END-TO-END PREDICTION OF WELD PENETRATION IN REAL TIME BASED ON DEEP LEARNING

Wenhua Jiao
University of Kentucky, wenhua.jiao@uky.edu
Author ORCID Identifier:
https://orcid.org/0000-0003-3705-3686
Digital Object Identifier: https://doi.org/10.13023/etd.2020.142

END-TO-END PREDICTION OF WELD PENETRATION IN REAL TIME BASED ON DEEP LEARNING

DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy in the
College of Engineering
at the University of Kentucky

By
Wenhua Jiao
Lexington, Kentucky
Director: Dr. Yuming Zhang, Professor of Electrical and Computer Engineering
Lexington, Kentucky
2020

ABSTRACT OF DISSERTATION

END-TO-END PREDICTION OF WELD PENETRATION IN REAL TIME BASED ON DEEP LEARNING

Welding is an important joining technique that has been automated/robotized. In automated/robotic welding applications, however, the parameters are preset and are not adaptively adjusted to overcome unpredicted disturbances, which cause these applications to not be able to meet the standards from welding/manufacturing industry in terms of quality, efficiency, and individuality. Combining information sensing and processing with traditional welding techniques is a significant step toward revolutionizing the welding industry. In practical welding, the weld penetration as measured by the back-side bead width is a critical factor when determining the integrity of the weld produced. However, the back-side bead width is difficult to be directly monitored during manufacturing because it occurs underneath the surface of the welded workpiece. Therefore, predicting back-side bead width based on conveniently sensible information from the welding process is a fundamental issue in intelligent welding.

Traditional research methods involve an indirect process that includes defining and extracting key characteristic information from the sensed data and building a model to predict the target information from the characteristic information. Due to a lack of feature information, the cumulative error of the extracted information and the complex sensing process directly affect prediction accuracy and real-time performance. An end-to-end, data-driven prediction system is proposed to predict the weld penetration status from top-side images during welding. In this method, a passive-vision sensing system with two cameras to simultaneously monitor the top-side and back-bead information is developed. Then the weld joints are classified into three classes (i.e., under penetration, desirable penetration, and excessive penetration) according to the back-bead width. Taking the weld pool-arc images as inputs and corresponding penetration statuses as labels, an end-to-end convolutional neural network (CNN) is designed and trained so the features are automatically defined and extracted.

In order to increase accuracy and training speed, a transfer learning approach based on a residual neural network (ResNet) is developed. This ResNet-based model is pre-trained on an ImageNet dataset to process a better feature-extracting ability, and its fully connected layers are modified based on our own dataset. Our experiments show that this transfer learning approach can decrease training time and improve performance.

Furthermore, this study proposes that the present weld pool-arc image is fused with two previous images that were acquired 1/6s and 2/6s earlier. The fused single image thus reflects the dynamic welding phenomena, and prediction accuracy is significantly improved with the image-sequence data by fusing temporal information to the input layer of the CNN (early fusion). Due to the critical role of weld penetration and the negligible impact on system implementation, this method represents major progress in the field of weld-penetration monitoring and is expected to provide more significant improvements during welding using pulsed current where the process becomes highly dynamic.

KEYWORDS: End-to-End, Weld Penetration, ResNet, Transfer Learning, Dynamic Welding Phenomena, Early Fusion

Wenhua Jiao

(Name of Student)

02/15/2020

Date

END-TO-END PREDICTION OF WELD PENETRATION IN REAL TIME BASED ON DEEP LEARNING

By
Wenhua Jiao

Dr. Yuming Zhang
Director of Dissertation

Dr. Aaron Cramer
Director of Graduate Studies

02/15/2020
Date

ACKNOWLEDGMENTS

The following dissertation, while an individual work, benefited from the insights and direction of several people. First, I sincerely thank my advisor Dr. Yuming Zhang, for his invaluable guidance, encouragement, and constructive suggestions. In addition, I am grateful to Dr. Michael T. Johnson, Dr. Dan Ionel, and Dr. Wei Li for their timely and instructive support in my study as my committee members, allowing me to complete this project on schedule. Next, I would like to give my thanks to all of my colleagues in the Welding Research Lab: Chao Li, Qiyue Wang, Yongchao Cheng and Rui Yu for their help and suggestions for my work.

In addition, I want to give my thanks to my parents for their support and endless love throughout my life. Finally, I would like to express my gratitude to my wife for her encouragement, supports and love, and to also thank my son and daughter for the happiness and warmth they bring to my life.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iii

TABLE OF CONTENTS iv

LIST OF TABLES vi

LIST OF FIGURES vii

CHAPTER 1. INTRODUCTION 1

1.1 Background 1

1.2 Objectives and Approach 3

1.3 Dissertation Outline 5

CHAPTER 2. LITERATURE REVIEW 8

2.1 Welding Process Analysis 8

2.2 Welding Process Sensing 11

2.2.1 Pool Oscillation Method 11

2.2.2 Ultrasonic Method 12

2.2.3 Infrared Based Method 12

2.2.4 Arc Sensing 13

2.2.5 Vision Based Sensing 13

CHAPTER 3. SYSTEM CONFIGURATION 16

3.1 Experimental Platform 16

3.1.1 Automatic GTAW System 17

3.2 Passive Vision System 19

3.3 Observation Results 20

3.4 Image Sequence 21

CHAPTER 4. DEEP LEARNING 23

4.1 Introduction 23

4.2 Convolutional Neural Network 24

4.2.1 Basic CNN components 26

4.2.2 Convolutional Layers 26

4.2.3 Pooling Layers 31

4.2.4 Activation Function 33

4.2.5 Fully Connected Layer ... 36
4.2.6 Architecture of CNN ... 38
4.3 Residual Neural Network ... 39
4.4 Transfer Learning ... 41
CHAPTER 5. TIME INFORMTION FUSION IN CNNS ... 44
5.1 Single Frame Model ... 45
5.2 Late Fusion ... 46
5.3 Early Fusion ... 47
5.4 Slow Fusion ... 49
5.5 Summary ... 51
CHAPTER 6. DATA DESIGN ... 54
6.1 Image Sequence Design ... 56
6.2 Labels ... 60
CHAPTER 7. MODEL TRAINING ... 64
7.1 CNN Model and ResNet Model ... 64
7.2 Hyperparameter ... 67
7.3 Tricks for Training Neural Networks ... 68
7.3.1 Loss Function ... 68
7.3.2 Optimizer ... 71
7.3.3 Learning Rate ... 76
7.3.4 Batch Normalization ... 77
7.4 Summary ... 79
CHAPTER 8. RESULTS AND ANALYSIS ... 80
8.1 Comparison of Results ... 82
CHAPTER 9. CONCLUSION AND FUTURE WORK ... 87
9.1 Conclusion ... 88
9.2 Future Work ... 88
REFERENCES ... 90
VITA ... 98

LIST OF TABLES

Table 3.1 Welding parameters applied 18
Table 3.2 Camera configuration applied 20
Table 6.1 Welding parameters applied 54
Table 6.2 Labels and penetration status 55
Table 7.1 The architecture of the CNN (early fusion) 65
Table 7.2 The architecture of the 18-layer ResNet 66
Table 8.1 Confusion matrix 85

LIST OF FIGURES

Figure 1.1 The general workflow of prediction process 2
Figure 2.1 Welding process: (a) partial penetration; (b) full penetration;
(c) excessive penetration... 9
Figure 2.2 Weld arc cone and length 10
Figure 3.1 System flow chart 17
Figure 3.2 Automatic GTAW system 18
Figure 3.3 Passive vision system 19
Figure 3.4: Passive vision images. (a) top-side image (b) back-side image 20
Figure 3.5 The sequence of top-side image 21
Figure 4.1 The element of convolution process 28
Figure 4.2 Convolution process 29
Figure 4.3 The mean pooling 31
Figure 4.4 The max pooling 32
Figure 4.5 The stochastic pooling 33
Figure 4.6 Sigmoid activation function and gradient curve 34
Figure 4.7 tanh activation function and gradient curve 35
Figure 4.8 ReLU activation function and gradient curve 35
Figure 4.9 The architecture of fully connected layer 36
Figure 4.10 Canonical form of a residual neural network 40
Figure 4.11 The performance of neural network model with transfer learning 41
Figure 5.1 The architecture of single-frame model 45
Figure 5.2 The architecture of late-frame model 46
Figure 5.3 The multiple frames convolved with CNN 48
Figure 5.4 The 2D and 3D convolution operations 50
Figure 6.1 Image-sequence design 57
Figure 6.2 Three frames are merged as an RGB image 58
Figure 6.3 The process of back-side image to convert to the width of back-side bead
width 60
Figure 6.4 The calibration experiment 61
Figure 6.5 Model fitting between area and back-side bead width 62
Figure 7.1 Model 1 and Model 2 70
Figure 7.2 The effect of gradient descent with different initial points 72
Figure 7.3 The effect of gradient descent with different learning rates 73
Figure 7.4 SGD vs. GD 74
Figure 7.5 Different training processes with different learning rates 76
Figure 8.1 Training accuracy with different model and different data 80
Figure 8.2 Training loss with different model and different data 81
Figure 8.3 The single image data and image-sequence data are trained by CNN 82

Figure 8.4 The single-image data and image-sequence data are trained
by ResNet with transfer learning 83
Figure 8.5 The image-sequence data are trained by ResNet with
transfer learning and CNN 84
Figure 8.6 The prediction results from the test experiment with 65A 86
Figure 8.7 The prediction results from the test experiment with 100A 86
Figure 9.1 The all works of this study 87

CHAPTER 1. INTRODUCTION

1.1 Background

As one of the most popular welding technologies, gas tungsten arc welding (GTAW) has been playing an important role in some critical application cases. This is due to its stability and high-quality joints in practical manufacturing, such as automobiles and pressure vessels. With the development of industrial technology, more and more welding productions are done primarily by automatic GTAW equipment. However, the path, fixture, and parameters of automatic welding are all pre-set and are hardly changed in the manufacturing process. This imperfection cannot meet the increasing requirements in welding/manufacturing industry related to quality, efficiency, and individuality. Highly skilled human welders are often still preferred over automatic welding, because automatic equipment cannot analyze and control the weld penetration status. The back-bead width is a major parameter that can be used to analyze and control weld penetration [1]. Its real-time monitoring requires a complex setup that is not suitable for welding operations. Therefore, it has become urgent for automatic GTAW equipment to have intelligent analysis capability.

Skilled welders can control the weld penetration by observing the welding process occurring on the top side of the work-pieces to adjust parameters (e.g., current, welding speed, arc length, and torch orientation). The mechanism of welder’s behaviors was utilized to develop a general method to predict weld penetration, many studies have been conducted to predict weld penetration using different characteristic information from the welding process. They typically (1) sense observable phenomena from the welding process using different sensors or phenomena; (2) define and extract characteristic information from sensed phenomena; and (3) build a model to correlate the extracted characteristic information to the penetration state [2,3].

The general method as shown in Figure 1.1 meets with some inescapable problems: the characteristic information is defined by humans and therefore greatly depends on an individual's opinions, views, and abilities, which means some key data that determining the target information may be missed. In addition, the extracting process will add

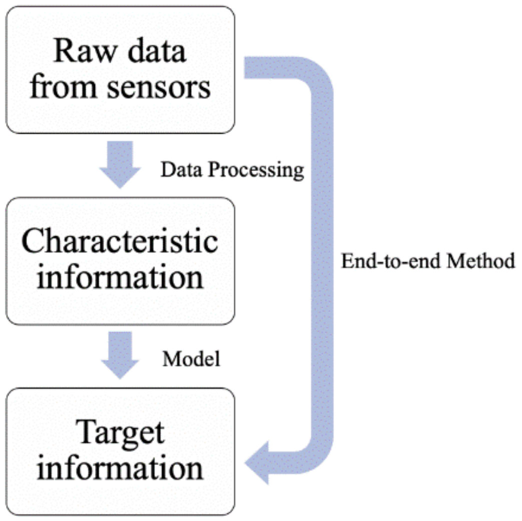


Figure 1.1: The general workflow of prediction process

complexity and accumulated errors to the whole system. Furthermore, the welding process is typically nonlinear and complex, so a linear numerical model may not possess sufficient accuracy to describe the nonlinear behavior of the welding process; most numerical models are designed for certain welding applications and are not a universal solution [4, 5].

Deep learning has developed quickly in recent years, and the weld penetration prediction by deep learning models has become an important issue in the welding field. The artificial neural network (ANN) is one of the most widely intelligent algorithms that can be used to predict the bead geometry and penetration problems [2], and the back propagation artificial neural network (BPNN) is applied to predict penetration depth, back-bead width, and bead geometry size [3, 6]. However, all these methods have the issue, namely that prediction accuracy is not high; this is often due to missing characteristic information. Complex setups, extensive data pre-processing, and results with unsatisfactory accuracy are often the causes. In order to address this general challenge, this dissertation applies a deep-learning based method to automatically extract the information. The major remaining challenge to acquire adequate information from the welding process is therefore reduced.

Skilled welders can judge weld penetration by observing welding phenomena during the welding process. Thus, the welding community believes that images from observable welding scenes contain sufficient information to predict weld penetration. While earlier efforts were made to follow the aforementioned procedure to initially propose characteristics, the deep learning method was recently applied; with a focus on using a convolution neural networks (CNN) [7, 8, 9, 10] to directly map images of the penetration [11, 12]. Training for the parameters, which includes the convolutional kernels and weights in fully connected layers and feature extraction and reasoning, is automatically done. However, these images of observable weld scenes and weld-penetration states are synchronously collected for use in training CNN models. They all assume that the current weld penetration can fully determine the final weld penetration, but they miss the temporal dynamic information and incompletely interpret skilled-welder operation.

1.2 Objectives and Approach

As can be seen from a previous analysis, the general method has some limitations. Thus, the goal of this dissertation is to establish an efficient-yet-relatively-simple end-to-end prediction system to learn weld-penetration status in complex welding processes by deep-learning methods. In this system, the characteristic information from the welding process is defined and extracted by a data-driven process, and the deep-learning model can directly predict the weld-penetration status from knowledge learned from the characteristic information. Therefore, the objectives of this dissertation are:

- to understand the skilled-welder response to the welding process and simulate skilled-welder intelligence to learn knowledge from the welding process by the deep-learning method
- to establish an effective sensing and welding system that can simultaneously know the back-bead width by automatically observe the welding process (i.e., a welder's view)
- to design a dataset that reflects dynamic welding phenomena to effectively simulate the welding process
- to build an effective and fast deep-learning model that will accurately predict weld penetration status in real time

Technical challenges include sensing the welding process from the high-light welding arc, designing a training dataset that includes temporal information (i.e., dynamic welding phenomena), and training the deep-learning model to accurately predict the weld penetration status from complex welding process information in real time. To address these, we propose the following approaches: (a) a passive vison system with two installed cameras to simultaneously capture the top-side and back-side images; (b) fusing the present weld pool-arc image with two previous images acquired 1/6s and 2/6s earlier (the fused single image will reflect the dynamic welding phenomena); and (c) the CNN model is trained with the fused weld pool-arc images as input data and the penetration status as labels. To improve the prediction model, we also introduce a residual neural network (ResNet) and a transfer-learning method. Welding experiments were conducted in a variety of welding conditions to synchronously collect the necessary data-pairs to train the deep-learning model. Results show that this method significantly improved prediction accuracy, and the trained model was verified for robustness and versatility.

1.3 Dissertation Outline

In this dissertation, an intelligent welding system that automatically collects data and an end-to-end prediction system that effectively predicts weld penetration status in real time and controls the welding process based on the prediction results. The dissertation is organized as follows.

Chapter 1: Introduction

The background and motivation of this research is discussed, and the research objectives are proposed.

Chapter 2: Literature Review

Analysis of the welding process is surveyed, and the physical evolution of this process and weld-penetration sensing are discussed; these include pool oscillation, infrared, ultrasonic, arc sensing, and vision-based sensing methods.

Chapter 3: System Configuration

An automatic GTAW welding platform is built, and a passive vision system is designed in the platform that includes two cameras that will simultaneously capture the top-side and back-side of the welding process. The platform is driven and controlled by a computer with a stepper motor. Due to the special characters of the top-side images, we introduce the image-sequence analysis which will also be discussed in this chapter.

Chapter 4: Deep Learning

In this chapter, deep-learning methods are discussed to explain how they use convolution pixel information for purposes of prediction, then the role and advantage of a ResNet are derived by the vanishing-gradients problem of a CNN. Furthermore, we propose a transfer-learning method to improve the training speed and performance of neural networks by pre-training the ImageNet dataset.

Chapter 5: Time Information Fusion

We will discuss methods to fuse temporal information in the deep-learning models. Compared with the different fusion methods, the early fusion method best meets our design requirements and is also suitable for the characteristics of the input data.

Chapter 6: Data Design

Chapter 6 offers a detailed description of the data design for an input dataset and labels. In this case, the top-side image is the static-welding-process information at a certain moment, but it misses the dynamic and temporal information that is needed to analyze dynamic weld phenomena. So, we design the image-sequence data as an input dataset, and the weld penetration status is defined by a range of back-bead width values that is calculated according to the back-side image.

Chapter 7: Model Training

Different parameters and function settings are discussed in this chapter, and the optimal model is established. The CNN- and ResNet-model-based transfer-learning trains the single-frame images and image-sequence data to show the different results from the different models and different datasets.

Chapter 8: Results and Analysis

In Chapter 8, we compare the different models with different training datasets to analyze the advantage of image-sequence and ResNet-model-based transfer learning.

Chapter 9: Conclusion and Future Work

The main findings and contributions are summarized, and the future work to improve the proposed research is discussed.

CHAPTER 2. LITERATURE REVIEW

Welding is one of the most important joining techniques, and it has been automated/robotized. In practical welding, the weld penetration as measured by the back-side bead width is a critical factor in determining the integrity of the weld. However, the bask-side bead width is difficult to directly monitor during manufacturing, as it occurs beneath the surface of a welded piece. Therefore, predicting back-side bead width from conveniently sensible information from the welding process has become a fundamental issue in intelligent welding. Skilled welders can control weld penetration by observing the welding process and adjusting parameters (e.g., current, welding speed, arc length, and torch orientation) as needed. The mechanisms of a welder's behavior were utilized to develop a general method to predict weld penetration; thus, various methods were studied to monitor the welding process. The dynamic welding process is analyzed in the following subsection, then the sensing methods are reviewed.

2.1 Welding Process Analysis

The development of weld penetration is a complex process, but the physical evolution is clear. When welding starts, the solid metal melts and the liquid metal expands due to thermal expansion; this causes the surface to start to rise, as shown in Figure 2.1(a). Under continuous application of heat, the arc is formed and the volume of the liquid weld pool increases. As a result, the surface area of the weld pool increases and the penetration depth deepens until the liquid weld pool completely penetrates the solid metal.

The width of the back-side weld bead (W_d) will increase after the solid metal is fully melted, as shown in Figure 2.1(b), and the W_d is the key information to determine the penetration state. The weld quality is considered qualified when W_d reaches a certain width in actual production. If the liquid weld pool continuously increases, the back-side weld pool surface will become more convex due to gravity.

When the convex volume is greater than the thermal expansion volume, the top-

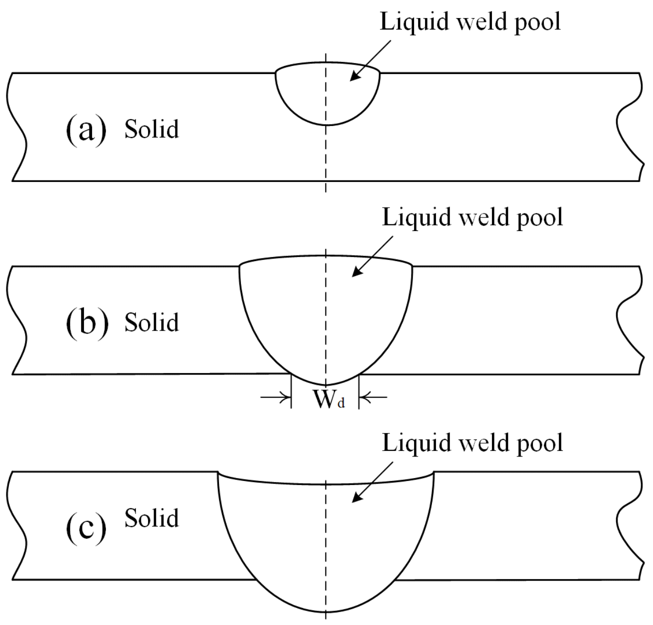


Figure 2.1: Welding process (a) partial penetration; (b) full penetration; (c) excessive penetration

side surface will become concave, as shown in Figure 2.1(c). The analysis discussion illustrates that the top-side and back-side weld pool must follow the physical relationship, and using the observable weld scene from the top-side to predict back-side bead width is justifiable. In addition, the dynamic evolution of the top-side weld scene is even more informative and can better correlate to what occurs on the back-side.

We note that welders not only see the surface of the weld pool, but also the welding arc. Until recently, the arc was regarded as strong light interference that affected effective image processing used to extract more critical information, such as the weld pool boundary. Part of the efforts were to filter out the arc. Possible information from the arc that was relevant to the development of the weld pool was ignored. In fact, because heat is directly applied by the arc, the arc cone directly affects energy-density distribution and the heated area that is related to the direction and speed of the weld-pool extension.

As shown in Figure 2.2, as the arc voltage increases, the arc length becomes longer, which leads to a wider arc cone and a broader arc-heating area. Conversely, when the arc voltage decreases, the arc length is shorter, the arc cone becomes narrower, and the heating area is more focused [13]. The surface of the weld pool will constantly change with the development of the weld pool, and the deformed surface also affects the arc cone. Hence, we collect the top-side image which includes both the arc and the weld pool as the raw information of the relevant weld phenomena.

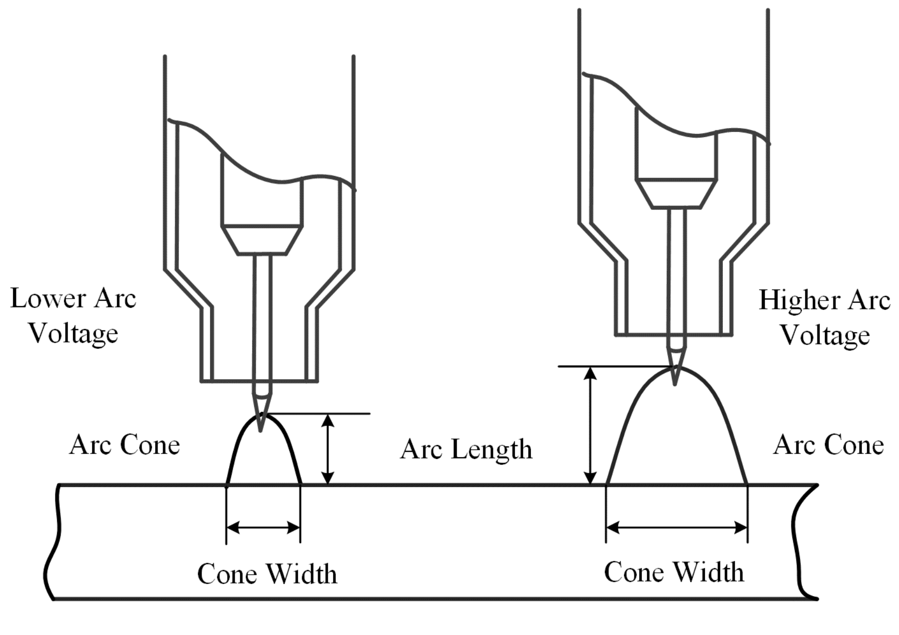


Figure 2.2: Weld arc cone and length

While complex raw data increases redundant information, it also increases the difficulty of extracting characteristic information and computational complexity. However, the deep-learning-based data-driven approach is expected to be capable of effectively analyzing top-side images despite the increased complexity.

2.2 Welding Process Sensing

Sensing the welding process should provide information associated with the weld penetration. Extensive work has been done to sense the welding process, including pool oscillation [14–23], ultrasonic sensing [24–29], infrared sensing [30–32], arc sensing [33–35], and vision sensing methods [36–51]. The review regarding these sensing methods is presented in the following.

