP Colour	90.1	90.1	88.9	51.3	33.9	27.9	88.8	90.5	89.0	41.9	37.7	28.8

Table 4.2: *A summary of TCP classifiers' performance on salient region in size of 256×256 with HOG feature extraction. HOG's cell sizes are defined as 4×4, 8×8 and 16×16. The bolded items indicate the best performance for each individual scenario. If there is no significant difference between the various models in one scenario, we bolded all the results that suitable.*

Image Size 256 x 256	mAP (%)											
	GongBi and XieYi			Landscape (Tang, Yuan, Ming, Qing)			Landscape, Figure, and Flower-and-Bird			7-Class Object Figure and Flower-and-Bird		
Cell Size	4x4	8x8	16x16	4x4	8x8	16x16	4x4	8x8	16x16	4x4	8x8	16x16
SVM+HOG	80.9	90.0	83.2	50.1	57.5	52.3	80.9	89.0	83.4	44.3	52.3	47.9
SVM+HOG+ TCP Colour	81.1	**91.2**	84.6	51.3	**61.6**	53.1	81.5	91.0	84.1	47.1	53.4	50.9
SVM+ HOG+Sobel	78.9	89.5	84.0	48.7	59.1	50.5	78.7	88.8	82.2	40.5	50.7	46.4
SVM+HOG+ Sobel+TCP Colour	78.9	89.3	85.1	50.0	60.1	50.3	79.0	90.2	83.5	42.8	50.7	49.8
CNN	91.4			30.8			90.7			57.9		
CNN+HOG	88.9	90.3	87.9	50.2	52.2	30.5	89.3	90.0	88.2	46.3	55.8	37.8
CNN+HOG+ TCP Colour	89.0	90.0	87.7	50.1	53.1	28.7	89.0	89.3	88.2	46.5	56.1	37.2
CNN+ HOG+Sobel	89.7	90.3	89.0	52.0	54.2	29.3	90.4	90.5	89.2	48.1	57.3	40.3
CNN+HOG+ Sobel+TCP Colour	90.4	90.5	89.1	51.7	54.9	27.9	90.3	90.6	89.3	49.3	**59.5**	41.0

Table 4.3: *A summary of TCP classifiers' performance on salient region in size of 512×512 with HOG feature extraction. HOG's cell sizes are defined as 4×4, 8×8 and 16×16. The bolded items indicate the best performance for each individual scenario. If there is no significant difference between the various models in one scenario, we bolded all the results that suitable.*

Image Size 512 x 512	mAP (%)											
	GongBi and XieYi			Landscape (Tang, Yuan, Ming, Qing)			Landscape, Figure, and Flower-and-Bird			7-Class Object Figure and Flower-and-Bird		
Cell Size	4x4	8x8	16x16	4x4	8x8	16x16	4x4	8x8	16x16	4x4	8x8	16x16
SVM+HOG	77.5	82.1	87.7	41.6	48.8	51.2	78.4	82.3	85.3	43.3	45.1	42.1
SVM+HOG+ TCP Colour	79.5	83.4	89.3	44.2	50.7	52.9	81.5	84.4	86.0	44.4	47.4	43.5
SVM+ HOG+Sobel	74.5	80.7	86.4	40.0	50.0	48.4	78.7	81.1	83.2	41.2	43.3	40.6
SVM+HOG+ Sobel+TCP Colour	75.1	82.2	87.9	45.1	51.1	50.9	79.0	82.2	83.5	42.3	46.3	42.6
CNN	91.4			30.8			**90.7**			57.9		
CNN+HOG	87.8	91.7	88.0	49.1	58.4	42.3	84.2	89.5	83.7	36.9	44.7	32.6
CNN+HOG+ TCP Colour	88.1	92.0	87.6	48.9	58.8	41.9	84.1	89.3	84.1	37.3	45.0	30.4
CNN+ HOG+Sobel	88.5	91.8	88.3	51.7	59.8	45.9	84.1	90.5	84.9	39.2	46.3	34.5
CNN+HOG+ Sobel+TCP Colour	89.2	**92.9**	88.1	52.3	**60.4**	44.8	85.4	**90.8**	86.1	40.4	46.6	35.0

In the TCP domain, the VGG-16 outperformed SVM in most contexts, especially in a larger dataset. Vice versa, SVM could achieve overwhelming performance in a small-scale dataset, as shown in Landscape with Dynasty dataset. But VGG-16 presented its strength in object recognition. Both SVM and VGG-16's were strong coherent with feature representation used as input.

Question 2: Which features better perform in the TCP classification, HOG (hand-crafted representation), DL feature map (DL representation), or combined TCP cross-feature?

SVM has shown its sensitive to colour enhancement. However, Sobel's edge kernel decreased the performance of SVM when classifying TCP data. A similar conclusion was driven in the natural image domain from Dalal [59]. The reason we promoted Sobel's edge kernel is to enhance edge detection to reduce the impacts of water-and-ink diffusion but received a negative response from SVM. The reason is that SVM is a linear based model, but Sobel's is non-linear. Applying Sobel kernel to the hand-crafted representation is not only increasing the computation complexity but preventing the model from linearly fitting.

Combining DL representation with hand-crafted representation can improve the the performance of CNN model. Although the VGG-16 is not as sensitive to colour as SVM, TCP colour can slightly improve its classification performance. Enhancing edge knowledge and refining it with colour information in VGG-16 can significantly improve the classification performance. This might because the CNN model has low sensation on colour over stridden convolutions, especially

		2-Style	LFF	L_Dynasty	TCP Object
Best	Model	CNN	SVM	SVM	CNN
	Features in Model	HOG, Sobel, TCP Colour	HOG, TCP Colour	HOG, TCP Colour	HOG, Sobel, TCP Colour
	Cell Size	8	8	8	8
	Window Size	512	256	256	256
Worst	Model	SVM+	SVM	CNN	CNN
	Features in Model	HOG, Sobel	HOG, Sobel	HOG, Sobel, TCP Colour	HOG, TCP Colour
	Cell Size	4	16	16	16
	Window Size	512	128	128	128

Table 4.4: *A table to highlight the best and worst performance in feature evaluations for TCP style learning to classify in a total of 97 experiments. All listed include the classifier model, its associating features, descriptor's size, and window size.*

after Conv3. Though local edge information will weaken during convolutions, there is research [49] proved that enhancing edge feature could improve higher-level features, such as shape and contour, which can lead to an improvement in accuracy performance.

Question 3: By implementing TCP domain knowledge to image representation, is it true that it significantly improves classification results?

Based on our research, the answer is "yes". When applying TCP domain knowledge to both for SVM and VGG-16 classification models, they achieved better performance. However, the effective features which significantly improved the models are different, while SVM's is TCP colour, and VGG-16's is the TCP cross-feature combined with TCP colour palette and the Sobel's kernel.

4.6 Summary

This chapter proposes two research topics related to computational aesthetics in the TCP domain: image representation and classification.

The first half of the research undertaken in this chapter focuses on finding effective image representation, which is most suitable for TCP image data. Detailed evaluations of TCP feature extraction and image representation are provided with evidence resulting from the TCP classifications we properly constructed in the second half of this chapter.

The second half of this chapter introduces two representative classification models: the traditional hand-crafted ML model, SVM, and VGG-16 Deep CNN model. Both of them are state-of-art classification algorithms in their era, respectively.

The achievements in this chapter are outlined as follows:

- Constructing four structured TCP datasets with manual processing, filtering, redundancy removing and annotation. They are TCP style-based binary-class dataset with GongBi and XieYi schools; three-class TCP content-based dataset with landscape, figure and flower-and-bird; the dynasty-base landscape style dataset and the 7-class TCP object dataset. These are datasets that can be used in other research applications in our project, discussed in chapter 5 and chapter 6.

- Introducing the sliding window for pre-processing data, which enlarges the

number of images when the classifier has to face a small dataset limitation.

- Reviewing of traditional hand-crafted feature extractor - Histogram of Oriented Gradient (HOG). And computing hand-crafted representation using HOG for TCP classification.

- Applying TCP domain knowledge by defining TCP colour palette encoding; and Sobel's edge kernel, which reduces the impact of water-and-ink diffusion on the TCP image edge information. And evaluating their effectiveness and influences on TCP classifiers.

- Proposing a mathematical procedure that is suitable for crossing DL representation with hand-crafted representation. It is a Hash code mapping transformation with Hamming distance and similarity thresholding. By implementing this algorithm in feature extraction, our project obtains TCP cross-features consisting of DL representation, HOG, and two enhanced features with TCP domain knowledge.

- Explaining the SVM hand-crafted ML classifier and VGG-16 DL classifier methodology and comparing them with different types of image representation as input. Simultaneously setting up end-to-end training and testing on all combinations of the classifier models and feature representations that our project is concerned with, along with variations in sizes of descriptor and sliding window. Evaluation of all 97 experiments is summarised and discussed.

Chapter 5

DL Object Detection in Traditional Chinese Painting

5.1 Motivation

Computational aesthetic tends to model machine to have similar performance as human behaviours. As shown in chapter 4, both hand-crafted ML model and DL model can achieve relatively good performance on recognising TCP art style, while all model can achieve more than 91 % in mAP. However, object recognition with hand-crafted ML model (SVM) and Complex Backbone Network (CBN) DL model (VGG16) do not perform as they generally do in the natural image domain. Can the performances of object recognition be improved by Deep CNN stage-wise development in object detection? This chapter presents experimental investigations on this TCP object detection problem.

There are two main challenges in applying DL to detecting objects in Chinese Paintings: there does not exist an official well-structured Chinese Painting Image Database and associated Ontology; most of the popular DL object detection algorithms are natural-image-trained.

The first one cannot be solved because of licences. But our project insists on the dataset generated and consisted in chapter 4 to provide horizontal comparisons.

In Deep CNN models, higher layer convolutional feature capture high-level semantic knowledge. Low-level visual information is processed in lower layers and then processed in higher layers [132]. With differing low-level visual information, the high-level convolutional features may be too coarse when we project our RoIs from the feature map to the original image. This may be problematic with TCPs, as one might expect XieYi school images' characteristics hugely differ from natural images as shown in figure 5.1.

Deep CNN is known as an ML model that has strong transferability across domains. But there is a few evidence that Deep CNN-based object detectors can keep their success in the TCP domain, comparing to ones in the natural image.

We propose our own Deep CNN architecture, named Assembled Region Proposal Network (A-RPN) object detection models, in the following sections. The A-RPN are invented to investigate the transferability of Deep CNN models from the natural image domain to the TCP domain. And provide pieces of the experimental evidence on its improvement with TCP cross-feature.

Figure 5.1: *The image on the left is a XieYi painting of a black horse, while the the right one is a natural image of a black horse. As shown in the figure, color of an object in a natural image is solid. And the color shading can show differences of brightness and shadow. However, highlight in XieYi painting is demonstrated as white space. Also, it is not necessary to have fully connected edges in XieYi painting, such as the legs of the horse are not connected to the body.*

5.2 Two-stage Object Detector with TCP

As the state-of-the-art object detection architecture, two-stage object detection architecture separates the object location task from the object classification task. Respectively, one region selector generates the region proposal and a classifier to the regions. A two-stage object detector can produce high detection accuracy, but they are slow in detection speed.

The choice of region selectors [215, 216, 86, 217] determines computational cost and storage space of the model. One bottleneck of two-stage object detectors is to find a satisfying region selecting method. Therefore, Ren et al. propose Region Proposal Network (RPN) [26] , which integrates region proposals extraction with shared convolution features of the whole image in the detection network. This implementation allows region proposals to be computed on the GPU, which means

the computational cost is minimised.

The RPN [26] is an extra fully convolutional network added after the last Conv layer of the backbone Deep CNN. After initialising by feature map, RPN can then creates rectangular object proposals with objectness scores. In original Faster R-CNN models, the RPN makes K predictions based on scales and aspect ratios of every sliding region of the feature map. One fully-connected layer defines boxes with four coordinates, presents in 4 $\times$ K vector. Another fully-connected layer identifies whether the box contains an object, a 2 $\times$ K vector returned. RoI is the integration of these two vectors. There are N$\times$N$\times$K RoIs generated from the RPN when the initial feature map has size N$\times$N.

The RPN reduces computation-intensive caused by a large number of proposals in the traditional methods [26] and significantly boosts the localisation accuracy and speed. However, the RoI pooling layer between the RPN and the object classification network converting the feature map from multiscale to fixed-size. It suppresses the translation invariance of the network and conduciveness on object classification. Then the object detector is not position-sensitive to small objects. If the model is not position-sensitive, it might miss to localise objects in flower-and-bird paintings. Flower-and-bird paintings often contains small objects. The ratio between objects' sizes against the image's size can be minimal in the TCP domain. For example, the birds and the person in HuiZong's painting (figure 2.2). This characteristic of TCPs pushing the two-stage object detector to improvement and remaining questions:

- **Question 1:** How to balance translational invariance and translational transformation in TCP object detection?

- **Question 2:** And how to retain position-sensitive to small objects?

The answer can be stimulated by R-FCN [27]. It is an approach to share fully-convolution and introduce position-sensitive RoI pooling so that translational variances are blended into the Conv layers. R-FCN transmits and convolutes throughout the whole image in all learnable layers with spatial information encoded.

Another is Mask R-CNN [135] extended from Faster R-CNN [26] that promotes a Mask network. The Mask network integrates RoIs' object detection and prediction segmentation in parallel. Each RoI is predicted with a pixel-level mask by using bilinear interpolation, known as RoIAlign [135].

Inspired by devoting effort from researchers in the computer vision community, our project proposes a new two-stage object detection architecture that adjusting it with TCP's characteristics in the next section 5.3.

5.3 Assembled Region Proposal Network (A-RPN)

This section promotes an adjusted two-stage object detection architecture with TCP's characteristics, called Assembled Region Proposal Network (A-RPN) [25].

The architecture of our TCPs Object Detector A-RPN is a VGG-16 [77] backbone base model, adopting the popular two-stage object detection strategy of

Faster R-CNN [26], as shown in Figure 5.2. Comparing to very 'deep' backbone architecture ResNet (ResNet has 152 layers) [78], VGG-16's depth is appropriate given the limitation of TCP data. Moreover, choosing VGG-16 can better synchronise with the end-to-end pre-trained TCP Deep CNN resulting from chapter 4.

The A-RPN has three components:

- A general Region Proposal Network (RPN) [26] that return RoIs and Intersection over Union (IoU) scores. (The IoU is defined as the region of interest union with the ground truth bounding box.)
- A Chinese-Kitten RPN (CN-Kitten RPN) concatenates both lower-level and high-level features to generate small object sensitive proposals and refines the RoIs. (The reason for naming the model as CN-Kitten RPN is because the draft model was confusing in recognising TCP kittens as described in the appendix.)
- A Detection Network R-FCN that takes proposal RoIs, and classifies object labels and background.

All the sub-networks within our A-RPN are fully convolutional with respect to RoIs. More details will be described in the coming paragraphs.

The A-RPN first load a hierarchical feature map through shared convolutional layers, which forms a pre-trained resembling VGG-16, and use this feature map *M0* to initialise the general RPN, the CN-Kitten RPN, and the Detection Network

R-FCN [27].

Differing from the original work of Faster R-CNN [26], A-RPN has two regional proposal networks: a general RPN and a CN-Kitten RPN

The general **RPN** generates coarse object proposals from the high-level convolutional layer conv5. Each of these regions is associated with an Intersection over Union(IoU) score, which estimates the probability that the current RoI contains an object. The NMS [218] with a default threshold as 0.7 is applied to the set of RoIs so that only the top 2K scored proposals (RoIs-set1) are returned when K is the number of prediction coherent with scales and aspect ratios of the window. RoIs-set1 is then passed to the next stage, CN-Kitten RPN.

CN-Kitten RPN is an enhanced RPN that combines multi-layer features knowledge and RoI pooling layer to refine RoI proposals. This approach is designed to increase the model's sensitivity to small-scale object detection inspired by the principles of Feature Pyramid Networks[219]. CN-Kitten RPN integrates features from both Conv4 and Conv5 as FCN-16s [136] to serve as input feature maps during convolutions. The upsampling filter is applied on Conv5 to obtain transposed convolutional layer [107, 108] Conv5'. Then L2-normalise each layer per spatial location and re-scales it with the same resolution as Conv4 so that multi-layer features can be concatenated. After normalising Conv4, it is concatenated with Conv5' to obtaining a fine-grained feature map, named convCNK. The final step is to reduce the dimension of the combined feature map to 256 channels with 1×1 convolutoinal layer. Then, we can build a fully-connected layer

for computing an alternative set of 2K scored proposals, RoIs-set2 as explained in figure 5.2. Simultaneously, we applied a bilinear interpolation on convCNK, and obtained a semantic segmentation heat-map of the entire image for later refinement.

