Automatic Identification of 2D Materials Based on Machine Learning Method

by

Quanjin Wang

Department of Physics

Bachelor of Arts

University of Colorado Boulder

Committee Members:

Dr. Daniel Dessau - Thesis Advisor

Department of Physics

Dr. John Cumalat - Honors Council Representative

Department of Physics

Dr. Sean Shaheen - Outside Reader

Department of Electrical, Computer & Energy Engineering

April 8, 2024

Abstract

As highly anticipated quantum materials of low dimensionality, 2D materials have recently seen significant advances in their fabrication, characterization, modification, and application in research. The research on 2D materials is fascinating, and one of the most attractive directions relates to the creation of heterostructures with on-demand and unique quantum properties through the stacking of various 2D materials. However, finding and identification of 2D materials are challenging as the essential part of 2D material research. Although we can artificially estimate the number of 2D materials layers based on optical microscopy (OM), looking for sufficient and suitable 2D materials for further research is still time-consuming and labor-intensive. Given the excellent performance of machine learning in the field of computer vision, we design a method to rapidly and accurately identify 2D materials upon OM. We utilize the architecture of the Mask Region-Based Convolutional Neural Network (Mask R-CNN) to complete the model training with datasets of 2D materials images. Additionally, we optimize hyperparameters to maximize the evaluation results of the trained models. Subsequently, we successfully apply the trained models to predict optical images of 2D materials, outputting the prediction image with bounding boxes (position), segmentations (shape), and categories (monolayer, few-layers, and thick-layers). Based on the feasibility of models and algorithms, we program software for 2D materials identification for universal and convenient usage. Finally, we design a Machine-Learning-Based 2D Material Transfer System with a system software, that

successfully achieves real-time identification of 2D materials, processes large batches of images, and realizes precise control of the motorized stage through simple usage of the system software. This thesis is upgradeable; we also discuss potential development directions in datasets, the function of 2D materials auto scanning, and different architectures of machine learning.

Acknowledgments

Firstly, I would like to thank Professor Daniel Dessau for his kindness, patience, and invaluable assistance as my advisor for 3 years. I deeply appreciate his willingness to provide me with the opportunity to obtain a position in the research and offer strong academic support.

I also wish to thank the members of the Dessau's Group, with a special mention of Mr. Zack Sierzega. Zack has offered many valuable suggestions and guidance throughout this thesis and has become a good friend in my life.

I would like to express my deep gratitude to my parents, whose support, enlightenment, and upbringing have been foundational to my life and academic journey. Their endless dedication has fueled my pursuit of dreams, and I am eternally grateful for everything they have done.

Finally, I profoundly appreciate Yanyun, my beloved, whose unwavering encouragement and hope have been power for me to move forward. I am immensely thankful for her irreplaceable tenderness and love, which complete my life and make me shine.

Contents

Chapter 1

Introduction

Two-dimensional (2D) materials are crystalline solids composed of single or few layers of atoms or molecules. These materials exhibit substantial dimensions in two directions (width and length) compared to their thickness in the third dimension. The unique structural characteristic of 2D materials endows them with special and excellent optical, electrical, and mechanical properties, which cannot be found in their bulk structures. Serving as quintessential examples of low-dimensional quantum materials, monolayers—or specifically, the single-layer configurations of 2D materials—are the most considered and focal variety in fabrication, characterization, modification, theoretical calculation, and exploration of application around condensed matter physics. This focus is not solely due to their pronounced quantum effects and adaptability to simulations but also relates to physicists' pursuit of the ultimate limit of materials' thinness.

Since graphene, the first discovered monolayer of graphite, was produced by simple mechanical exfoliation using tape in 2004, the exploration and discovery of various monolayers have experienced explosive growth over the past 20 years, their category span across transition metal dichalcogenides, organic materials, magnetic materials, and more, exhibiting a range of electronic properties including insulators, semiconductors, and even superconductors. One of the monolayers' most fascinating characteristics is that the interlayers' Van der Waals interactions allow the flexible stacking of different

monolayers to build up 2D structures manually. According to the consideration of lattice matching and symmetry, strategically engineered stacking displacements and relative angles between monolayers can profoundly modify their original electronic characteristics. A prime illustration is the “magic angle” of 1.1 degrees in bilayer graphene, which induces superconductivity, showcasing the profound impact of interlayer orientation on material properties. Additionally, the assembly of monolayers reveals more complex underlying theories and phenomena, such as interlayer interactions, moiré patterns, and a spectrum of quantum effects and topological concepts. The variability in properties and band gaps across different monolayers, when stacked to form heterostructures, offers researchers innovative avenues for designing materials with bespoke, superior qualities tailored to specific applications [1]. The practical implications of research on monolayers are vast, leading to significant enhancements in optical and electrical devices, microcircuits, and quantum technologies.

In the wide field of 2D materials research, finding a monolayer is the initial and essential step. Therefore, developing a universal methodology for the rapid and precise identification of monolayers is important, which could facilitate high quality, high efficiency, and standardization in 2D materials research.

1.1 Microscopy in 2D Materials Research

There are several methods to identify the thickness of 2D materials in research, the two most outstanding methods are atomic force microscopy (AFM) and optical microscopy (OM). AFM is a powerful and widely applied technique for characterizing micro-samples in different scientific fields. The fundamental principle of AFM is touching the sample surface with a finely tuned mechanical

probe and moving. Simultaneously, a laser beam is directed at the probe to detect any tiny motion of it, and the reflected light will be captured as data for analysis. Then, this data is then utilized to generate high-resolution images and three-dimensional models of the samples by computer. Therefore, researchers can utilize AFM to obtain a precise and reliable measurement of the thickness of 2D materials and verification of monolayers. However, AFM is not suitable for the identification of monolayers because it requires significant time and skill to operate. Additionally, AFM is only capable of measuring small areas of samples due to the limit of the movement of the tiny probe.