2.2.1 Pool Oscillation Method

Sensing weld penetration by weld pool oscillation behavior, which is based on the oscillation frequency of the weld pool, is related to weld-pool geometry. In 1972, Kotecki et al. [14] discovered weld pool oscillation behavior and the relationship between the weld pool diameter and the natural frequency of the weld pool oscillation in the GTAW process. Richardson and his colleagues [15] then proposed a method to predict the back-bead width based on the natural frequency of weld pool. However, these results had poor accuracy, and it was unclear how the method could be applied in the moving welding process.

The abrupt change in the oscillation frequency of the weld pool during the transition from partial to full penetration was found and applied to monitor and control the weld penetration by Xiao and Ouden [17, 18]. Anderson [19] developed a synchronous weld pool oscillation method to control and monitor weld penetration status. Yudodibroto [22] further implemented the weld pool oscillation method to control weld penetration during the GTAW process with the addition of cold filler wire. However, the pool oscillation method had some issues that resulted in low prediction accuracy due to the moving speed of the arc torch; additionally, the work-piece surface, surface dirt, and oxidation of the work piece changed the natural frequency of the weld pool oscillation. Furthermore, the sensing device and data processing needed to control the weld penetration by sensing the pool oscillation method is complex. Therefore, the pool oscillation method is designed for specific welding applications and is not suitable for general welding applications.

2.2.2 Ultrasonic Method

Ultrasonic methods are used to find the boundaries of the weld pool in the work piece [24–26]. Developed ultrasonic sensing methods can locate and track the welding seam to ensure the arc torch is in the correct position during the welding process. Recently, various non-contact ultrasonic sensing methods have been developed in which weld penetration depth is determined by calculating the receiving time of the reflected ultrasonic wave [23, 28]. However, the ultrasonic wave transmission speed is different in different materials. Therefore, ultrasonic methods only can be used in uniform material that contain a low percentage of impurities, and the surface of work piece must be clean. In addition, ultrasonic methods need special calibration and the equipment is expensive, which make this an imperfect sensing method for welding applications.

2.2.3 Infrared Based Method

Infrared-based methods use infrared sensors to measure thermal distribution in the welding process by monitoring the weld parameters, including weld bead width, penetration depth, and torch position [30–32]. However, infrared sensors are very expensive, and sensing accuracy can be affected by the environment, such as sun, light, or other lighting conditions. This limits applicability of the infrared-based method; thus, it is not suitable for industrial welding applications.

2.2.4 Arc Sensing

Arc sensing is widely used to track welding seams and control arc stability by monitoring arc behavior in the welding process [33, 34]. The advantage of arc sensing is that the system uses the arc itself as a sensor, instead of the external sensors. However, less-useful information and easy interference make this sensing method less effective in controlling weld penetration [35].

2.2.5 Vision Based Sensing

The top-side of the weld pool provides many important information for understanding the welding process and is the source of information that welders can obtain in the welding process. As discussed above, the top-side of the weld pool includes the arc cone and the weld pool surface (which will constantly change with development of the weld penetration), and they are closely related to weld penetration. Vision-based sensing is an observation method to collect information from the top-side of the weld pool; this is the same information a welder can obtain, and it is widely used in industrial welding due to its cheap cost, easy setup, reliability, and high accuracy.

Pioneering work on vision-based sensing to observe the weld pool was proposed by Rokhlin and Guu [36, 37]. They used radiography to measure depth penetration according to the understanding that the radiation increases with the depression depth. The principle of this method is to measure the material’s thickness. However, it is difficult to extract the shape of the weld pool surface for full penetration, which causes pool surface deformation to occur.

A stereovision system was proposed by Minch [52] to measure the weld pool by using the high frame rate. Two cameras synchronously captured images when the arc was off, and the 3D weld pool surface was reconstructed by stereo image process algorithms. Zhao [44] developed a method to reconstruct the weld pool surface shape from a shading algorithm with a single weld pool image. Researchers at the University of Kentucky Welding Research Laboratory have done pioneering work in weld-pool monitoring and welding penetration state control based on structured laser lights. By taking advantage of the mirror-like reflective property of the weld pool, they captured images of a structured laser (including multi-stripes [38], dot matrix [40, 42, 43], and grid matrix [53]) reflected by the weld pool and designed corresponding image processing algorithms to reconstruct the 3D surface of the weld pool.

With the general methods to predict weld penetration status, the sensing information always eliminates the influence of the bright arc light, because no method can analyze the arc information with the images, and the arc light blocked out the sight of the weld pool, which was regarded as interference. Now the deep-learning-based data-driven approach can analyze and solve this issue, so the images that include more information are better.

The passive vision system was created with the process and development of deep-learning in the field of welding. Chen's [54] research shows that the arc and weld pool surface are both important features that affect the weld penetration, and Jiao [11] proposed an end-to-end prediction system by processing top-side images from the passive vision system; the latter method has an easy setup and is not concerned about extracting and defining characteristic information from the sensing data, and the raw sensing data, including the weld pool surface and arc cone, can be directly used as input data. This method is highly suitable for industrial applications, so we will continue to use this method and optimize it in this dissertation.

CHAPTER 3. SYSTEM CONFIGURATION

Skilled welders can control and estimate weld penetration by observing the top-side of the welding process; this mechanism proves that top-side images contain enough characteristic information to estimate the weld penetration status. In this chapter, a passive vision system is used to observe the welding process in real time, similar to the welder's perspective on an automatic GTAW platform. The top-side images will be input data, and the back-side images will be used to estimate the back-bead width as label data, utilizing a relationship function between the pixels of the back-side images and the actual back-bead width. The deep-learning model can then directly output the prediction results by training the raw top-side data.

3.1 Experimental Platform

The experimental platform, which includes an automatic GTAW system and the passive vision system, is a simple but fully automated welding platform, meaning all welding operations and information collections are controlled by computer programs. The system flow chart is shown in Figure 3.1: The computer controls the two cameras and collects image information with a PCI expansion card (IEEE 1394), and a data acquisition card (PCI 6229) is used to order the welding power supply and retrieve feedback information from the welding power supply. The stepper motor is controlled by the motor controller with the order of the computer by the data acquisition card. All the collected information is transmitted to the computer and saved in the specified path, and the computer programs are edited with Python software.

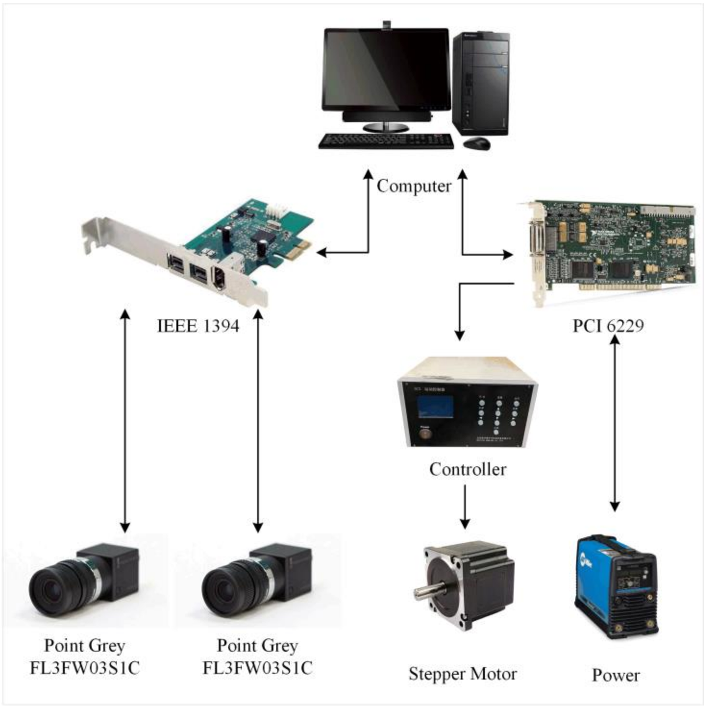


Figure 3.1: System flow chart

3.1.1 Automatic GTAW System

As shown in Figure 3.2, the automatic GTAW system was designed on the automatic platform, and the weld torch was set perpendicular to the automatic platform. The center of the platform is a rectangular hole, which is convenient for adding shielding gas and observing weld penetration states via camera during the welding process. For the consistency of the experiment, a stepper motor drives the platform to achieve and control the position and movement of the work piece. In the automatic GTAW system, two shielding gas pipelines are used to prevent the front and back of the work piece from becoming oxidized during the welding process, which also leads to better welding effect. The power supply is Miller PM200 DC, which is able to output direct current up to 200 amperes. In experiments, each workpiece/sheet is welded in 12 spots that are spaced 2cm apart. For the more complete samples under different conditions, the ranges of welding currents and welding times are relatively large. Cameras capture the images during the entire welding process from partial penetration to excessive penetration under different welding conditions. The preset parameters are shown in Table 3.1.

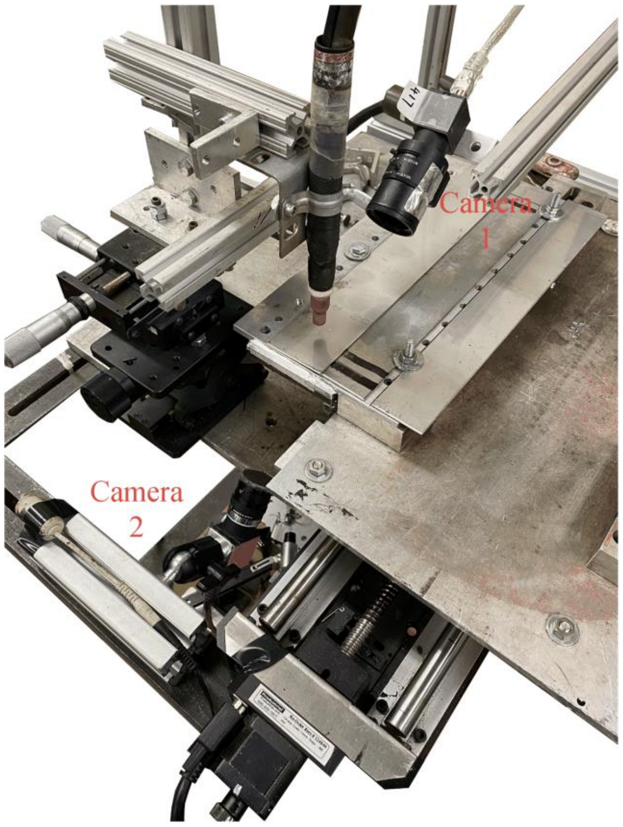


Figure 3.2: Automatic GTAW system

Table 3.1: Welding parameters applied

Welding Parameters	Value
Welding Type	GTAW
Welding Current (A)	60-110
Welding Time (S)	4-12
Tungsten Diameter (mm)	2.4
Shielded Gas	Argon
Gas Flow (L/Min)	7
Workpiece Material	304L

3.2 Passive Vision System

In the automatic platform, the welding torch and two high-speed cameras are set to remain stationary, and the work piece is moved by the step motor, which is shown in Figure 3.3. This design ensures that the cameras can always capture high-resolution paired images from top-side and back-side of the work piece. In order to eliminate the strong interference from the arc radiation, Camera 1 (Point Grey FL3FW03S1C) uses a 685nm center-wavelength band-pass optical filter. The same filter is used by Camera 2 for the same reason: the filter will eliminate interference light from the strong arc and keep the arc cone.

The configurations for both cameras are shown in Table 3.2. The passive vision system can capture the full top-side images of the welding process to ensure that the collected information has sufficient valid information. Furthermore, the passive vision system is very easy to setup to reduce the negative impact of the external system in the

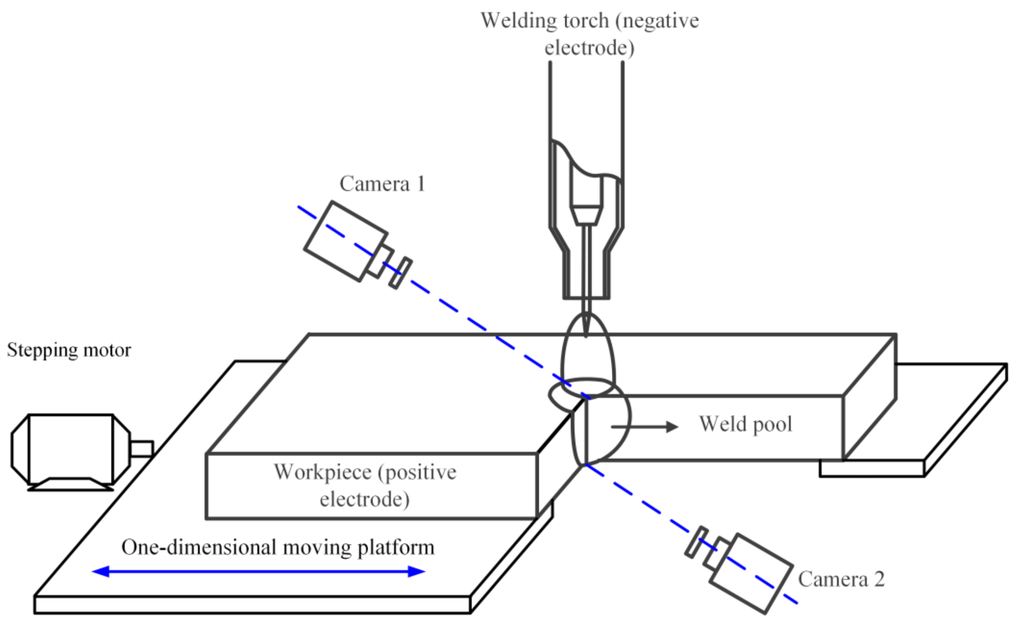

Welding torch (negative electrode)
Camera 1
Stepping motor
Weld pool
Workpiece (positive electrode)
One-dimensional moving platform
Camera 2

Figure 3.3: Passive vision system

welding process and design difficulties. The passive vison system is different from the active vision system with the structured laser, which is easily disturbed by the environment, and the captured images includes more information than those of active vision system.

Table 3.2: Camera configuration applied

Configuration	Value
Filter Center (nm)	685±2
Filter FWHM (nm)	10±2
Image Size (Pixel)	480×640
Format	Mono8
Frame Rates (FPS)	30
Shutter Time (ms)	0.08
Sharpness	3000
Gain	0
Gamma	2.5

3.3 Observation Results

In experiments, the automatic platform drives a 1.85mm-thick stainless steel sheet

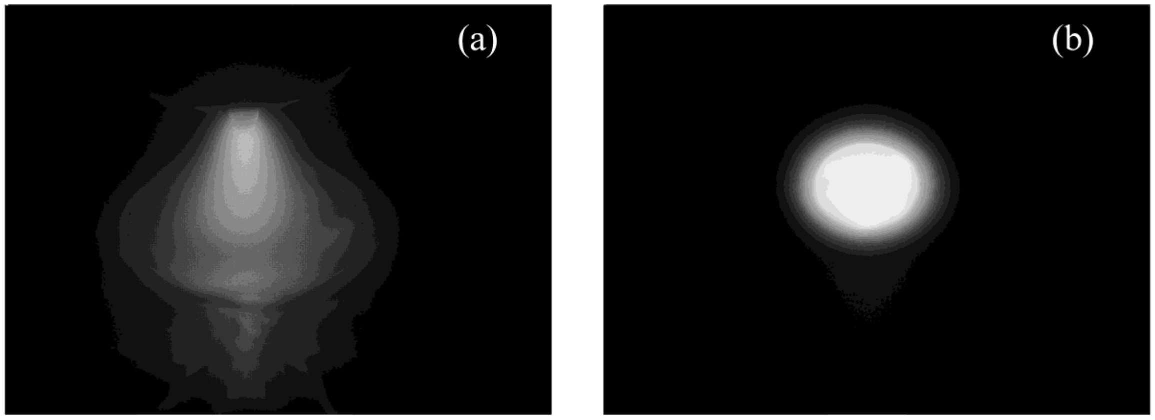


Figure 3.4: Passive vision images. (a) top-side image (b) back-side image

to move into position and be spot-welded using direct current GTAW. As shown in Figure 3.4, both-sides images of the welding pool are obtained through the passive vision system. In Figure 3.4(a), the top-side images are obtained at 0.08ms camera exposure time. Most of arc light was suppressed, and the arc shape and weld pool side edge can be clearly observed. In Figure 3.4(b), the back-side images are obtained synchronously with the top-side images, and the back-side geometry is captured as desired. The designed passive vision system performs well in the automatic GTAW system, and the observation results meet the design requirements. Over 120 welding experiments were automatically conducted with these parameters, and 28,494 images pairs were collected as a raw dataset. All these image pairs were divided into training, validating, and testing datasets with the size of 22,794, 2,849, and 2,851, respectively.

3.4 Image Sequence

An image sequence contains more information than any of its constituent images. For this reason, image-sequence analysis has been used in computer vision for a long time. Analyzing image sequences is very effective for object detection, especially for relatively moving objects [55]. Additionally, camera noise can corrupt individual images, but such noise can be suppressed using an image sequence to reliably detect low-contrast objects

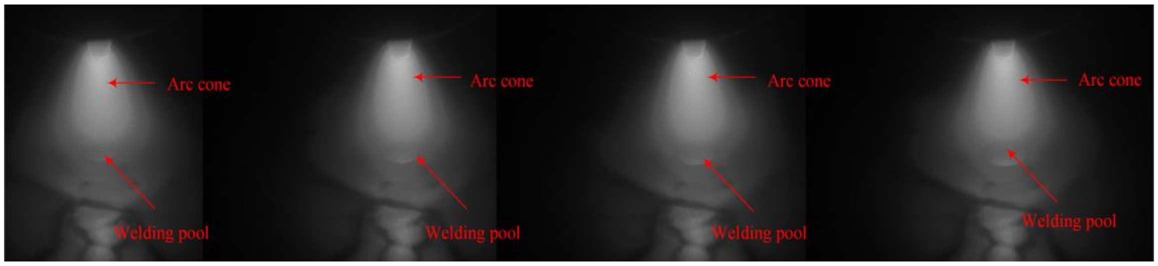


Figure 3.5: The sequence of top-side image

[56]. For our particular application, the analysis of the weld pool dynamic evolution also illustrates the necessity of using dynamic weld phenomena.

In our case, the top-side images have the following characteristics: dark chroma, low contrast, and the development of the weld pool and arc is a slow process. Thus, the top-side images in the same process have a high degree of similarity. Furthermore, characteristic information in the welding process is dynamic while the surrounding environment is static, as can be seen from the sequence of top-side images in Figure 3.5. The development of the weld pool and arc cone can be observed with this sequence, but the harsh imaging conditions make extracting image features a challenge. Directly processing the image by a CNN without first extracting image features is therefore a correct choice. In this paper, we will design an image-sequence dataset to increase the temporal information in the raw information, which enables CNNs to extract dynamic feature information. In addition to widen the image information, we also deepened the model to better use the high-level information to improve the prediction accuracy.

The main connectivity pattern category for fusing temporal information through a CNN includes single frame, early fusion, late fusion, and slow fusion [17]. These contents involve the knowledge of deep learning, which will be explained in detail in Chapter 5. And the deeper CNN-based model will be discussed in Chapter 4.

CHAPTER 4. DEEP LEARNING

4.1 Introduction

Deep-learning algorithms are a specialized subset of Artificial Intelligence (AI). Deep learning is inspired by information processing and distributed communication nodes in biological systems, which can transmit the intelligence to a computer by using multiple layers to progressively extract higher-level features from the input data and learn the potential relationship to output target information without being explicitly programmed. Deep-learning architectures such as deep neural networks, recurrent neural networks, deep belief networks, and CNNs have been applied in many fields, including computer vision, natural language processing, machine translation, medical-image analysis, and so on.

CNNs are the best among the learning algorithms for image content understanding and have achieved good results in related tasks such as segmentation, classification, detection, and retrieval [55, 56]. The success of CNNs has drawn attention from all walks of life. In the industry, large companies such as Google, Microsoft, and Facebook have established active research groups to explore CNN new architecture [59]. Currently, most front-runners in image processing and computer vision competitions use deep CNN-based models.

In 2011, CNN was applied on graphics processing units (GPUs) to win an image recognition contest that broke through the deep-learning performance [60]. The success of the CNN model promoted the development of image analysis in various fields and has introduced a new possibility for solving weld-penetration problems. The architecture of a CNN is similar to the connection pattern of the visual cortex in the human brain [61], so it can quickly and efficiently learn the features from the images without much preprocessing. Hence, we propose an end-to-end method to directly predict the target information from the raw data without the need for hand-engineering to define and extract characteristic information. In such a process, the characteristic information is automatically defined and extracted by model, and the model can achieve high prediction accuracy due to the reduced likelihood of missing characteristic information. The powerful performance of a CNN offers a possible way to build such an end-to-end method.

4.2 Convolutional Neural Network

The advantage of a CNN is that it can exploit spatial or temporal correlations of data. The structure of the CNN is divided into multiple learning stages, consisting of convolutional layers, nonlinear processing units, and sub-sampling layers [62]. CNNs are feed-forward multilayer hierarchical networks that are similar to fully connected neural networks; each layer uses a set of convolution kernels to perform multiple transformations [63]. Convolution operations extract useful features from locally correlated data. Assigning the output of the convolution kernel to a nonlinear processing unit (activation function) not only helps to learn the abstraction, but also embeds the nonlinearity in the feature space. This nonlinearity generates different activation modes for different reactions, thereby promoting the learning of image semantic differences. The output of the nonlinear activation function is usually performed after subsampling, which helps to summarize the results and also causes the input to be unaffected by geometric distortions [64]. A CNN has automatic feature-extraction capabilities, which reduce the need to synthesize separate feature extractors [65]. Therefore, CNNs with small processors can learn good internal representations from raw pixels. The important attributes of a CNN are hierarchical learning, automatic feature extraction, multi-tasking, and weight-sharing [64–66].

CNNs first attracted people's attention in 1989 through Yann LeCun's processing of grid-like topological data (images and time-series data) [69]. CNNs' architectural design is inspired by the work of Hubel and Wiesel and largely follows the basic structure of the visual cortex in primates [70], [71]. The popularity of CNNs is largely due to its hierarchical feature-extraction capabilities. The hierarchical organization of CNN simulates the deep hierarchical learning process of the neocortex of the human brain and dynamically learns features from raw data [72]. Because a CNN has a multi-layered structure, it can extract low-, mid-, and high-level features. High-level features (more abstract features) are a combination of low- and mid-level features.

From the late 1990s to 2000, CNNs made several improvements in learning strategies and architectures, which makes a CNN scalable to large, heterogeneous, complex, and multi-class problems. Innovations of the CNN include modifications of basic components, parameter and hyperparameter optimization strategies, regularization units, design patterns, connectivity between layers, and so forth. CNN-based applications became popular after AlexNet performed well on the ImageNet dataset [9]. CNNs have proposed major innovations since 2012, primarily in the reorganization of processing units and the design of new blocks. In addition, Zeiler and Fergus proposed the concept of layer-wise visualization of a CNN [73], which improved the feature-extraction stages and changed the trend of extracting low spatial resolution features in deep architectures such as a visual geometry group (VGG) [74]. Currently, most new architectures are built on the simple and isomorphic topology principles introduced by VGGs. On the other hand, the

Google Group introduced the idea of splitting, transforming, and merging; the corresponding block is known as an inception block. An inception block gives the concept of branching in a layer for the first time, which allows abstracting features on different spatial scales [75]. In 2015, the concept of skip connection for deep CNN training introduced by ResNet was widely known [76]. Subsequently, the concept was used by most subsequent networks, such as Inception-ResNet, WideResNet, ResNeXt, etc. [75–77].

4.2.1 Basic CNN components

In the end-to end prediction system, the input is the top-side image that includes both the weld pool and the arc, and a CNN is used as the prediction model. A CNN is a kind of neural network that builds higher levels of functionality from groups of pixels commonly in images. The images are then weighted using scoring on these features to generate the final classification results [69]. Usually, a CNN consists of a series of convolution layers, pools layers, and fully connected layers. In some cases, the full-connection layer is replaced by the global-average pooling layer. In addition to different mapping functions, different adjustment units such as batch normalization and dropout are added to optimize the performance of the CNN [80]. The arrangement of CNN components plays a critical role in the design of the new architecture to improve performance. This section briefly discusses the role of these components in the CNN architecture.