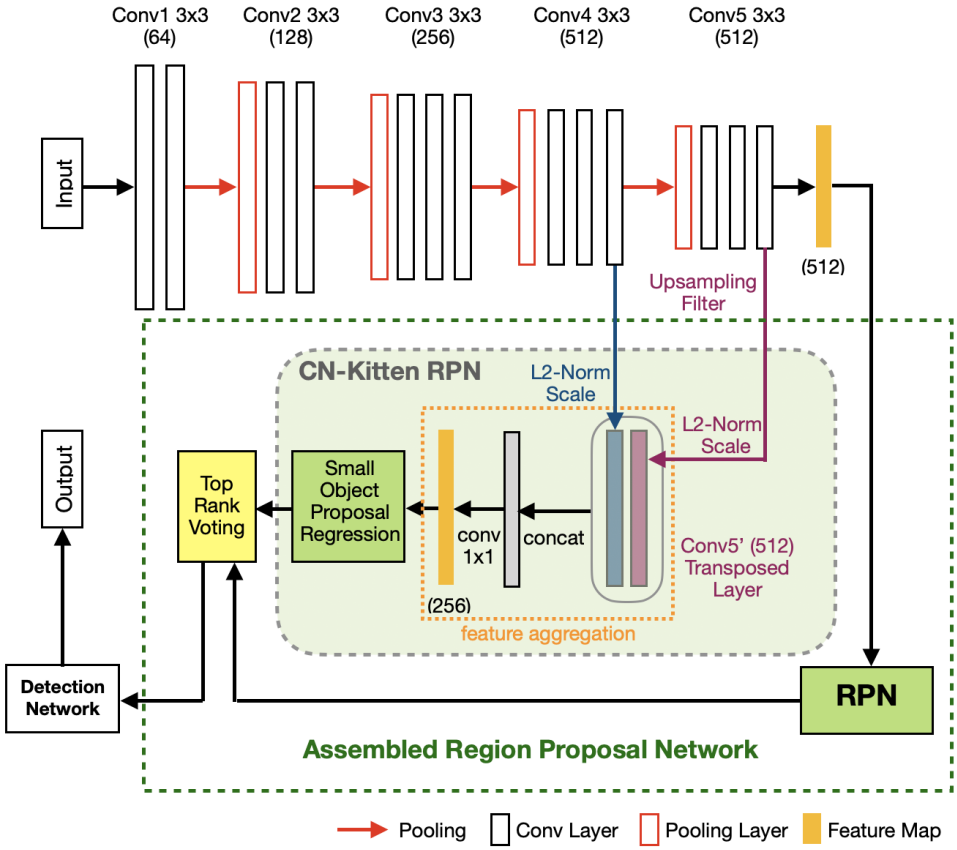

Figure 5.2: *Assembled Region Proposal Network(A-RPN) architecture. The Conv4 and Conv5 represent the last convolutional layers of the Conv4 block and Conv5 block in the figure, respectively. For the CN-Kitten RPN, a de-convolutional is used to upsampling Conv5 so that the features (Conv5') can have the same resolution as Conv4 (but the number of channels stays the same). Before concatenating Conv4 and Conv5', we proposed to use L2-normalisation and scale each feature map extracted from these layers. This is to make sure the downstream values of the pooled features are at reasonable scales when training is initialised. (The re-scale process can be implemented with a learnable "scale layer" that is initialised to 20 in Caffe.) Finally, the dimension of the combined feature is reduced to 256 channels with 1×1 convolutional layer. The small object proposal re-gression is the sliding window process on convCNK with a single scale of 32×32 pixels which return RoIs-set2.*

The region proposals (RoIs) processed in our A-RPN are divided into two sets. One set comes from the generated 2K proposals in the general RPN (RoIs-set1): the other is the small-scale region boxes selected by the sliding window run on convCNK (RoIs-set2). Comparing to Conv5, convCNK can fit small objects better and better utilise fine-grained features. Due to this characteristic, convCNK is used to better developing small-scale object proposals. The same NMS operation in RPN was applied to reduce redundancy. It returned us another set of 2K proposals, RoIs-set2.

We observed no significant drop in recall with small objects (32×32 pixels) comparing to Faster R-CNN when running on the same dataset. In other words, this is a feasible solution to observe locations of small scale objects in TCPs.

Pioneering DL object detector, such as R-CNN family [97, 134, 26] consisted of two sub-networks (two stage-wise). They are discomposed by RoI pooling layer [26], a shared 'fully convolutional' object detection subnetwork. The RoI-wise image classification subnetwork does not share computation. However, this causes a dilemma between optimising translation variance for object detection and translation invariance for image classification. Object translation inside an anchor should produce a meaningful estimation of how well it detects an object. ResNet [78] addresses this dilemma by inserting an RoI pooling layer and a considerable number of region-wise layers in convolution. This region-specific operation deals with the puzzle but causes low efficiency and increased computation. There is fast, accurate, and low computation cost fully shared and convolutional architecture,

known as the region-based fully convolutional networks (R-FCNs) [27]. As a member of the FCN [136] family, they construct a set of position-sensitive score maps to incorporate translation variance.

To addresses the dilemma between optimising translation variance for object detection and translation invariance for image classification, the A-RPN replaces the general RoI pooling layer in Faster R-CNN with a position-sensitive RoI pooling layer that is coherent with a top rank-voting model.

The model projects sets of anchors and refines their IoU scores with the sematic segmentation heat-map. Then a top rank-voting model applied to cross between RoI-set1 and RoI-set2. All anchors were ranked with their IoU scores from the highest toward the lowest into a descending list. The top rank-voting model tracks from the beginning of the list and tries to achieve two tasks:

- To project the selected anchor to the semantic segmentation heat-map and compute an additional IoU score with an overlapped percentage. The overlapped percentage is the ratio between area in common to both segmentations and the total area.

- To remove redundant or highly similar anchors in two sets. The overlapped percentage is calculated by comparing the anchor with one in the opposite set. If the overlapped percentage is higher than 0.7, two anchors have merged by averaging their coordinates. This procedure kept repeating until the model obtains proposals with the highest pair of IoU scores. Eventually, 300 proposals are selected after the top rank voting strategy.

The proposal regions are then entered into the detection network. The detection network identifies and regresses the bounding box of regions likely to contain classes. Classification is applied to each proposed region. Regions are then ranked according to the highest confidence object detection score to finalise the ranked list. Unlike in the original Faster R-CNN [26], we employ R-FCN [27] on the Deep CNN to construct the detection network.

R-FCN uses a bank of specialised convolutional layers to encode as score maps position information concerning a relative spatial position [27]. All the FC layers are removed; instead, all learnable weight layers are computed on the entire image. The final convolutional layer returns a bank of position-sensitive score maps for C + 1 categories (C object categories + 1 background). Each set of score maps for one particular class represents a $k \times k$ spatial grid describing relative position information (top-left, top-centre, top-right etc.) These shared sets of score maps are then used to perform average voting as described in a position-sensitive RoI pooling layer [27]. Selective pooling only returns one score out of $k \times k$ on class prediction. As there are no learning layers, our model reduces the computation cost but have a competitive mAP.

5.4 Experiment and Evaluation

This section sets up two scenarios to verify the performance of our A-RPN architecture. They are:

- **Object Detection without TCP Domain Knowledge:** The first scenario uses pre-trained natural image feature maps from ImageNet [21] to initialise the model. This is to do computational aesthetic study on the Deep CNN's **transferability** from the natural image to the TCP domain without highlighting TCP sensitive features. This scenario is designed to simulate how a human being makes object recognition decisions in TCP only with a natural image learning background.

- **Object Detection with TCP Domain Knowledge:** The second scenario is to apply TCP domain knowledge to our A-RPN architecture. This scenario is an investigation of how TCP domain knowledge influences Deep CNN's performance in object detection? To enable our A-RPN to get TCP domain knowledge, we use the feature maps from the Deep CNN backbone network (VGG) which was pre-trained in chapter 4 to initialise the model.

5.4.1 Object Detection without TCP Domain Knowledge

5.4.1.1 Experiment Setting

This section evaluates the object detection performance of A-RPN on natural image data and TCP data without TCP domain knowledge.

Our experiment introduces the TCP object dataset we generated in chapter 4. The 7-class TCP object dataset contains 1,400 images in seven classes, including

human, horse, cow, bird, plant, cat, and dog. Each image in this dataset is annotated with class labels and segmentation knowledge (bounding box) as ground truth. The number of objects in each image is not equivalent. For comparison with object detection in natural images, we employed the natural image dataset from PASCAL VOC2007 [98]. And regenerate it with the same set of classes contained in the 7-class TCP object dataset, which are randomly selected subsets with these classes for 200 images each. Since the PASCAL VOC2007 only has "potted plant", we randomly selected 100 images from the set. Then, we concatenated them with another 100 images randomly selected from the Oxford 102 flowers dataset [220].

We separately trained DL models on features from both the natural image and our 7-class TCP object dataset and evaluated their predictions. We compared our A-RPN's outputs with two popular DL object detection models: single shot detector method YOLO2 and Faster R-CNN to further investigate image-level classifiers.

Since the scenario is to evaluate A-RPN without TCP domain knowledge, the pre-trained hierarchical feature map used to initialise our detector is a hierarchical feature map pre-trained from VGG 16 in PASCAL VOC2007 [98].

We always keep one-third of our data as the test set and divided the remaining data into 90% training set and 10% validation set. And all the methods were trained and tested on the same data.

The DL models involved are YOLO2 [138], Faster R-CNN [26], and A-RPN.

(The reason why we did not test with YOLO3 [139] is that we failed to manage training and fine-tuning the object detector in the limited period.)

5.4.1.2 Results

In this scenario, we run six experiments: Single Shot detectors on natural images, YOLO(N), and on Chinese paintings, YOLO(P); Faster R-CNN on natural images, Faster R-CNN(N), and on Chinese paintings, Faster R-CNN(P); A-RPN on natural images, A-RPN(N), and on Chinese paintings, A-RPN(P). The results are shown in table 5.1 and figure 5.3, figure 5.4, figure 5.5.

Methods	Natural Image	Chinese Painting
YOLO 2	71.37%	58.48%
Faster R-CNN	73.22%	59.98%
A-RPN	**75.25%**	**61.85%**

Table 5.1: *The mAP comparison of YOLO2, Faster R-CNN, A-RPN object detection performance in 7-class TCP object dataset.*

Table 5.1 shows that all the DL models can achieve more than 70% mAP (mean Average Precision) on natural image object detection. A-RPN achieved the highest classification performance and has significantly higher mAP ($P < 0.02$) than YOLO2 and Faster R-CNN. (Using the McNemar test, the Z-test statistic against YOLO2 and Faster R-CNN are respectively -3.7905 and -2.4188. The P-value are 0.000075 and 0.007786, which means the results are significant).

Object recognition performance significantly drops when applied to TCPs without TCP domain knowledge. The YOLO2 model has an mAP performance

drop of 12.9% while Faster R-CNN falls 13.2% and A-RPN drops 13.4%. The statistical difference in performance between the three methods of TCP object recognition is slight. But A-RPN has proved that it outperforms the other two models on TCP data. Details are below.

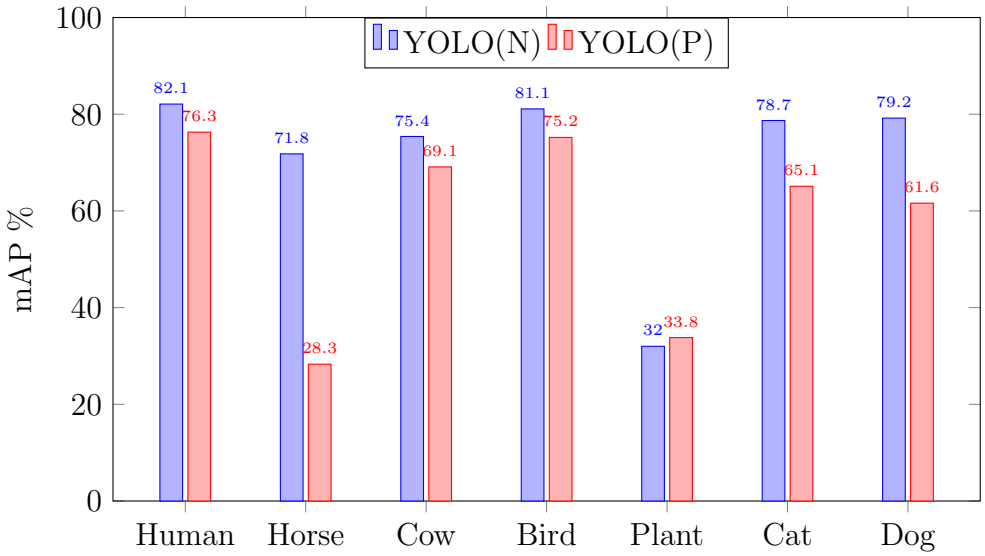

Figure 5.3: *One Shot Object Detector YOLO2 performance of natural image and Chinese painting with classes specification*

The success of object recognition varies greatly between object classes (figure 5.3, figure 5.4, figure 5.5). The performance on classes "Human" and "Bird" was stable and accurate for all methods. Performance on class "Plant" was stable but had low mAP in all models. There are a couple of factors that might lead to this performance. For the natural images, the "Plant" class dataset is constructed from potted plant data and outdoor flower data. These categories are generally separated in the ImageNet dataset, which means the pre-trained network might require more training to narrow the gap in diversities. The second reason is

that most of the "Plant" objects in TCP exist in flower-and-bird paintings as introduced in section 2.2. Sometimes, they could be drawn at the centre but they may act as background (as small objects) with birds and figures. Another reason, plants do highly possible overlap with each other in both cases.

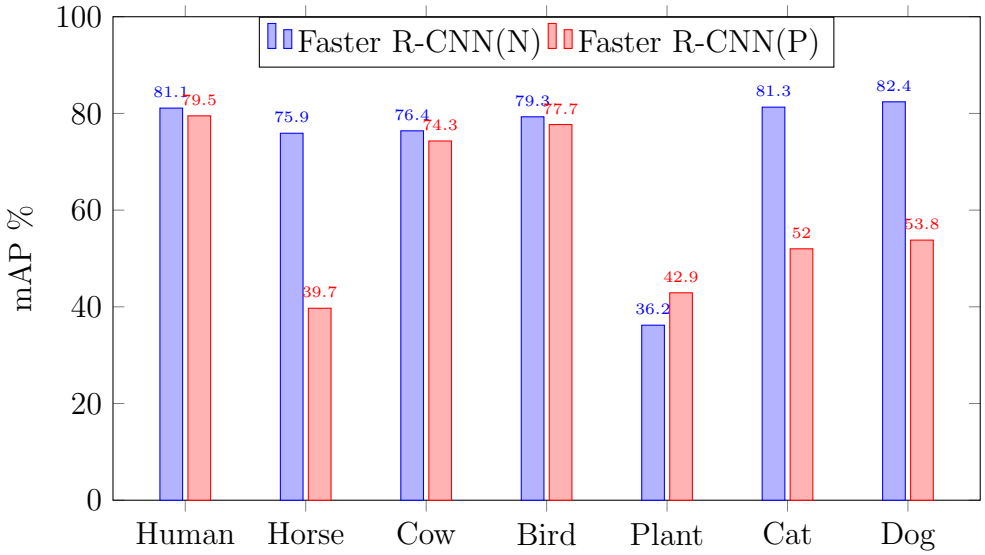

Figure 5.4: *Faster R-CNN Object Detector performance of natural image and Chinese painting with classes specification*

Class "Cow" is more easily recognized on the natural image dataset, but the difference is relatively small (YOLO2: 6.3%, Faster R-CNN:2.1%), while A-RPN increases with 1.8% mAP. All three models confused "Cat" and "Dog" in Chinese paintings. For class "Cat", the drops were 13.6% for YOLO2, 29.3% for Faster R-CNN, and 27.2% for A-RPN. For class "Dog", the drops were 17.6% for YOLO2, 28.6% for Faster R-CNN, and 31.4% for A-RPN. These are the only two cases that the YOLO2 method that has higher accuracy than our A-RPN on TCPs. The largest drop in performance was in the class "Horse" in TCPs: only A-RPN

	Prediction							
Truth	**Human**	**Horse**	**Cow**	**Bird**	**Plant**	**Cat**	**Dog**	**N/A**
Human	**0.82**							**0.18**
Horse		**0.35**	0.03	**0.18**	0.02		**0.27**	**0.15**
Cow		0.09	**0.72**				0.01	**0.18**
Bird				**0.78**	0.01	0.03		**0.18**
Plant				0.04	**0.38**	0.01	0.01	**0.56**
Cat				0.02	0.01	**0.51**	**0.38**	0.08
Dog		0.01	0.09			**0.29**	**0.51**	0.10
Extra RoI								

Table 5.2: *Heat-Map for A-RPN classification without TCP domain knowledge. Miss-classification scenarios with error more than 15% have been highlighted in the table. The N/A column represents the scenario that – there is no bounding box detected over the current ground truth bounding box. The Extra RoI represents the scenario that bounding box is detected an object but the ground truth is false.*

could achieve more than 40% mAP, while their mAPs on Nature Image data are all above 70%.