On the contrary, OM is a simple, efficient, and nondestructive technique that enables rapid characterization of 2D materials over a large area [2]. The fundamental principle of OM depends on observation, just using a digital microscope with a computer to view the optical image of 2D materials. Monolayers usually appear translucent and have low color saturation compared to thick layers. Based on the analysis of color contrast values, we can distinguish the number of layers in a large area of samples. Moreover, researchers can achieve quick estimation of monolayers by observing the optical images through short-term training. Figure 1.1 illustrates the comparison of the identification of graphene by OM and AFM. Within the red dashed rectangle in Figure 1.1a and c, the measurements by AFM respectively are 0.4 nm, 1.7 nm, and 2.1 nm, which are relevantly corresponding to 1 layer, 5 layers, and 6 layers measured by OM. As a result, OM has emerged as a universal, popular, and dependable method in the identification of monolayers in current 2D materials research, not only because of the characteristics of high-speed and large-area scanning but also the adaptability to most

laboratories due to its relaxed demand in instruments.

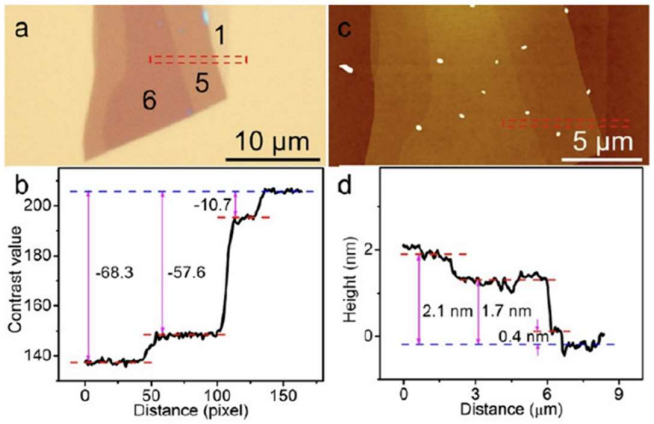


Figure 1.1: Optical (a) and AFM height (c) images of the graphene flake on the silicon chip. Corresponding to contrast value plot (b) and thickness measurement (d). Adapted from [2].

Nonetheless, in the real case of 2D materials research, the identification of monolayers becomes complicated and challenging. Here, the optical images of MoS_2 and graphite flakes on SiO_2/Si substrates, which were captured by the digital microscope, are shown (See Figure 1.2). It is obvious to

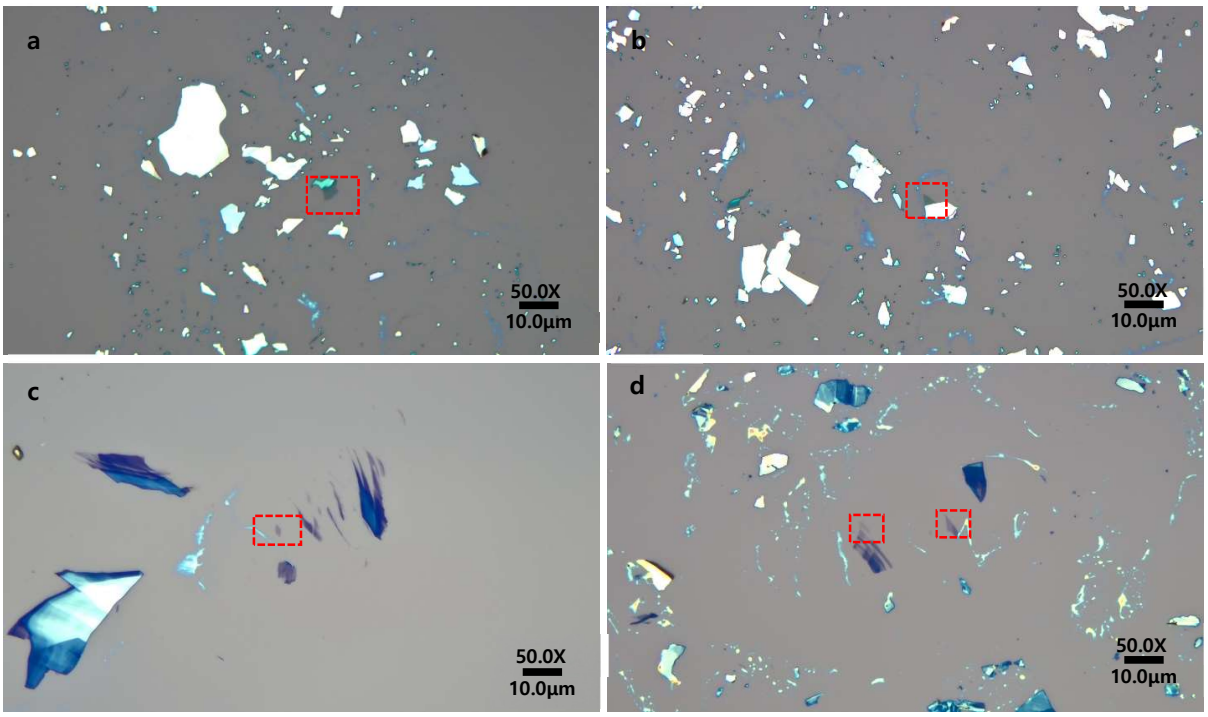


Figure 1.2: Optical images of MoS_2 flakes (a and b) and optical images of graphite flakes (c and d). The corresponding monolayers are within the read dashed rectangles.

see that there are numerous irregularly shaped flakes in the view, most of them are too thick to be utilized as 2D materials. Within the areas marked by red dashed rectangles, monolayer flakes can be discerned. Although mechanical exfoliation can yield monolayers at a modest rate (about 5 – 10 flakes of monolayers per substrate), they are separately distributed across the silicon substrates. Researchers must carefully scan every region of the silicon substrate through the fine-tuning of the microscope's stage. This process is fraught with the risk of overlooking such tiny and inconspicuous monolayers among the multitude of flakes, especially during long-time searching periods. In addition, some of the discovered monolayers are too small to be selected for stacking heterostructures, researchers need to spend several days or even weeks to find sufficient and suitable monolayers for further research. This manual search for monolayers using OM is time-consuming and labor-intensive, significantly dragging down the efficiency of research endeavors.

To avoid the excessive consumption of time and manpower, and the unnecessary waste of materials because of missing monolayers, we are eager to design a solution to upgrade OM to standardize a convenient, efficient, and accurate methodology for finding monolayers. This improvement will allow researchers to dedicate their focus to the study of monolayers rather than the repetitive and tedious task of searching for monolayers.