4.2.2 Convolutional Layers

The role of the convolution layer is to extract feature values, which can be understood as using a filter (convolution kernel) to filter each small area of the image to obtain the feature values of these small areas. In specific applications, there are often multiple convolution kernels extracting the features of the same image at the same time to achieve the extraction of multiple features. Here, the convolution kernels can also be called filters. Different filters will get different output data; this is the equivalent to using different filters to extract the specific information you want about the image (i.e., color, depth, or contour). The neurons in each layer of the CNN are arranged in three dimensions; they are arranged in a rectangular parallelepiped, with width, height, and depth. For RGB images, this convolution layer contains three filters, namely three sets of parameters. Each filter can convolve the raw input image to obtain a feature map, and three filters can obtain three feature maps. As for how many filters a convolutional layer can have, this can be freely set. In other words, the number of filters in the convolutional layer is also a hyperparameter. We can think of the feature map as image features extracted through convolution transformation. The three filters extract three different sets of features from from the raw image, also called three channels (channel).

Here is a simple example to explain how to calculate the convolution and explain some important concepts and calculation methods of the convolutional layer. Suppose there is a 5*5 image, and a 3*3 filter is used to convolve to get a 3*3 feature map, as shown in Figure 4.1. First, numbering each pixel of the image, using $x_{i,j}$ to represent the i-th row and j-th column of the image; each weight of the filter is numbered by $w_{m,n}$, which is the m-th row and n-th column weight, and w_b represents the bias of the filter. When numbering each element of the feature map, use $a_{i,j}$ to represent the i-th row and j-th column of the feature map; use $\mathcal{F}$ to represent the activation function; in this example, the

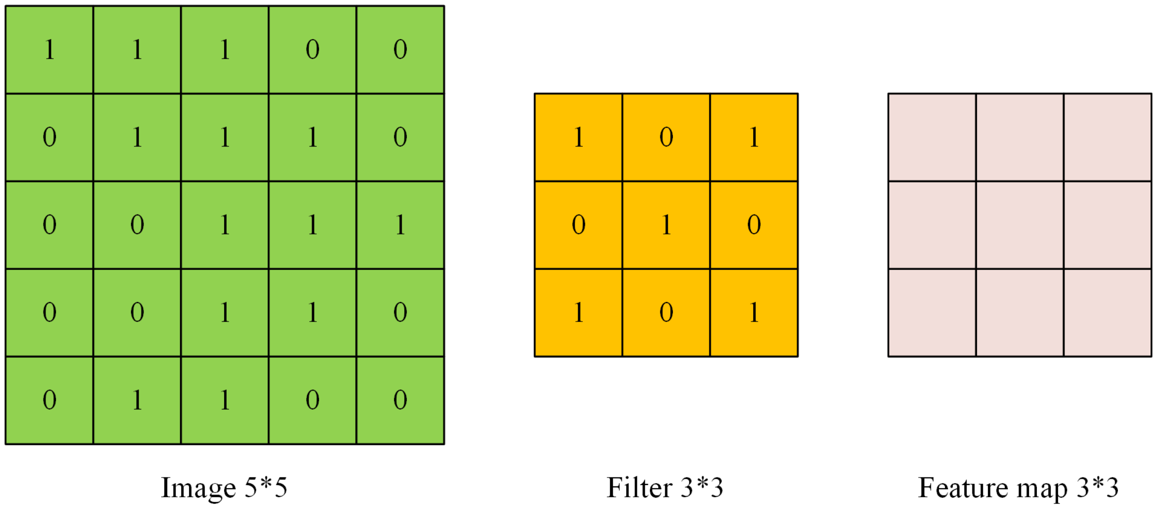


Figure 4.1: The element of convolution process

Rectified Linear Unit (ReLU) function is selected as the activation function). The convolution is then calculated using Formula 4.1:

$$a_{i,j} = \mathcal{F}(\sum_{m=0}^{2}\sum_{n=0}^{2} w_{m,n}x_{i+m,j+n} + w_b) \tag{4.1}$$

For the upper-left element $a_{0,0}$ of the Feature map, the convolution calculation method is:

$$a_{0,0} = \mathcal{F}(\sum_{m=0}^{2}\sum_{n=0}^{2} w_{m,n}x_{m+0,n+0} + w_b)$$

$$= \mathrm{relu}(w_{0,0}x_{0,0} + w_{0,1}x_{0,1} + w_{0,2}x_{0,2} + w_{1,0}x_{1,0} + w_{1,1}x_{1,1} + w_{1,2}x_{1,2} + w_{2,0}x_{2,0} + w_{2,1}x_{2,1} + w_{2,2}x_{2,2} + w_b)$$

$$= \mathrm{relu}(1 + 0 + 1 + 0 + 1 + 0 + 0 + 0 + 1 + 0)$$

$$= \mathrm{relu}(4)$$

$$= 4$$

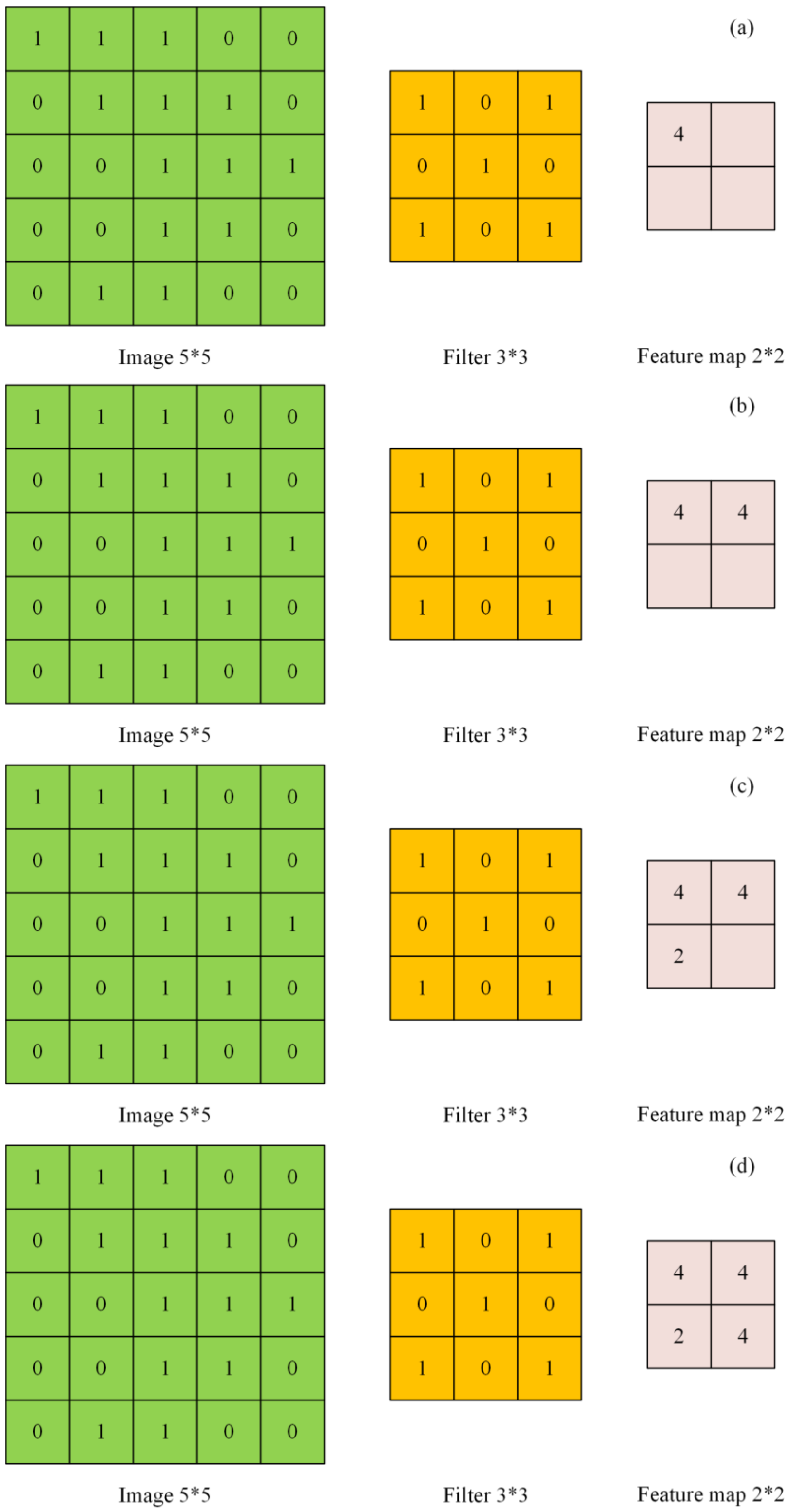


Figure 4.2: Convolution process

The result is shown in Figure 4.2 (a); if the stride is set to be 2, the feature map is calculated as Figure 4.2. This process shows that the size of Feature map is based on the stride and size of the image. In Figure 4.2, the 2*2 Feature map is calculated when the stride is set as 2; the relationship is as follows:

$$W_2 = (W_1 - \mathrm{F} + 2\mathrm{P})/S + 1 \tag{4.2}$$

$$H_2 = (H_1 - \mathrm{F} + 2\mathrm{P})/\mathrm{S} + 1 \tag{4.3}$$

where W_2 is the width of the Feature map after convolution, W_1 is the width of the image before convolution, F is the width of the filter, P is the number of zero padding (i.e., the number rounds of zero around the raw image). S is the stride, H_2 is the height of Feature map after convolution, and H_1 is the height of the image before convolution.

According to the above analysis, the convolution layer can extract the feature information of a 2D image by convolution, in the same way the convolution layer can also extract the feature information of depth image (i.e., a multi-channel image). If the image depth before convolution is *D*, then the corresponding filter depth must also be *D*. Formula 4.1 can be extended to get the convolution formula with depth greater than 1 as follows:

$$a_{i,j} = \mathcal{F}(\sum_{d=0}^{D-1}\sum_{m=0}^{F-1}\sum_{n=0}^{F-1} w_{d,m,n}x_{d,i+m,j+n} + w_b) \tag{4.4}$$

where *D* is the depth of the image, and *F* is the size of the filter (the width or height-both are same). $w_{d,m,n}$ represents the weight of the *d*-th layer, *m*-th row and *n*-th column of the filer. $x_{d,i,j}$ represents the pixels in layer *d*, row *i*, column *j* of the image.

Segmenting the image into smaller pieces helps extract feature information. Different features are extracted from images using sliding kernels (i.e., a filter) with the same weight. Convolution operations can be further classified into different types depending on the type and size of the filter, the type of padding, and the direction of the convolution [69]. In addition, if the kernel (filter) is symmetrical, the convolution operation will become a related operation.

4.2.3 Pooling Layers

The limitation of the feature map output of convolutional layers is that they record the precise location of the features. This means that small movements of feature locations in the input image will result in different feature maps. This can happen when re-cropping, rotating, shifting, and making other minor changes to the input image. A common method to solve this problem in signal processing is called down sampling, in which a lower-resolution version of the input signal that still contains large or important structural elements without the fine details that may not be helpful for the task is created. The main

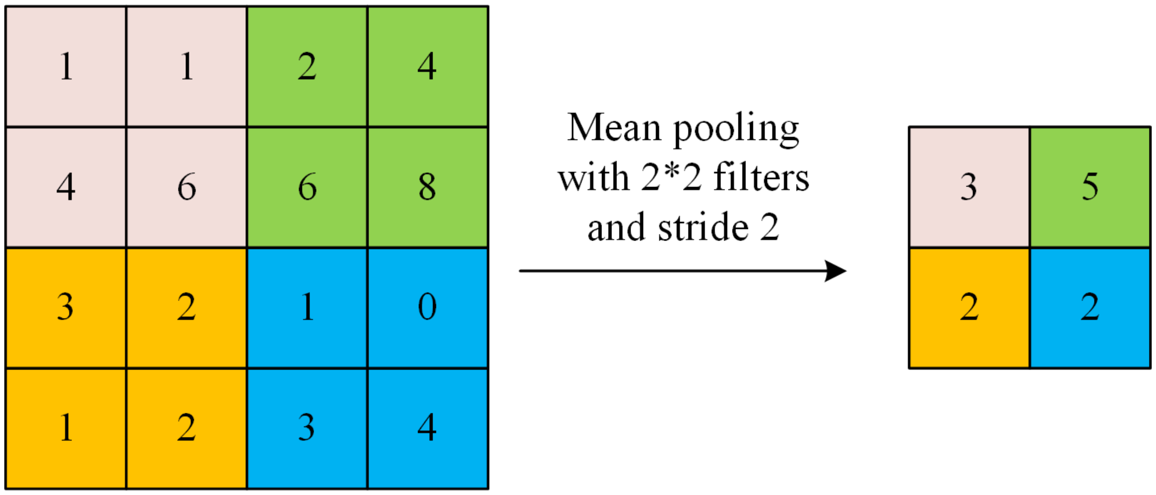


Figure 4.3: The mean pooling

function of the pooling layer is down sampling, which further reduces the number of parameters by removing unimportant samples from the feature map, reducing the tendency of overfitting. There are many pooling methods are used in a CNN: mean pooling, max pooling and stochastic pooling.

Mean pooling calculates the average of the elements for each patch on the feature map; this process is shown in Figure 4.3. Mean pooling will smoothly extract features and bring all features into the next layer, which means all values are used for the Feature map and output, so this is a generalized computation. Mean pooling is suitable for instances when all the data are important for the output, and causes the model to be more robust to space translations in the data.

Max pooling calculates the maximum of the elements for each patch on the feature

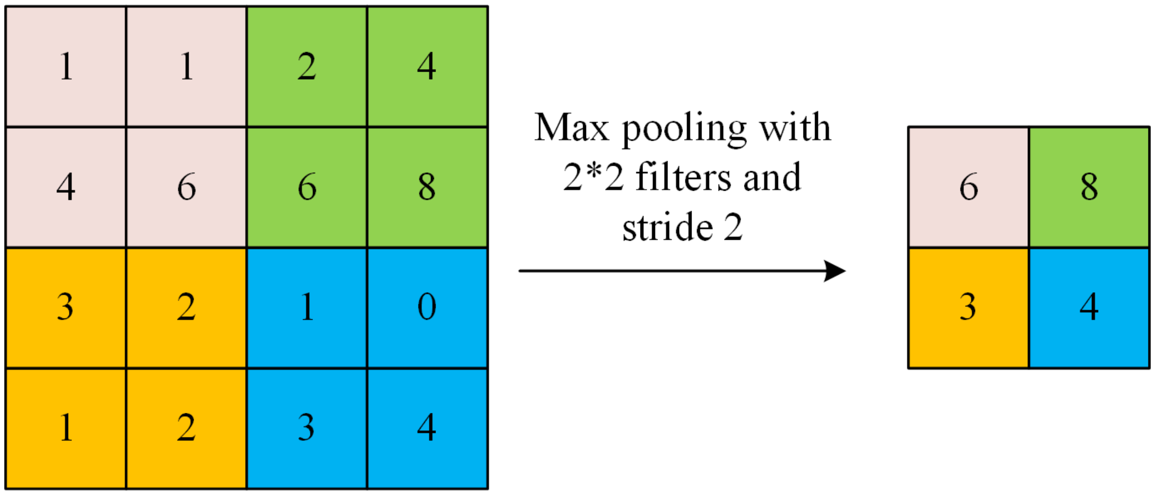


Figure 4.4: The max pooling

map; this process is shown in Figure 4.4. Max pooling is good for extracting the most important features like boundary location, chromatic aberration, and so on. Max pooling is too sensitive for some patterns in feature map, it is more informative to observe the maximum presence of different features than their average presence [81].

Stochastic pooling picks every possibly value for each patch on the feature map, unlike max pooling, which dismisses the smaller values. In each patch, first calculate the probability of the value based on Formula 4.5.

$$p_{i,j} = \frac{a_{i,j}}{\sum_{k\epsilon R} a_k} \tag{4.5}$$

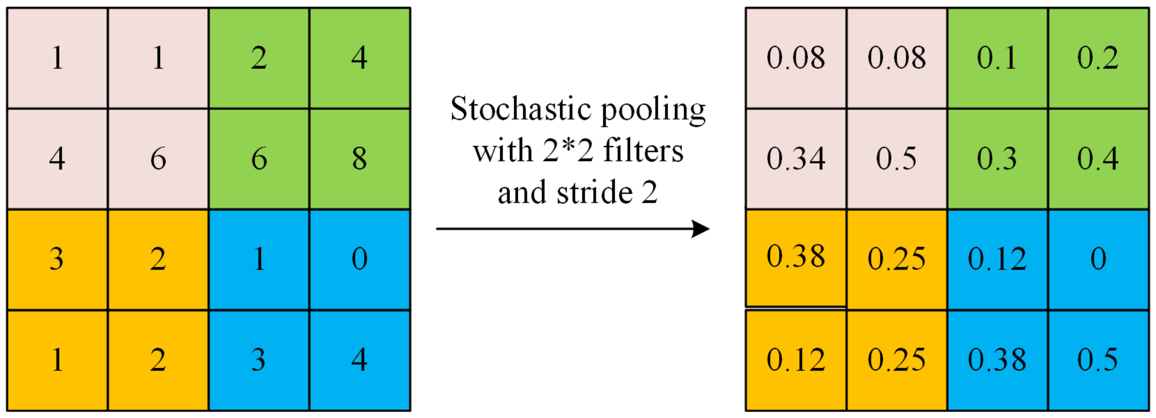


Figure 4.5: The stochastic pooling

Then, the output is chosen by the probabilities, as shown in Figure 4.5. Max pooling can be thought of as a special stochastic pooling, where the probability of the maximum value is 1 and others are 0. In stochastic pooling, the smaller values also have a chance of being chosen as output.

4.2.4 Activation Function

The activation function is a decision function that helps to learn complex patterns. Selection of an appropriate activation function can speed up the learning process. In a multilayer neural network, there is a functional relation between the output of the upper node and the input of the lower node. The output is normally a linear combination of the inputs, which is the primitive perceptron. By introducing nonlinear functions as an activation functions, the deep neural network expression ability is more powerful; it is no longer a linear combination of inputs, but can approximate almost any function. In the literature, different activation functions such as sigmoid, tanh, maxout, ReLU, and variants of ReLU such as ELU and PReLU. The sigmoid, tanh and ReLU functions are the most widely used in current CNN applications.

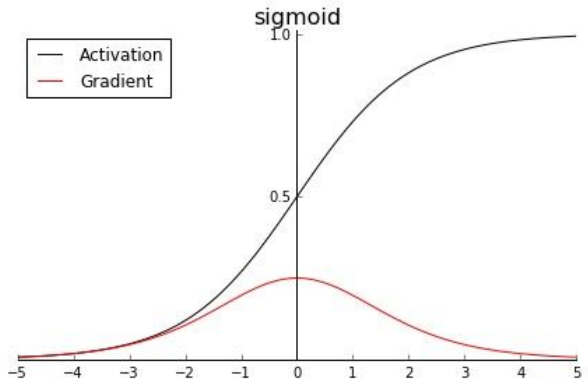


Figure 4.6: Sigmoid activation function and gradient curve

The sigmoid function is also called the logistic activation function. It compresses real values into the range of 0 to 1, as shown in Figure 4.6. This function can also be used in the output layer of prediction probability. This function converts large negative numbers to 0 and large positive numbers to 1. The mathematical function can be calculated with Formula 4.6:

$$\sigma(x) = \frac{1}{1 + e^{x}} \tag{4.6}$$

There are three main drawbacks of the sigmoid function: (1) the gradient disappears, and the rate of change becomes flat when the sigmoid function approaches 0 and 1. (thus, the gradient of the sigmoid approaches 0); (2) it is not zero-centric; and (3) calculations are costly

The tanh activation function is also called the hyperbolic-tangent activation function. Similar to the sigmoid function, the tanh function uses truth values. Unlike sigmoid, the output of the tanh function is zero-centric because the interval is between -1 and 1. The tanh functions can be imagined as two sigmoid functions put together which as shown in Figure 4.7. In practice, the tanh function takes precedence over the sigmoid function. Negative inputs are treated as negative values, zero input values are mapped to

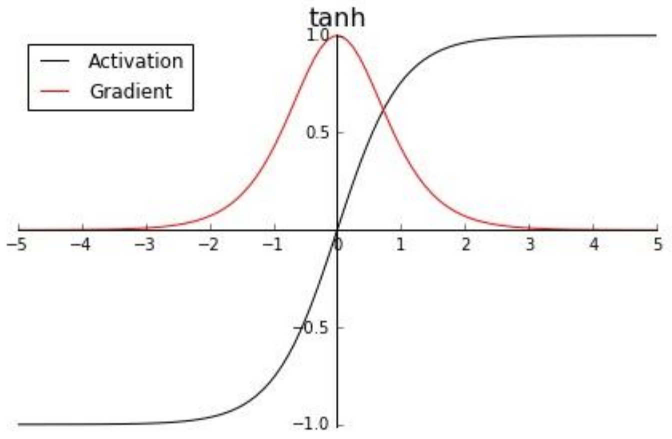


Figure 4.7: tanh activation function and gradient curve

near zero, and positive inputs are treated as positive values. The only disadvantage is that the tanh function also has the vanishing gradient problem.

Rectified linear unit (ReLU) solves the problem of vanishing gradients commonly found in sigmoid and tanh, and is also the fastest activation function for computing gradients. The mathematical formula is:

$$\sigma(x) = \max(0, x) \tag{4.7}$$

when x <0, the output is 0; and when x> 0, the output is x. This activation function makes the network converge more quickly. It does not saturate; thus, it can combat the vanishing gradient problem, at least in the positive region (when x> 0), so the neuron does not

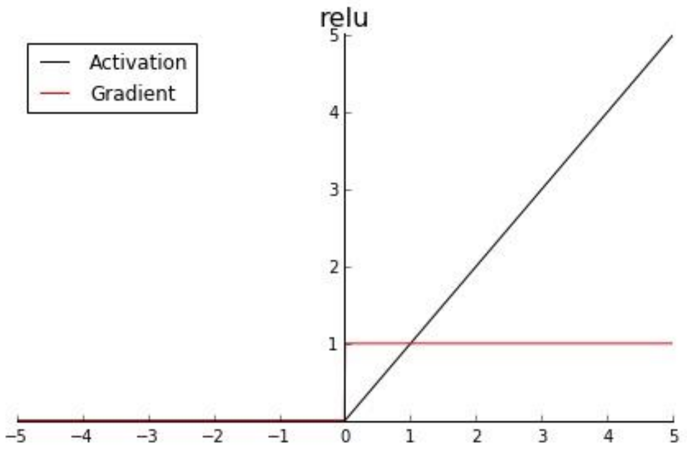


Figure 4.8: ReLU activation function and gradient curve

propagate all zeros in at least half of the region, as shown in Figure 4.8. Due to the use of simple thresholding, ReLU calculations are very efficient. But ReLU neurons also have some disadvantages: (1) it is not zero-centric; and (2) the activation value is always zero when the input continues to be negative.

4.2.5 Fully Connected Layer

In the CNN structure, after multiple convolutional layers and pooling layers, one or more fully connected layers are connected. Each neuron in the fully connected layer is fully connected to all neurons in the previous layer. Fully connected layers can integrate class-specific local information in convolutional or pooling layers. In order to improve the performance of CNN networks, the activation function of each neuron in the fully connected layer generally uses the ReLU function. Unlike the convolution and pooling layers, the fully connected layer is a global operation. It outputs the final classification result by globally analyzing the input from features of the preceding layers.