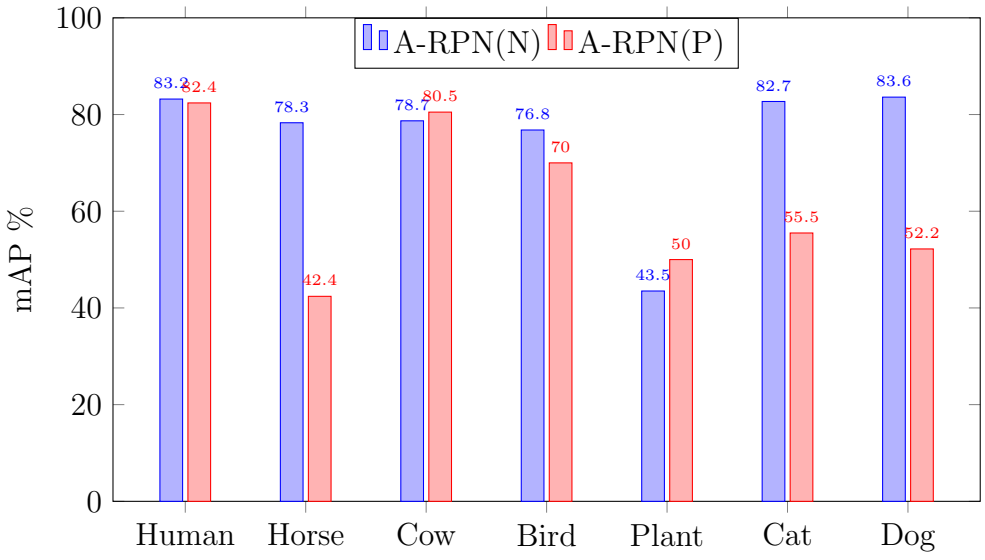

Figure 5.5: *A-RPN Object Detector performance of natural image and Chinese painting with classes specification*

In the experimental scenario that is without TCP domain knowledge, the heat-map table 5.2 shows that our assemble region proposals network architecture has limitation when allocating object-bounding boxes. There are five classes (out of seven in total) failed to achieve a certain detecting rate, above 85%, when localising objects of TCP art in the images. Also, the model confused itself on distinguishing "Cat" and "Dog". Moreover, mis-classifying "Horse" into "Bird" was a general case.

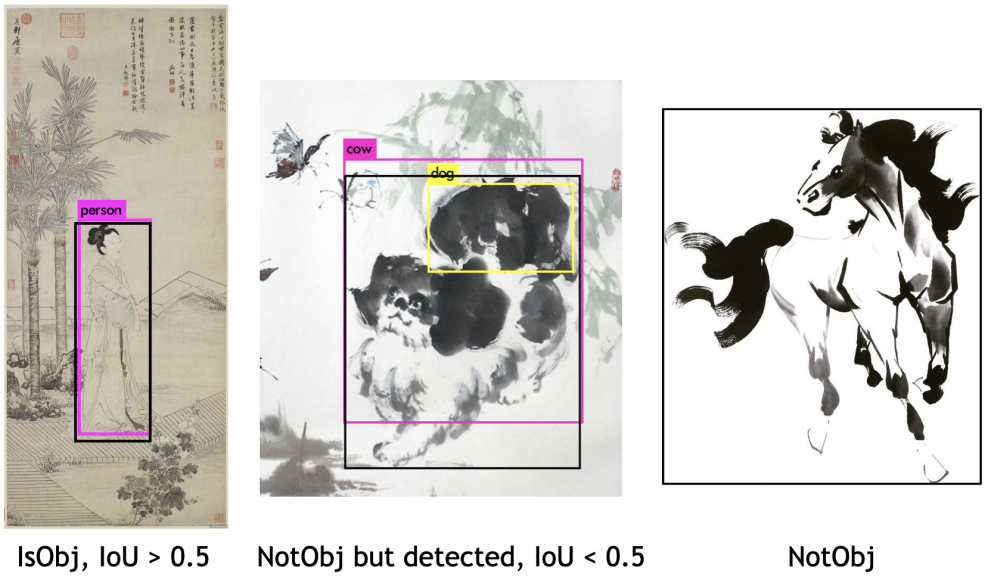

Figure 5.6: *In this figure, we show some bounding box detection samples (if the threshold is 0.5), when IsObj and NotObj represent the predicted binary outcome of identifying whether the current region contains an object. The black outline is the ground true bounding box of an object. Left: object detected with IoU score greater than the threshold; Middle, NotObj, but detected (IOU < 0.5); Right, the object is detected as NotObj.*

To further analyze object segmentation ability of the model, we introduced the concept of **Intersections over Unions (IoU)** (examples are given in figure 5.6). We set a threshold 0.5 (can be any value above 0.5) as the IoU ratio in an RPN:

if the detector has a score greater or equals to this threshold, then the bounding box is marked as an object, vice versa. After verifying object segmentation with IoU in our model, we found that only 63% of the ground truth bounding boxes were detected with ratios above threshold. Around 22% of the objects were not detected, especially in the "Plant" class. Furthermore, there were at least 15% out of all cases, which our A-RPN generated a bounding box over an area that does not contain any object.

5.4.2 Object Detection with TCP Domain Knowledge

5.4.2.1 Experiment Setting

This section is designed to evaluate the A-RPN's object detection performance on TCP data with TCP domain knowledge.

This scenario intends to address a question – whether TCP domain knowledge can improve machine aesthetic learning when detecting and recognising objects of the TCP art. Therefore, the A-RPN is initialised with the features map resulting from our pre-trained VGG-16 in chapter 4. This is a VGG network trained from scratch with feature extractors that observed to improve TCP classifications in chapter 4. The TCP cross-features selections were based on experiments, so that we expected they are TCP sensitive. Then, we validate the results by comparing the results from previous scenario, which is A-RPN tested without TCP domain knowledge.

This experiment keeps the TCP objects dataset we employed in the last section. The 7-class TCP object dataset integrated with seven classes: human, horse, cow, bird, plant, cat, and dog. The total number of images is 1400. The number of images is even in each class, but the number of objects in each image is not equivalent. Every individual image is annotated with class labels and segmentation knowledge (bounding box) as ground truth. We always keep one-third of our data as the test set and divided the remaining data into 90% training set and 10% validation set. And all the methods were trained and tested on the same data.

5.4.2.2 Results

In this scenario, we run an experimental comparison. We applied TCP domain knowledge to the A-RPN and obtained object detection results for the 7-class TCP object dataset. The summaries of the comparison results of this experiment are given in table 5.3 and figure 5.3.

Methods	A-RPN w/ TCP	A-RPN w/o TCP
Prediction (mAP)	69.18%	61.85%
isObj IOU$\geq$0.5	84.74%	63.24%
NotObj IOU$<$0.5	12.79%	22.14%
NotObj but Detected	2.47%	14.62%

Table 5.3: *The mAP and IoU detection rate comparison for A-RPN object detection performance with 7-class TCP object dataset.*

Table 5.3 shows that A-RPN's overall performance gained significant improvement by applying TCP domain knowledge to the initialisation of A-RPN object detector. The A-RPN without TCP domain knowledge was beated. The A-RPN with TCP domain knowledge boosted the object detection performance with a 7.33% increment in mAP. For class "Horse", A-RPN, with awareness of TCP domain knowledge, overwhelmingly outperformed the model that has no TCP knowledge background. Every individual class in the dataset achieved competitive enhancement when detecting objects of TCP art, but the class "Plant" was not able to get great improvement.

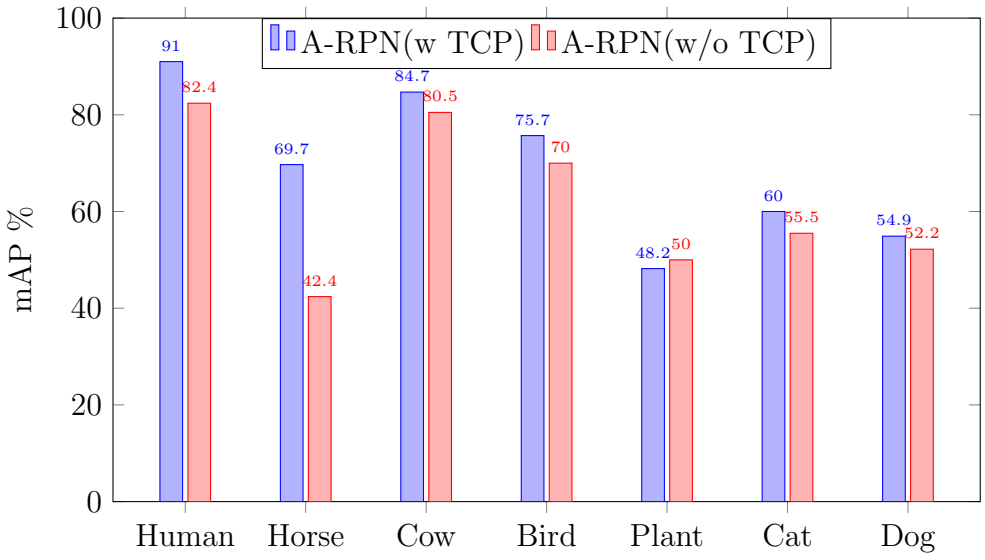

Figure 5.7: *A-RPN Object Detector performance with and without TCP domain knowledge with classes specification*

Compared the heat-map table 5.4 with table 5.2, we can conclude that to apply TCP domain knowledge to the Deep CNN object detector can increase the position-sensitive of the model in object detection. Our region proposal network

	Prediction							
Truth	**Human**	**Horse**	**Cow**	**Bird**	**Plant**	**Cat**	**Dog**	**N/A**
Human	**0.915**							0.085
Horse		**0.645**	0.06	0.05	0.015	0.045	0.07	0.115
Cow		0.045	**0.83**	0.005		0.045	0.04	0.035
Bird		0.045	0.005	**0.795**	0.065		0.005	0.085
Plant				0.105	**0.47**			**0.425**
Cat		0.015	0.005			**0.645**	**0.27**	0.065
Dog		0.035	0.01	0.005		**0.245**	**0.62**	0.085
Extra RoI	0.005	0.025	0.035	0.005		0.03	0.04	

Table 5.4: *Heat-Map for A-RPN classification with TCP domain knowledge. Miss-classification scenarios with error more than 15% have been highlighted in the red. The N/A column represents the scenario that – there is no bounding box detected over the current ground truth bounding box. The Extra RoI represents the scenario that bounding box is detected an object but the ground truth is false.*

achieved an 85% detecting rate when allocating object-bounding boxes with TCP domain knowledge. Only one class (out of seven in total) failed to target the state-of-the-art detecting rate (85%) for object localisation (Plant class). Moreover, by improving the A-RPN network learning strategy with TCP domain knowledge, the situation that the network generated bounding boxes over area that does not contain any object has been reduced to only 2%.

5.5 Discussion

This section presents a discussion refers to the collections of experimental comparisons in section 5.4.1 and section 5.4.2. We first describe the investigate of DL's transferability in object detection from the natural image domain to the

TCP domain. We provide an answer to a computational aesthetic research question – can machine recognise objects of art as human being can without training in art? Then, we evaluate the influence to DL model when applying TCP domain knowledge. Will this enhance the learning process as humans trained to perform better with domain knowledge?

- **Transferability from Natural to TCP**

Putting to one side the differences between human and DL image recognition, it is interesting to consider whether in principle, images in TCPs are harder to recognize using DL. There are a number of possible reasons for the drop in performance of DL in TCPs. First of all, DL models require large training sets, but the number of TCPs that exist in the world is quite limited, and their usage generally involve licenses. Therefore the size of the data set we used potentially prevented the DL models to be fully trained.

Another related reason may be the initialisation of the layer M0 feature map. When the RPN is initialized, we use a pre-trained CNN layer trained on natural images. Natural images differ in a number of ways from images in TCPs. For example, they inherently include perspective, while most Chinese paintings do not use perspective. In TCPs objects are often depicted in a highly abstract manner, easily comprehended by the human visual system, but quite different from natural images.

A third possible cause for reducing performance may be ineffective feature formation when we compute feature maps during convolution. Elgammal et

al.[221] demonstrated that shifts in artistic style could be analyzed and categorized according to five binary characteristics. One was whether the work was 'linear' (contour-led) or 'painterly' (reliant more on brushstrokes denoting light and shadow).[221, 222] Chinese paintings are produced using specialized tools, materials and techniques, which may limit the number of possible low-level features. For example, the majority of Chinese paintings are only black and white. In image processing, white color is often treated as 'no color', with a probability of being an object of zero. Furthermore, CNN models have their own texture representations [219, 107, 108]. Lower layer preserve color and small-scale structure. However, color information is discarded in the upper layers, and edge information is more important.

- **DL with TCP domain knowledge**

DL object detection in the TCP domain is facing two challenges that lead to deterioration in performance. One is the concept of "Empty Space" (Designing of White Space) in TCP. Another is the insufficiency edge features cause by water-and-ink diffusion or TCP abstract manner "Qi".

Many principles of TCP derive from Daoism [223]. For example, empty space is an important concept and a symbol of the void or nothingness. The most crucial text in Daoism states: 'Having and not having arisen together' (Laozi 2). TCP has also been influenced by Buddhism, which emphasizes that 'What is the form that is emptiness, what is the emptiness that is form' (Paramita Hridaya Sutra). These beliefs have led TCPs to stress the concept of Designing White

Space — if one's mind can reach there, there is no need for the touch of any brush and 'formless is the image grand' (Laozi 4). Related to this, an essential canon of TCP describes its rhythmic vitality as Qi, a metaphysical concept of cosmic power: with Qi empty space is not blank; it is alive, like air. This prominent characteristic of TCP turns its treatment of empty space into solid space. Taking the horse in XieYi style in figure 5.1 as an example, the absence of precise edges depicting the outline of a horse feed the human imagination. The horse's body is fragmented into solid ink colour blocks and "white spaces". This might not be difficult for a human to identify it as a horse. However, machine vision analysis can be precise to pixels. Some algorithms use similarity to cluster pixels to generate feature representations or to achieve classification and segmentation. The information delivered by a "white space" and a solid ink colour block can have huge differences, especially when the background is white in the painting.

We hypothesize that the abstract nature of TCPs may fundamentally restrict the ability of DL systems to recognize objects in TCPs. Objects in TCPs do not have fully connected edges as example given in figure 5.1 in chapter 5. When CNN edge detection filters are applied on non-edge pixels in low-level layers, the result matrices are filled with really small numbers or even zeros. Convolutions are then computed based on these inputs. In the CNN texture representation strategy, the smaller the edge detection matrix is, the lower the convolution value. Besides, this CNN problem gets worse during stridden convolutions. Therefore, the application of CNNs can weaken the contribution from ambiguous edge pixels

and further separates edges that should be treated as connected in TCPs.

By applying the TCP cross-features model driven from chapter 4, the A-RPN is enhanced its sensitivity of edge detection and the interpretation of TCP colour. Also, initialisation with TCP domain knowledge further fine-tuning the model in end-to-end training. In conclusion, our A-RPN can achieve better prediction on recognizing TCP objects. This explains that the DL model can be improved by learning domain knowledge to the specific domain.