1.2 2D Materials Transfer System

After decades of progress in 2D materials engineering and research, a mature, reliable, and flexible 2D material transfer system has been established. Basically, the transfer system is designed based on OM, which includes a digital microscope, a high-precision stage (move in XYZ direction and θ rotation), a heating station, and a computer with a display (See Figure 1.3). To meet specific research requirements, the system sometimes incorporates a floating table to minimize vibrations during delicate operations and a glove box enclosure to protect sensitive and easily degradable samples.

Figure 1.3: Photograph of a standard 2D materials transfer system setup. Image link: https://www.hq2d.com/

Enhancements and modifications to the 2D materials transfer process are increasingly prevalent. For instance, researchers have designed to replace costly digital microscopes by employing more accessible alternatives, such as zoomable cameras, optical columns, and lenses. Additionally, the adoption of universal storage containers for constructing makeshift gloveboxes facilitates the study of various 2D materials under limited laboratory conditions. Despite the varied scenarios and objectives that these innovations address within 2D materials research, they share a unified goal: to achieve a methodology that is convenient, efficient, cost-effective, user-friendly, and high-performing [3, 4].

This orientation of system designing is advantageous to reduce the entry barrier to the 2D materials field and encourage the widespread popularization of 2D materials research.

Considering the principles of universality and adaptability in 2D materials transfer systems, the machine learning method becomes a suitable and anticipated solution to enhancing OM. As a software-based strategy, the machine learning method can be integrated into any hardware system equipped with a computer. This compatibility provides the flexibility for further upgrading and adjustment. Furthermore, it provides opportunities to excavate the fascinating and powerful collaboration between machine learning and 2D materials research, extending to more physics-centered interdisciplinary research.

1.3 Machine Learning

With the exponential growth in computational capabilities, alongside the progress in algorithms and architectures, machine learning (ML) has seen remarkable development in recent years, evolving from theoretical concepts to practical applications across different fields. True to its name, ML enables computers to learn through experience and data, automatically improving the ability to complete tasks and refine problem-solving algorithms for inferential outcomes.

Physics, a discipline strongly linked to analysis, calculation, and simulation, is increasingly recognizing the potential of ML. Although the conception and application of ML in physics are in their infancy, physicists have begun to make some pioneering efforts on complex physical systems, quantum computing, and the prediction of material properties by applying ML [5]. As the continuous evaluation of ML, its influence on and integration with physics are anticipated to deepen, including theory and

experiments. Exploring possible applications of ML in physics is meaningful and rewarding, which not only provides a novel method to study physics efficiently and effectively but also facilitates both fields of physics and ML to achieve a win-win situation.

The task of identifying 2D materials represents a typical challenge of analysis and inference, which aligns perfectly with the specialty of ML. Also, the field of computer vision task, derived by ML, has realized significant achievements in speed, multi-function, and accuracy over the years. This outcome emphasizes the feasibility of applying ML to improve OM in 2D materials research.

1.3.1 Convolutional Neural Network

Artificial Neural Network (ANN), as the cornerstone of deep learning, has become one of the most popular and successful models in the region of ML. Inspired by the human brain's working principle and mimicking connections observed in biological synapses, an extensive network of nodes is established, interconnected via weights to facilitate data transmission (See Figure 1.4). These

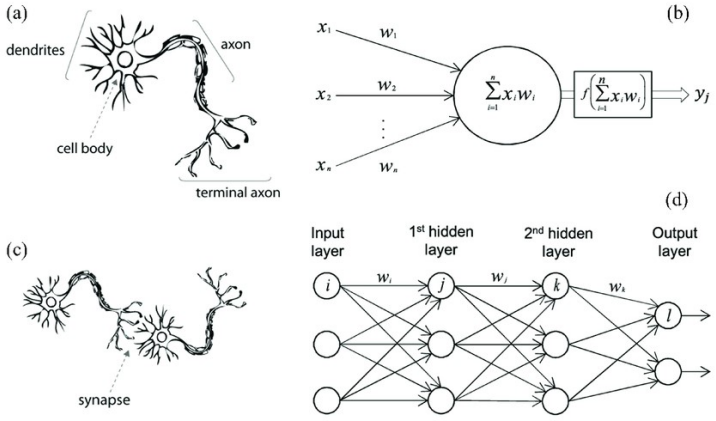


Figure 1.4: Diagram of comparison between biological neurons and artificial neural network. (a, b) are single neuron and artificial neuron; (c, d) are biological synapse and ANN. Adapted from [6].

weights continuously update throughout the extensive and repetitive training process with provided datasets, until they form a sophisticated network that connects inputs and outputs. These trained weights enable the computer to understand and interpret the underlying relationship between questions and answers to complete the assigned task [6].

To address various computer tasks, diverse ANNs have been developed. Among them, Convolutional Neural Networks (CNN) have excellent performance in the field of computer vision, including vision analysis, recognition, and generation. The convolutional layer is the key component of CNNs, which is responsible for feature extraction, including edges and textures of objects in the input image. This process merits a thorough understanding.

An input image can be treated as a matrix of pixels. According to the RGB channels, the input image is converted into a tensor (a collection of matrices). Each of these matrices will be processed by a small $(n \times n)$ matrix, known as the weight or convolutional kernel. The convolutional kernel will systematically "scan" the image matrix, performing dot product between $(n \times n)$ segments of the image matrix and the convolutional kernel to calculate new single values. After the convolutional kernel "scans" the entire image matrix, the aggregated values compose a new, smaller matrix termed feature map. With the addition of convolutional layers, the convolutional kernel keeps refining and becomes increasingly relevant to generate meaningful and smaller feature maps that contain the information and features of the objects in the original image.

The operation of convolutional primarily is linear algebra, which can be expressed by the equation below:

$$Z^{l+1}(i,j) = [Z^l \otimes w^{l+1}](i,j) + b = \sum_{k=1}^{K_l} \sum_{x=1}^{f} \sum_{y=1}^{f} \left[Z_k^l(s_0 i + x, s_0 j + y) w_k^{l+1}(x,y) \right] + b \qquad (1.1)$$

Here, Z^l and Z^{l+1} respectively represent the input matrix (image) and output matrix (feature map). The convolutional kernel w and the bias vector b are automatically adjusted during the operation, which plays a crucial role in influencing the efficacy of feature extraction.