The specific architecture of a fully connected layer is shown in Figure 4.9. In this

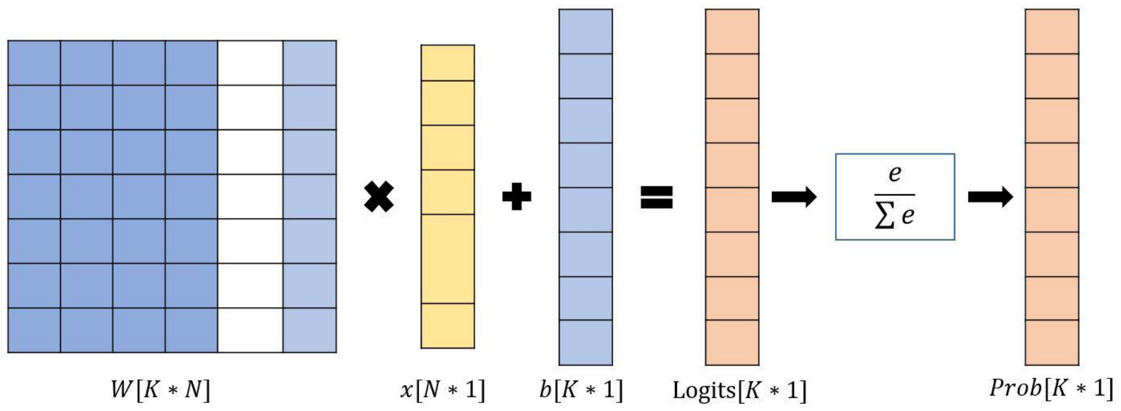


Figure 4.9: The architecture of fully connected layer

process, where W_{K*N} is the weight matrix of the neurons, K is the number of the categories to be classified, and *x* is the result of flattening the output of the previous layers and is also the input vectors of fully connected layer. The fully connected layer multiplies the weight matrix by the input vectors and adds the bias to map the $N(-\infty, +\infty)$ real numbers into $K(-\infty, +\infty)$ real numbers (fractions). And the softmax will map the $K(-\infty, +\infty)$ real numbers into $K(0, 1)$ real numbers (probabilities), while ensuring the components will add up to 1, so the results can be interpreted as probabilities [82]. The mathematical function is as shown in Formulas 4.8 and 4.9:

$$z = \begin{bmatrix} z_1 \\ \vdots \\ z_K \end{bmatrix} = \begin{bmatrix} w_1^T \\ \vdots \\ w_K^T \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \tag{4.8}$$

$$\hat{y} = \text{softmax}(\text{z}) = \text{softmax}(W^T x + b) \tag{4.9}$$

where $\hat{y}$ is the output probabilities of the softmax function, and *b* is the bias. The softmax function is as follows:

$$\text{softmax}(z_j) = \frac{e^{z_j}}{\sum_K e^{z_j}} \tag{4.10}$$

From the Formula 4.9, it can be seen that the fully connected layer analyzes the importance of the features in each dimension and obtains the scores of each category by the weighted sum of the features, and the softmax function then maps the scores to probabilities. Finally, the result of the neural networks is the class that has the highest probability. This is a whole process of the logistic regression, and the process of linear regression will not include the softmax function, as the output of the linear regression is continuous.

4.2.6 Architecture of CNN

CNNs are designed to simulate the human visual system. However, there are billions of neurons in the human brain. Even in current hardware, billions of neurons are large and difficult to implement. Therefore, researchers tend to design different architectures for different cases. Different improvements in CNN architecture have been proposed from 1989 to present. These improvements can be categorized as parameter poetization, structural reformulation and so on. However, the main development direction is to build deeper network architectures. An important specialty of a CNN is local connection, which means that each neuron does not perceive the entire image, but perceives a certain locality. Here the "local" scope is called the receptive field. Therefore, from shallow layers to deep layers, the receptive field is increased and some detailed features are removed due to the effect of stride or pooling. The CNN goes from extracting the underlying local features to being able to extract globally semantic features. In addition, the greater number of layers mean there are more parameters that can be adjusted, and the greater freedom of network adjustment, leads to a better fitting effect of the network.

Deeper networks mostly have the following two advantages: (1) Better fitting features and more powerful expression capabilities; deeper models mean better nonlinear expression capabilities, which can learn more complex transformations and can fit more complex feature inputs. (2) If the network is deeper, the things to be done at each layer are simpler, and it is possible to better learn layer by layer.

However, it also brings about the following problems: (1) Vanishing gradient and the exploding gradient problems. (2) The overfitting problem: As the number of network layers deepens and the number of parameters increases, the fitting ability of the neural network becomes stronger, which means that the functions it expresses will become more complicated. If simple problems such as binary classification problems are used with the deeper network, it will be too complicated and can easily cause overfitting. For complex problems such as image and language problems, the deeper network will get better results due to its complexity. (3) The degradation problem: As the number of network layers increases, the degradation problem will occur. The essence of the problem is that the networks cause overfitting due to the loss of information. This problem was originally proposed in the CNN network. In the structure of the CNN, each layer will produce an effect similar to lossy compression after passing the convolution kernel. If the convolution is performed multiple times, it will inevitably lead to an excessively high degree of abstraction and information loss, which will eventually lead to larger training errors and degradation problems. The deep residual neural network (ResNet) proposed by He et al. [76] solved this problem. Therefore, to improve our prediction model the ResNet is applied in this dissertation.

4.3 Residual Neural Network

The architecture and properties allow a CNN to achieve better ability on the vision problem. The state of the art CNN model goes deeper and deeper, the deep CNN architectures are based on the assumption that as the depth increases, the network can better approximate the objective function through a large number of nonlinear mappings and richer feature hierarchies [83]. Network depth is an important factor in the success of supervised training. Theoretical research shows that deep networks can represent certain types of functions more effectively than shallow structures [84]. However, simply stacking layers together will lead to the vanishing gradient problem and consumes a lot of computing power; the performance will be saturated, or may even start to rapidly degrade. In 2016 He et al. [76] proposed a residual neural network (ResNet) to revolutionize CNN performance by introducing the concept of residual learning in a CNN and designing an effective method to train deep networks, which also can avoid the problem of vanishing

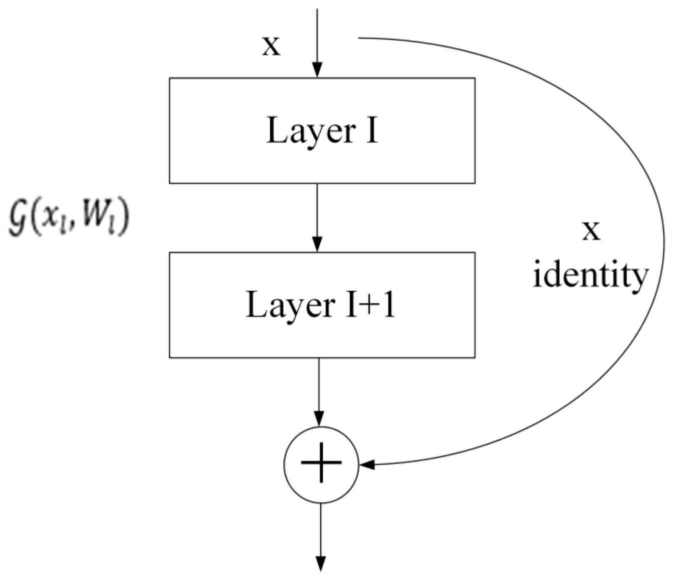


Figure 4.10: Canonical form of a Residual neural network

gradient. The core idea of ResNet is utilizing shortcut connections to skip some layers, as shown in the Figure 4.10. The mathematical formula of a ResNet is expressed as follows:

$$x_{l+1} = \hbar(x_l) + \mathcal{G}(x_l, W_l) \tag{4.11}$$

$$\mathcal{G}(x_l, W_l) = x_{l+1} - x_l \tag{4.12}$$

where $\mathcal{G}(x_l, W_l)$ is a transformed signal and x_l is an input of l^{th} layer without transformation. In Equation 4.11, W_l shows the W number of kernels from input layer to output layer, where the residual block can consist of one or more hidden layers. The input x_l is applied to the activation function $\hbar(.)$ and added to transformed signal $\mathcal{G}(x_l, W_l)$ through short-cuts, thus results in an aggregated output x_{l+1}. The ResNet believes that the residual function is easy to optimize and can improve the accuracy of depth.

The ResNet introduces short-cut connections in layers to achieve cross-layer connections. Compared with the gates of normal networks, this layer is independent of data and has no parameters. In ResNet, the residual information is always passed and the identity shortcut is never closed. Residual links (shortcut connections) speed up the convergence of deep networks, enabling ResNet to avoid gradient descent problems. The CNN and ResNet models are used as the base ideals for prediction models. As discussed above, due to the special characters of top-side images, I improved the performance of the prediction system from two aspects. One is to increase the width of the data to increase the temporal information (image sequence), another is to increase the depth of the model to improve the ability of image analysis (ResNet model).

4.4 Transfer Learning

An adequate model structure does not necessarily mean that good results will be obtained, because training a model well requires huge data and tedious parameter

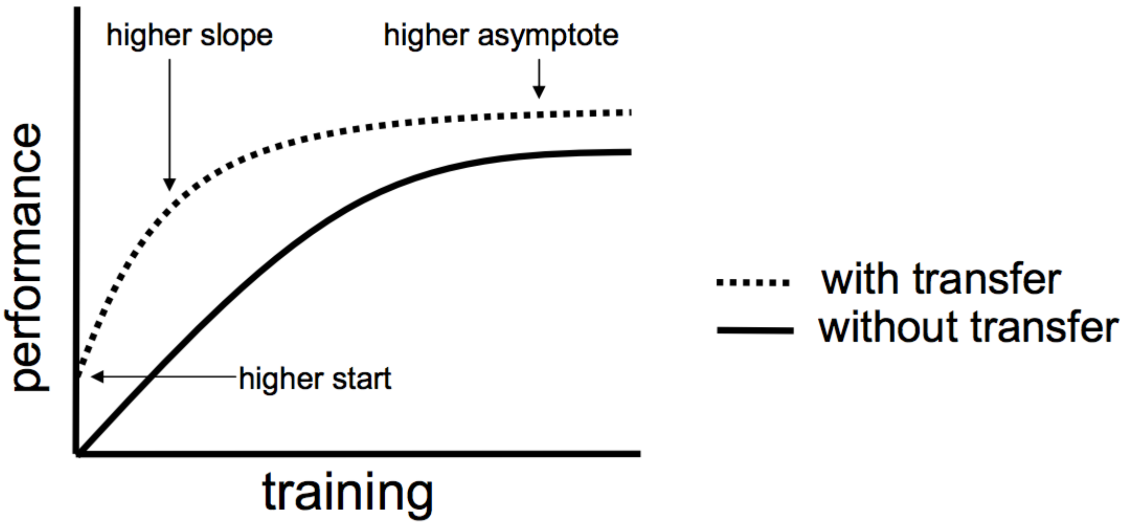


Figure 4.11: The performance of neural network model with transfer learning [86]

adjustment process. The end-to-end prediction system requires to be used simply and universally and the transfer learning is an effective and versatile method to help. Transfer leaning focuses on overcoming the isolated learning model and applying knowledge acquired from one task to solve related tasks [85]. It is useful to train neural network models with insufficient data by storing the weights of neural network from the pre-train data. This method has been proved to be suitable to solve practical problems in computer vision and natural language processing, because there is often not enough data in practical applications. The approach improves the baseline and final performance of the neural network model, and it also reduces the model-development time which is shown in Figure 4.11[86].

There are usually two way to train in transfer learning including: the develop model approach and the pre-train model approach.

The develop model approach includes the following steps: (1) select source task (choosing a related predictive modeling problem with an large amount of data where there are some relationship with the input data, output data, and/or concepts learned); (2) develop source model (developing a skillful model for the first task); (3) reuse mode (the model should fit on the source task and then be used as the starting point for a model on the task of interest); (4) tune model (the model needs to be adapted on the input and output data available for the task of interest).

The pre-trained model approach is easier than develop model approach. First, a pre-trained source model is chosen from the available model, then pre-trained source model is trained with a large data which has relationships with the task data. Next, the tuning model is adapted to the data available for the task of interest. In this dissertation, the ResNet model will be used to extract the high-level features from the input data, considering the training difficulty of the ResNet, which will thus be trained using the pre-train method approach to improve the accuracy and training speed in this dissertation.

CHAPTER 5. TIME INFORMTION FUSION IN CNNS

As discussed above, this system can automatically collect continuous image information of the top-side and back-side during the welding process. The continuous welding image information contains more temporal information than the static image. In this project, the temporal information can help to identify and analyze the characteristics of the welding process and enable the system to more accurately predict the penetration status. The CNN has been proven to be able to learn and interpret image features, and the input data in this dissertation is image information, so the CNN model is the only and optimal choice for analyzing image sequences. Using CNN to analyze continuous images with time sequence information, the network not only accesses the appearance information in a single static image, but also the complex time evolution. In this case, there are several challenges in extending and applying CNNs: (1) It is difficult to label continuous images on a large scale; the data-design and labeling issues will be discussed in Chapter 6. (2) A CNN requires a long training period to effectively optimize millions of parameterized models. When the processing object is time-sequence images, the amount of calculation is greater because the network must process multiple frames of images at a time, not just one image. This challenge will be solved by the data design, which reduces the useless images, and the transfer-learning approach to reduce the model-development time. (3) How can CNN connect and take advantage of this time information? This chapter will discuss multiple CNN architectures to analyze this problem. Each architecture uses a different method to fuse information across time domains, which are the following frame fusion strategies.

5.1 Single Frame Model

The single-frame model analyzes the single-frame image, which is derived from one whole image sequence or a video to achieve the classification goal. This process randomly selects a frame image from the video or image sequence and converts the video classification into an image classification. Using this method as a baseline, the classification effect is still possible in some specific cases, because some behaviors can be

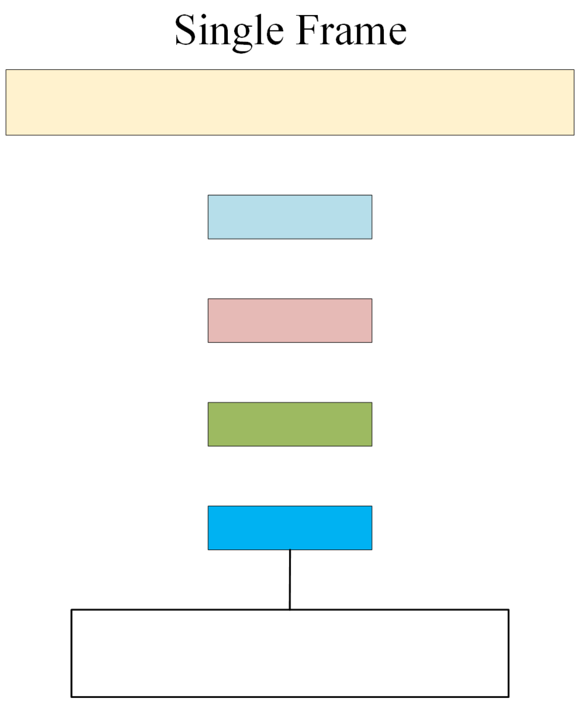


Figure 5.1: The architecture of single frame model

judged through a single frame of screen, but some behaviors are not. The single frame model is the same as the normal CNN model, which is the forward convolution recognition operation of a single frame. The architecture of single frame model is shown in Figure 5.1. This approach is similar to such methods as ConvLSTM [87] and CNN-RNN [88], which use the convolution layers to extract characteristic information from each frame of video or image sequence, and then the LSTM [89] or RNN [90] model will learn the temporal relationship between the outputs of the convolution layers. However, RNN and LSTM have better capabilities for analyzing and fusing temporal information than the full connectivity layer, because the LSTM and RNN is proposed to predict the results with the temporal information.

5.2 Late Fusion

The late-fusion method convolves two frames of the same video or image sequence separated by a certain time, then fused in the first fully connected layer to compute global motion characteristics; because the fusion function is after the convolution layers, it is called late fusion. Therefore, the previous convolutional layer does not detect any motion, but the first fully connected layer can receive timing information, and the global motion characteristics are calculated by comparing the output of both frames. The late-fusion model can also be seen as the fusion of two single-frame models at the fully connected

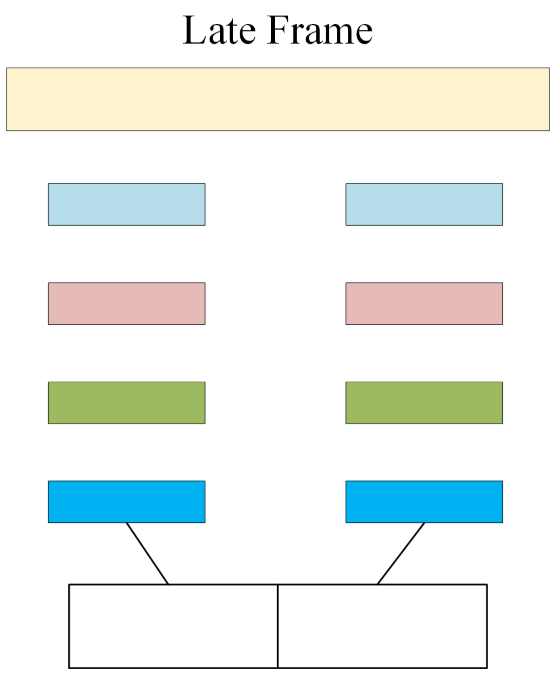


Figure 5.2: The architecture of single frame model

layers, and is shown in Figure 5.2.

The two-stream convolution network is another evolution of the late-fusion model, which also places two separate convolution networks (ConvNet); one spatial-stream ConvNet is used to extract the image information from the single frame and output the image classifications, and another temporal stream ConvNet is used to extract the motion characteristics from the continuous frames. The structures of both streams are the same, and the probabilistic output scores can be separately predicted by the softmax function, then the output scores from the two streams are fused by a support-vector machine (SVM) [91].

The late fusion is widely used in the prior art due to its simplicity and variability. However, a big disadvantage of the late-fusion model is that it is too expensive in terms of learning effort, because each model requires a separate supervised learning stage. In addition, combinatorial representation requires an additional learning stage. Another disadvantage of this method is that correlations may be lost in the mixed feature space [92].

5.3 Early Fusion

The early fusion method uses continuous multiple frames for prediction, and fuses them in the first convolution layer. The filter size ($\mathtt{W} * \mathtt{H} * \mathtt{C}$) on the first convolution layer is extended to $\mathtt{W} * \mathtt{H} * \mathtt{C} * \mathtt{T}$, where the *W* is the width of the input image, *H* is the height of the input image, *C* is the channel number of the input image and *T* is the number of fused frame, which means the filter size will extend through the full depth of the input volume. The filter will convolve across the width and height of the input data, the convolutional layer can more clearly see the direction and speed of the motion in the pixel level, and so early fusion is also referred to as fusion in feature space. The unimodal features extracted from the different frames which are all in the same video or image sequence are integrated into a single large feature vectors set for training, and the

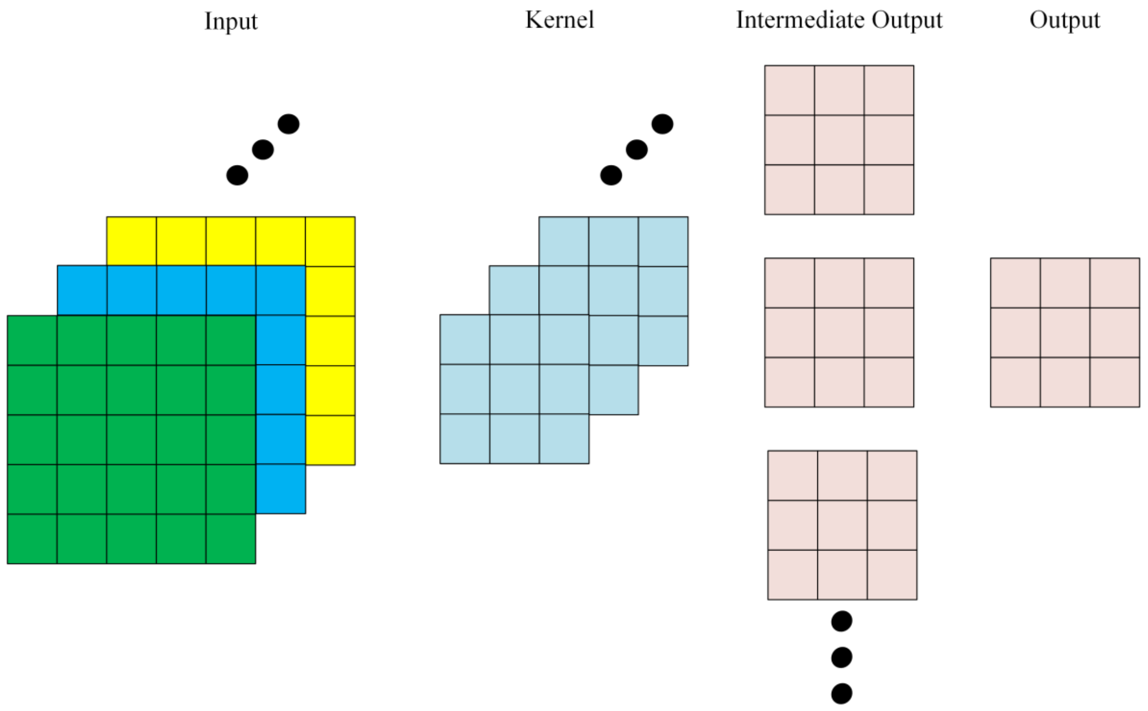


Figure 5.3: The multiple frames convolved with CNN

normalization is performed to keep the features on the same scale. Classifiers are trained for the target results using these large feature vector sets.

Early fusion captures the true essence of continuous frame fusion as all features are grouped together in a unified form. The convolution layer convolves the consecutive frames at the same time, so that the feature map is connected to multiple adjacent consecutive frames in the previous layer, and the motion information is captured sequentially. This process is shown in Figure 5.3. This process shows that the shape of theinput image as (input_height, input_width, input_channels, input_depth); here the input-depth is 3, which means that three consecutive frames are used as input instead of three channels, and the shape of kernel size is (kernel_height, kernel_width, kernel_channels, kernel_depth). The same parameters of kernels are used in the same channel, it is same as that the weight used by each channel of the kernel in a normal convolution. The weight of different channels may be different. In early-fusion process, the input of continuous frames is overall convolved once.

The early-fusion method was performed by Snoek et al. [92] to combine the visual and textual in the feature level. The visual vector contains pixel decision values for all pixels in the most representative image segment, which is combined with histogram-based text features extracted from the voice record. SVM classifiers are then trained for each semantic concept. The results show that six out of 20 concepts have better performance with early fusion than late fusion. Zou et al. [93] tracked people in a cluttered environment by fusing multi-modal measurements, which take advantage of the correlation between the visual movement of walking people and the corresponding footsteps at the feature level. A delayed neural network (TDNN) is used to train and combine audio and visual features.

Many early fusion evolution methods have been proposed to deal with the problem of the multiple information fusion, and different methods address different problems.