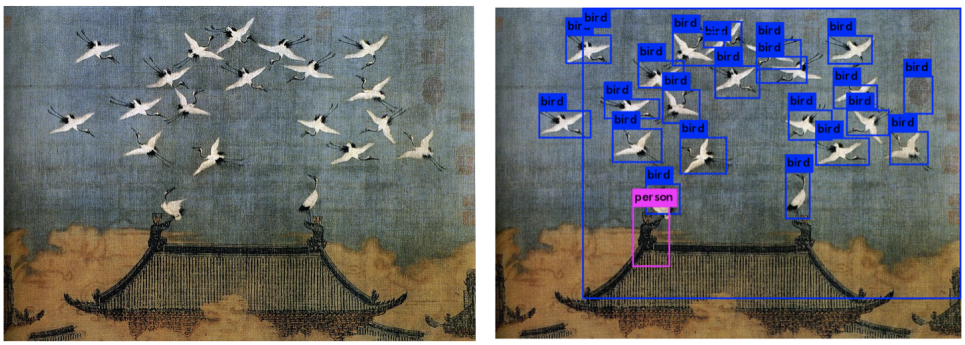

Figure 5.8: *An example of testing on small object detection with A-RPN*

The figure 5.8 shows a successful example of small object detection with A-RPN, but the object augmentation problem remains unsolved in our promoted architecture. Object augmentation handling is a technique of altering the existing data to generate more data so that the object detection process can be artificially expanded to available data, such as re-scaled data or rotated data. However, this is not implemented in our project due to the limitation of time, examples are shown in figure 5.9. The reason for this unexpected performance is that ML models that we used are sensitive to scales of images. Similar to evidence from

chapter 4 classification, the salient region size impacts feature learning, either positive or negative. Two rotated inputs at the bottom more emphasis that object augmentation is one shortcoming of our A-RPN.

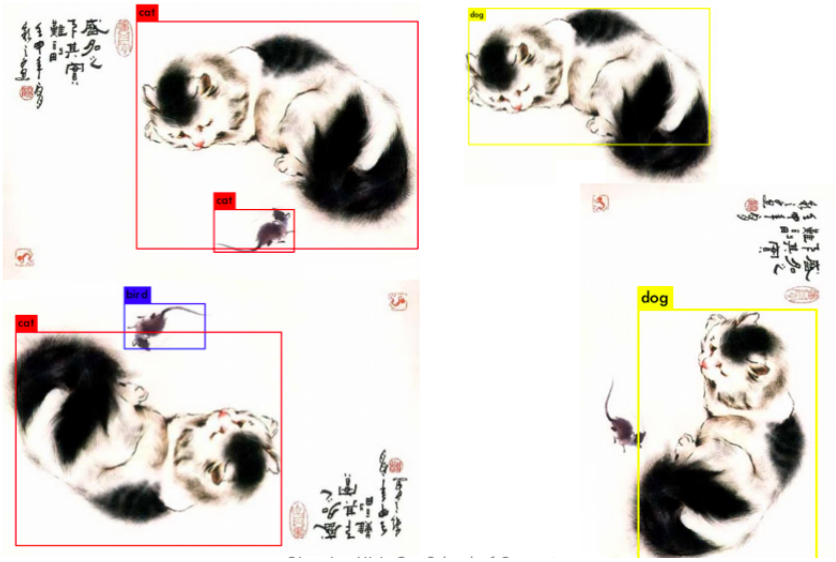

Figure 5.9: *TCP Object Augmentation problem in Deep CNN, when the upper right corner is the detection running on the ground truth bounding box*

5.6 Summary

This chapter introduces and investigates the object detection task related to computational aesthetics in the TCP domain. After briefly describing the methodology of the two-stage DL object detector and TCP object detection challenges in section 5.2, we propose our two-stage DL object detector A-RPN and evaluate its performance in both the natural image and TCP domains. The achievements we obtained in this chapter are outlined below:

- Proposing two-stage object detection architecture A-RPN in section 5.3.

This A-RPN introduces a CN-Kitten RPN, which imports transposed convolutional layers on the concatenation of feature maps. Then the approach is to balance translational invariance and translational transformation. The A-RPN consists of a general RPN and a CN-Kitten RPN and replaces RoI pooling with top-rank voting and position-sensitive pooling. Position sensitivity on small object detection is then enhanced (shown in both sections 5.4.1 and section 5.4.2).

- Evaluating transferability of DL object detector from the natural image domain to the TCP in section 5.4.1. Three sets of experimental comparisons are set up and show the negative responses on domain transformation.

- Investigating how TCP domain knowledge influences Deep CNN model A-RPN on object detection experiments and the model returns positive feedback in the section 5.4.2.

Chapter 6

Traditional Chinese Painting Style Transfer

6.1 Motivation

The way in which human beings learn artistic aesthetics and establish a structured understanding of particular art styles is a gradual process. From classifying and recognising artistic styles to analysing the content within, these all provide the basis for developing painting skills. So can the machine also replicate this sequence of advancing process? The answer to this question is what our computational aesthetic system needs to provide.

In chapter 4, our computational aesthetic system has shown its ability to learn TCP image representation and use this domain knowledge to recognise TCP style better. In chapter 5, our model develops better content recognition,

object detection in our case, with TCP knowledge. Therefore, we challenge our computational aesthetic system to "draw" a TCP painting in this chapter.

We describe the methodology of two milestone applications, Gram-based Style Transfer and Patch-based Style Transfer in NST. Then we promote an enhanced NST application by applying TCP domain knowledge sensitivity.

6.2 Neural Style Transfer with TCP style

6.2.1 Gram-based Style Transfer

As stated in section 3.3.1, before DL bring style transfer applications to the new era, traditional style transfer replying to texture synthesis transformation [147, 148, 149, 150, 151, 152, 153].

The first Deep CNN style transfer model proposed by Gatys et al. [155, 5, 162] is a parametric texture modelling with summary statistics in IOB-NST. The researchers provide a solution by matching content and style's statistics with a minimal loss function. This is implemented by using Gram matrices to represent the style features.

In Gram-based style transfer, the content loss of a chosen content layer is the Mean Squared Error between the feature map F of the content image and the feature map P of the generated image:

$$L_{content} = \frac{1}{2}\sum_{ij}(F_{ij}^{l} - P_{ij}^{l})^2 \tag{6.1}$$

Minimising the content-loss, the model drives the target image feature activation towards the activation of the content image. The smaller the content-loss, the more similar the images. Contours of the content are then transferred to the target image with selections of layers.

Similar approach is employed to the style layers. Gatys's [155, 5] model measures feature in the activated style layers and copy this activation pattern to the target image. The Gram-matrix, comprising correlated features, is introduced to the calculation of tensors output by style layers. Each Gram-matrix is a dot-products between feature activations' vectors of a style layer. If the feature map is F, the Gram matrix G can be formulated as:

$$G_{ij} = \sum_{k} F_{ik} F_{jk} \tag{6.2}$$

If the Gram-matrix has a tiny value close to zero, these two features are not activated simultaneously for the given style image. Target image on this style activation pattern will only be generated when the Gram-matrix returned a large value. Then the style loss L_{style} function is calculated as the Mean Squared Error for the Gram-matrices:

$$L_{style} = \frac{1}{2} \sum_{l=0}^{L} (G_{ij}^{l} - A_{ij}^{l})^2 \tag{6.3}$$

If feature maps from two different images produce the same Gram matrix at a particular layer, the model assumes they have the same representation. Therefore, when the model has two content representations with the same Gram matrix, the

images are expected to hold the same style. Gatys [155, 5] suggests that the combination of features both from shallow and deep layers can lead to the best performance for the Gram-based style transfer model.

A pre-trained CNN generally initialises the Gram-based style transfer model, then takes a pair of content image and style image as inputs. To generate the output result, the model is to minimise the network losses such that the style loss, content loss and the total variation loss were at a minimum. The total loss is written as a weighted sum of both the style and content losses:

$$L_{total} = \alpha L_{content} + \beta L_{style} \tag{6.4}$$

By retrieving the minimal of the above equation, the model resembles the content image to have stylish attributes from the style image and secures pixel-wise smoothness in the output result.

6.2.2 Patch-based Style Transfer

The Gram-based method combines both low-level features and high-level features to get impressive NST result. However, this approach only considers global correlations but ignores local pixel information. Li and Wand [167] employ Markov Random Field (MRFs) and CNN to replace the Gram matrices. The researchers apply the nearest neighbour calculation with neural patch size 3×3 on both content and style Conv layers, while the neural patch is a sub-area of an image with the same size as the convolutional filter. This is known as the fundamental model

of Non-parametric with MRFs in IOB-NST, a NST model that deliver style by optimizing images. Champandard [10] enable a more accurate semantic match with this model by incorporating a segmentation mask over the MRF Loss.

When x_c is the guidance content image, x_s is the style image, and x is the target mixed image at the current Conv layer. The $\Phi(x)$ represents x's feature map. Therefore, the MRFs loss function is formulated as the below equation set:

Style Loss function: The E_s denotes the style loss function, which is MRFs constraint in this context. Then the list of all local patches extracted from $\Phi(x)$ is defined as $\Psi(\Phi(x))$. Each neural patch is indexed by i of the size $m = h \times w \times C$, where h is the height, w is the width, and C is the number of channels for the current layer.

$$E_s(\Phi(x), \Phi(x_s)) = \sum_{i=1}^{m} ||\Psi_i(\Phi(x)) - \Psi_{NN(i)}(\Phi(x_s))||^2 \tag{6.5}$$

The nearest neighbour $NN(i)$ is computed using normalised cross correlation to achieve stronger invariance. And in Champandard's [10] research, the weighted semantic map is taken into account to a further improvement. Where m_s is the size of the semantic map.

$$NN(i) := \underset{j=1,2,\ldots,m_s}{\operatorname{argmin}} = \frac{\Psi_i(\Phi(x)) \cdot \Psi_j(\Phi(x_s))}{|\Psi_i(\Phi(x))| \cdot |\Psi_j(\Phi(x_s))|} \tag{6.6}$$

Content Loss function: E_c guides as the content loss, taking the minimal value of the squared Euclidean distance between $\Phi(x)$ and $\Phi(x_c)$:

$$E_c(\Phi(x), \Phi(x_c)) = ||\Phi(x) - \Phi(x_c)||^2 \tag{6.7}$$

The total loss as a weighted sum of both the style and content losses:

$$L_{total} = \alpha L_{content} + \beta L_{style} \tag{6.8}$$

6.2.3 TCP Sensitive Patch-based NST

Semantic segmentation is a natural step located at classification as a progression from coarse to fine inference. It consists of predicting localisation or detection for a whole input with classes or additional information regarding a spatial location. Making dense predictions inferring labels for every pixel can lead to a fine-grained inference.

From Li and Wand's and Champandard's research, semantic segmentation has demonstrated its strength in increasing NST generation. Therefore, our project proposes a patch-based NST to modelling our TCP computational aesthetic learning system to "draw" TCP artwork.

Since our model aims to achieve style transfer between natural image and TCP, simultaneously, we expect our model can cover all NST scenarios such as TCP to TCP; a detected object in the content image to TCP object, etc. We import all previous research outcomes in chapter 4 and chapter 5.

The section 2.2 and section 4.5.3 indicates that TCP has a rigorous insistence on the unity of style across local to global. Therefore, our TCP sensitive patch-based NST employs global style loss to the original model from Champandard's. Meanwhile, the model proposes content loss control with a Target Mask

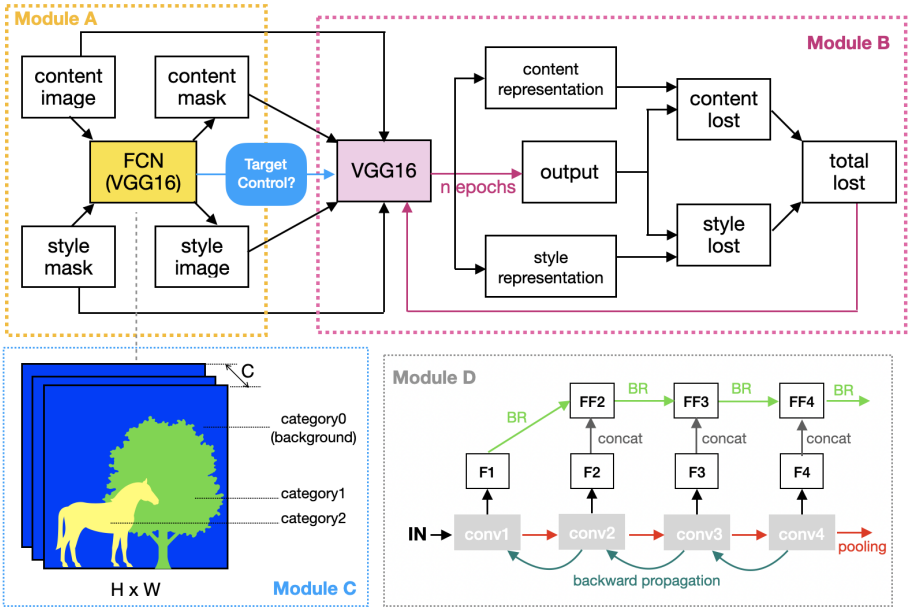

Figure 6.1: *This system diagram provides an overview of our TCP NST model. Both the FCN and the VGG16 are initialised with the pre-trained VGG backbone network (train-from-scratch) in chapter 4. Module A is the automated segmentation function. It returns semantic masks for content input and style input. Module B describes the system flows of the TCP sensitive patch-based NST. The blue block encounters between module A and module B is the binary parameter that activates the area targeted control. Module C is an example of semantic mask results from module A, which denotes the mask matrix to enable the area targeted control. The calculation of content representation and style representation in module B require fused feature maps of content and style. The deep convolutional aggregation process computes the fused feature map FF^N is visualised in Module D, while F1 denotes F^1 in equation 6.11. (Module D only shows 4 conv blocks as the limitation of space.)*

Matrix. This is to enable the NST model to have area control on the content loss. For example, only apply NST on a detected object and set all other pixels as background.

The system flow of our promoted TCP sensitive patch-based model is shown in figure 6.1. All initialisations of CNN are with our pre-trained TCP backbone CNN.

6.2.3.1 Local Style loss and Global Style Loss

The TCP sensitive patch-based NST model computes style loss from both local and global perspectives.

Local Style Loss function E_s is defined as:

$$E_s(\Phi(x), \Phi(x_s)) = \sum_{i=1}^{m} ||\Psi_i(\Phi(x)) - \Psi_{NN(i)}(\Phi(x_s))||^2 \tag{6.9}$$

The nearest neighbour $NN(i)$ is computed with normalised cross correlation:

$$NN(i) := \underset{j=1,2,...,m_s}{\text{argmin}} = \frac{\Psi_i(\Phi(x)) \cdot \Psi_j(\Phi(x_s))}{|\Psi_i(\Phi(x))| \cdot |\Psi_j(\Phi(x_s))|} \tag{6.10}$$

The FF_s is the fused feature map of style, and FF_x is the fused feature map of the target image. The aggregation process to obtain the fused feature map is defined as follows:

Let F^N denotes the output feature map of $convN$, $BR(J, K)$ represents the bilinear resizing that downsamples J to have the same resolution as K with channel remains unchanged, $\oplus$ denotes the concatenate operation on channels, FF^N is the interactive aggregation from $conv1$ to $convN$.

$$FF^N = \begin{cases} BR(F^1, F^2) \oplus F^2, N = 2 \\ BR(FF^{N-1}, F^N) \oplus F^N, N > 2 \end{cases} \tag{6.11}$$

To compute global correlation matrices of the fused feature maps, the fused feature map has to be reshaped as $C \times A$. C represents the channels, and $A = h \times w$ denotes the height h and the width w in the current layer.

$$G_s =< FF_s, {FF_s}^T > \quad and \;\; G_x =< FF_x, {FF_x}^T > \tag{6.12}$$

Global Style Loss is defined by the global correlation matrices of style and the target images. The equation is:

$$L_{globalStyle}(G_s, G_x) = \frac{1}{4C^2A^2} \sum^{h \in H} \sum^{w \in W} (G_{shw} - G_{xhw})^2 \tag{6.13}$$

6.2.3.2 Local Targeted Content Loss

In the traditional content loss function, E_c is the minimal value of the squared Euclidean distance between $\Phi(x)$ and $\Phi(x_c)$. By enabling area targeted control, M_t denotes as a targeted mask matrix, filled with $[0, 1]$. Its offset non-targeted mask matrix is written as M_{nt}. E is an all-one matrix, such that:

$$E = M_t + M_{nt} \tag{6.14}$$

While doing encoding, mask is $M = [M_1, M_2, \ldots M_n]$, and $n \in category = [1, 2, \ldots n]$. The **Local Targeted Content Loss** is formulated as:

$$L_{targetContent}(\Phi(x), \Phi(x_c)) = \sum_{i=1}^{m} \frac{||\Psi_i(\Phi(x))M_t - \Psi_i(\Phi(x_c))M_t||^2}{H_t W_t C_t} \tag{6.15}$$

Then the **Non-targeted Content Loss** is defined as:

$$L_{nonTargetContent}(\Phi(x), \Phi(x_c)) = \sum_{i=1}^{m} \frac{||\Psi_i(\Phi(x))M_t - \Psi_i(\Phi(x_c))M_t||^2}{H_i W_i C_i - H_t W_t C_t} \tag{6.16}$$

6.2.3.3 Weighted Total Loss

In conclusion, our total loss function contains a targeted area content loss weight by α, a non-targeted area content loss weight by β, a local style loss weight by γ, and a global style loss weight by σ, which is:

$$L_{total} = \alpha L_{targetContent} + \beta L_{nonTargetContent} + \gamma L_{style} + \sigma L_{globalStyle} \tag{6.17}$$

For experimental comparison purposes to better evaluate the contribution of employing the global style loss in TCP NST, we derive another total loss function:

$$L_{total} = \alpha L_{content} + \beta L_{style} + \gamma L_{globalStyle} \tag{6.18}$$

which consists of a content loss weight by α, a local style loss weight by β, and a global style loss weight by γ.