The rectified linear unit (ReLU) layer is the activation function in CNNs. It determines whether a neuron should be activated, that is, to transmit information to the next layer of the network. Different from the linear calculation in convolutional layers, ReLU layers introduce necessary non-linear factors to CNNs. Within the domain of the ReLU layer, $f(x) = 0$ is applied for any negative input x, and $f(x) = x$ for any non-negative input, as shown in Equation 1.2. This simple but effective strategy significantly enhances the complexity and expressive power of networks for understanding complicated patterns and relationships.

$$f(x) = \max(0, x) \tag{1.2}$$

The max pooling layer is utilized to further highlight the significant features (greater values), and reduce redundant features (smaller values). Equation 1.3 gives an example of max pooling processing:

$$\begin{bmatrix} 0 & 2 & 6 & 5 \\ 7 & 5 & 1 & 3 \\ 3 & 1 & 8 & 6 \\ 2 & 5 & 1 & 2 \end{bmatrix} \xrightarrow{Max\ Pooling} \begin{bmatrix} 7 & 6 \\ 5 & 8 \end{bmatrix} \tag{1.3}$$

In this process, we convolve a 2×2 kernel over a 4×4 input feature map to generate a new 2×2 feature map. This operation realizes downsampling while continuously extracting features at the same time, which reduces the amount of data for analysis and speeds up the convergence of the model.

Approaching the end of processing, the dimension of feature maps has been sufficiently reduced, and the number of feature maps has been increased through the repeated operations by three kinds of

layers above. Subsequently, feature maps can be flattened into a one-dimensional vector. The classification of objects in the image is operated by the fully connected layers, which integrate the features extracted from previous layers and allocate the weights, helping the SoftMax activation function to realize the final probabilistic distribution [7]. The architecture of CNNs in this thesis is depicted in Figure 1.5, which provides a visual representation of the discussed concepts.

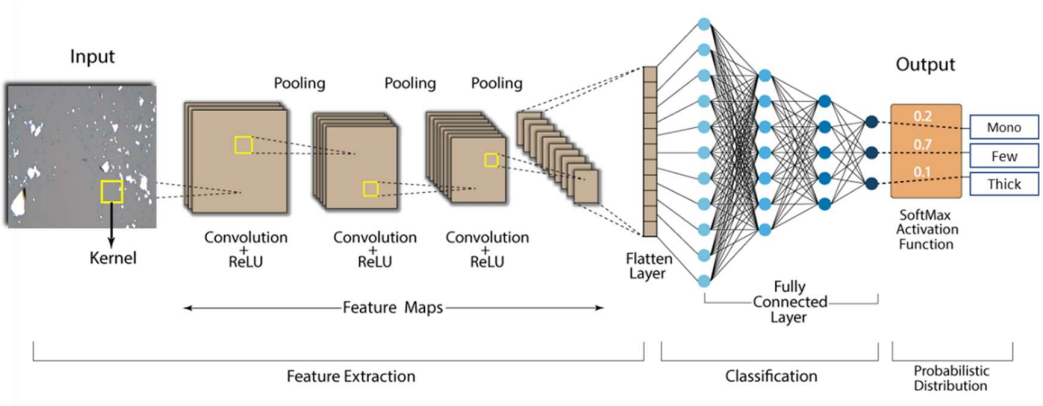


Figure 1.5: Diagram of CNN's architecture for identification of the optical 2D materials image.

1.3.2 Mask Region-Based Convolutional Neural Network

The Mask region-based convolutional neural network (Mask R-CNN) is a successful and powerful extension with the CNN as a backbone network. Its capabilities have been recognized due to applications in various visual fields. Mask R-CNN has excellent performance in realizing object detections, classifications, and pixel-level instance segmentations, which can generate high-quality segmentation masks for each object to indicate its specific location and shape [8]. These outstanding functionalities are perfectly in line with the objective of this thesis, which is exploring a method to detect and locate monolayers in optical images and videos. Therefore, Mask R-CNN is the most suitable choice of architecture for training models and detecting monolayers in this thesis. The

architecture of Mask R-CNN in this thesis is shown in Figure 1.6.

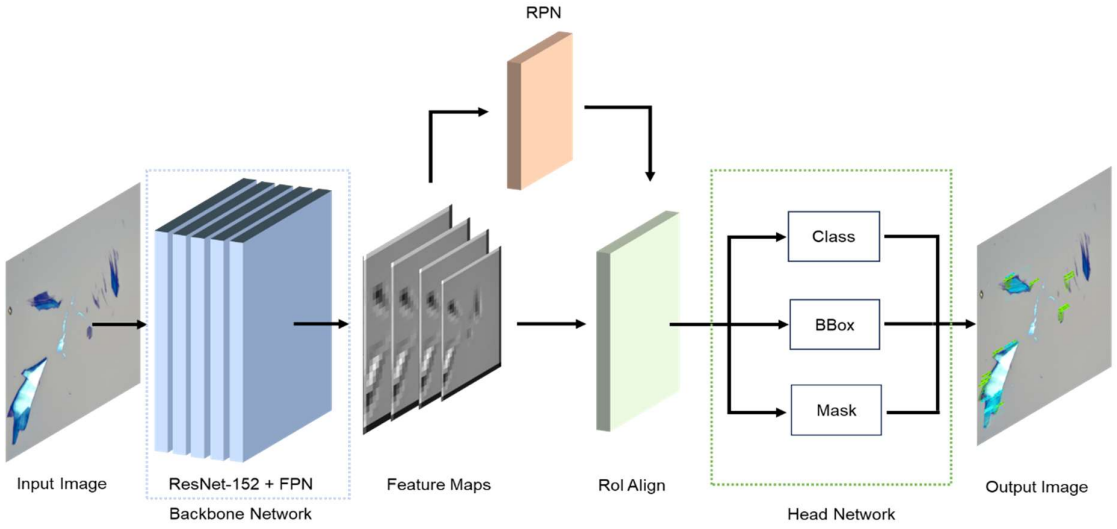


Figure 1.6: Architecture of Mask R-CNN for classification and segmentation of 2D materials

Remarkably, the backbone network in Mask R-CNN is an advanced evolution of plain CNNs, which is known as the Residual Neural Network (ResNet). ResNet innovatively introduces the residual blocks (refer to Figure 1.7) into the CNN framework, realizing the skip connections between layers [9]. The approach solves the problems of accuracy degeneration and gradients vanishing or exploding that can occur with the increasing of layers in CNNs, which networks can be "deeper" for training more accurate models. Currently, the architectures in use include ResNet-50 (50 convolutional layers), ResNet-101, and ResNet-152, with this thesis specifically applying ResNet-152.