5.4 Slow Fusion

Slow fusion is a balanced hybrid method between early fusion and late fusion. This method slowly fuses temporal information through the whole network, so that the deeper network layers can gradually obtain more global information form the space and time dimensions of the input data. Slow fusion uses 10 consecutive frames as input, and the 10 frames (0, 10) are divided into (0, 4), (2, 6), (4, 8) and (6, 10) in the form of length as 4 and stride as 2. Then the four parts of the local time sequence information are divided into two groups and early fusion is performed on the same convolution layer. Finally, the probabilistic output scores from the two groups are further fused on the fully connected

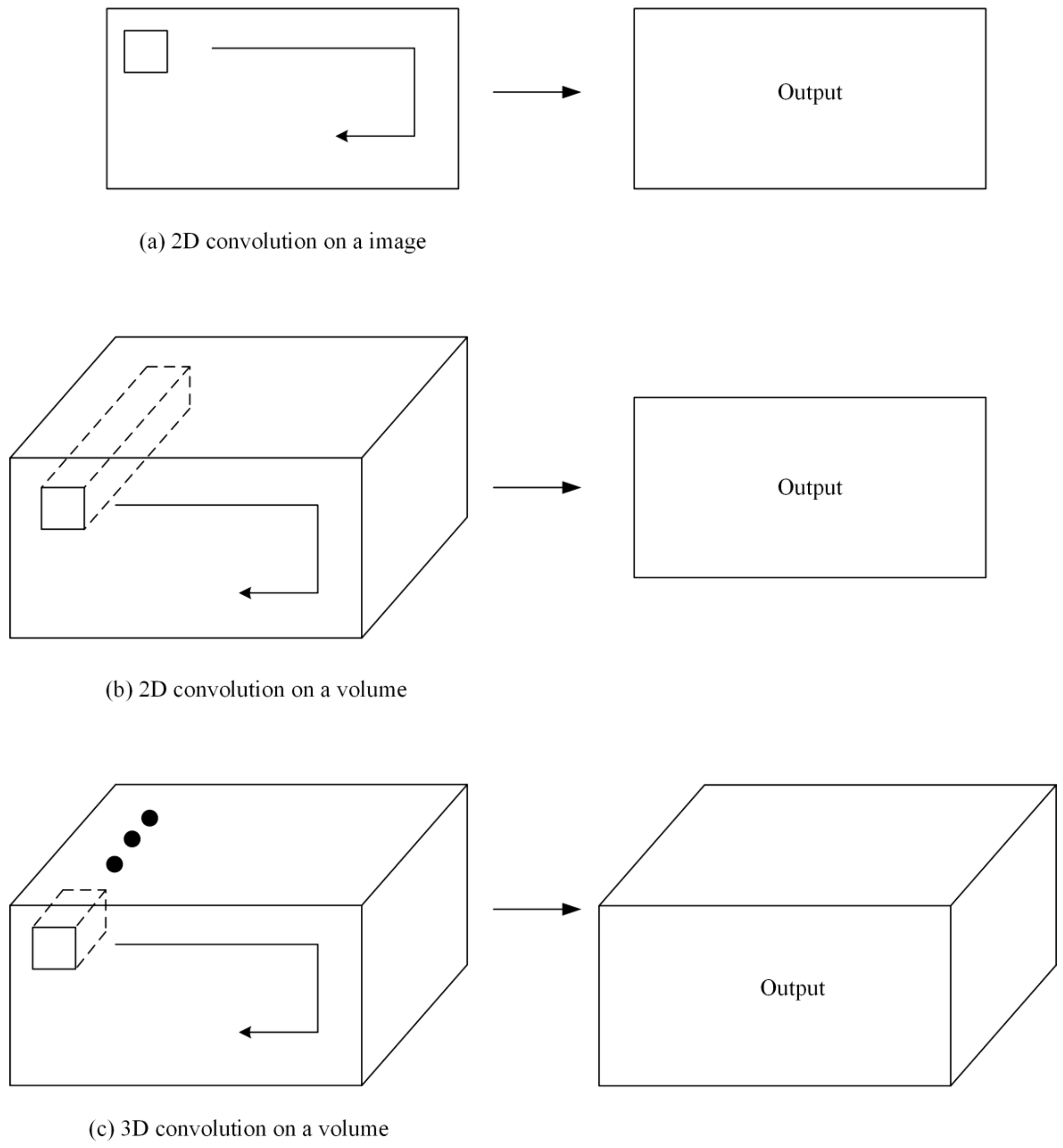

Output
(a) 2D convolution on a image
Output
(b) 2D convolution on a volume
Output
(c) 3D convolution on a volume

Figure 5.4: The 2D and 3D convolution operations

layer. Overall, the process improves the receptive field from 4 frames to 8 frames and finally raised to 10 frames. Obviously, this idea is very deep learning and slowly fusion. This method is the changed 3D-CNN, Figure 5.4 shows that the differences between 3D-CNN and 2D-CNN (general CNN). In the field of image processing, the objects to be convolved are still images, so using a 2D convolution network is sufficient. In the field of video or image sequence understanding, in order to retain temporal information at the same time, it is necessary to learn the spatiotemporal characteristics at the same time. If 2D-CNN is used to process the video or image sequence, the motion information between consecutive multiple frames cannot be considered, because if applying 2D convolution on a video volume also results in an image. In this process, a cube formed by stacking multiple consecutive frames at the same time is convolved with a 3D kernel. Through this construction, the feature map on the convolutional layer is connected to multiple consecutive frames of the previous layer, thereby capturing motion information. Furthermore, the 3D-kernel moves slowly in the three-dimensional space of continuous images to ensure learning and extracting more spatiotemporal characteristics, this process can learn more information than the early fusion.

Slow fusion has better performance than the other three fusion methods, because it can learn and extract the spatiotemporal information better due to the sliding 3D-kernel operation and two-times fusion. However, it also has some disadvantages such as training difficulty, expensive computation, large memory consumption, and so on.

5.5 Summary

Single frame is a method of single-image classification, so we will not consider using it here. Early fusion involves fusing multiple features that are extracted from early layers in the CNN, then using the fused features to train a predictor (later layers). Late fusion uses separate networks to predict, then fuses the prediction results from these reworks together. The slow-fusion model needs a much larger dataset and longer computation time, and it is difficult to predict the penetration state in real time. Early fusion appears to be more appropriate than the other two choices for our application, because the goal of our application is to find the hidden dynamic features in the feature space, so the neural network can use the dynamic features to predict the weld penetration status. Early fusion is an early attempt by researchers to conduct multi-modal fusion. Since only a common model needs to be trained, by associating the underlying features of each mode with the learning correlation, the complexity is controllable and the input information sources are the same, which will not increase the difficulty of the original data fusion.

The key first step in our application is to find the feature information; it is difficult to find these important features, which are hidden in low-contrast images by using a single image. Therefore, we need to use the method of time-information fusion to extract temporal and spatial information to help the network discover these key dynamic features by the early fusion. When the multiply frames enter the convolution layer, the multiply kernels convolve the multiply frames at the same time and superpose the intermediate outputs to get a vector, which contains pixel decision values that make it easy to find the dynamic information. This process the multiply frames are analyzed by the same receptive field, the dynamic features are easily discovered through comparison previous frame in the process. The second step is to train this feature information. In our application, we believe that it is more interesting to use deeper neural networks for training, because the network can better extract deep-level information from complex features by the deeper neural networks, instead of simply fusing the probabilistic output scores of the previous network. Therefore, we adopt early fusion and apply a ResNet to increase the number of network layers in order to better predict the results from the features.

The disadvantages of these methods cannot be ignored. In the next chapter, we will focus on how to use clever data design to avoid these problems, such as incomplete fusion information in early fusion and excessive training data in 3D-CNN.

CHAPTER 6. DATA DESIGN

A series of experiments were designed to perform spot welding using the gas GTAW process. Each 1.85mm-thick stainless steel sheet was welded 12 spots with 2cm distance apart. For completeness and reasonableness of the experiments, the range of welding current and time is designed to simulate a wide range of conditions. In each experiment, one stainless steel sheet was subjected to 12 complete spot weldings in sequence from low current to high current, and the corresponding welding time was from long to short. After each spot was completed, the current drops to 15 amps and the platform moved to the next spot-welding position. This ensured that each spot welding could be continuously performed without the need to restart the arc every time. At the same time,

Table 6.1: Welding parameters applied

Welding Parameters	Value
Welding Type	GTAW
Welding Current (A)	60-110
Welding Time (S)	4-12
Tungsten Diameter (mm)	2.4
Shielded Gas	Argon
Gas Flow (L/Min)	7
Workpiece Material	304L

the shielding gas argon was always output to protect both sides of the welding process. All these processes are automatically performed by the system, and the detailed parameters are shown in Table 6.1.

During each experiment, cameras can capture images of the entire welding process from partial penetration to excessive penetration. 28,494 images pairs haven been collected as raw data from more than 120 experiments, and all the images pairs are segmented into training, validation, and test datasets with sizes of 22,794, 2,849, and 2,851respectively. The experiment data is stored separately in a folder that is named in the experiment sequence. Furthermore, each image is named in the time when it is collected, so as long as the name of the corresponding image is detected, it can be discriminated whether the image pair is synchronized. Other parameters such as current, voltage, and speed of the welding process are all saved as a .csv file in each experiment folder, which is to prepare for future data analysis.

The collected images are vision information from both sides of the welding process, which simulates the visual experience of a welder during the welding operation, and the welding-situation pictures which a welder can see being directly collected as input data for the training model. For completeness and authenticity of the image information, none images collected will be subjected to any preprocessing. Generally, many designers use image enhancement processes such as noise addition, horizontal flip, and rotation for image recognition and classification projects. Image enhancement processing cannot increase the information of the original image; it can only enhance the ability to discern certain information, and this processing will definitely lose other information, and the quality of the enhanced image will be difficult to quantify.

In this project, the system predicted the weld perpetration status through the complete and useful information of the top-side images of the welding process; the relationship between top-side images and weld penetration status is a complex physical process. The design purpose of this system is to automatically define and extract features through data-driven, so there is no need to enhance certain features of interest. If the process of image enhancement is also adopted, the original physical relationship may be broken, and we cannot synchronously change the back-side image to conform to the original physical change process of welding. Therefore, the original images will be directly used as input data. Of course, in this process, we need to consider how to effectively improve the quality of the data and enrich the data information. We will use the image sequence as the input data to increase the spatial and temporal information of the data.

6.1 Image Sequence Design

In a typical CNN model, the input images are passed through the network one by one. The image generator yields (N, W, H, C) data, where *N* is the batch size, *W* and *H* are the width and height of the images, and *C* is the number of channels (3 for RGB images, and 1 for grayscale images). The image sequence dataset needs several frames in a sequence and the data dimension becomes (N, W, H, C, F), where *F* is the number of frames in a sequence. In this project each experiment data is a large image sequence, if the large image sequence (over 500 frames) is directly inputted the model, it will be a disaster for the model. It is impossible for a prediction model to analyze so many frames at the same time, and as discussed in Chapter 5, some information will be lost if the early fusion model convolves too many frames.

Another method is to divide the whole image sequence into several parts, it is also a problem to analyze each small image sequence separately. How to effectively segment these image sequences is a very important issue, because there are relationships between the segmented image sequences, and the corresponding connections of the sequence may be segmented after the segmentation. In this case, the most critical time information is the moment when the penetration state changes, and we do not know when this critical point is. If the information of the critical point is divided, it will lose its design significance.

The 3D kernel function provides a good ideal, which can effectively control the size of the input data and only needs to control the stride size to avoid the problem of image sequence segmentation. In our system the frame rate of camera is 30 frames per second, so we set every sequence to be 1/3 s which has 10 frames (grayscale images), and the stride is set to 1 to prevent missing critical information. In this application, the welding

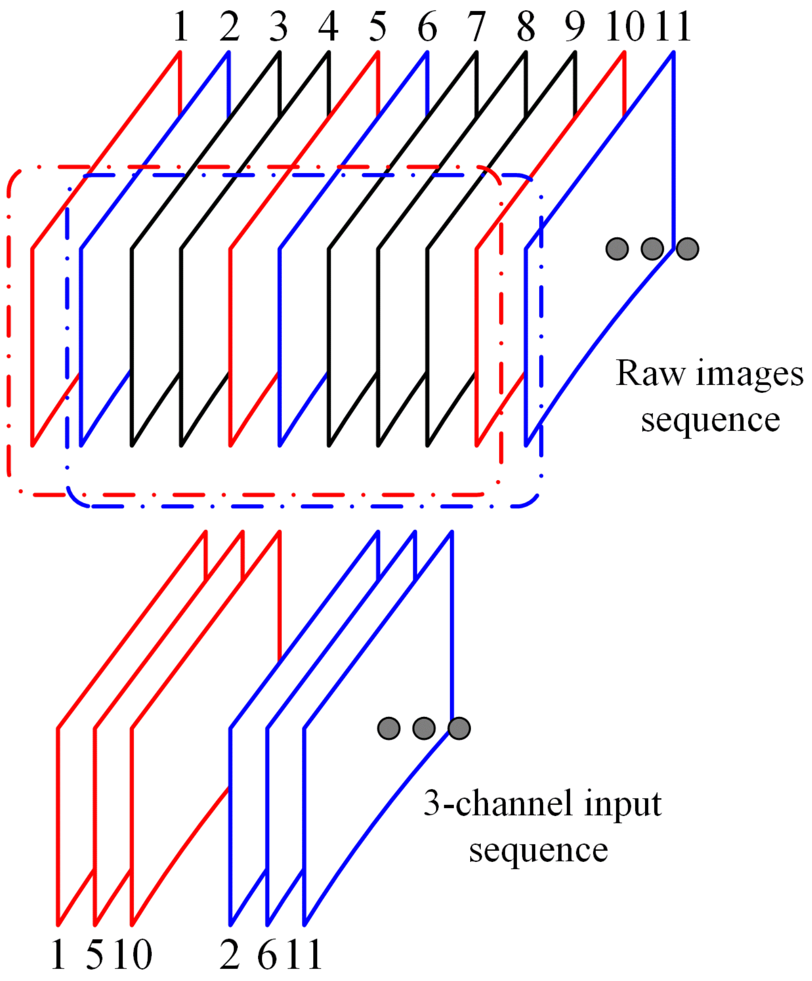


Figure 6.1: Image sequence design

development is very slow, so every frame in the sequence is not equally important as it is in other applications such as motion analysis and video classification [94], [95]. We pick the last image (Image 10), the image in the middle of the sequence (1/6 s earlier or Image 5) and the first image (Image 1) in the sequence as the sample of the sequence. The sampled sequence is used as the input of the CNN. The image sequence design is shown in Figure 6.1.

The three red frames are one input sequence data, and the three blue frames are the second input sequence data with the stride as 1. With this design, the early fusion can also have the same fusion effect as 3D-CNN, and the sliding specifies a size of ten frames which is a suitable time width. What's more, in order to reduce too many calculations, three frames are taken to represent the spatiotemporal information in every ten-frame sequence. The designed image-sequence data can meet the requirements of quickly and effectively identifying feature information.

This image-sequence design is necessary for the analysis of subtle changes. Since the angle and position of the camera are exactly the same in all experiments, the welding

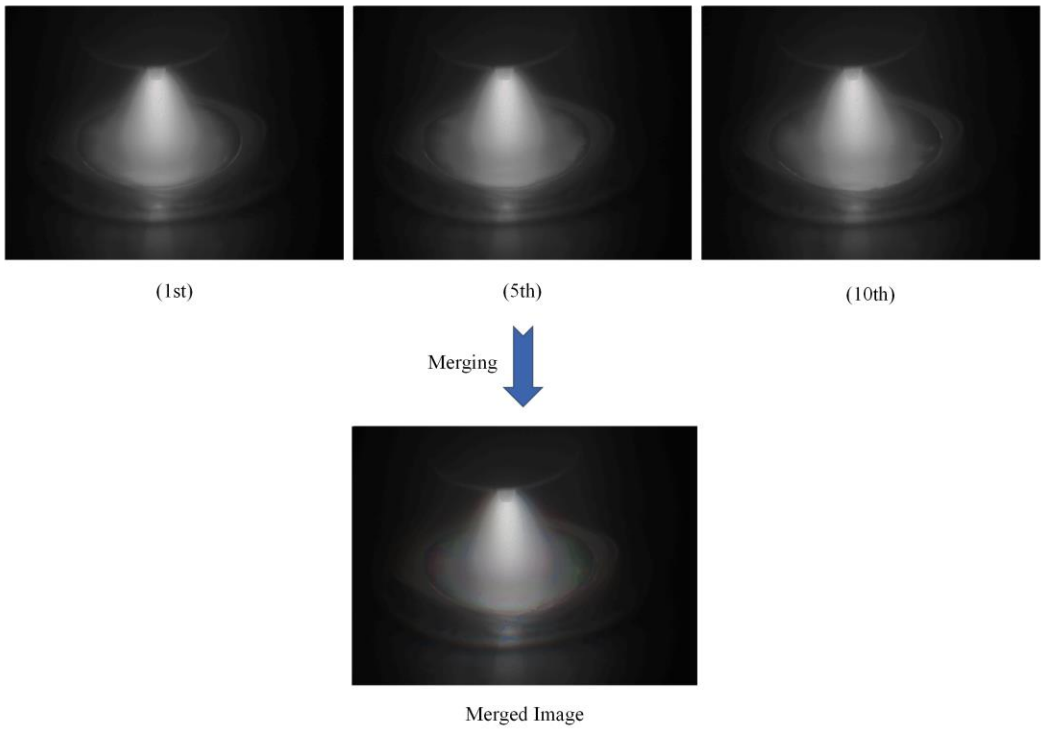


Figure 6.2: Three frames are merged as an RGB image

pool, weld arc and other information are in the same corresponding position for all the images. According to the above design, three frames of grayscale images are merged into 3 channels to form an RGB image. As shown in Figure 6.2, it is easy to find other colors around the boundary of the welding pool and the shape of the arc. By comparing the colors, we can clearly see the local characteristic information and dynamic information. Similarly, when three frames of images enter the same convolution layer and are calculated by the multiply convolution kernels, it is easy to find the pixel changes at the same location by superimposing the filter maps. This local dynamic information is the feature that needs to be found and analyzed. Therefore, all the raw images are converted into 3-channel images according the above design, that the three frames are in the same receptive field as same as merged in 3 channels. And the merged images are named as the 10th image's name. Taking into account the requirements for real-time prediction of the penetration status during the welding process, the first input merged images must wait until the tenth frame is acquired from the beginning of the process, and each subsequent merged images input only needs to wait for the most recent frame acquisition, because the previous 9 frames have already been obtained. Hence, we named these merged images in the time of the last frame acquisition, so the corresponding label is also the penetration status of the last frame during the training process.

6.2 Labels

The back-side images are used as the penetration state labels for training the CNN model. The penetration state is defined by the range of the real back-bead width. However, it is not possible for this to be measured simultaneously with the top-side images in real time during the welding process. Therefore, we calculate the back-bead width value through a model that correlates the back-side image to the real back-bead width. The conversion is demonstrated in Figure 6.3. To this end, we first define a bright area using a

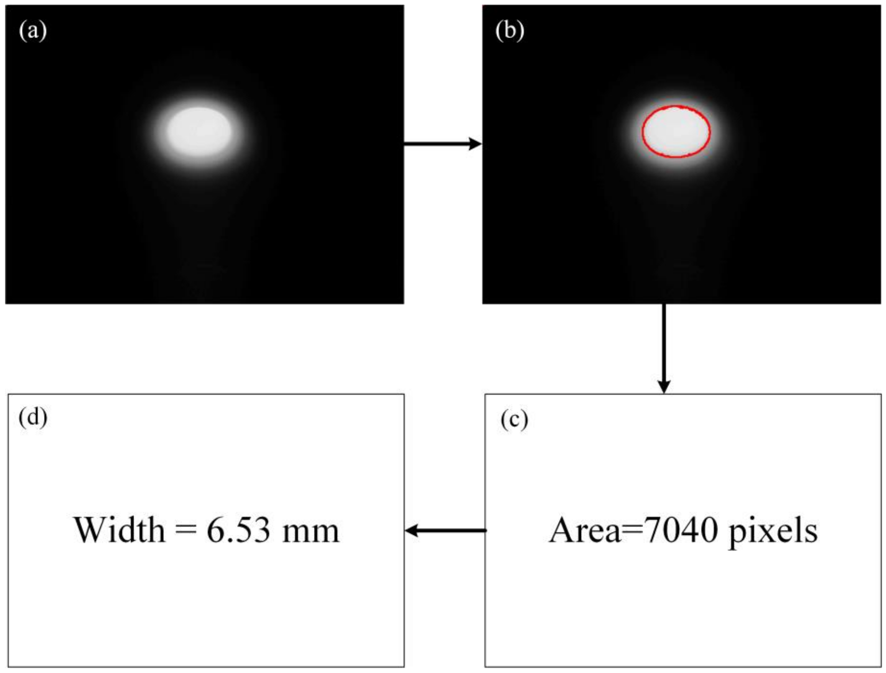


Figure 6.3: The process of back-side image to convert to the width of back-side bead width

threshold, then calculate it from the back-side image. In this dissertation, we set the threshold at 170. The relationship between the area and the actual back-bead width was then established through calibration experiment. In order to reduce the measurement error, we also conducted 6 sets of experiments with different times and different currents, then accurately measured the widths of the back-side images, as shown in Figure 6.4. The back-side welding pool generally is irregularly round, so its four diameter data are measured separately, then the average value is taken as the width of the back-side welding pool. Since the square of the actual back-bead width is proportional to the area (number of pixels), and we can propose Formula 6.1:

$$Width = k * \sqrt{Area - Error} \qquad 6.1$$

where k and Error are unknown parameters determined by least squares approximation, the fitting curve is shown in Figure 6.5. The coefficient of determination is 0.9879, which indicates that the regression predictions approximate the real data well. The back-bead width can be calculated from the back-side images with the formula 6.1, and the penetration state is classified according to the back-bead width. If the threshold is changed, the relationship will be changed but the bright area will also be changed such that the same

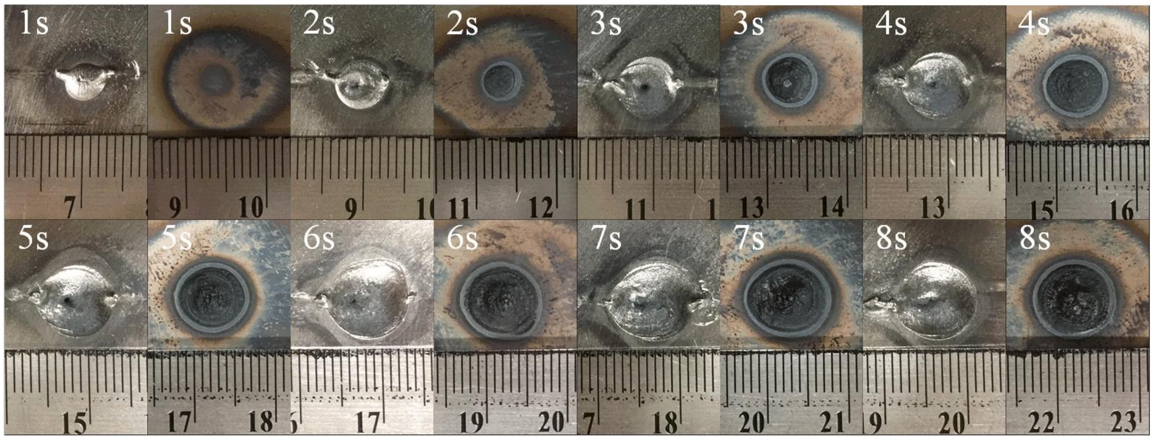


Figure 6.4: The calibration experiment

width of the back-side is converted.

The goal to predict the weld penetration to assure that the weld penetration is appropriate and as desired. As such, we can classify the weld penetration into three categories: under penetration including partial penetration where the back-side bead width is zero, desirable penetration and excessive penetration. Table 6.2 lists the classification criteria used in this dissertation. Such criteria may change per application such that the model will be changed, but the effectiveness of our proposed methods should not. Per the criteria in Table 6.2, the back-side width calculated from the back-side image and the relationship in Figure 6.5 is classified into one of the three penetration categories that will be used as a training label for the CNN model.

With the merged images-sequence data and the labels, we create three independent

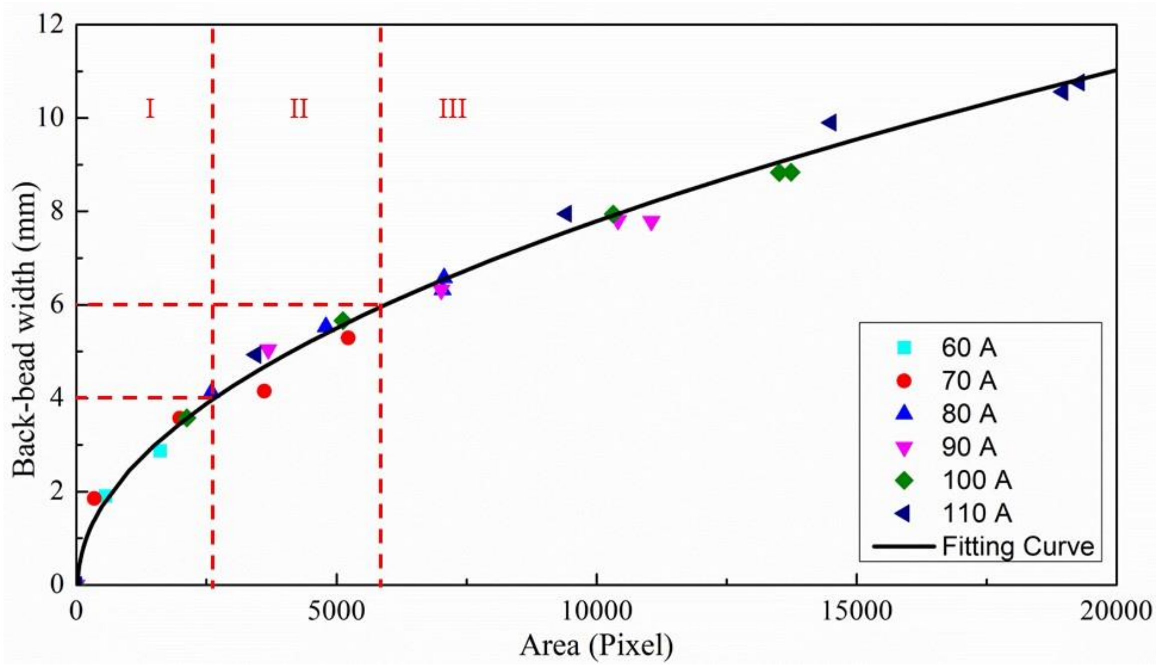


Figure 6.5: Model fitting between area and back-side bead width

datasets (training dataset, validation dataset and test dataset) to ensure the training results of neural networks are convincing. The three datasets are completely different, because if the images of the test dataset are also used for training, we can't judge whether this neural network really learned or just remembered all the images. Therefore, we divided 120 sets of experimental data into three parts. The first 96 sets of data were used as training data, and there is a total of 21,823 merged image-sequences. The next two sets of data, each with 12 spot welding experiments which are divided into verification data (2,797 merged image-sequences) and test data (2,794 merged image-sequences). The three datasets are respectively saved in three .txt files with the corresponding label information, and the detailed path and name of the merged image-sequences. In this way, during the training process the model only needs to read the .txt file to obtain all the training merged image-

Table 6.2: Labels and penetration status

Label	**Number of images**	**Back-bead width(mm)**	**Penetration status**
0	11474	< 4	Under penetration
1	5268	4~6	Desirable penetration
2	11755	> 6	Excessive penetration

sequences and labels.