An experimental comparison will be included in section 6.3

6.3 Experiment and Evaluation

In this section, we set up four different scenarios: TCP to TCP, natural image (landscape) to TCP, natural image (object) to TCP, natural image (object) to TCP with excluding the background. The background can be set offline by assigning value 0 to the M_{nt}.

The NST experimental comparisons employed three IOB-NST models: Champandard's [10], our TCP Sensitive Patch-based NST (TCP NST) without targeted

area control, the TCP Sensitive Patch-based NST with targeted area control. We aimed to obtain 300 generated images from each individual model. Each output image was generated after 500 epochs on running a NST model.

Unlike classification or object detection, the NST does not have a standard mathematical explanation of its performance evaluation. The evaluation in this research remains an open but important problem. There are two major approaches of evaluation methodologies that might be suitable in the NST research. One is a qualitative evaluation determined by the observers' aesthetic judgments. The other is the quantitative evaluation, such as time complexity. We promote three types of evaluation of our TCP sensitive patch-based NST model. The first one is Fréchet Inception Distance (FID), the second is a machine-based qualitative evaluation, and the second one is the speed-per-iteration quantitative evaluation.

To evaluate the performance, we introduce the concept of Fréchet Inception Distance (FID). FID represents the distance between the feature vector of the generated image and the feature vector of the style image. The closer the distance, the better the effect of the generated model, that is, the sharpness of the image is high and the diversity is rich.

$$\textbf{FID} = ||\mu_r - \mu_g||^2 + Tr(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}) \tag{6.19}$$

where Tr is the trace of a matrix, $X_r \sim \mathcal{N}(\mu_r, \Sigma_r)$ and $X_g \sim \mathcal{N}(\mu_g, \Sigma_g)$ are the 2048-dim activations of the InceptionV3 pool3 layer. μ_r is the mean of style image feature. μ_g is the mean of generated photo's feature. Σ_r is the covariance

matrix of real photo's feature. Σ_g is the covariance matrix of generated photo's feature

Figure 6.2 shows the FID curve against the epochs of generation three different NST models. The Champandard's is the state-of-the-art model. As shown in the figure, the larger the number of epochs, the smaller the FID. Both TCP NST models we promoted perform better than the Champandard's. And the TCP NST with target control achieves slightly better results than the TCP NST without target control, as the first one's FID is steadily decreasing. However, as the difference between them is not large and the figure shows a dramatic drop in FID of the TCP NST without target control at the 350th epoch. We further introduce the machine-based qualitative evaluation to compare these two models.

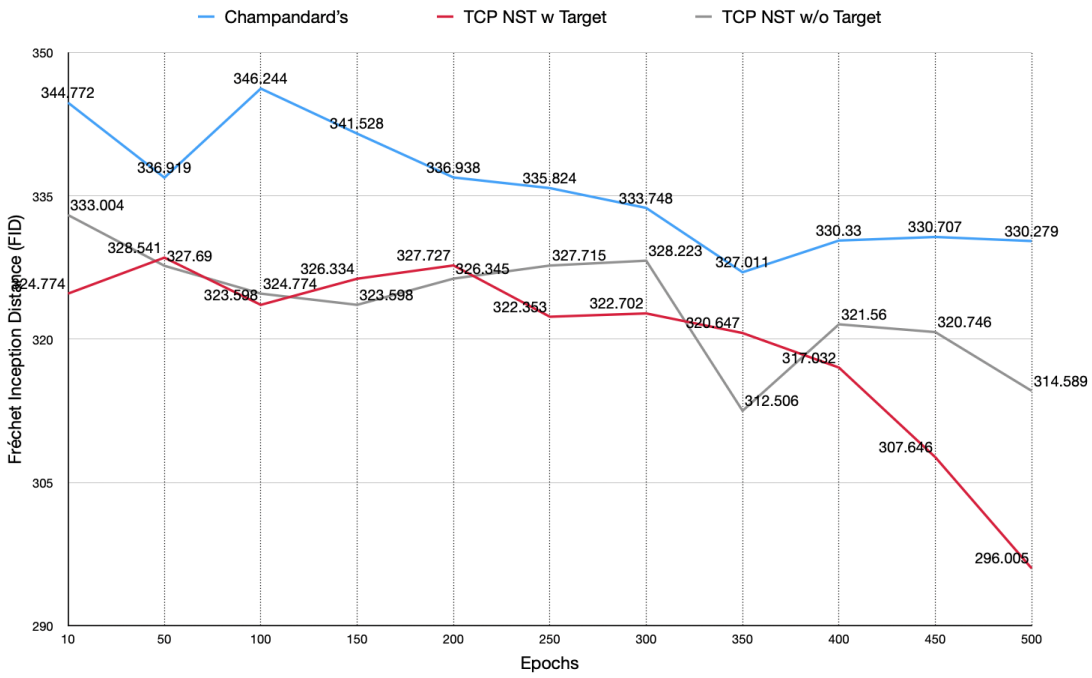

Figure 6.2: *FID curve against number of epochs to show which model obtain better effect of TCP generation. All models are analysed with 300 images and 500 epochs. We computed the FID every 50 epochs to deliver the figure.*

Machine-based qualitative evaluation is a evaluation method widely used in computational aesthetics [15] area. It can be designed as a classifier to distinguishing natural images and artworks, in our case, TCP images. Our project retrieved the GongBi-and-XieYi dataset and Deep CNN with TCP cross-features employed in chapter 4. Next, the project trained a natural-TCP image classifier with the TCP dataset and 3872 natural images randomly selected from ImageNet. This classifier ends with 93.5% accuracy on recognising natural images and TCP images.

In order to retain the consistency, especially for the speed evaluation, we reshaped all images to 512×320 pixels with bicubic interpolation. Then we evaluated 300 TCP artworks generated from our TCP NST model. Results are shown in table 6.1.

	Accuracy(%) and Speed(s)	
	TCP style recognition	Speed per iteration
TCP NST w Target	85.32	0.775
TCP NST w/o Target	89.78	0.533

Table 6.1: *TCP NST's classification accuracy and artwork generation speed. The speed per iteration is the time required for running one single epoch of generation.*

According to table 6.1, although the TCP NST with target control can obtain better FID, it is more time-consuming than the TCP NST without target control. The accuracy difference is 4.46% but the p-value is 0.502, which does not suggest the classification in predictions is significant.

Therefore, we determine not to make any judgement on which TCP NST we promoted in our project is the best at this stage. But we will provide some example outputs and discussion of them.

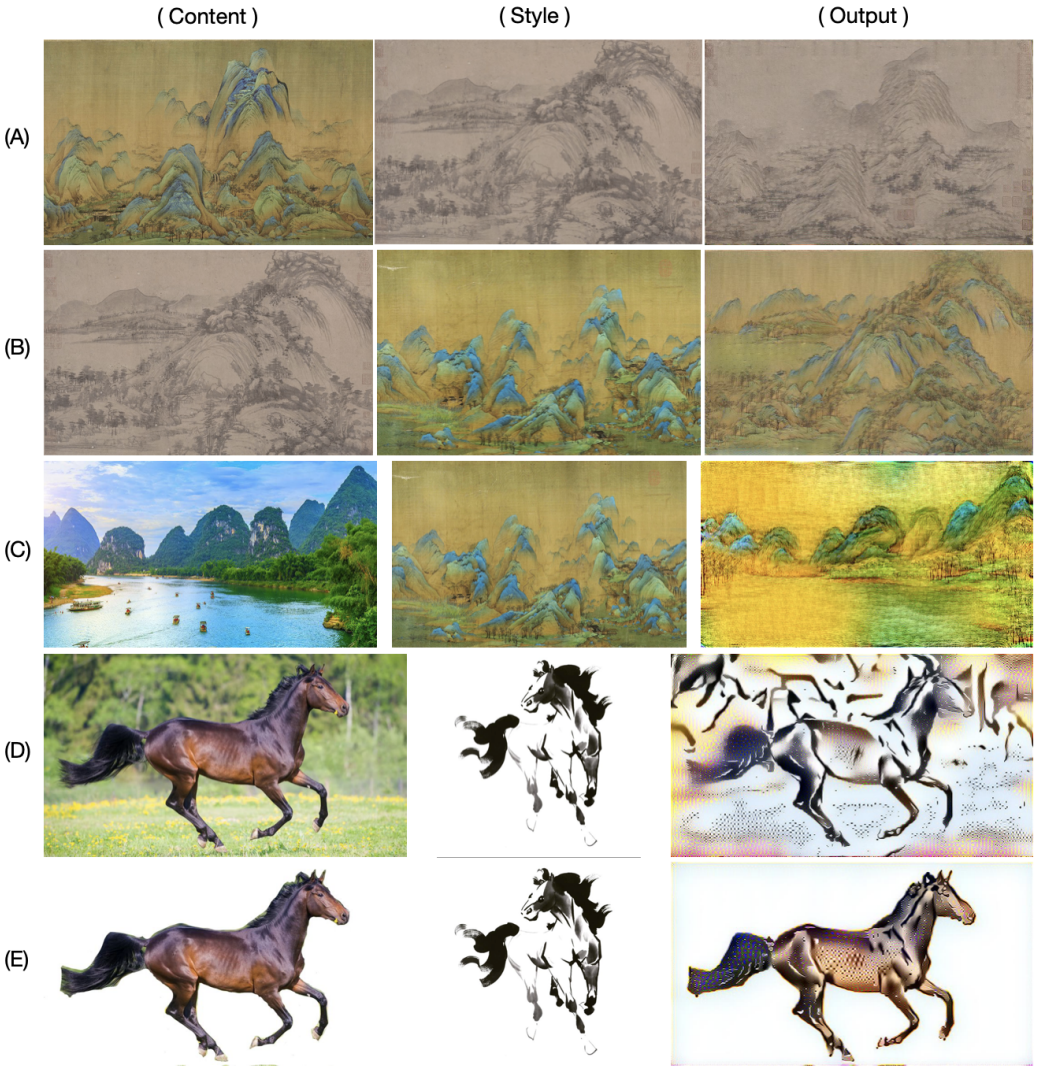

Figure 6.3: *Examples outputs for four different scenarios with our TCP Sensitive Patch-based NST, while (A) and (B) are both TCP to TCP style transfer. (C) is natural image to TCP transfer. (D) is a horse photo to TCP transfer example of TCP NST with target control but keep the background of the object. (E) is a horse photo to TCP transfer example of TCP NST with target control and set the background as empty.*

Figure 6.3 shows examples outputs of four different scenarios resulting from

our TCP sensitive patch-based model with targeted area control. Both A and B are TCP-to-TCP style transfer, while A is GongBi to XieYi, and vice versa, B is XieYi to GongBi. Their FID scores between the output and the style are 188.33 and 158.76 respectively. Compared to the average FID values in figure 6.2, this pair of examples show that the TCP NST does a good job on landscape transferring between GongBi and XieYi styles.

C is a style transfer from a natural landscape photo to a TCP landscape painting with an FID score of 243.21. The outcome in C compared with A and B suggested that our model might be sensitive to the brightness of the content image. We modified the brightness of the content image B1 along with the brightness of the style image B2 and computed the FID scores for brightness adjusted generations. As shown in table 6.2, brightness level does have an impact on NST. This experiment offers an alternative hypothesis that less difference of brightness between the content image and style image might produce better generation.

Brightness	FID
B1	234.207
B1-0.33*(B2-B1)	222.579
B1-0.66*(B2-B1)	179.265
B2	147.630

Table 6.2: *Example C in figure 6.1 with brightness adjustment and the respective FID scores between the outputs and the style image.*

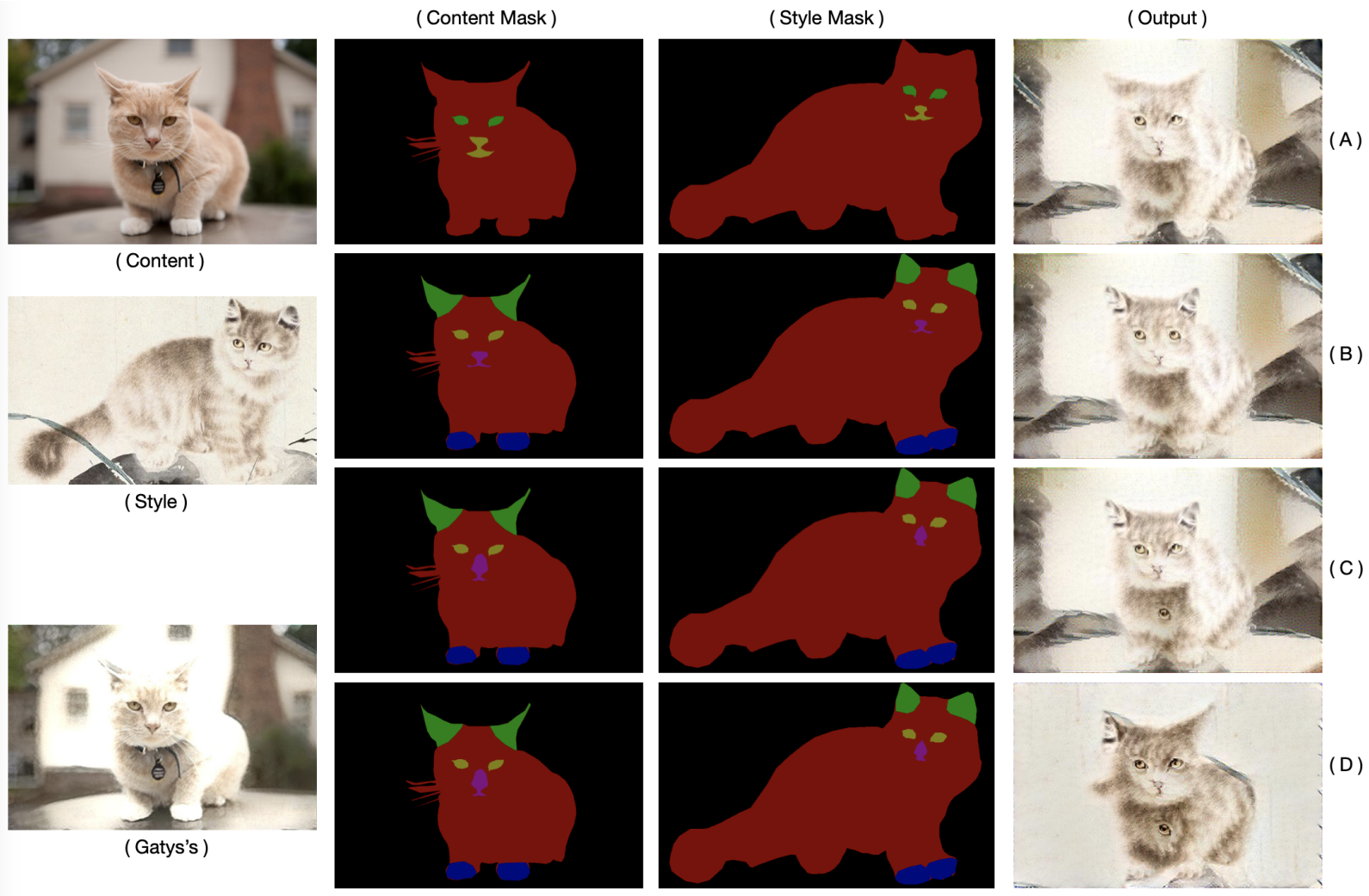

Figure 6.4: *NST comparisons with saliency detection results and example results returned by 4 different IOB-NST: Champandard's[10], our TCP-NST model without targeted area control, TCP-NST model with targeted area control, TCP-NST model with targeted area control and set background offline. Gatys's[5] result is provided as a reference*

D in figure 6.3 is a natural object, a horse, "painting" in TCP XieYi style with FID value as 292.87 without setting the background as off. We tended to set example E as the background to be off. But the result is a negative response of re-drawing object in TCP style, as its FID is 579.74. (There is another pair of on-or-off background controlling examples of an object, a cat, in GongBi style. The results are shown in figure 6.4.)