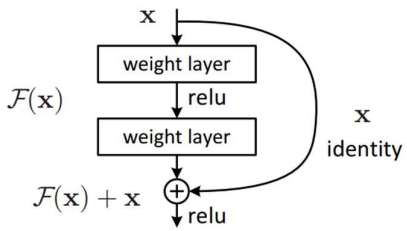


Figure 1.7: Diagram of residual blocks applied between layers. Adapted from [9]

The Feature Pyramid Network (FPN) can seamlessly integrate with ResNet to reserve feature maps from greater size (higher dimensional) to smaller size (lower dimensional) during the convolutional process. This strategy allows the identification of larger objects in abstract feature maps and smaller objects in detailed feature maps, which effectively utilize d various levels of semantic features. Then, the Region Proposal Network (RPN) scans the feature maps of varying sizes by a sliding window to generate anchors. These anchors are classified as objects and backgrounds based on their calculated scores, respectively corresponding to flakes of 2D materials and silicon substrates in optical images. Regions of Interest (RoI) are defined by filtering anchors with the cooperation of Non-Max Suppression (NMS) to refine selections. According to data from prior networks, the RoI Align technique precisely extracts relevant features from feature maps for each RoI. Finally, head networks utilize activation functions to process data for specific tasks, enabling the accurate determination of object classifications, bounding box coordinates for localization, and pixel-wise segmentations.

Chapter 2

Data and Evaluation

Datasets serve as the foundation for algorithm performance, supporting both the training and evaluation in deep learning. This is particularly true for computer vision tasks including Mask R-CNN, where datasets contain images and their respective annotations. To ensure optimal algorithm performance, images must capture the full spectrum of details and features related to the detection environment and objects. Meanwhile, annotations must provide precise and detailed information about their corresponding images. In addition to the quality of datasets, the quantity of datasets is also important. The sufficient datasets can provide comprehensive algorithm training, which improves the algorithm's ability to generalize in various detection scenarios. The typical composition of datasets covers training data and validation data. The training data are utilized to train the model weights. In contrast, the validation data serve for the evaluation of trained models, and they are not included in the training data.

In this thesis, optical images of 2D materials captured via OM are distinct from common datasets in Mask R-CNN training. Firstly, 2D material flakes often manifest as simple geometric shapes with different colors, which means they have relatively limited features for providing algorithm information. Secondly, optical images of 2D materials are acutely responsive to the research environment, different objective lens, illumination intensity, and observational instruments result in a completely different

appearance of 2D materials, especially for colors (See Figure 2.1). Consequently, the selection of 2D material datasets is pivotal to the outcomes of this research.

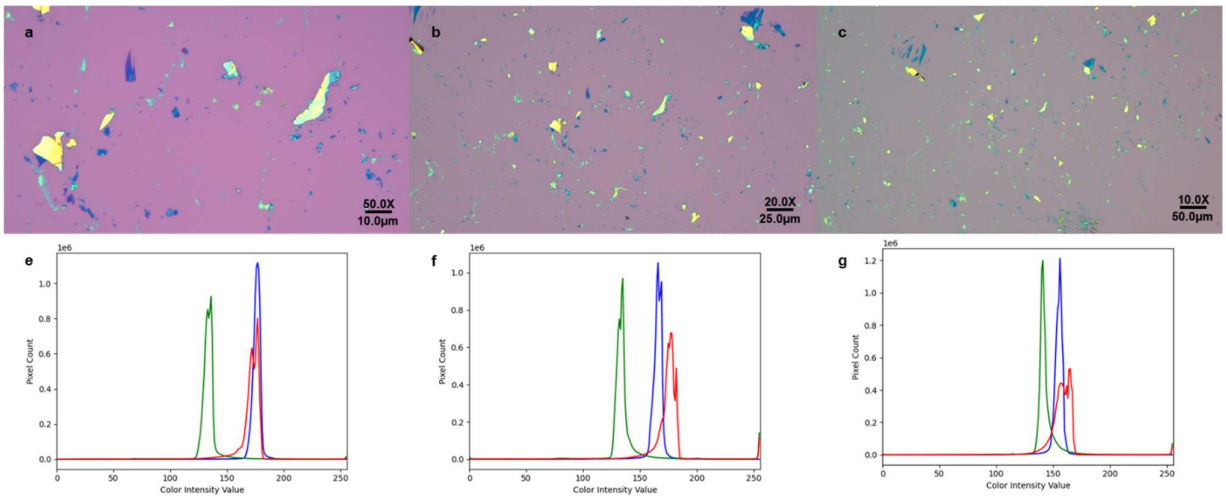


Figure 2.1: Optical images of the same area of graphite sample under identical illumination intensity but at different magnifications. From (a) to (c), the magnifications are 50.0X, 20.0X, and 10.0X. (e) to (f) present color histograms for (a), (b), and (c), respectively, showing the reducing of red pixels.

2.1 Common Objects in Context

Common Object in Context (COCO) is a widely recognized and universal format of annotation for datasets, which was initially developed by Microsoft to facilitate category detection, instance spotting and instance segmentation in the field of computer vision [10]. COCO is designed for easy readability, allowing users to effortlessly understand and edit annotations through JSON files. Furthermore, the development team of COCO provides a robust Python API library, which supports the objectives of generalization and practical application in this thesis.

Given challenges in assembling a large scale of comprehensive and precise datasets in a limited time frame—challenges that include the collection of optical images of 2D materials and the manual labeling of these images – I choose to apply datasets from Dr. Masubuchi's group for algorithm training.