CHAPTER 7. MODEL TRAINING

Early fusion can immediately expand the entire time window to fusion information at the pixel level. This is achieved by modifying the kernels of the first convolutional layer in the single-frame model by expanding the size to W*H*C*F. In this dissertation, we set the same weight of kernel for each frame, because the designed image sequence can be easily found in the dynamic features in the same receptive field, and the designed image sequence reduces the redundant image. Hence, the early fusion model can learn and extract the dynamic features in the first convolution layer with the designed image sequence, and the short time window of the designed image sequences also help our model attain the advantage of slow fusion, which can gradually learn and extract more dynamic information. Thus, the kernel size also can be expanded to be W*H*(C*F), which means the 3 frames are converted into 3-channels, this matches with our design sequence data. The network can then accurately detect local motion directions and speeds [96]. With the early fusion method, we designed a nine-layer CNN model and an eighteen-layer ResNet model to train three independent datasets. This chapter discusses how to train neural networks, with a focus on hyperparameter settings and tricks for training neural networks.

7.1 CNN Model and ResNet Model

Different CNN structures may be constructed per the specific tasks in computer vision. Normally, 5 - 10 layers are sufficient for relatively simple image classification problems. In this paper, a nine-layer CNN architecture that includes three convolutional layers, three max-pooling layers and three fully-connected layers are used. The final output is the classification of the weld penetration state is predicted by one softmax regression layer. As shown in Table 7.1, the CNN (early fusion) takes 480*640*3 merged image-

Table 7.1: The architecture of the CNN (early fusion)

Layer Name	**Kernel size**	**No. of Filters**	**Output Size**	**CNN (early fusion)**
Conv1	5*5	32	476*636	Batch Normaization, ReLU
Pool1	3*3	-	158*212	MaxPool
Conv2	3*3	64	156*210	Batch Normaization, ReLU
Pool2	3*3	-	52*70	MaxPool
Conv3	3*3	128	50*68	Batch Normaization, ReLU
Pool3	3*3	-	16*22	MaxPool
FC1	-	-	1080	-
FC2	-	-	64	-
FC3	-	-	3	softmax

sequence as input, and its Conv1 has 32 kernels of 5*5 (3-channel) to fuse the temporal information at the pixel level by convoluting over the 3-channel raw images. A 3*3 Maxpool layer follows the Conv1 to reduce the computation cost of the whole neural network. The similar processes with different kernel size and kernel number are applied on Conv2, Pool2, Conv3 and Pool3 respectively to further extract the features. The CNN model is a tradeoff between speed and accuracy: the 5*5 kernel is used in Conv1 to better fuse information from 3-channel images and the 3*3 kernels are used in Conv2 and Conv3 to reduce memory usage and compute faster [20]. Then, the data is flattened and connected with three fully-connected layers to output the classification of the weld penetration state.

The input data is the merged top-side images. There are many important features (such as edges, shapes, colors, time information etc.) under the arc light. To extract features with better precision the network must perform more convolutions. The vanishing gradient issue must be addressed. We explore by using ResNet 18-layer in the prediction model. The detailed architecture of the 18-layer ResNet is shown in Table 7.2. In the ResNet model the first convolution layer convolutes the input data with a 7*7 kernel, and all the follow-up convolution layers with 3*3 kernels. The short-cuts connections used to transmit the activations with a stride 2 after the first convolution layer. Moreover, optimization function, loss function and activation function are the same as the CNN model. The total parameters become very large after 17 convolution layers, which increases the training difficulty. Hence, we use the transfer learning method to pretrain ImageNet data by the 18-layers ResNet model, then we use the transferring information to train the top-side images. This method significantly improved the sample efficiency and performance.

Table 7.2: The architecture of the ResNet 18-layer

Layer Name	Output Size	ResNet 18-layer
Conv1	240*320	7*7, 64, stride 2
Conv2	120*160	3*3 MaxPool, stride 2
		$\begin{bmatrix} 3*3, & 64 \\ 3*3, & 64 \end{bmatrix} * 2$
Conv3	60*80	$\begin{bmatrix} 3*3, & 128 \\ 3*3, & 128 \end{bmatrix} * 2$
Conv4	30*40	$\begin{bmatrix} 3*3, & 256 \\ 3*3, & 256 \end{bmatrix} * 2$
Conv5	15*20	$\begin{bmatrix} 3*3, & 512 \\ 3*3, & 512 \end{bmatrix} * 2$
	1*1	Average pool, 3-d fc, softmax

7.2 Hyperparameter

Machine learning models are definitions of mathematical formulas in which many parameters need to be learned from the data. In other words, the model is fitted to the data by training the model using existing data. However, there is another parameter that cannot be directly learned from the regular training process. These parameters represent the "high-level" properties of the model and they are called hyperparameters. Hyperparameters are usually determined before the actual training process begins. The hyperparameters related to the network structure include the size of the input image, kernel size and pooling size.

In this dissertation, the size of the input merged image-sequence is 640*480*3. High-resolution input images mean richer information, which is helpful for the CNN to obtain more details and more robust features. For this case, high resolution is key for the low contrast and harsh imaging conditions. In the CNN model, the 5*5 kernel was used in the first convolution layer to better fusion and extract features, and the 3*3 kernels were used to reduce the parameters in the next convolution layers, which is similar to the setting of the 18-layer ResNet, just change the kernel size to 7 * 7 in the first convolutional layer. Since the network is deeper, increasing the kernel size can effectively reduce the calculation amount of the entire network. The ILSVRC winners commonly keep the kernel size at 3*3 or 5*5, which is often kept larger in the first convolution layer [74, 75] because the size is less important in the first layer, and there are fewer input channels.

Furthermore, the Maxpool function is used after each convolution layer for the CNN model. Due to the inputs, there are 3 sequential grayscale images, and the feature information is brighter pixels in the grayscale images, Maxpool is useful when the background of the images is dark and the target information is lighter pixels. Therefore, on Pool1 the Maxpooling is used to extract the extreme features and give the neural network robustness to position variance. Similarly, the Maxpool function is used after the first convolution layer in ResNet, and after all the convolution layers the global average pool is used to reduce the tendency of overfitting by reducing the total number of parameters in the model.

7.3 Tricks for Training Neural Networks

Training deep learning neural networks is very difficult because it requires knowledge and many experiences to optimize model and proper training. This section we discuss some tricks for training neural networks. The lost function is discussed first, which can optimize the parameter values. The optimization methods are important approaches to improve the training model. The learning tricks, including learning rate and decay learning rate, are used in our training process. Finally, the batch normalization, which controls the overfitting is discussed.

7.3.1 Loss Function

The loss function is used to estimate the difference between the predicted output value and the real value of the model, which can help researchers optimize the model [97]. The smaller the value of the loss function, the better the robustness of the model. The loss functions used by different models are generally different. Mean squared error loss (MSE) and mean absolution error loss (MAE) are the most commonly used loss functions in machine learning and deep learning for regression tasks [98]; we will not discuss these much here due to this dissertation being a classification task.

Cross entropy loss is a workhorse of basic loss function for the classification problem [99]. This case is a multi-classification task, the real value y_i is a one-hot vector [100], and the compression of the model output is the softmax function. The softmax function limits the output range of each dimension to (0, 1), and the output sum of all dimensions is 1. The probability distribution is shown as Formula 7.1:

$$\mathrm{p}(y_i|x_i) = \prod_{k=1}^{K} \left(\hat{y}_i^k\right)^{y_i^k} \tag{7.1}$$

where the $\hat{y}_i$ is output value, y_i is the real value, and $k \in K$ is one class of K categories.

Assuming the data points are independently and identically distributed, the negative log-likelihood can be obtained as:

$$\mathrm{NLL(x,y)} = L_{CE} = -\sum_{i=1}^{N} \sum_{k=1}^{K} y_i^k \log\left(\hat{y}_i^k\right) \tag{7.2}$$

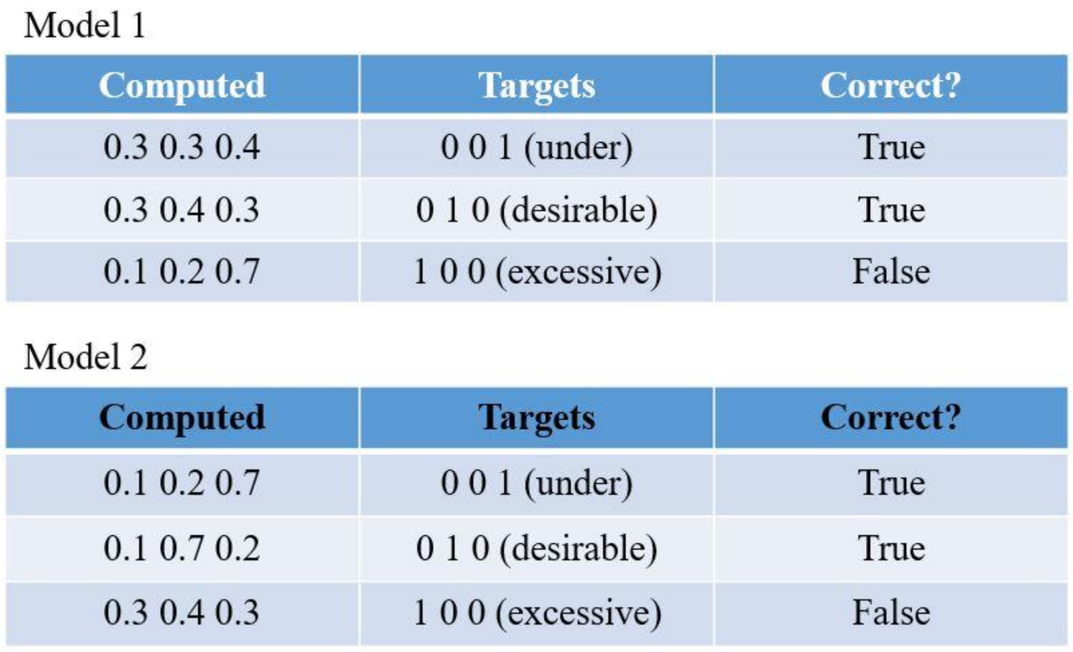
Due to the real value y_i is a one-hot vector, the output of other classes is 0, and the target class is 1, thus, the above formula can also be written as:

$$L_{CE} = -\sum_{i=1}^{N} y_i^{c_i} \log\left(\hat{y}_i^{\hat{c}_i}\right) \tag{7.3}$$

The integer c_i is the target class of x_i. When this cross-entropy loss function is applied to multi-classification, it is also called softmax loss or categorical cross entropy loss.

In this case, the labels have three categories, under penetration, desirable penetration, and excessive penetration. Assuming that there are two classification models, both of these models use the softmax function to obtain the probability value for each prediction, which are shown in Figure 7.1.

Model 1 correctly predicts Samples 1 and 2 with a very weak advantage, and incorrectly predicts Sample 3. Model 2 accurately predicts Samples 1 and 2, and incorrectly predicts Sample 3, but it is not too far from the truth. We then used Formula 7.3 to calculate the value of the loss function in the above models.


Model 1

Computed	Targets	Correct?
0.3 0.3 0.4	0 0 1 (under)	True
0.3 0.4 0.3	0 1 0 (desirable)	True
0.1 0.2 0.7	1 0 0 (excessive)	False

Model 2

Computed	Targets	Correct?
0.1 0.2 0.7	0 0 1 (under)	True
0.1 0.7 0.2	0 1 0 (desirable)	True
0.3 0.4 0.3	1 0 0 (excessive)	False

Figure 7.1: Model 1 and Model 2

Model 1:

Sample 1: $LOSS = -(0 * \log 0.3 + 0 * \log 0.3 + 1 * \log 0.4) = 0.91$

Sample 2: $LOSS = -(0 * \log 0.3 + 1 * \log 0.4 + 0 * \log 0.3) = 0.91$

Sample 3: $LOSS = -(1 * \log 0.1 + 0 * \log 0.2 + 0 * \log 0.7) = 2.3$

ALL $$L_{CE} = \frac{0.91 + 0.91 + 2.3}{3} = 1.37$$

Model 2:

Sample 1: $LOSS = -(0 * \log 0.1 + 0 * \log 0.2 + 1 * \log 0.7) = 0.35$

Sample 2: $LOSS = -(0 * \log 0.1 + 1 * \log 0.7 + 0 * \log 0.2) = 0.35$

Sample 3: $LOSS = -(1 * \log 0.3 + 0 * \log 0.4 + 0 * \log 0.4) = 1.2$

ALL $$L_{CE} = \frac{0.35 + 0.35 + 1.2}{3} = 0.65$$

From this process, we find that the cross-entropy loss function can capture the difference of prediction effects between Model 1 and Model 2. During the backpropagation training, the cross-entropy loss avoids the $(\hat{y}_i) * (1 - \hat{y}_i)$ term. Thus, the weight changes do not become smaller and smaller, and training will not stall out. With these advantages of the cross-entropy loss, it is used in the CNN and ResNet models, which helps both models achieve better results.

7.3.2 Optimizer

The training process of the neural network is to find the relationship between layers; the parameters are the weights of the layers, and the process of finding the parameters is called learning. Therefore, the purpose of the neural network is to constantly update the parameters to minimize the value of the loss function, then to find the best results. The gradient descent (GD) is a method of constantly updating parameters to find solutions, and it is also one of the most popular and common ways to optimize neural networks.

In calculus, taking the partial derivative for the parameters of the multivariate function, and the gradient is simply the vector of the partial derivative in each parameter, which is shown in Formula 7.4.

In geometry, the gradient vector is where the function change increases the fastest. Conversely, along the opposite direction of the gradient vector, the gradient decreases the fastest, so it is easier to find the minimum value of the function. In Formula 7.4, $f(x_1, x_2, \ldots, x_n)$ is a surface equation by the parameters $x \epsilon \mathbb{R}^n$, and the partial derivative

$$\nabla f(x_1, x_2, \ldots, x_n) = (\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \cdots, \frac{\partial f}{\partial x_n}) \tag{7.4}$$

$\frac{\partial f}{\partial x_1}$ is the slope. In the neural network, the gradient can be used to find the minimum point of loss function, then to find the best results of the network. However, an easy way for this to be used in deep learning will be difficult. Because deep learning models have many neurons and layers and different connection methods, the loss function often has many parameters. Furthermore, according to the type of the loss function, the gradient is usually nonlinear to the end. All these factors will lead to difficult calculations.

The concept of GD method is to constantly adjust the stride and direction from the initial point, which is randomly selected, then trying to find the optimal solution. The formula of gradient descent is as follows:

$$w^{t+1} = w^t + \eta * \left(-\nabla L(w^t)\right) = w^t - \eta * \nabla L(w^t) \tag{7.5}$$

where w^t is the initial point, and w^{t+1} is the target point, according to the direction towards the minimum of loss function, thus, $\nabla L(w^t)$ should be negative, and η is the learning rate.

According to the above introduction, the GD method must first determine whether the loss function is differentiable. If the loss function is a non-differentiable function, then it needs to be handled by convex optimization. If the loss function is a differentiable, the

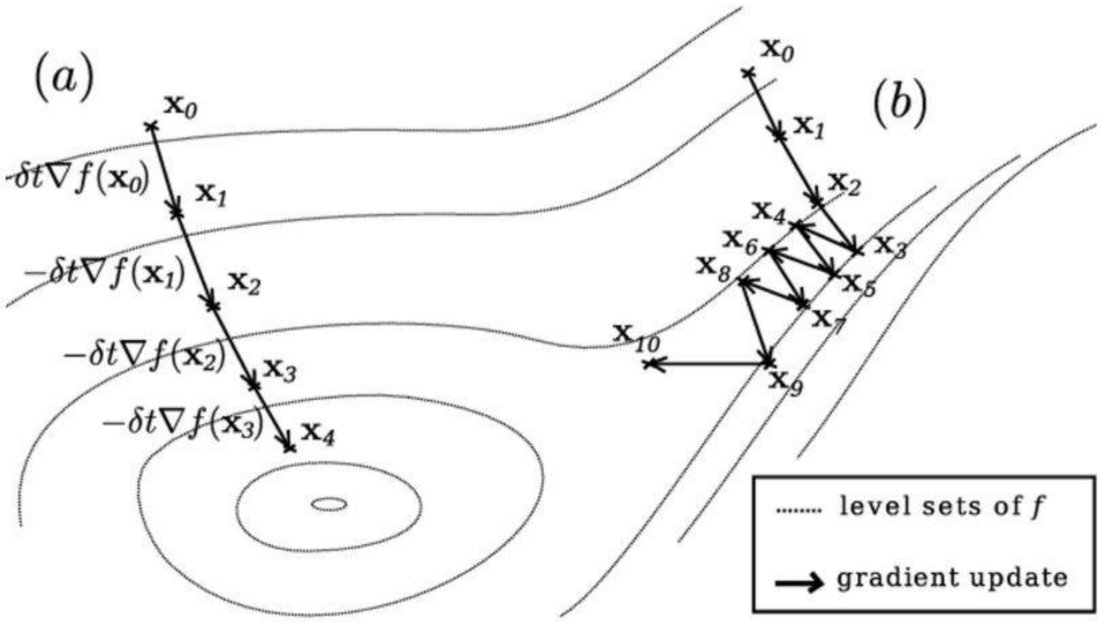


Figure 7.2: The effect of gradient descent with different initial points

initial point and learning rate both will affect the effect of the GD.

First, the GD process will change with the initial point as shown in Figure 7.2. The process (a) and process (b) have different paths, both of which try to find the minimum

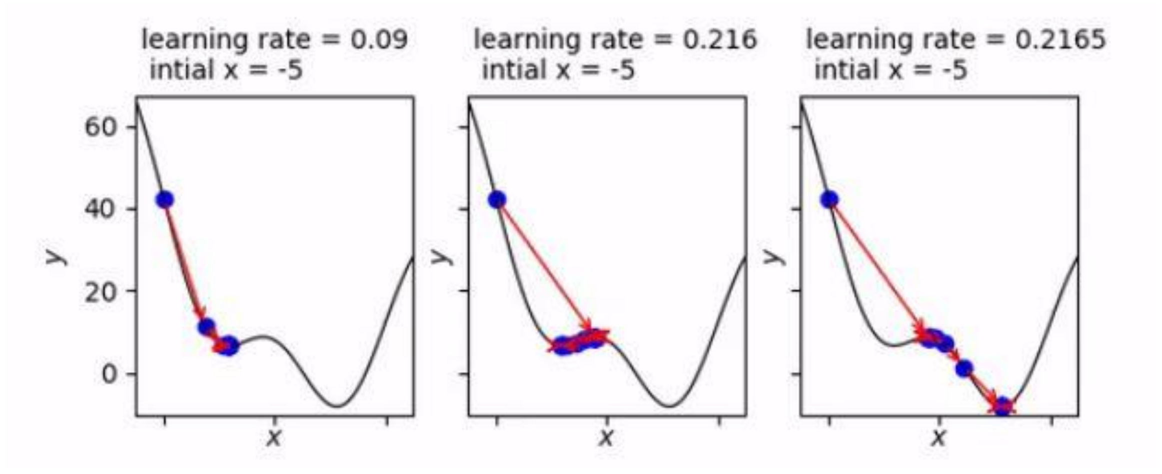


Figure 7.3: The effect of gradient descent with different learning rates [97]

point of the curved space, and the efficiency of both paths is different.

Furthermore, the learning rate also affect the GD, as shown in Figure 7.3 [97]. If the learning rate is small, the number of iterations will be many and will easily fall to the local minimum. If the learning rate is big, the amplitude of the iteration is large and not stable enough. So the best way is to adjust the learning rate with each iteration.

A variant of gradient descent is stochastic gradient descent (SGD), which is to run a training example or a mini-batch examples and then calculate the gradient or the average of the gradient of the small batch and update it once [101]. And this example or the examples of the small batch is randomly selected, the formula can be written as:

$$\mathrm{L}(w^t) = -\sum_{i=1}^{N} y_i^{c_i} \log\left(\hat{y}_i^{\hat{c}_i}\right) = -\sum_{i=1}^{N} \mathrm{y}(x^i, w^t) \log\left(\hat{y}_i^{\hat{c}_i}\right) \tag{7.6}$$

$$\nabla \mathrm{L}(w^t) = -\nabla \sum_{i=1}^{N} \mathrm{y}(x^i, w^t) \log\left(\hat{y}_i^{\hat{c}_i}\right) \tag{7.7}$$

The SGD method is different from the GD method; all parameters must be calculated to generate a gradient. SGD only uses one sample or a mini-batch sample to calculate the gradient. As long as enough samples are randomly selected, the final expected value will be similar to the result of the GD method. Before updating the gradients, the SGD method only uses one training example during the training process. When the training set is larger (deep learning always needs a large training set), the SGD can be faster than GD and can also be used to learn online. However, the process will oscillate toward the

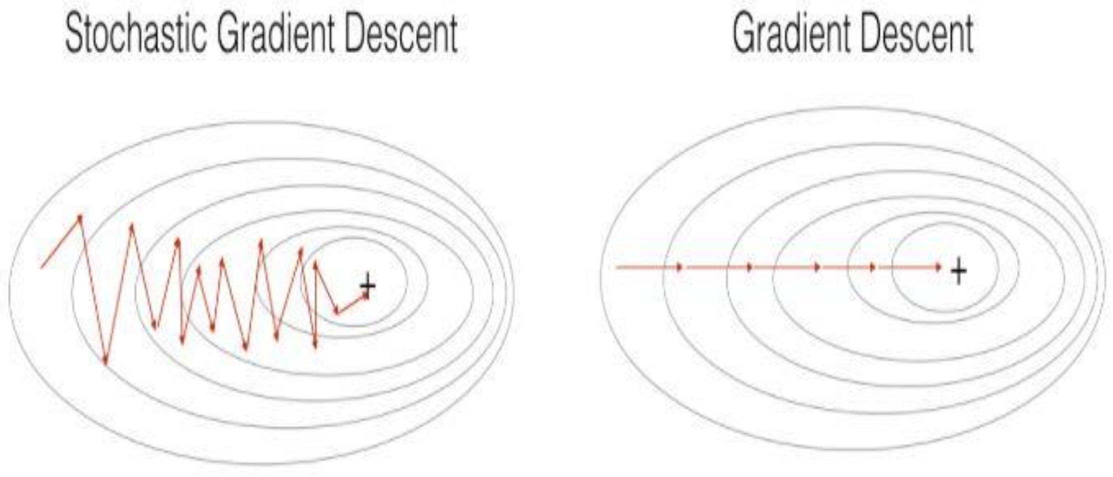


Figure 7.4: SGD vs GD [102]

minimum point rather than converge smoothly, as shown in Figure 7.4 [102].

SGD's fluctuation will help it jump local minimum to look for the global minimum. However, when the learning rate is large, the fluctuation will also be large, which is not easy to converge. Therefore, a better way to use SGD is by slowly decreasing the learning rate to help it achieve better convergence behavior.

Another widely used method is adaptive moment estimation (Adam) [103]. Adam will compute adaptive learning rate for each parameter to adjust the better learning rate for the training process. In addition to storing the exponentially decaying average of past squared gradients (v_t), the Adam also keeps exponentially decaying average of past gradients (m_t). The Adam combines the advantages of AdaGrad [104] and RMSProp [105] and the formula v_t and m_t as follows:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla \mathrm{L}(w^t) \tag{7.8}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla \mathrm{L}(w^t))^2 \tag{7.9}$$

The m_t and v_t integers are the estimates of the first moment and the second moment of the gradient, as is the name of the algorithm. The Equation 7.8 - 7.11 shows the updating process for the parameters.

$$\widehat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{7.10}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{7.11}$$

$$w^{t+1} = w^t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\widehat{m}_t \tag{7.12}$$

where the $\nabla \mathrm{L}(w^t)$ is the gradient, and as the first moment m_t and second moment v_t are initialized as 0, they are biased towards zero, especially when the decay rates are small [103]. In generally, the default values of 0.9 for β_1, 0.999 for β_2, and 10^{-8} for ϵ.