In order to better evaluate the on-and-off target area controlling TCP NST, we manually prepared a set of segmentation masks that provide the ground truth outlines of objects in the images. This is to make sure an object of interest could be "cut-off" precisely in all situations. And this is particularly important for segmenting masks from TCP paintings, especially XieYi paintings. Simultaneously, our project has automatically segmentation function on sub-parts learning of objects which is a FCN [136] model enhanced from chapter 4. In other words, the sub-parts of an object is learned while the outline is defined.

Besides the computational aesthetic system provides its answer on the NST results, the figure 6.4 offer human observers to make aesthetic decision on our TCP NST model. Gatys's[5] and Champandard's[10] research are listed as references. The content mask and style mask are saliency detection results respect to content images and style images. Ordering from A to D, the outcomes are: Champandard's, TCP NST without target area control, TCP NST with target area control, TCP NST with target area control and set background off. The respective FID scores are 137.72 for A, 136.31 for B, 143.79 for C, 97.84 for D.

From FID aspect, the output of D holds the most similar feature vector to the original style image. But from the human aspect, both C and D's outputs have an extra eye in front of the chest of the cat. Especially it is obvious that there is no sub-part returned in their content masks. Where the eye comes from remains a question.

Unfortunately, our project cannot provide an objective and confident solution to solve the NST problem in the TCP domain. Neither figure 6.3 figure nor 6.4 can claim that our TCP NST model is able to return one-hundred-percent positive response. Though style recognition is competitive, the art generation speed is slow compared to some state-of-art NST models verified before [6].

6.4 Summary

For the final stage of our computational aesthetic learning system, we employ an NST application. This chapter summarises the core mathematic behind two milestone IOB-NST models, Gram-based style transfer and patch-based style transfer. Then we introduce TCP domain knowledge to re-design a TCP style transfer application that might be more in keeping with a human being's level of satisfaction. We outline our achievements as follows:

- Proposing a TCP NST model that combines local style loss and global style loss to better synchronise the image style. This model also invited Target Mask Matrix to the content loss calculation. The targeted area control

enables the TCP NST model to have more precise content representation to improve detail generation.

- Evaluation of the TCP NST model's outcomes is provided but not as positive as expected. Generated TCP artworks are entered back into our computational aesthetics learning model and return reasonable responses. But far more can be improved in the future.

Chapter 7

Conclusions and Future Work

In this thesis, we address computational aesthetics learning in the TCP domain. We focus on imitating the human visual learning process on TCP and trying to provide computational aesthetic decisions and outcomes, respectively, recognising objects of arts and generating TCP artworks. Computational aesthetics learning regularly starts with extracting essential aesthetics information, and this is implemented by our image representation descriptor. Based on the TCP image representations, our TCP computational aesthetics learning system achieves good recognition results in TCP styles, content schools, and objects. The system also presents reasonably good quality TCP generative artworks in some cases, such as landscape. We provide our answers through theoretical analysis and empirical investigations on extensive experiments. Conclusions regarding computational aesthetics learning on TCP and potential directions of future work are summarised in this chapter.

7.1 Conclusions

In the following, we will summarise our approaches and draw conclusions from research studies which are described in Chapters 4, 5 and 6. We will also outline how our project met the objectives stated in section 1.2.

- **To investigate effectiveness of features when constructing the precise image representation of TCP.**

Our computational aesthetics learning system can learn and interpret TCP formal aesthetics style as image representations in the computer vision domain. The image representation descriptors proposed in section 4.3 provide the solutions. These image representation descriptors deliver both hand-crafted representations (HOG) and DL representations (VGG feature maps). In order to represent TCP aesthetics information more precisely, TCP domain knowledge is applied to the descriptors. HOG with Sobel kernel is proposed to reduce the impact of water-and-ink diffusion on edges, and TCP colour palette encoding based on Basic Color Terms [7] is invited to interpret the logic of culture. We also promote a cross-feature mathematical transformation to concatenate hand-crafted and DL representations in section 4.3.4, which is Hash mapping encoding with Hamming distance and similarity thresholding. All these TCP related feature extractions improve feature selection performance at different levels. And our descriptors have shown their successes to achieve the effectiveness of feature selections. Extensive experiments in section 4.5 provide evidence to these

investigations.

- **To develop TCP Classifiers when distinguishing TCP's style and content.**

Our computational aesthetics learning system can classify TCP art styles and contents. This function is implemented by a pair of TCP classifiers employed in section 4.4, a hand-crafted ML model SVM and a DL model VGG-16. We evaluate these TCP classification models in section 4.5 and obtain large amounts of empirical evidence to draw some conclusions. First, both SVM and VGG-16 can achieve competitive results in TCP image classifications. Our computational aesthetics learning system can achieve 92.9% mAP on classifying GongBi and XieYi paintings; around 91% on identifying main contents in TCP; 61.6% on distinguishing dynasties of landscape paintings; and approximately 59% when recognising seven types of TCP objects. Second, the TCP classification performance when identifying styles or contents can be improved by entering the image representations with TCP knowledge defined in section 4.3. SVM is sensitive to TCP colour palette encoding, and VGG with cross-features can achieve better performance. Last, the DL model outperforms the hand-crafted ML model in most contexts, but SVM can achieve better results in a small dataset.

- **To propose a TCP object detector that can recognise the basis (content element) of computational aesthetics.**

Our computational aesthetics learning system can recognise objects of art in

TCP. A two-stage object detection architecture A-RPN is proposed to implement this function in section 5.3. This architecture consists of a general RPN and a CN-Kitten RPN. The CN-Kitten RPN concatenates feature maps with transposed Conv layers to balance translational invariance and translational transformation. We replace the RoI pooling with a top-rank voting model and position-sensitive pooling to enhance position sensitivity on small objects. The A-RPN outperforms some state-of-the-art models such as YOLO2 and Faster R-CNN when detecting TCP objects. This performance can be further improved by applying TCP domain knowledge. A-RPN with TCP knowledge achieves 69% mAP in TCP object detection task. Empirical evidence is shown in both section 5.4.1 and section 5.4.2.

- **To establish a TCP's style sensitive style transfer algorithm.**

Our computational aesthetics learning system can generate TCP artwork using content information from a content image and style representation from a style image. A TCP NST algorithm is proposed in chapter 6 with a combined calculation of global style loss and local style loss. We promote Target Mask Matrix to enable area control on content loss computation in section 6.2.3. Therefore, the TCP NST algorithm can achieve more precise saliency detection. Our system can produce good quality TCP artworks. Also, these outcomes can be recognised by the computational aesthetics learning system. Results are shown in section 6.3.

7.2 Future Work

In spite of the contributions we had made in this research study, our project has some limitation and unexplored problem. Concerning our research objectives, the computational aesthetics learning system we promoted managed to achieve the key results but not accomplished all of them. In this section, we will discuss its limitations and suggest potential directions for future work.

- Human visual learning on artistic aesthetics is a gradual process. Our computational aesthetics learning system was designed to imitate this. However, according to Hawkins' theory [224], "what cortical columns do is to attach reference frames to objects in the world and also to abstract concepts." Human brains can attach frames to external objects in "what" columns and use "where" columns to internal mapping them to all related reference in our body. This means any information our brains process will activate all the knowledge in every domain. Our system has been designed to share TCP knowledge by employing unifying initialisation to all CNNs involved (in classification, object detection, and NST), but their communications were lost after target-specific training. How to retain the interconnection among all? This seems to be a question that a P.h.D cannot answer.

- Essential TCP skeleton features can be used to define the contours of objects and brushstroke identifications. Its importance was proved by previous research studies [186, 193]. However, constructing an effective TCP skeleton

feature extractor is difficult. The influence factors of water-and-ink diffusion can be diverse based on materials' chemical structures. This is the reason that most existing TCP brushstroke identification is determined concerning a single artist. Furthermore, the TCP concept, "Designing White Space", leads to the fact that most TCP objects do not have fully-connected edges and clear contour. But recently, SketchGAN [225] was proposed to solve the problem of sketch completion. This model can recognise and edit incomplete sketch after identifying their types. This might be a sensible solution to the problem of TCP incomplete contour. In conclusion, building a TCP skeleton feature extractor is a valuable research direction we should carry on.

- In section 4.3.4, we define a mathematical transformation that allows the image representation descriptor to concatenate hand-crafted and DL representation. We use $[0, 1]$ Hash mapping is because it is straight forward and easy to fit popular DL activation functions (Sigmoid and ReLUs) so that the computational cost can be reduced. However, these formulations can be rewritten in Euclidean space to keep higher feature space instead of hashing. Simultaneously, the Hamming distance can be replaced, for example, by chi-square calculation in unmapped data.

- The A-RPN is not perfect. Although it can achieve competitive results, the additional RPN, CN-Kitten RPN, slows the model down. The detection

network, which is determined in the A-RPN model, can be used to generate a semantic segmentation mask of the detected object. These semantic segmentation masks can be entered as inputs to the TCP NST model. However, the current model is trained on only seven object classes. In order to fully connect A-RPN and the TCP NST, more training should be employed in the future.

- As stated in 1.1, good aesthetics need to meet satisfaction in terms of both psychological and physical requirements. There is a statement [224], "intelligent machines will not have human-like emotions and drives unless we purposely put them there." A machine has no understanding of what is satisfaction. Therefore, most NST algorithms promoted now-a-day remain on the stage of "mapping" content to style, so do ours. We used to have a designed (figure A.2) that allowed the NST to "learn" a generic representation for a specific TCP style and generate artworks without any content image, but we failed to implement it because of the due to time also the complexity of involving relational knowledge graph or NLP which might be needed. This is another potential direction of studying NST and computational aesthetics.

As a new discipline, there are many topics in computational aesthetics that remained un-investigated. We could not attempt all of them in our thesis, neither list every possible research direction for future work in this section. But we will carry on related research in the future.

Bibliography

[1] Bruno A Olshausen and David J Field. Natural image statistics and efficient coding. *Network: computation in neural systems*, 7(2):333–339, 1996.

[2] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[3] Benjamin Bergner. Automatic landmark detection for preoperative planning of hip surgery using image processing and convolutional neural networks. *PhD Theis Doi: 10.13140/RG.2.2.19674.18884*, 03 2018.

[4] Frieder Nake. Database of digital art. `http://dada.compart-bremen.de/item/agent/68`, 1960s. [Online; accessed 10-March-2021].

[5] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[6] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*, 26(11):3365–3385, 2019.

[7] Brent Berlin and Paul Kay. *Basic color terms: Their universality and evolution*. Univ of California Press, 1991.

[8] Joost Van De Weijer, Cordelia Schmid, and Jakob Verbeek. Learning color names from real-world images. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[9] Coffee Cream的博客. 支持向量机（support vector machine, svm）. `https://blog.csdn.net/coffee_cream`, 2020. [Online; accessed 7-January-2022].

[10] Alex J Champandard. Semantic style transfer and turning two-bit doodles into fine artworks. *arXiv preprint arXiv:1603.01768*, 2016.

[11] Richard L Gregory. *Eye and brain: The psychology of seeing*, volume 80. Princeton university press, 2015.

[12] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

[13] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[14] Bruno A Olshausen and David J Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, 37(23):3311–3325, 1997.

[15] L Neumann, M Sbert, B Gooch, W Purgathofer, et al. Defining computational aesthetics. *Computational aesthetics in graphics, visualization and imaging*, pages 13–18, 2005.

[16] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.

[17] Ian Roberts. *Mastering composition : techniques and principles to dramatically improve your painting*. North Light Books, Cincinnati, Ohio, 1st edition, 2008.

[18] Immanuel Kant. *Kritik der urteilskraft*, volume 39. Meiner, 1913.

[19] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Kailyn Schmidt, et al. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv 407007, doi:10.1101/407007*, 2018.

[20] Martin Schrimpf, Jonas Kubilius, Michael J Lee, N Apurva Ratan Murty, Robert Ajemian, and James J DiCarlo. Integrative benchmarking to advance neurally mechanistic models of human intelligence. `https://www.cell.com/neuron/fulltext/S0896-6273(20)30605-X`. Neuron, 2020.

[21] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[22] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[23] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[24] Elliot J Crowley and Andrew Zisserman. The art of detection. In *European Conference on Computer Vision*, pages 721–737. Springer, 2016.

[25] Qianqian Gu and Ross King. Deep learning does not generalize well to recognizing cats and dogs in chinese paintings. In *International Conference on Discovery Science*, pages 166–175. Springer, 2019.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[27] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

[28] George David Birkhoff. *Aesthetic measure*. Cambridge, Mass., 1933.

[29] Gary Greenfield. On the origins of the term computational aesthetics. In *The Eurographics Conference on Computational Aesthetics in Graphics, Visualization and Imaging*, pages 9–12, 2005.

[30] Nick Zangwill. Aesthetic Judgment. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, spring 2021 edition, 2021.

[31] Wikipedia. Norm (philosophy) — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Norm%20(philosophy)&oldid=967950914`, 2021. [Online; accessed 10-March-2021].

[32] Donald K. McNamee. The documents of 20th-century art: The tradition of constructivism ed. by stephen bann (review). *Leonardo*, 9:78 – 78, 1976.

[33] Martin Heidegger, Gregory Fried, and Richard Polt. *Introduction to Metaphysics: Second Edition*. Yale University Press, 2014.

[34] 陈师曾. 中国绘画史. BEIJING BOOK CO. INC., 2014.

[35] Shuqiang Jiang and Tiejun Huang. Categorizing traditional chinese painting images. In *Pacific-Rim Conference on Multimedia*, pages 1–8. Springer, 2004.

[36] Xinming Wu, Guofeng Li, and Ye Liang. Modeling chinese painting images based on ontology. In *2013 International Conference on Information Technology and Applications*, pages 113–116. IEEE, 2013.

[37] Dasha Bogdanova, Jennifer Foster, Daria Dzendzik, and Qun Liu. If you can't beat them join them: handcrafted features complement neural nets for non-factoid answer reranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 121–131, 2017.

[38] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *arXiv preprint arXiv:1406.2199*, 2014.

[39] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4305–4314, 2015.

[40] Shichao Zhao, Yanbin Liu, Yahong Han, Richang Hong, Qinghua Hu, and Qi Tian. Pooling the convolutional layers in deep convnets for video action recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(8):1839–1849, 2017.

[41] John Heil and John Fergusson Heil. *From an Ontological Point of View*. Oxford University Press on Demand, 2003.

[42] Giuliana Ramella and Gabriella Sanniti di Baja. Color histogram-based image segmentation. In *International Conference on Computer Analysis of Images and Patterns*, pages 76–83. Springer, 2011.

[43] Fatih Kurugollu, Bülent Sankur, and A Emre Harmanci. Color image segmentation using histogram multithresholding and fusion. *Image and vision computing*, 19(13):915–928, 2001.

[44] Michael J Swain and Dana H Ballard. Color indexing. *International journal of computer vision*, 7(1):11–32, 1991.

[45] James F Blinn and Martin E Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19(10):542–547, 1976.

[46] B Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91—97, March 1981.

[47] Bela Julesz. Texton gradients: The texton theory revisited. *Biological cybernetics*, 54(4):245–251, 1986.

[48] Suraya Mohammad. *Textural Measurements for Retinal Image Analysis*. The University of Manchester (United Kingdom), 2015.

[49] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[50] Theo Gevers and Arnold WM Smeulders. Color-based object recognition. *Pattern recognition*, 32(3):453–464, 1999.

[51] Bartlett W Mel. Seemore: combining color, shape, and texture histogramming in a neurally inspired approach to visual object recognition. *Neural computation*, 9(4):777–804, 1997.

[52] Padmavati Shrivastava, KK Bhoyar, and AS Zadgaonkar. Image classification using fusion of holistic visual descriptions. *International Journal of Image, Graphics & Signal Processing*, 8(8), 2016.

[53] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001.

[54] Radhakrishna Achanta, Francisco Estrada, Patricia Wils, and Sabine Süsstrunk. Salient region detection and segmentation. In *International conference on computer vision systems*, pages 66–75. Springer, 2008.