In 2019, they achieved significant progress in the deep learning analysis of 2D materials images and released their COCO-formatted datasets for public study [11]. These datasets of 2D materials contain a rich variety of optical images of different 2D materials, accompanied by their corresponding COCO format annotations. Using these verified datasets, which have been proven their effectiveness in algorithm training, reduces the potential negative impact of substandard datasets on this thesis and provides a clear framework for the development of new 2D materials datasets.

The datasets of 2D materials explored and applied for training algorithms include Graphite, MoS_2 (Molybdenum disulfide), and WTe_2 (Tungsten ditelluride). Figure 2.2 shows the overall structure of the datasets, and Table 2.1 presents a summary of the statistic of datasets for this thesis.

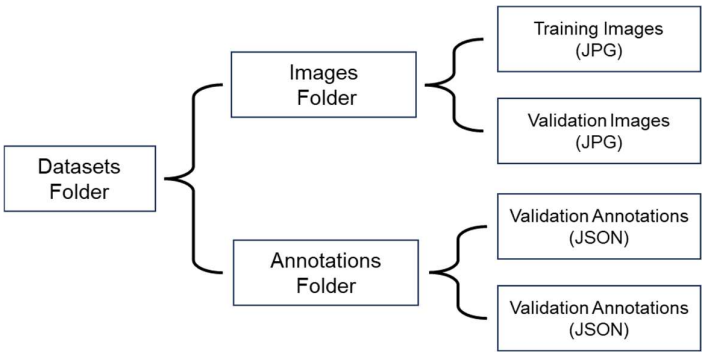


Figure 2.2: Diagram of the overall structure of datasets for algorithm training

Name	# of Images Total	# of Images Training	# of Images Validation	# of Objects (Total)	# of Objects (Mono)	# of Objects (Few)	# of Objects (Thick)
Graphite	862	664	198	4,805	1,858	2,081	866
MoS_2	569	480	89	839	239	523	77
WTe_2	375	309	66	1,053	148	582	323

Table 2.1: Statistic of datasets for algorithm training. Adapted from [11].

To comprehend the process that algorithms learn from datasets, it is essential to focus on the structure of COCO-formatted annotations. As mentioned above, an annotation file is a large dictionary in a JSON format. In this thesis, an annotation file contains three major sections: image information, category information, and annotation information. The image information records the basic characteristics of the optical image of 2D materials, including dimension (heights and width), identification number (id), and file path. The category information records the names and IDs of object classifications. Specifically, this thesis identifies three categories of 2D materials: Mono, Few, and Thick. Lastly, the annotation information captures details of the labeled object, including pixel area position (bounding box), shape (segmentation), and classification (category ID). These groups of information complement each other, which realizes the loading of data in algorithms. Figure 2.3 shows an example of an optical image of graphene and its annotations.

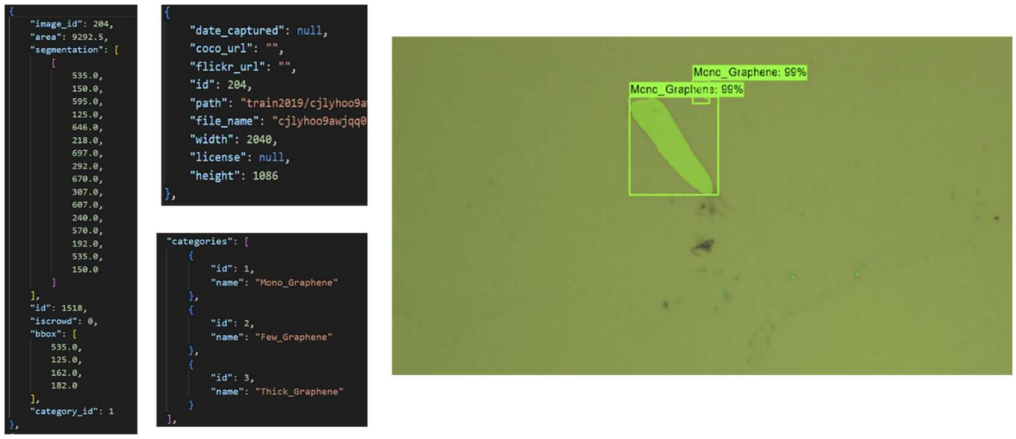


Figure 2.3: Optical image of Graphene and its annotations. Image Source: [11].

Notably, there are significant differences between optical images from Dr. Masubuchi's datasets and those taken in our lab. These variations come from the use of different observational instruments and conditions, which in turn influence the overall color patterns of optical images. Figure 2.4 illustrates this contrast by comparing training images from datasets with our laboratory images, and

their respective color histograms follow. According to the principle of convolutional layers introduced in Chapter 1.31, algorithms are trained on RGB channels of input images. Different colors of training images may hinder the generalization ability of trained models, which needs further exploration in the result section.