In this case, when we compared SGD and Adam in our CNN and ResNet models, we observed that the performance of SGD was better than the Adam, but the convergence speed of Adam was faster. We think the adaptive learning rate will not be correct for each parameter during the training process, so the incorrect learning rates will disturb the performance of Adam. Hence, in this case, we choose the SGD to be an optimizer which is also in line with the initial setting of the VGG and ResNet [76], [106]. The role of the optimizer is not only to optimize the parameters in the current network, but also to optimize the information in the back propagation. When the model code is written by PyTorch, the gradients need to be manually set to 0 before the backpropagation, because the PyTorch will accumulate the gradients.

7.3.3 Learning Rate

As discussed above, the learning rate controls the degree of gradient adjustment. A small learning rate helps the neural network converge to the global minimum, but it will take a lot of time because the weight of the network has only a few adjustments updates in each parameters. A smaller learning rate is also more likely to trap the neural network in the local minimum, because the smaller learning rate cannot jump out of the local minimum. However, a higher learning rate may also bring undesirable consequences. A

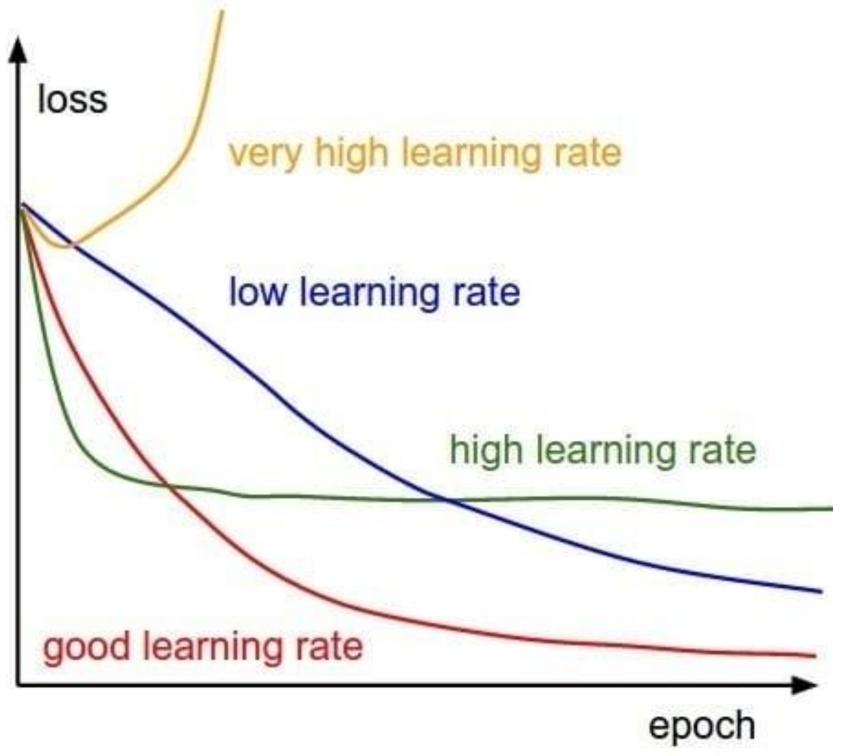


Figure 7.5: Different training processes with different learning rates [97]

high learning rate can almost never reach the global minimum because it is likely to skip the extreme value. In this way, the gradient will fluctuate greatly and make the network difficult to converge. Therefore, it is difficult to properly set the learning rate. The Figure 7.5 demonstrates the different situations with different learning rates [97].

The selection strategy of the learning rate is constantly changing during the network training process. At the beginning, the parameters are relatively random, so we should choose a relatively large learning rate so the loss declines faster. After training for a period of time, the update parameter should have a smaller amplitude, so the learning rate will decay; there are many ways to decay, such as discrete staircase, exponential decay, and 1/*t* decay. Here we will not discuss more about the decay methods. Another problem is how to determine the initial learning rate. An inefficient way is to try each level of learning rate to run the network, and observe the loss results. A relatively reasonable learning rate can be chosen, but this method is too time-consuming.

Leslie N. Smith proposed using a cyclical learning rate that is widely used in the deep learning such as ResNet, AlexNet and GoogLeNet [107]. This method eliminates the way to experimentally find the best global learning rate, instead of monotonously reducing the learning rate. This method uses a periodic learning rate to keep the learning rate between reasonable boundary values, thereby improving classification accuracy, and usually with fewer iterations.

As discussed above, the performance of Adam is slightly worse than the SGD in this case, due to the adaptive learning rate that may add noise when it gets an incorrect learning rate in the training process. Thus, we gave up the adaptive learning rate and chose the cyclical learning rate, which brought about good results in this case.

7.3.4 Batch Normalization

During the training process, the parameters in the network are constantly updated with the gradient decreases. When the parameters in the previous layer change slightly, these subtle changes are amplified with the network deepens. In addition, the change of parameters causes the input distribution of each layer to change, and the upper layer network needs to constantly adapt to these distribution changes, which makes model training difficult. This phenomenon is called internal covariate shift. The internal covariate shift brings about two problem in the neural network: (1) the network needs to be constantly adjusted to adapt to changes in the distribution of input data, which reduces learning speed, and (2) the training process fall easily into gradient saturation, which slows down the network convergence speed.

In this case, we used the batch normalization (BN) to avoid internal covariate shift and improve the speed, stability and performance of the training model [108, 109]. The principle states that when the parameters are passed from the previous layer to the next layer, a normalization process is performed before entering the next layer of the network. This method is used to make each dimension feature mean 0 and variance 1, the formula is as follows:

$$\hat{x}^k = \frac{x^k - E(x^k)}{\sqrt{Var(x^k)}} \tag{7.13}$$

where the man $E()$ and variance $Var()$ are calculated based on the batch data.

However, if only the normalization formula is used, then the features learned by the next layer of network will be distorted. In this way, the feature distribution learned in the previous layer is broken, so the transformation and reconstruction are added at the end of the BN process to retain the features learned by the previous layer, the step follows as:

$$y^k = \gamma^k \hat{x}_i^k + \beta^k \tag{7.14}$$

where the parameters γ^k and β^k are learnt in the subsequent optimization process.

In the CNN model, the characteristics of the convolutional neural network correspond to a whole feature response map, so when doing BN the response map should be used as a unit instead of according to each dimension. In this case, we followed this logic to add BN between the convolutional layers and set the size of BN to be

M*W*H*(C*F), where the *M* is the batch size. With this setting the results of the CNN are in line with forecast.

7.4 Summary

In this chapter, the setting of the training model is discussed. For the performance view, a batch normalization (BN) and Rectified Linear Unit (ReLU) function followed each Conv to improve the performance of the CNN model. BN had the effect of stabilizing the learning process and dramatically accelerating the neural network training speed by reducing internal covariate shift, and reducing generalization error [108]. ReLU was used as activation function; comparing with other activation functions such as tanh and sigmoid, ReLU can overcome the vanishing gradient problem and allow the model to learn faster and perform better [110]. In addition, the CNN model is trained using mini-batch back-propagation algorithm. The loss function is set as the cross-entropy for this multi-class classification problem, and SGD optimization is used to reduce the computation cost and improve the convergence rate in the training process [101]. In the ResNet model, the optimization function, the loss function and the activation function are the same as the CNN model.

CHAPTER 8. RESULTS AND ANALYSIS

In this chapter, we will discuss the results of the CNN and the ResNet with normal single image and merged image-sequence. All trainings were done on a person computer with an Intel(R) Core (TM) i7-6700K CPU and an NVIDIA GeForce GTX 1080 GPU. Both models respectively trained single images and merged image-sequence continues 20 epoches with the batch size as 30 (Due to the limit of RAM, 30 is highest number of batch size by our computer) and the training accuracies are shown in Figure 8.1. Compared with the CNN model, the accuracy of ResNet with transfer learning had as higher start and higher asymptote, no matter what kind of data we used, the ResNet model shows similarly excellent performance. Comparing these two different types of data, single-image data often has a better start, but the performance of the two types of data is

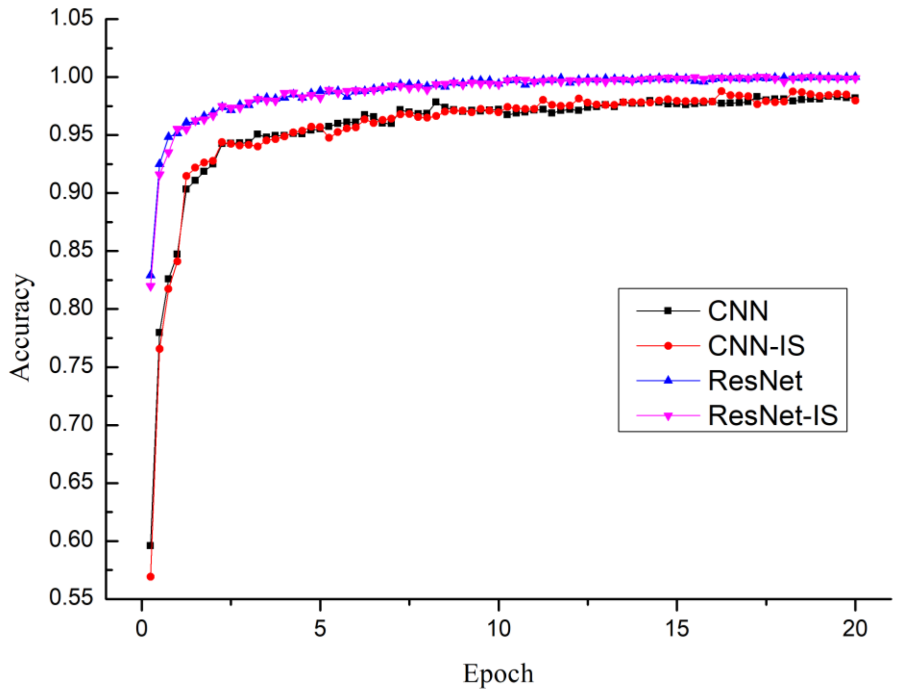


Figure 8.1: Training accuracy with different model and different data

not much different as training continues. We believe that the content of single image data is less, and it is simpler than the data which contained time series information (IS).

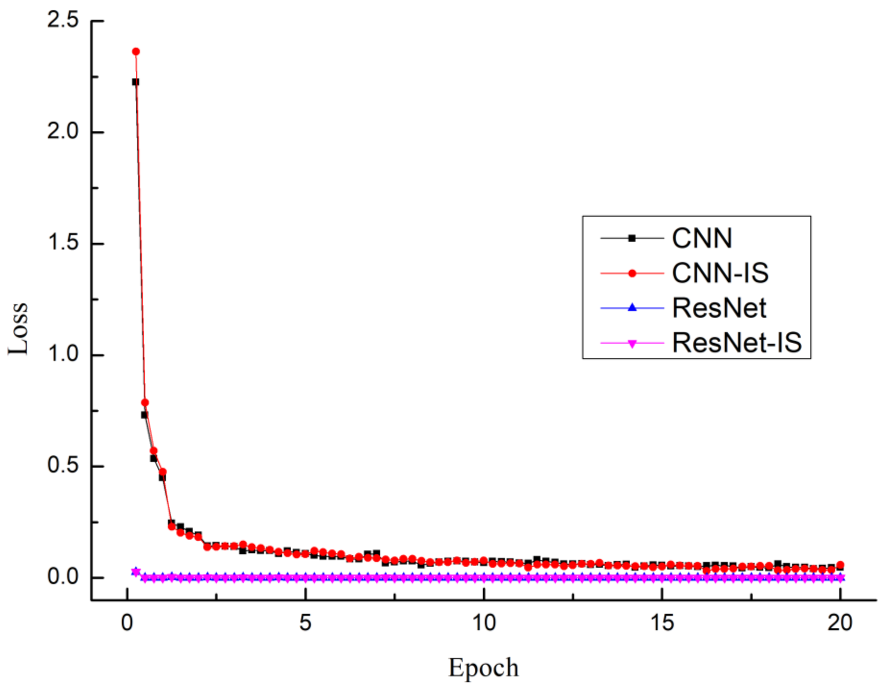


Figure 8.2: Training loss with different model and different data

Therefore, the model training single image set can converge faster. The training losses are shown in Figure 8.2, the training loss of the CNN model is quickly reduced in first two epochs and does not significantly change after the third epoch. However, the training losses of the ResNet model are very small, and there is almost no obvious change in the entire training process. The phenomenon mainly depends on the training model, and the training data has little effect on it.

The training process shows that the ResNet model has better performance with transfer learning. However, the training data and prediction accuracy have not yet been divided, they need to be compared with the verification results to be fully analyzed. After each epoch training, the model will be in eval mode; at this time, all weights of layers no longer change, and predict the validation data and then return the validation accuracy and validation loss. After this, each batch training results and each epoch validation results would be save in the same .csv file.

8.1 Comparison of Results

From the training results, it is known that ResNet has a better performance, so here we will focus on comparing and analyzing the valuation test, which can be found the

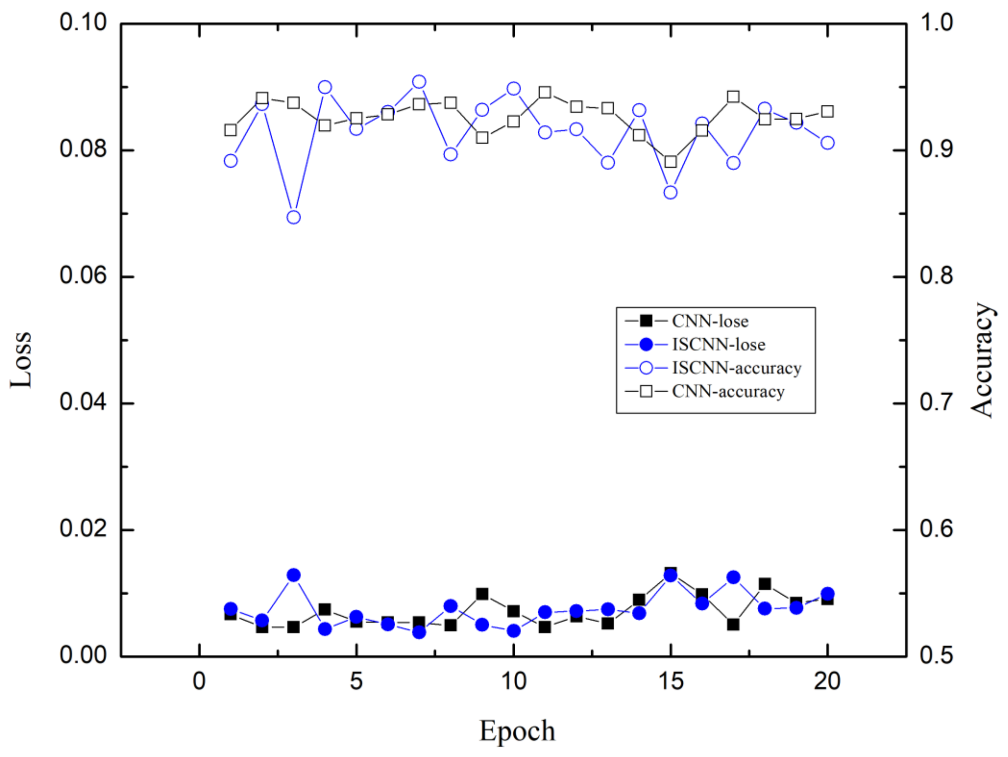


Figure 8.3: The single image data and Image sequence data are trained by CNN

differences of the both input. The single image data and image-sequence data are trained by the CNN which is shown in Figure 8.3. The validation accuracy of CNN with single image data is relatively volatile, the best accuracy is 94.2% in Epoch 17. The validation accuracy of CNN with image sequence data (CNN-IS) has big fluctuations, but the big fluctuations is relative. The accuracy range is between 84.7% and 95%, and the best

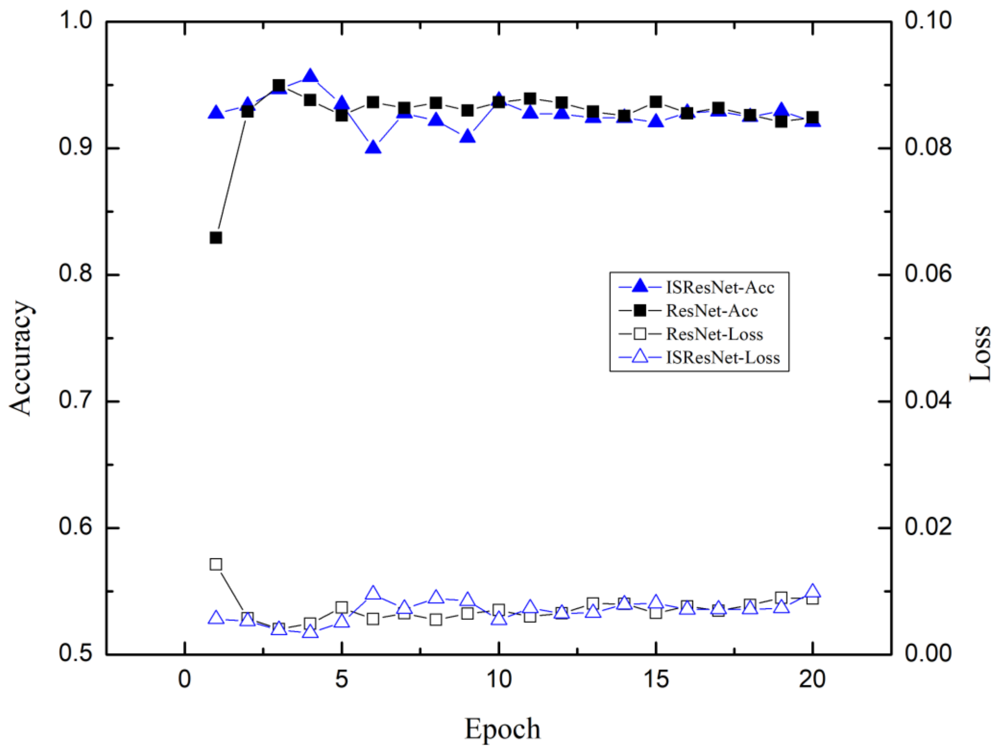


Figure 8.4: The single image data and Image sequence data are trained by ResNet with transfer learning

accuracy is still better than single image data. The best performance appears in Epoch 7.

The Figure 8.4 shows that the results of both datasets are trained by ResNet with transfer learning. Obviously, ResNet is more suitable for complex image data, such as image sequence data. The loss of ResNet with image-sequence data (ISResNet) does not case significant fluctuation in the whole process, and the loss is very small at the beginning. Again, the accuracy of ISResNet is very good from the start. The best accuracy appears in Epoch 4 and the value is 95.6%. Comparing the input data by both the CNN and ResNet, the image-sequence data has better accuracy in both models. This shows that the image sequence has more information, and that dynamic information can be effectively extracted

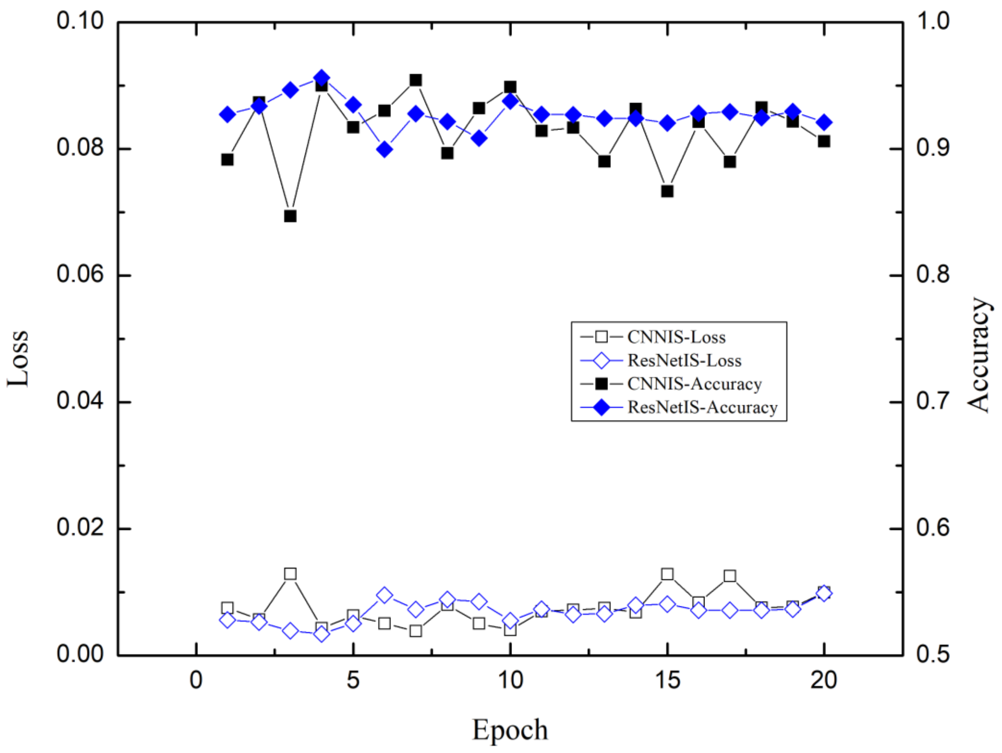


Figure 8.5: The image sequence data are trained by ResNet with transfer learning and CNN

and used by the reasonable model with our data design.

Furemore, we again compare the results of both model with image-sequence data, which is shown in Figure 8.5. It can be observed that the accuracy and loss of ISCNN case more obvious fluctuations. We believe that when the CNN model processes complex image data, because its depth is not enough to fully learn the knowledge from the multiply-channel image, and the network parameters change greatly, so the prediction results are relatively fluctuating.

As compared above, in Epoch 4 the best performance appears where the accuracy is 0.956 and the loss is 0.00344, and the ResNetIS model is saved as the test model. The prediction results by the saved ResNetIS model with the testing data is 94.3% while it is

Table 8.1: Confusion matrix

		Actual class		
		0	1	2
Predicted Class	0	1075	42	0
	1	10	430	102
	2	0	4	1131

92.3% for the model using single image. The testing results are shown in Table 8.1, and the predicting results during the test welding experiment, where the weld penetration increases with time of the arc application, are shown in Figure 8.6. The inaccurate results only occur at the borders of the classes. When compared with Figure 8.7, we find that the test experiment with low current has better prediction results. The possible reason for this is that the intensive arc from the high welding current affects the quality of the captured images. The prediction using the trained model is considered real-time. The prediction process takes 0.0137s, and they system requires 0.024s for the computer to transfer the image sequence to the CNN and for the CNN to output the classification of the weld penetration. It is less than the time of acquiring one image, which is 0.033 s.