[55] Paul L. Rosin. A simple method for detecting salient regions. *Pattern Recognition*, 42(11):2363–2371, 2009.

[56] Christoph H Lampert, Matthew B Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[57] Jianguo Zhang, Marcin Marszałek, Svetlana Lazebnik, and Cordelia Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International journal of computer vision*, 73(2):213–238, 2007.

[58] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 27(10):1615–1630, 2005.

[59] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[60] Long Zhu, Yuanhao Chen, Alan Yuille, and William Freeman. Latent hierarchical structural learning for object detection. In *2010 IEEE computer*

society conference on computer vision and pattern recognition, pages 1062–1069. IEEE, 2010.

[61] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[62] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[63] Juho Kannala and Esa Rahtu. Bsif: Binarized statistical image features. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 1363–1366. IEEE, 2012.

[64] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *Proceedings. international conference on image processing*, volume 1, pages I–I. IEEE, 2002.

[65] Zhenhua Guo, Lei Zhang, and David Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE transactions on image processing*, 19(6):1657–1663, 2010.

[66] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[67] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010.

[68] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.

[69] Jun Yang, Yu-Gang Jiang, Alexander G Hauptmann, and Chong-Wah Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 197–206, 2007.

[70] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[71] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[72] Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu, Yann Cun, et al. Learning convolutional feature hierarchies for visual recognition. *Advances in neural information processing systems*, 23:1090–1098, 2010.

[73] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.

[74] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[75] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[76] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[77] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[78] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[79] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[80] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[81] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[82] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[83] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *2011 International conference on computer vision*, pages 2556–2563. IEEE, 2011.

[84] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

[85] Andrea Vedaldi, Varun Gulshan, Manik Varma, and Andrew Zisserman. Multiple kernels for object detection. In *2009 IEEE 12th international conference on computer vision*, pages 606–613. IEEE, 2009.

[86] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.

[87] Yinan Yu, Junge Zhang, Yongzhen Huang, Shuai Zheng, Weiqiang Ren, Chong Wang, K Huang, and T Tan. Object detection by context and boosted hog-lbp. In *ECCV workshop on PASCAL VOC*, 2010.

[88] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[89] Huaizu Jiang, Jingdong Wang, Zejian Yuan, Yang Wu, Nanning Zheng, and Shipeng Li. Salient object detection: A discriminative regional feature integration approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2083–2090, 2013.

[90] Elhadi Adam, Onisimo Mutanga, John Odindi, and Elfatih M Abdel-Rahman. Land-use/cover classification in a heterogeneous coastal landscape using rapideye imagery: evaluating the performance of random forest and support vector machines classifiers. *International Journal of Remote Sensing*, 35(10):3440–3458, 2014.

[91] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European conference on computer vision*, pages 446–461. Springer, 2014.

[92] Shivani Agarwal and Dan Roth. Learning a sparse representation for object detection. In *European conference on computer vision*, pages 113–127. Springer, 2002.

[93] Marius Muja and David G Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2227–2240, 2014.

[94] Johan AK Suykens and Joos Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[95] Hans Graf, Eric Cosatto, Leon Bottou, Igor Dourdanovic, and Vladimir Vapnik. Parallel support vector machines: The cascade svm. *Advances in neural information processing systems*, 17:521–528, 2004.

[96] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[97] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014.

[98] Mark Everingham and John Winn. The pascal visual object classes challenge 2007 (voc2007) results. In *URL http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2007/index.html.*, 2007.

[99] Mark Everingham, Luc Van Gool, CKI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge 2012 (voc2012) results. In *URL http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html*, 2011.

[100] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[101] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[102] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[103] Yann LeCun et al. Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, 20(5):14, 2015.

[104] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[105] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[106] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[107] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. Deconvolutional networks. 2010.

[108] Matthew D Zeiler, Graham W Taylor, Rob Fergus, et al. Adaptive deconvolutional networks for mid and high level feature learning. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 1, page 6, 2011.

[109] Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Detnet: Design backbone for object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 334–350, 2018.

[110] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[111] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[112] Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

[113] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[114] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[115] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[116] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[117] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015.

[118] Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1013–1020. IEEE, 2018.

[119] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[120] Jiantao Qiu, Jie Wang, Song Yao, Kaiyuan Guo, Boxun Li, Erjin Zhou, Jincheng Yu, Tianqi Tang, Ningyi Xu, Sen Song, et al. Going deeper with embedded fpga platform for convolutional neural network. In *Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 26–35, 2016.

[121] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2820–2828, 2019.

[122] Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, and Jian Cheng. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4820–4828, 2016.

[123] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[124] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[125] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[126] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

[127] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6848–6856, 2018.

[128] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

[129] Robert J Wang, Xiang Li, and Charles X Ling. Pelee: A real-time object detection system on mobile devices. *arXiv preprint arXiv:1804.06882*, 2018.

[130] Amir Gholami, Kiseok Kwon, Bichen Wu, Zizheng Tai, Xiangyu Yue, Peter Jin, Sicheng Zhao, and Kurt Keutzer. Squeezenext: Hardware-aware neural network design. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1638–1647, 2018.

[131] Chenxi Liu, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. Progressive neural architecture search. In *Proceedings of the European conference on computer vision (ECCV)*, pages 19–34, 2018.

[132] Tsung-Yu Lin and Subhransu Maji. Visualizing and understanding deep texture representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2791–2799, 2016.

[133] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[134] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[135] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017.

[136] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[137] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[138] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[139] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[140] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[141] Cheng-Yang Fu, Wei Liu, Ananth Ranga, Ambrish Tyagi, and Alexander C Berg. Dssd: Deconvolutional single shot detector. *arXiv preprint arXiv:1701.06659*, 2017.

[142] Zhen Liu. Kernelized deep convolutional neural network for describing complex images. *arXiv preprint arXiv:1509.04581*, 2015.

[143] Amir Semmo, Daniel Limberger, Jan Eric Kyprianidis, and Jürgen Döllner. Image stylization by oil paint filtering using color palettes. In *Proceedings of the workshop on Computational Aesthetics*, pages 149–158, 2015.

[144] Gustav Theodor Fechner. *Vorschule der aesthetik*, volume 1. Breitkopf & Härtel, 1876.

[145] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[146] Yongcheng Jing, Yang Liu, Yezhou Yang, Zunlei Feng, Yizhou Yu, Dacheng Tao, and Mingli Song. Stroke controllable fast style transfer with adaptive receptive fields. In *ECCV*, 2018.

[147] Bruce Gooch and Amy Gooch. *Non-photorealistic Rendering*. CRC Press, 2001.

[148] Thomas Strothotte and Stefan Schlechtweg. *Non-photorealistic Computer Graphics: Modeling, Rendering, and Animation*. Morgan Kaufmann, 2002.

[149] Paul Rosin and John Collomosse. *Image and video-based artistic stylisation*, volume 42. Springer Science & Business Media, 2012.

[150] Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346, 2001.

[151] Iddo Drori, Daniel Cohen-Or, and Hezy Yeshurun. Example-based style synthesis. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–143. IEEE, 2003.

[152] Oriel Frigo, Neus Sabater, Julie Delon, and Pierre Hellier. Split and match: Example-based adaptive patch sampling for unsupervised style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 553–561, 2016.

[153] Michael Elad and Peyman Milanfar. Style transfer via texture synthesis. *IEEE Transactions on Image Processing*, 26(5):2338–2351, 2017.

[154] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340, 2001.

[155] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[156] Li-Yi Wei, Sylvain Lefebvre, Vivek Kwatra, and Greg Turk. State of the art in example-based texture synthesis. In *Eurographics 2009, State of the Art Report, EG-STAR*, pages 93–117. Eurographics Association, 2009.

[157] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1033–1038. IEEE, 1999.

[158] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 479–488, 2000.

[159] Bela Julesz. Visual pattern discrimination. *IRE transactions on Information Theory*, 8(2):84–92, 1962.

[160] David J Heeger and James R Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238, 1995.

[161] Javier Portilla and Eero P Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International journal of computer vision*, 40(1):49–70, 2000.

[162] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. *arXiv preprint arXiv:1505.07376*, 2015.

[163] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3828–3836, 2015.

[164] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017.

[165] Vishal M Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine*, 32(3):53–69, 2015.

[166] Shaohua Li, Xinxing Xu, Liqiang Nie, and Tat-Seng Chua. Laplacian-steered neural style transfer. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1716–1724, 2017.

[167] Chuan Li and Michael Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2479–2486, 2016.

[168] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[169] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*, volume 1, page 4, 2016.

[170] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6924–6932, 2017.

[171] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[172] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *European conference on computer vision*, pages 702–716. Springer, 2016.

[173] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

[174] Dongdong Chen, Lu Yuan, Jing Liao, Nenghai Yu, and Gang Hua. Stylebank: An explicit representation for neural image style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1897–1906, 2017.

[175] Tian Qi Chen and Mark Schmidt. Fast patch-based style transfer of arbitrary style. *arXiv preprint arXiv:1612.04337*, 2016.

[176] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network. *arXiv preprint arXiv:1705.06830*, 2017.

[177] Yi-Lei Chen and Chiou-Ting Hsu. Towards deep style transfer: A content-aware perspective. In *BMVC*, 2016.

[178] Roey Mechrez, Itamar Talmi, and Lihi Zelnik-Manor. The contextual loss for image transformation with non-aligned data. In *Proceedings of the European conference on computer vision (ECCV)*, pages 768–783, 2018.

[179] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[180] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[181] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[182] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[183] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018.

[184] Danging Zhang, Binh Pham, and Yuefeng Li. Modelling traditional chinese paintings for content-based image classification and retrieval. In *10th International Multimedia Modelling Conference, 2004. Proceedings.*, pages 258–264. IEEE, 2004.

[185] Shuqiang Jiang, Qingming Huang, Qixiang Ye, and Wen Gao. An effective method to detect and categorize digitized traditional chinese paintings. *Pattern Recognition Letters*, 27(7):734–746, 2006.

[186] Meijun Sun, Dong Zhang, Zheng Wang, Jinchang Ren, and Jesse S Jin. Monte carlo convex hull model for classification of traditional chinese paintings. *Neurocomputing*, 171:788–797, 2016.

[187] Meijun Sun, Dong Zhang, Jinchang Ren, Zheng Wang, and Jesse S Jin. Brushstroke based sparse hybrid convolutional neural networks for author classification of chinese ink-wash paintings. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 626–630. IEEE, 2015.

[188] Qingyu Meng, Huanhuan Zhang, Mingquan Zhou, Shifeng Zhao, and Pengbo Zhou. The classification of traditional chinese painting based on cnn. In *International Conference on Cloud Computing and Security*, pages 232–241. Springer, 2018.

[189] Elliot J Crowley and Andrew Zisserman. The state of the art: Object retrieval in paintings using discriminative regions. 2014.

[190] Yu-Chi Lai, Bo-An Chen, Kuo-Wei Chen, Wei-Lin Si, Chih-Yuan Yao, and Eugene Zhang. Data-driven npr illustrations of natural flows in chinese painting. *IEEE transactions on visualization and computer graphics*, 23(12):2535–2549, 2016.

[191] Daoyu Lin, Yang Wang, Guangluan Xu, Jun Li, and Kun Fu. Transform a simple sketch to a chinese painting by a multiscale deep neural network. *Algorithms*, 11(1):4, 2018.

[192] Zhizhong Wang, Lei Zhao, Wei Xing, and Dongming Lu. Glstylenet: Higher quality style transfer combining global and local pyramid features. *arXiv preprint arXiv:1811.07260*, 2018.

[193] Bin He, Feng Gao, Daiqian Ma, Boxin Shi, and Ling-Yu Duan. Chipgan: A generative adversarial network for chinese ink wash painting style transfer. In *Proceedings of the 26th ACM international conference on Multimedia*, pages 1172–1180, 2018.

[194] Tzutalin. Labelimg. Free Software: MIT License, `https://github.com/tzutalin/labelImg`, 2015.

[195] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[196] Junwei Han, Dingwen Zhang, Gong Cheng, Nian Liu, and Dong Xu. Advanced deep-learning techniques for salient and category-specific object detection: A survey. *IEEE Signal Processing Magazine*, 35(1):84–100, 2018.

[197] Sidheswar Routray, Arun Kumar Ray, and Chandrabhanu Mishra. Analysis of various image feature extraction methods against noisy image: Sift, surf and hog. In *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, pages 1–5. IEEE, 2017.

[198] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[199] Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. Collaborative Filtering of Color Aesthetics. In *Proc. Computational Aesthetics (CAe)*, 2014.

[200] Donald MacLeod and Jürgen Golz. A computational analysis of colour constancy. *Colour Perception: Mind and the Physical World*, 01 2012.

[201] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18(7):1512–1523, 2009.

[202] Fahad Shahbaz Khan, Rao Muhammad Anwer, Joost Van De Weijer, Andrew D Bagdanov, Antonio M Lopez, and Michael Felsberg. Coloring action recognition in still images. *International journal of computer vision*, 105(3):205–221, 2013.

[203] Fahad Shahbaz Khan, Rao Muhammad Anwer, Joost Van De Weijer, Andrew D Bagdanov, Maria Vanrell, and Antonio M Lopez. Color attributes for object detection. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3306–3313. IEEE, 2012.

[204] Fahad Shahbaz Khan, Joost Van de Weijer, and Maria Vanrell. Modulating shape features by color attention for object recognition. *International Journal of Computer Vision*, 98(1):49–64, 2012.

[205] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost Van de Weijer. Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1090–1097, 2014.

[206] Spencer W Thomas. Efficient inverse color map computation. In *Graphics Gems II*, pages 116–125. Elsevier, 1991.

[207] Robert W Floyd. An adaptive algorithm for spatial gray-scale. In *Proc. Soc. Inf. Disp.*, volume 17, pages 75–77, 1976.

[208] Lale Akarun, Y Yardunci, and A Enis Cetin. Adaptive methods for dithering color images. *IEEE transactions on image processing*, 6(7):950–955, 1997.

[209] Dogan Ozdemir and Lale Akarun. Fuzzy algorithms for combined quantization and dithering. *IEEE Transactions on Image Processing*, 10(6):923–931, 2001.

[210] Wang Xiaofeng, Zhang Fei, Geng Guohua, et al. 3d model retrieval algorithm based on range image [j]. *Computer Engineering and Applications*, 48(7):197–200, 2012.

[211] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[212] Bernhard Scholkopf, Kah-Kay Sung, Christopher JC Burges, Federico Girosi, Partha Niyogi, Tomaso Poggio, and Vladimir Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE transactions on Signal Processing*, 45(11):2758–2765, 1997.

[213] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1449–1457, 2015.

[214] Jawad Nagi, Frederick Ducatelle, Gianni A Di Caro, Dan Cireşan, Ueli Meier, Alessandro Giusti, Farrukh Nagi, Jürgen Schmidhuber, and Luca Maria Gambardella. Max-pooling convolutional neural networks for vision-based hand gesture recognition. In *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, pages 342–347. IEEE, 2011.

[215] Pablo Arbeláez, Jordi Pont-Tuset, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 328–335, 2014.

[216] Philipp Krähenbühl and Vladlen Koltun. Geodesic object proposals. In *European conference on computer vision*, pages 725–739. Springer, 2014.

[217] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European conference on computer vision*, pages 391–405. Springer, 2014.

[218] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.

[219] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[220] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

[221] Ahmed Elgammal, Bingchen Liu, Diana Kim, Mohamed Elhoseiny, and Marian Mazzone. The shape of art history in the eyes of the machine. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[222] Xi Shen, Alexei A Efros, and Aubry Mathieu. Discovering visual patterns in art collections with spatially-consistent feature learning. *arXiv preprint arXiv:1903.02678*, 2019.

[223] Miranda Shaw. Buddhist and taoist influences on chinese landscape painting. *Journal of the History of Ideas*, 49(2):183–206, 1988.

[224] Jeff Hawkins. *A Thousand Brains: A New Theory of Intelligence*, volume 288. Basic Books, 2021.

[225] Fang Liu, Xiaoming Deng, Yu-Kun Lai, Yong-Jin Liu, Cuixia Ma, and Hongan Wang. Sketchgan: Joint sketch completion and recognition with generative adversarial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5830–5839, 2019.

Appendix A

Appendix

This chapter presents some history on recording the negative attempts on our project during the past four years.

A.1 Pilot Landscape classifier without TCP knowledge

As first attempt, we only had 100 landscape images per class per dynasty. We first employed a natural image pre-trained VGG and a binary-classifier SVM with HOG to our dataset. The results returned was disappointing. The VGG-16 falied to fully converge and was not able to provide sensible predictions. The SVM return reasonable predictions and we summarised them in table A.1.