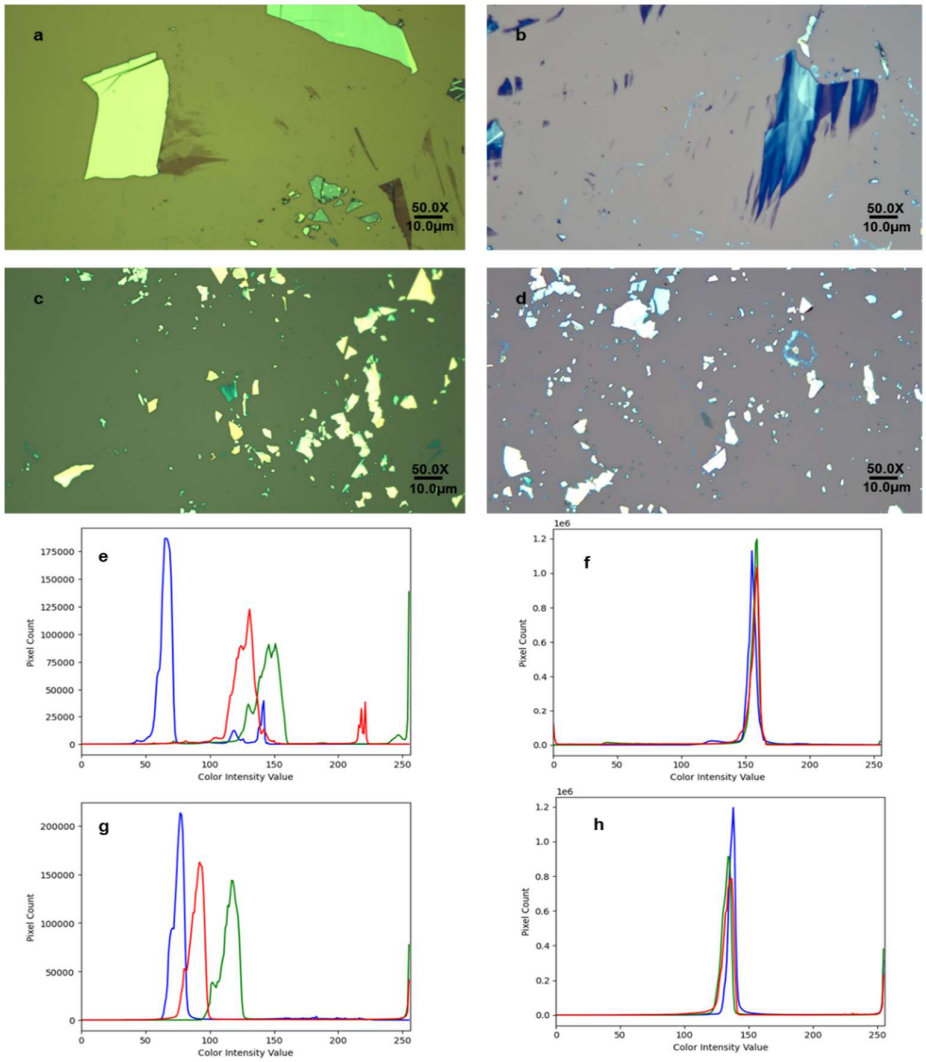


Figure 2.4: Comparisons between training images from datasets and our lab images. Image (a) and (e) are training images of Graphite and MoS_2. Image (b) and (d) are laboratory images of Graphite and MoS_2. The color histograms corresponding to these images, labeled as (e), (f), (g), and (h), illustrate the color distributions for (a), (b), (c), and (d), respectively. (a), (c) source: [11].

2.2 Evaluation

Neural Networks are described as “black boxes”. This term reveals the fact, that it is difficult to interpret the internal working principles and decision-making processes due to the complexity and opacity of neural networks. Therefore, evaluations are particularly important in the development and applications of neural networks, providing us a path to quantify the effectiveness of training and performance of trained models within this opaque context.

In this thesis, evaluation methods are based on the use of training data and validation data respectively. Visual results from evaluations play a crucial role in discussions in the following chapters, including fine-tuning hyperparameters, adjustment of training strategies, and accessing the performance and practical applicability of trained models.

2.2.1 Training Loss

Loss functions are utilized to measure the difference between predicted values from models and true values from datasets. The main purpose of loss functions is to optimize the training process. By pursuing the minimization of loss value, we can improve training strategies in a targeted manner.

The selection of loss function depends on the nature of the current task. In Mask R-CNN, the muti-task loss comprises the sum of classification loss, bounding box loss, and mask loss, as specifically mentioned in references [8, 12]:

$$L = L_{cls} + L_{box} + L_{mask} \tag{2.1}$$

The classification loss L_{cls} is given by the Equation 2.2:

$$L_{\text{cls}} = -\log p_u \tag{2.2}$$

Equation 2.2 is a log loss function, which is a commonly used loss for classification tasks. p_u represents the probability predicted by the model for the true category u. Next, the bonding-box loss function is defined as:

$$L_{box}(t^u, v) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t_i^u - v_i) \tag{2.3}$$

In this equation, smooth_{L_1} is specified as a L_1 loss:

$$\text{smooth}_{L_1}(x) \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{2.4}$$

Finally, L_{mask} is an average binary cross-entropy loss, which can be defined as:

$$L_{mask} = -\frac{1}{m^2} \sum_{1 \le i,j \le m} \left[y_{ij} \cdot log\left(\hat{y}_{ij}^k\right) + \left(1 - y_{ij}\right) log\left(1 - \hat{y}_{ij}^k\right) \right] \tag{2.5}$$

Here, y_{ij} represents the true mask value at coordinate (i, j) in the image, and $\hat{y}_{ij}^k$ is the predicted mask value for category k at the same position. Meanwhile, the entire mask is a RoI of size $(m \times m)$. So, the term $-\frac{1}{m^2}$ serves to normalize the loss value, making sure that the loss does not vary with different mask sizes.

The training loss is calculated during the training process. In this thesis, the training loss is used to assess whether the model fits well as the number of training epochs increases, with a lower training loss being targeted.

2.22 Mean Average Precision

Mean Average Precision (mAP) is a standard and comprehensive evaluation method in tasks such as classification and instance segmentation, especially for training by COCO formatted datasets.

Different from other metrics, mAP typically utilizes only validation data as its source for calculation. As discussed in the data section, validation data are never used during the training process, which means mAP can actually reflect the evaluation of models' practical performance in real-world tasks. Therefore, the mAP value serves as a crucial metric for adjusting training strategies and hyperparameters, selecting trained models for deployment, and presenting results.

The calculations of mAP are based on the concept of Intersection over Union (IoU) between actual values and predicted values, and the detailed introduction of mAP comes from the reference [13]. Figure 2.5 illustrates the principle of IoU:

$$\text{IoU} = \frac{y \cap \hat{y}}{y \cup \hat{y}} = \frac{\text{Actual Area} \cap \text{Predicted Area}}{\text{Actual Area} \cup \text{Predicted Area}} =$$

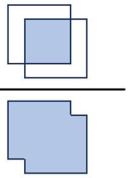


Figure 2.5: Equation and diagram of IoU calculation.