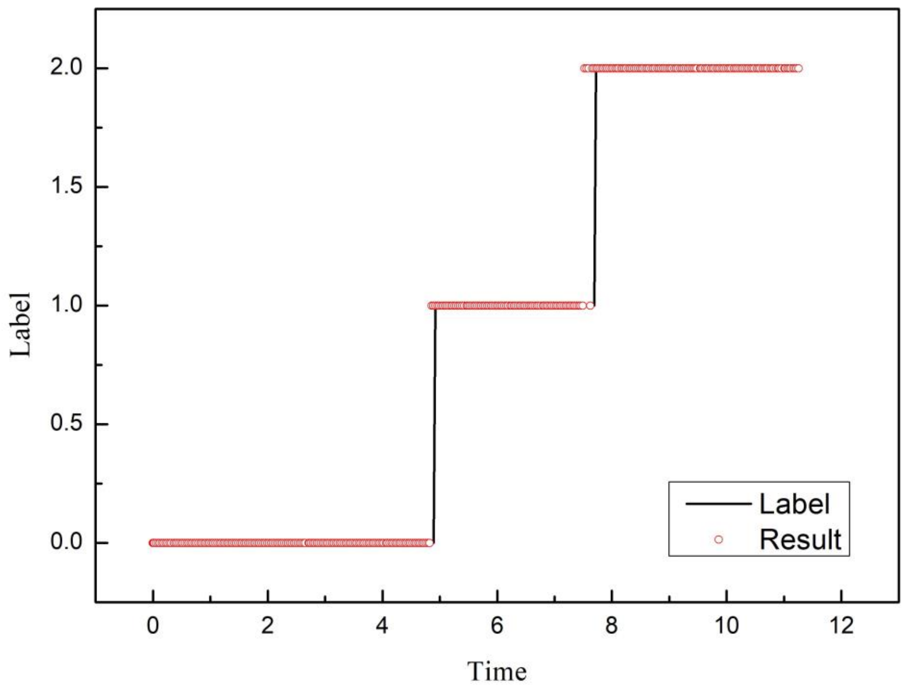


Figure 8.6: The prediction results from the test experiment with 65A

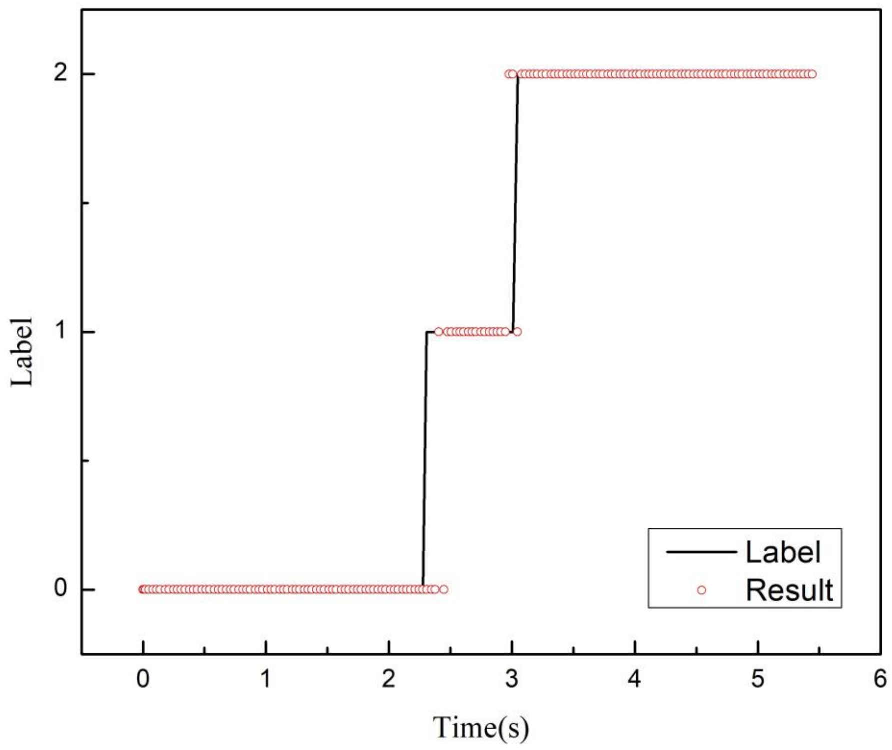


Figure 8.7: The prediction results from the test experiment with 100A

CHAPTER 9. CONCLUSION AND FUTURE WORK

In view of the special and complicated process of welding, we propose a passive vision system to capture the welding process images in real time, this is simpler, and more efficient, and reliable than the previous information collection method. A CNN and ResNet were applied to predict weld penetration based on the top-side image of the weld pool and arc from a passive vision system. The training dataset was produced from experiments under a variety of welding conditions. The weld penetration state was predicted by an end-to-end deep learning approach, and transfer learning was used to improve the training performance. What's more, due to the characteristics of the welding images such as low contrast and darker chroma, we designed merged image data that contained temporal

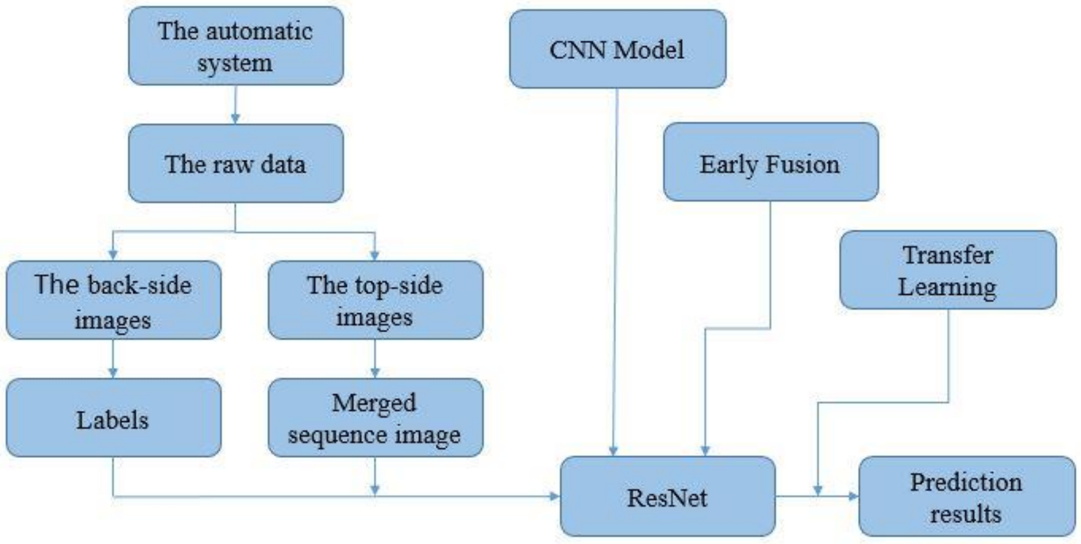


Figure 9.1: The all works of this study.

information. The image sequence was sent to multiple channels in sequence, and the same receptive field was then used to observe the dynamic information from the input image sequence, and the key features were extracted from the dynamic information to analyze and predict the welding penetration state. This method improves prediction accuracy from 92.3% to 94.3%. The all works of this study are shown in Figure 9.1.

9.1 Conclusion

(1) The top-side images where the weld pool and arc are passively imaged contain enough needed information to predict the weld penetration and can be used to directly train a complex prediction model;

(2) The transfer learning method significantly reduces the training difficulty for deep learning, and it can meet the training requirement even with a small amount of training data;

(3) The designed sequence of top-side images can effectively add temporal information and improve the accuracy of the prediction system;

(4) The early fusion approaches can enable the CNN and ResNet to quickly and accurately find the dynamic information in the sequence data;

(5) The whole process of the prediction system costs 0.024s, which is less than the time of acquiring one image. The prediction system meets the real-time requirement.

(6) The data-driven and end-to-end prediction system is an efficient and simple method to predict the weld penetration state from top-side images in the complex welding process.

9.2 Future Work

Now that we have proven the effectiveness and feasibility of the data-driven and end-to-end system, this is also the first attempt to understand and predict welding penetration by analyzing dynamic welding information. Due to the critical role of weld penetration and the negligible impact on system/implementation, this method represents a major progress in the important field of weld penetration monitoring and is expected to provide more significant improvements during welding when using pulsed current where the process becomes highly dynamic. What’s more, this model can be improved to increase the batch size and add the number of layers, such as slow fusion and two-stream fusion method, are worth experimenting with.

REFERENCES

[1] Y. K. Liu and Y. M. Zhang, “Model-Based predictive control of weld penetration in gas tungsten arc welding,” *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 3, pp. 955–966, 2014.

[2] D. S. Nagesh and G. L. Datta, “Prediction of weld bead geometry and penetration in shielded metal-arc welding using artificial neural networks,” *J. Mater. Process. Technol.*, 2002.

[3] B. Chen, J. Wang, and S. Chen, “Prediction of pulsed GTAW penetration status based on BP neural network and D-S evidence theory information fusion,” *Int. J. Adv. Manuf. Technol.*, 2010.

[4] C. Fan, F. Lv, and S. Chen, “Visual sensing and penetration control in aluminum alloy pulsed GTA welding,” *Int. J. Adv. Manuf. Technol.*, 2009.

[5] Y. M. Zhang, Z. N. Cao, and R. Kovacevic, “Numerical analysis of fully penetrated weld pools in gas tungsten arc welding,” *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.*, 1996.

[6] K. Manikya Kanti and P. Srinivasa Rao, “Prediction of bead geometry in pulsed GMA welding using back propagation neural network,” *J. Mater. Process. Technol.*, 2008.

[7] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks,” Dec. 2013.

[8] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “CNN features off-the-shelf: An astounding baseline for recognition,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2014.

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012.

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014.

[11] W. Jiao, Q. Wang, Y. Cheng, and Y. M. Zhang, “End-to-end prediction of weld penetration: A deep learning and transfer learning based method,” *J. Manuf. Process.*, 2020.

[12] Z. Zhang, G. Wen, and S. Chen, “Weld image deep learning-based on-line defects detection using convolutional neural networks for Al alloy in robotic arc welding,” *J. Manuf. Process.*, 2019.

[13] P. P. Thakur and A. N. Chapgaon, “A Review on Effects of GTAW Process Parameters on weld,” *Int. J. Res. Appl. Sci. Eng. Technol.*, 2016.

[14] D. Kotecki, D. Cheever, and D. Howden, "MECHANISM OF RIPPLE FORMATION DURING WELD SOLIDIFICATION," *Weld. J.*, 1972.

[15] R. J. Renwick and R. W. Richardson, "EXPERIMENTAL INVESTIGATION OF GTA WELD POOL OSCILLATIONS.," *Weld. J. (Miami, Fla)*, 1983.

[16] M. Zacksenhouse and D. E. Hardt, "Weld pool impedance identification for size measurement and control," *J. Dyn. Syst. Meas. Control. Trans. ASME*, 1983.

[17] Y. XIAO and G. DEN OUDEN, "Weld pool oscillation during GTA welding of mild steel," *Weld. J.*, 1993.

[18] A. J. R. Aendenroomer and G. Den Ouden, "Weld pool oscillation as a tool for penetration sensing during pulsed GTA welding," *Weld. J. (Miami, Fla)*, 1998.

[19] K. Andersen and G. E. Cook, "Synchronous weld pool oscillation for monitoring and control," *IEEE Trans. Ind. Appl.*, 1997.

[20] D. A. Hartman, D. R. DeLapp, G. E. Cook, and R. J. Barnett, "Intelligent fusion control throughout varying thermal regions," in *Conference Record - IAS Annual Meeting (IEEE Industry Applications Society)*, 1999.

[21] J. Ju, Y. Suga, and K. Ogawa, "Penetration control by monitoring molten pool oscillation in TIG arc welding," *Int. J. Offshore Polar Eng.*, 2004.

[22] B. Y. B. Yudodibroto, M. J. M. Hermans, Y. Hirata, and G. Den Ouden, "Influence of filler wire addition on weld pool oscillation during gas tungsten arc welding," *Sci. Technol. Weld. Join.*, 2004.

[23] M. Miller, B. Mi, A. Kita, and I. C. Ume, "Development of automated real-time data acquisition system for robotic weld quality monitoring," *Mechatronics*, 2002.

[24] N. M. Carlson and J. A. Johnson, "Ultrasonic sensing of weld pool penetration," *Weld. J. (Miami, Fla)*, 1988.

[25] R. Fenn, "ULTRASONIC MONITORING AND CONTROL DURING ARC WELDING.," *Weld. J. (Miami, Fla)*, 1985.

[26] D. E. Hardt and J. M. Katz, "ULTRASONIC MEASUREMENT OF WELD PENETRATION.," *Weld. J. (Miami, Fla)*, 1984.

[27] G. M. Graham and I. C. Ume, "Automated system for laser ultrasonic sensing of weld penetration," *Mechatronics*, 1997.

[28] B. Mi and C. Ume, "Real-time weld penetration depth monitoring with laser ultrasonic sensing system," *J. Manuf. Sci. Eng. Trans. ASME*, 2006.

[29] J. D. Aussel, A. Le Brun, and J. C. Baboux, "Generating acoustic waves by laser: theoretical and experimental study of the emission source," *Ultrasonics*, 1988.

[30] "Monitoring joint penetration using infrared sensing techniques," *Weld. J.*, 1990.

[31] S. Nagarajan, P. Banerjee, W. H. Chen, and B. A. Chin, "Control of the Welding Process Using Infrared Sensors," *IEEE Trans. Robot. Autom.*, 1992.

[32] H. C. Wikle, S. Kottilingam, R. H. Zee, and B. A. Chin, “Infrared sensing techniques for penetration depth control of the submerged arc welding process,” in *Journal of Materials Processing Technology,* 2001.

[33] D. Barborak, C. Conrardy, B. Madigan, and T. Paskell, “‘Through-arc’ process monitoring techniques for control of automated gas metal arc welding,” *Proc. - IEEE Int. Conf. Robot. Autom.*, 1999.

[34] B. J. Corlett, J. Lucas, and J. S. Smith, “Sensors for narrow-gap welding,” *IEE Proc. A Phys. Sci. Meas. Instrumentation. Manag. Educ. Rev.,* 1991.

[35] J. Wang, K. Kusumoto, and K. Nezu, “Microweld penetration monitoring techniques by arc sensing,” in *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM,* 2003.

[36] A. C. Guu and S. I. Rokhlin, “Computerized radiographic weld penetration control with feedback on weld pool depression,” *Mater. Eval.,* 1989.

[37] S. I. Rokhlin and A. C. Guu, “Computerized radiographic sensing and control of an arc welding process,” *Weld. J. (Miami, Fla),* 1990.

[38] Y. M. Zhang, H. S. Song, and G. Saeed, “Observation of a dynamic specular weld pool surface,” in *Measurement Science and Technology,* 2006.

[39] H. S. Song and Y. M. Zhang, “Three-dimensional reconstruction of specular surface for a gas tungsten arc weld pool,” *Meas. Sci. Technol.,* 2007.

[40] H. S. Song and Y. M. Zhang, “Measurement and analysis of three-dimensional specular gas tungsten arc weld pool surface,” *Weld. J. (Miami, Fla),* 2008.

[41] H. Song and Y. M. Zhang, “Error analysis of a three-dimensional GTA weld pool surface measurement system,” *Weld. J.,* 2009.

[42] W. Zhang, X. Wang, and Y. Zhang, “Analytical real-time measurement of a three-dimensional weld pool surface,” *Meas. Sci. Technol.,* 2013.

[43] W. J. Zhang, X. Zhang, and Y. M. Zhang, “Robust pattern recognition for measurement of three dimensional weld pool surface in GTAW,” *J. Intell. Manuf.,* 2015.

[44] D. B. Zhao, J. Q. Yi, S. B. Chen, L. Wu, and Q. Chen, “Extraction of three-dimensional parameters for weld pool surface in pulsed GTAW with wire filler,” *J. Manuf. Sci. Eng. Trans. ASME,* 2003.

[45] R. Kovacevic and Y. M. Zhang, “Sensing free surface of arc weld pool using specular reflection: Principle and analysis,” *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.,* 1996.

[46] R. Kovacevic and Y. M. Zhang, “Real-Time Image Processing for Monitoring of Free Weld Pool Surface,” *J. Manuf. Sci. Eng. Trans. ASME,* 1997.

[47] Y. M. Zhang, E. Liguo, and R. Kovacevic, “Active metal transfer control by monitoring excited droplet oscillation,” *Weld. J. (Miami, Fla),* 1998.

[48] G. Saeed and Y. M. Zhang, “Mathematical formulation and simulation of specular reflection based measurement system for gas tungsten arc weld pool surface,” *Meas. Sci. Technol.*, 2003.

[49] G. Saeed, M. Lou, and Y. M. Zhang, “Computation of 3D weld pool surface from the slope field and point tracking of laser beams,” *Meas. Sci. Technol.*, 2004.

[50] G. Saeed, Y. M. Zhang, and S. Cook, “A compact vision sensor for weld pool surface sensing,” *Int. J. Model. Identif. Control*, 2006.

[51] G. Saeed and Y. M. Zhang, “Weld pool surface depth measurement using a calibrated camera and structured light,” *Meas. Sci. Technol.*, 2007.

[52] C. M. Mnich, “DEVELOPMENT OF A SYNCHRONIZED, HIGH-SPEED, STEREOVISION SYSTEM FOR IN SITU WELD POOL MEASUREMENT.”

[53] X. Ma and Y. Zhang, “Gas metal arc weld pool surface imaging: Modeling and processing,” *Weld. J.*, 2011.

[54] Z. Chen, J. Chen, and Z. Feng, “Welding penetration prediction with passive vision system,” *J. Manuf. Process.*, vol. 36, pp. 224–230, Dec. 2018.

[55] T. S. Huang, Ed., *Image Sequence Analysis*, vol. 5. Berlin, Heidelberg: Springer Berlin Heidelberg, 1981.

[56] *Motion Analysis and Image Sequence Processing*. Springer US, 1993.

[57] X. Liu, Z. Deng, and Y. Yang, “Recent progress in semantic image segmentation,” *Artif. Intell. Rev.*, 2019.

[58] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, “Deep neural networks segment neuronal membranes in electron microscopy images,” in *Advances in Neural Information Processing Systems*, 2012.

[59] L. Deng and D. Yu, “Deep Learning: Methods and Applications Foundations and Trends R in Signal Processing,” *Signal Processing*, 2013.

[60] D. C. Cireşan, U. Meier, J. Masci, L. M. Gambardella, and J. Schmidhuber, “Flexible, high performance convolutional neural networks for image classification,” in *IJCAI International Joint Conference on Artificial Intelligence*, 2011.

[61] K. Fukushima, “Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position,” *Biol. Cybern.*, 1980.

[62] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, “What is the best multi-stage architecture for object recognition? BT - Computer Vision, 2009 IEEE 12th International Conference on,” in *Computer Vision, 2009 …*, 2009.

[63] Y. LeCun, K. Kavukcuoglu, and C. Farabet, “Convolutional networks and applications in vision,” in *ISCAS 2010 - 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, 2010.

[64] D. Scherer, A. Müller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* 2010.

[65] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” *J. Big Data,* 2015.

[66] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “A survey of deep neural network architectures and their applications,” *Neurocomputing,* 2017.

[67] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, “Deep learning for visual understanding: A review,” *Neurocomputing,* 2016.

[68] Q. Abbas, M. E. A. Ibrahim, and M. A. Jaffar, “A comprehensive review of recent advances on deep vision systems,” *Artif. Intell. Rev.*, 2019.

[69] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[70] D. H. Hubel and T. N. Wiesel, “Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex,” *J. Physiol.*, 1962.

[71] D. H. Hubel and T. N. Wiesel, “Receptive fields and functional architecture of monkey striate cortex,” *J. Physiol.,* 1968.

[72] Y. Bengio, “Learning deep architectures for AI,” *Found. Trends Mach. Learn.*, 2009.

[73] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics),* 2014.

[74] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[75] C. Szegedy *et al.*, “Going deeper with convolutions,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015.

[76] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.

[77] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017,* 2017.

[78] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, inception-ResNet and the impact of residual connections on learning,” in *31st AAAI Conference on Artificial Intelligence, AAAI 2017,* 2017.

[79] S. Zagoruyko and N. Komodakis, “Wide Residual Networks,” in *British Machine Vision Conference 2016, BMVC 2016*, 2016.

[80] J. Bouvrie, “Notes on convolutional neural networks,” *In Pract.*, 2006.

[81] N. Ketkar, *Deep Learning with Python*. 2017.

[82] “Pattern Recognition and Machine Learning,” *J. Electron. Imaging*, 2007.

[83] Y. Bengio, “Deep learning of representations: Looking forward,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013.

[84] G. Montúfar, R. Pascanu, K. Cho, and Y. Bengio, “On the number of linear regions of deep neural networks,” in *Advances in Neural Information Processing Systems*, 2014.

[85] “Spring Research Presentation | College of Physical and Mathematical Sciences.” [Online]. Available: https://web.archive.org/web/20070801120743/http://cpms.byu.edu/springresearch/abstract-entry?id=861. [Accessed: 27-Nov-2019].

[86] T. George Karimpanal and R. Bouffanais, “Self-organizing maps for storage and transfer of knowledge in reinforcement learning,” *Adapt. Behav.*, vol. 27, no. 2, pp. 111–126, Apr. 2019.

[87] X. Shi, Z. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo, “Convolutional LSTM network: A machine learning approach for precipitation nowcasting,” in *Advances in Neural Information Processing Systems*, 2015.

[88] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, “CNN-RNN: A Unified Framework for Multi-label Image Classification,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.

[89] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, 1997.

[90] C. L. Giles, C. B. Miller, D. Chen, H. H. Chen, G. Z. Sun, and Y. C. Lee, “Learning and Extracting Finite State Automata with Second-Order Recurrent Neural Networks,” *Neural Comput.*, 1992.

[91] C. Cortes and V. Vapnik, “Support-Vector Networks,” *Mach. Learn.*, 1995.

[92] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders, “Early versus Late Fusion in Semantic Video Analysis,” 2005.

[93] Xiaotao Zou and B. Bhanu, “Tracking Humans using Multi-modal Fusion,” 2006.

[94] J. K. Aggarwal and Q. Cai, “Human Motion Analysis: A Review,” *Comput. Vis. Image Underst.*, 1999.

[95] J. Y. H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G.

Toderici, “Beyond short snippets: Deep networks for video classification,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015.

[96] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and F. F. Li, “Large-scale video classification with convolutional neural networks,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014.

[97] A. Karpathy, “CS231n Convolutional Neural Networks for Visual Recognition,” *Stanford Univ.*, 2016.

[98] M. Mathieu, C. Couprie, and Y. LeCun, “Deep multi-scale video prediction beyond mean square error,” in *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[99] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, and Z. Wang, “Multi-class Generative Adversarial Networks with the L2 Loss Function,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[100] D. M. Harris and S. L. Harris, *Digital design and computer architecture, 2nd edition*. 2012.

[101] L. Bottou and O. Bousquet, “The tradeoffs of large scale learning,” in *Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference*, 2009.

[102] “Optimization methods.” [Online]. Available: https://datascience-enthusiast.com/DL/Optimization_methods.html. [Accessed: 05-Apr-2020].

[103] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

[104] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” in *COLT 2010 - The 23rd Conference on Learning Theory*, 2010.

[105] G. E. Hinton, N. Srivastava, and K. Swersky, “Lecture 6.5- Divide the gradient by a running average of its recent magnitude,” *COURSERA Neural Networks Mach. Learn.*, 2012.

[106] D. Mishkin, N. Sergievskiy, and J. Matas, “Systematic evaluation of convolution neural network advances on the Imagenet,” *Comput. Vis. Image Underst.*, 2017.

[107] L. N. Smith, “Cyclical learning rates for training neural networks,” in *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, 2017.

[108] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *32nd International Conference on Machine Learning, ICML 2015*, 2015, vol. 1, pp. 448–456.

[109] J. Collis, “Glossary of Deep Learning: Batch Normalisation,” *Medium*, 2017. .

[110] A. L. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” in *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

VITA

1. Wenhua Jiao, Qiyue Wang, Yongchao Cheng, YuMing Zhang, 2020 “End-to-end prediction of weld penetration: A deep learning and transfer learning based method” Journal of Manufacturing Processes, DOI: 10.1016/j.jmapro.2020.01.044
2. Wenhua Jiao, YuMing Zhang, 2020 “Real-time Prediction of weld penetration by image sequence” Welding Journal (In processing)
3. Qiyue Wang, Wenhua Jiao, Rui Yu, Michael T Johnson, YuMing Zhang. 2019 "Modeling of Human Welders’ Operations in Virtual Reality Human-Robot Interaction" IEEE Robotics and Automation Letters, DOI: 10.1109/LRA.2019.2921928
4. Qiyue Wang, Wenhua Jiao, Rui Yu, Michael T Johnson, YuMing Zhang. 2019 “Virtual Reality Robot-Assisted Welding Based on Human Intention Recognition” IEEE Transactions on Automation Science and Engineering, DOI: 10.1109/TASE.2019.2945607
5. Qiyue Wang, Wenhua Jiao, Peng Wang, Yuming Zhang. 2020 “A tutorial on deep learning-based data analytics in manufacturing: Through a welding case study” Journal of Manufacturing Processes,
6. Qiyue Wang, Yongchao Cheng, Wenhua Jiao, Michael T Johnson, YuMing Zhang. 2019 “Virtual reality human-robot collaborative welding: A case study of weaving gas tungsten arc welding” Journal of Manufacturing Processes, DOI: 10.1016/j.jmapro.2019.10.016
7. Yongchao Cheng, Qiyue Wang, Wenhua Jiao, Rui Yu, Shujun Chen, YuMing Zhang, Jun Xiao. 2020 “Detecting Dynamic Development of Weld Pool Using Machine Learning from Innovative Composite Images for Adaptive Welding” Journal of Manufacturing Processes,