	Predictions			
Dynasty	Tang	Yuan	Ming	Qing
Tang	67.3%	12.3%	11.1%	9.3%
Yuan	10.3%	57.3%	13.1%	19.3%
Ming	9.3%	15.7%	58.0%	17.0%
Qing	11.7%	20.3%	15.7%	52.3%

Table A.1: *Summary of predictions for four dynasties landscape painting classifier*

The performances was evaluated and validated with the form of confusion matrix, witch have been shown in detail is tables A.2, A.3, A.4, A.5, A.6 and

A.7. Further F1-values was computed and summarised in table A.8 as well.

	TRUE	
TEST	Tang	Yuan
Tang	63	37
Yuan	31	69

Table A.2: *Confusion matrix of Tang and Yuan*

	TRUE	
TEST	Tang	Ming
Tang	67	33
Ming	28	72

Table A.3: *Confusion matrix of Tang and Ming*

	TRUE	
TEST	Tang	Qing
Tang	72	28
Qing	35	65

Table A.4: *Confusion matrix of Tang and Qing*

	TRUE	
TEST	Ming	Yuan
Ming	53	47
Yuan	39	61

Table A.5: *Confusion matrix of Ming and Yuan*

	TRUE	
TEST	Ming	Qing
Ming	49	51
Qing	47	53

Table A.6: *Confusion matrix of Ming and Qing*

	TRUE	
TEST	Yuan	Qing
Yuan	42	58
Qing	61	39

Table A.7: *Confusion matrix of Yuan and Qing*

	F1-value			
Dynasty	Tang	Yuan	Ming	Qing
Tang	-	0.649	0.676	0.696
Yuan	0.669	-	0.587	0.414
Ming	0.702	0.552	-	0.500
Qing	0.674	0.396	0.519	-

Table A.8: *The F1-value computed for four dynasties landscape painting classifier*

When we first started this research project, we were facing a lack of data and unstructured dataset. We did spend a long period collecting data and annotating them manually. Even worse, the first dataset we constructed for landscape paintings did not contain all high-resolution images. Therefore, in the pilot attempt, the sliding window size was defined as 32×32.

We did not include the Song Dynasty in our dynasty classification because the TCP developed rapidly during this period. People did not manage to agree on relatively consistent aesthetics. The TCP art style of this dynasty is diverse.

A.2 Flower-and-Bird & Figure classifier for four dynasties

Another testing of the previous classification is to figure out a reliable algorithm on n distinguishing flower-and-bird and figure painting. This requires the TCP Style Learning system should have object detection ability. However, most existed object recognition aimed CNN have failed to segment objects on Chinese painting in most cases. The average accuracy is 32.7%, which is much lower than ground true.

	Accuracy	
Dynasty	Flower-and-Bird	Figure
Tang	41.2%	71.3%
Yuan	63.6%	62.7%
Ming	61.8%	64.8%
Qing	70.25%	56.9%

Table A.9: *Accuracy of TCP Flower-and-Bird & Figure classifier for four dynasties*

A.3 Pilot Object Detection on TCP

A.3.1 Methodology

For Chinese Painting object detecting, we used to use the origin VGG-16 [77] Faster R-CNN network [97] model. The detection was proceeded in two stages,

the Region Proposal Network (RPN) [26] and the Detector Network. Inspired by Crowley's research [24], the RPN was initialised with an architecture resembling VGG-VD-16. The aim of an RPN is to take an input image and predict up to 300 rectangular regions, RoIs. The set of RoIs resulting from the VGG-VD-16 based RPN was various in scales and aspect ratios. Each of these regions produced a score describing the possibility of it contains an object. Then, these proposal regions were passed to a pre-trained Fast R-CNN [134] network. This identified and regresses the bounding box of areas likely to contain classes. The VGG-16 Faster R-CNN [26] network was then applied to each painting in the test set. Images were ranked according to the score of the highest confidence detection window to finalise the ranked list. The detector networks was run on each single proposal regions to do classification. The training process could be described in below algorithm.

Algorithm 2 PilotObjectDetecting

procedure TRAININGPROCEDURE
    $M0 \leftarrow loadpretrainedcnnfeature_map$
    $M1 \leftarrow trainRPN(M0)$
    $P1 \leftarrow generateProposals(M1)$
    $M2 \leftarrow trainFastRcnn(M0, P1)$
    $M3 \leftarrow trainRpnFrozenConv(M2)$ ▷ RPN customer layer
    $P2 \leftarrow generateProposals(M3)$
    $M4 \leftarrow trainFastRcnnFrozenConv(M3, P2)$ ▷ Fast RCNN FC layer
    $P2 \leftarrow generateproposals(M3)$
    return $addRpnLayers(M4, M3.RPN)$

As mentioned in the previous chapter, we always have both natural images and Chinese paintings datasets. We trained the networks on features from these two datasets and evaluated them by dataset, respectively. We evaluated the outputs from the single-shot detector method YOLO2 on the test set of both the natural images and Chinese paintings datasets by comparing image-level classifiers. The classifier trained on paintings was expected to represent the 'best-case scenario' since there was no domain shift to the target domain. However, due to the lack of data, we failed to retrain all Conv features in CNN but only the last five layers. The pre-train Fast R-CNN, which was used to initialise the M0. M0 was a pre-trained ResNet feature map on natural images in our pilot testing model. The later experiment showed that this initialisation had effects on the outcomes. And results are summarised in the coming section.

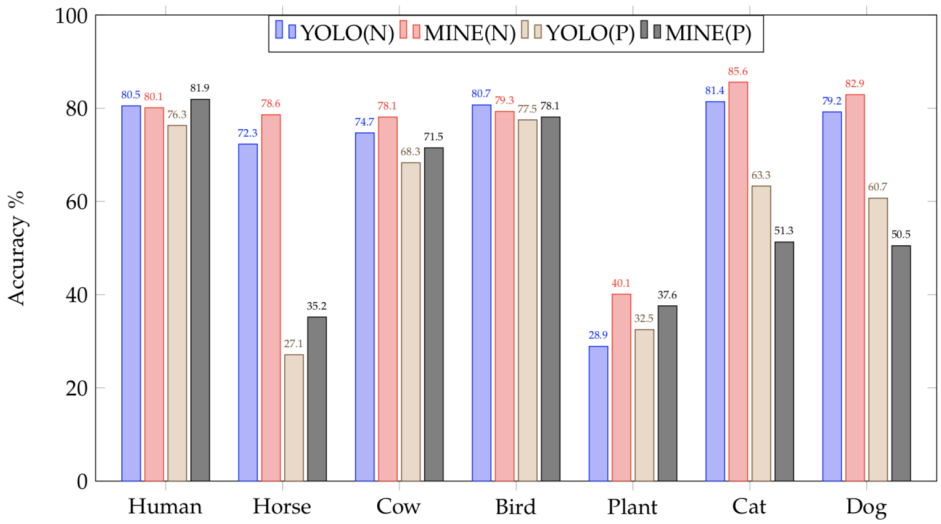

Figure A.1: *Out-of-date accuracy summary of TCP Object Detector with classes specification*

A.3.2 Experiment and discussion

This previous experiment was to evaluate the reliability and accuracy of the pilot object detection methods. We designed four scenarios: Pre-trained YOLO2 [138] algorithms on natural images evaluated on natural images, YOLO(N); Pre-trained YOLO2 algorithms on natural images evaluated on Chinese paintings, YOLO(P); Our object detector trained on natural images evaluated on natural images, MINE(N); Our object detector trained on Chinese paintings evaluated on Chinese paintings, MINE(P). After running all the above scenarios, the performance of object detectors was summarised in table A.10.

Methods	Accuracy
YOLO(N)	71.10%
MINE(N)	74.96%
YOLO(P)	57.96%
MINE(P)	58.01%

Table A.10: *Out-of-date accuracy comparison for Flower-and-Bird and Figure Classification in pilot approach*

From the results, both YOLO and our models achieved more than 70% accuracy on natural image object detection (YOLO:71.10%, MINE: 74.96%). However, the object recognition performances had dramatically drop when the testing domain was changed into Chinese Painting. According to table A.10, our pilot object detector confused with "cat" and "dog" in TCP. (Problem remains in our current model.) Also, it had big trouble on recognizing "horse". (Our A-RPN with TCP knowledge provides solution to this problem.)

A.4 Previous designs of NST

Since our computational aesthetics learning system is designed to imitate the human visual system on learning TCP, our previous system design targeted the model should have the ability to learn to unify style representation for NST. In the origin design, the aesthetic system should be able to generate two or more generic TCP style (representation) transformation. We expected our NST could produce generative TCP artworks without taking style image as input but with style class control. However, we failed. The old system design is shown in figure A.2.

Later, inspired by the ideas of [181, 183], the model has been re-designed. We expected the model to take both style images and content images as input, but to follow a GAN-based training methodology. A discriminant network is introduced to classify TCP styles (GongBi and XieYi), also the natural images. The discriminator should return a binary variable to identify the image is fake or true, and a vector that describes the style of the image, either GongBi, XieYi or natural. The generator should be able to extract the semantic mask of the content image and use style images as the training data to apply a similar approach to the content image with a conditional GAN framework [180].

The figure A.3 is not a fully correct system diagram, just keep it here as a record.

.....

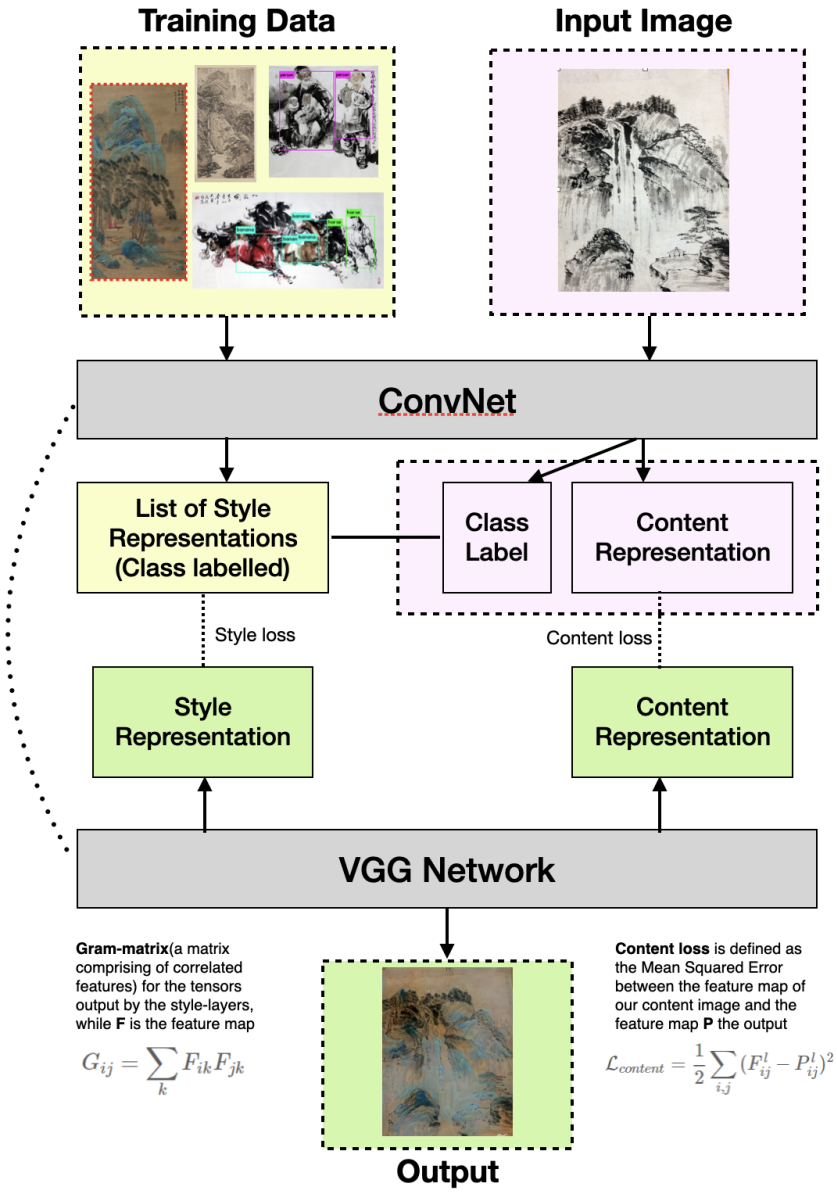

Figure A.2: *Out-of-date system diagram shows how to construct images whose feature maps at a chosen convolution layer match the corresponding feature maps of a given content image*

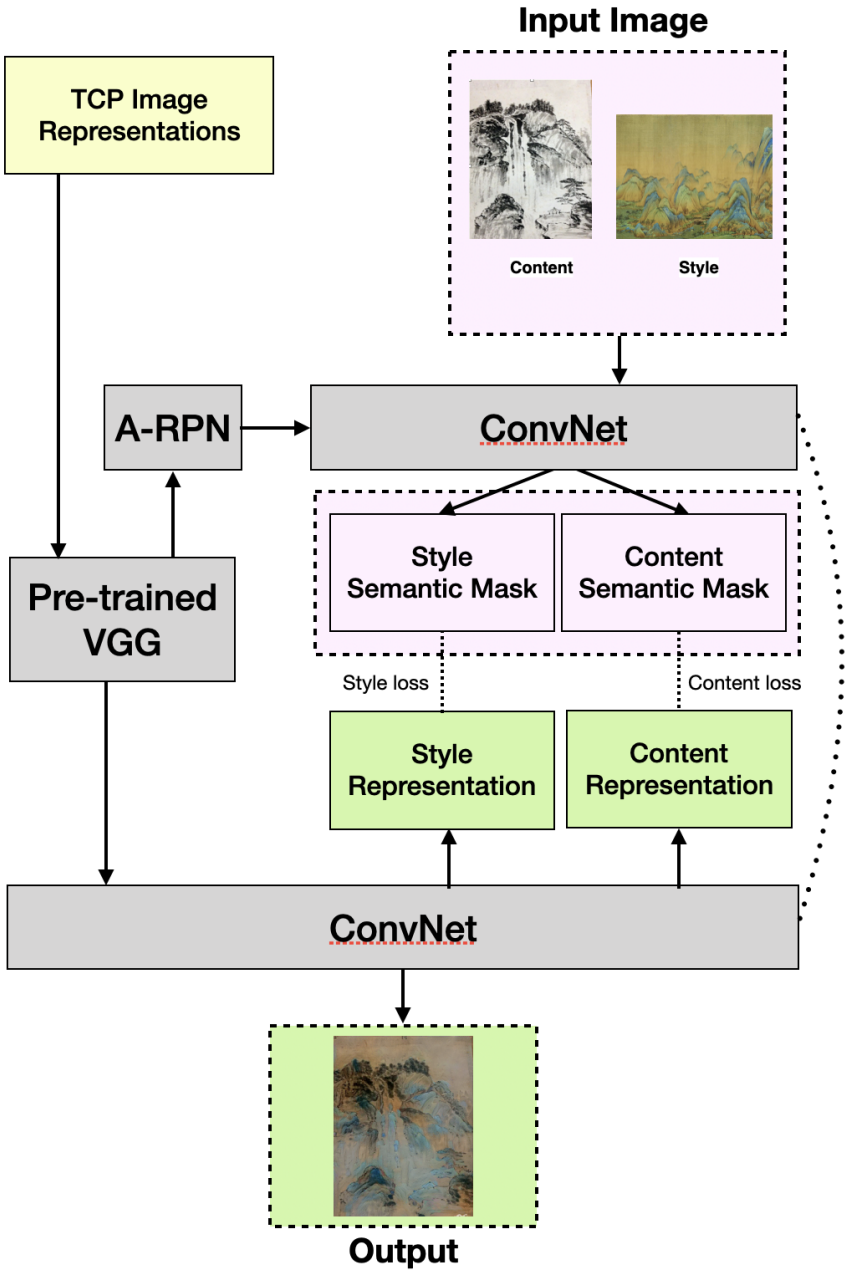

Figure A.3: *The system diagram shows how to construct images whose feature maps at a chosen convolution layer match the corresponding feature maps of a given content image with semantic information learning from a pre-trained detail segmentation network. The pre-trained VGG represents the VGG backbone network initialised with the train-from-scratch VGG in chapter 4. The A-RPN from chapter 5 holds the same initialisation and contribute to sub-parts detail learning of objects to refine segmentation masks. The pair of ConvNets is the discriminant network and generating network for a GAN model. (The dotted line that links ConvNets means the discriminant network and generating network are treated as one combined network in our design.)*