In this expression, IoU is applicable for bounding boxes and segmentations. When $\text{IoU} > 0.5$, we can define perdition as True Positive (TP). Conversely, when $\text{IoU} \leq 0.5$, we can define the prediction as False Positive (FP). Also, when an actual value is not successfully predicted, we define it as False Negative (FN). According to these definitions, we can define two critical terms. Precision describes the accuracy of identifying TP among all positive values, while Recall measures the ability to capture all TP within all the actual positive values. The equations below specify these ideas:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} = \frac{\text{TP}}{\#\text{Predictions}}, \text{Recall} = \frac{\text{TP}}{\text{TP+FN}} = \frac{\text{TP}}{\#\text{Ground Truths}} \quad (2.6)$$

By plotting a Precision-Recall (P-R) curve with Precision on the y-axis and Recall on the x-axis, we can calculate the Average Precision (AP) using the following equation:

$$AP = \int_0^1 p(r)dr \tag{2.7}$$

Finally, mAP is calculated by the mean AP value across all the categories, which is defined by Equation 2.8:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{2.8}$$

In this thesis, the number of categories $N = 3$, since the datasets contain three classifications of 2D materials.

To sum up, mAP values are comprehensive scores for models. A higher mAP indicates that the model can identify objects with high precision and accurately predict the areas of those objects. This feature aligns perfectly with the objectives of this thesis, in addition to reflecting the model's performance in real-world cases through mAP. Thus, mAP is the primary metric for deciding on training strategies and fine-tuning hyperparameters, as well as for selecting the final models to be applied. Additionally, mAP, in conjunction with training loss, helps to judge overfitting during the training process.

Chapter 3

Methodology

In this chapter, we discuss the methodology in this thesis, including the specific strategies and operational approaches applied to algorithms. The training and implementation of neural networks highly depend on the hardware's capacity, especially on the Graphics Processing Unit (GPU), which is mainly applied for algorithm training currently. The hardware configuration utilized for this thesis is listed in Table 3.1.

Hardware Name	Model
CPU	Intel - Core i7-13700K
GPU	GIGABYTE - GeForce RTX 4070 12GB
RAM	Corsair - DDR5 32GB 6400MHz

Table 3.1: List of hardware configuration

The architecture of the Mask R-CNN utilized in this thesis is built upon the PyTorch library [14], while calls to parallel computing via GPU are made through CUDA [15]. The chosen backbone network is ResNet-152, followed by FPN, RPN, RoI Align, and the head networks for classification, bonding boxes, and segmentations. More details about Mask R-CNN can be found in Chapter 1.32. After all, the entire trained model contains a total of 78,622,055 parameters. An overview of the model is provided in Appendix A.

3.1 Training Strategies

The proper application of training strategies can maximize the effective usage of hardware and datasets, which is important to the stabilization and efficiency of algorithm training. The training strategies and techniques applied in this thesis follow the guidelines provided in the reference [16].

The data augmentation is realized by the transform module at the beginning of dataset loading. When the optical images of 2D materials are converted into tensors, random flipping, and rotation are applied to 50% of them. This procedure increases the data volume, which is helpful to realize the generalization of models.

In this thesis, we also apply the transfer learning strategy by using pre-trained models [17], including the pre-trained model for ResNet-152 and the pre-trained model for COCO formatted Mask R-CNN, which are sourced from PyTorch database. These pre-trained models are based on large-scale data training, already learning the general features of various objects. Thus, the usage of pre-trained models can speed up the training process and reduce the requirements of datasets, which can improve the overall performance of the trained models.

To train models, we apply the warm-up strategy, where the learning rate gradually increases to the pre-set value in the first several epochs. Because the primal weights are randomly initialized, the warm-up strategy allows the model update to parameters more gently in the early stage, avoiding excessive fluctuations, and improving the stability of the entire training process. Furthermore, mixed precision training is applied [18], using 32-bit floating point for calculations that need full precision, and using 16-bit floating point for other computations. This strategy can significantly reduce

computational load while maintaining the precision of models, so that speech up the convergence. Finally, we utilize the learning rate scheduling throughout the entire training process. By reducing the learning rate with the increasing of epochs, models can be refined as they approach the later stage of training. This is also a useful strategy for stabilizing the training process and fine-tuning of models.

3.2 Hyperparameters and Evaluation Plots

The selection of training hyperparameters plays a decisive role in the performance of trained model. Table 3.2 lists the hyperparameters and their corresponding descriptions for this thesis.

Hyperparameter	Description
Epoch	One complete pass through the entire training dataset by the algorithm.
Batch size	The number of images in subsets that propagates through the algorithm before updating weights.
Learning rate	The hyperparameter that controls how much weights are adjusted each update.
Learning rate gamma	The discount factor that determines how much the learning rate decreases each time.
Learning rate step	The hyperparameter determines the epochs at which the learning rate is decreased.
Weight decay	The form of regularization that penalizes large weights to reduce weights in the algorithm.
Momentum	The hyperparameter replaces gradients with a leaky average over past gradients to speed up the training.

Table 3.2: List of hyperparameters and related descriptions

Among these hyperparameters, learning rate and batch size are closely monitored, as they directly influence models' convergence and the effectiveness of training [19]. In this thesis, we pay attention to the impact of various learning rates and batch size configurations on the evaluations as the number

of epochs increases. Apart from the settings of learning rate scheduling, batch size, and epoch, the configurations of other hyperparameters remain the same as the settings of common tasks, which are shown below:

$$\text{Learning rate gamma} = 0.1, \text{Weight decay} = 1\times10^{-4}, \text{Momentum} = 0.9 \quad (3.1)$$

As we discussed above, batch size refers to the number of images in subsets that will be trained by algorithm. For example, the training dataset of graphene contains 664 images. If batch size is 1, each image is a separate subset and trained individually. Also, if batch size is 2, there are 332 subsets, each containing 2 images, which are trained sequentially. Hence, the larger setup of batch size means more data is processed, which heavily relies on the performance and memory capacity of GPUs. Based on our GPU configuration, when the batch size is 1, GPU utilization is around 50%, whereas when the batch size is increased to 2, GPU utilization reaches 95%. According to resource saturation, the maximum value of batch is 2 in this thesis. Moreover, the larger batch size is usually accompanied by higher learning rates. A larger batch size means that more data is used for each weight update, which makes the calculated gradient estimate closer to the actual gradient of the entire dataset. Thus, the model can take a larger step, that is, a higher learning rate, in each update. The simplest method to find the corresponding learning rate for a batch size is to multiply the learning rate by k when multiplying the batch size by k [20]. Combining the random search method, we select the combination of batch size = 1 and initial learning rate = 0.004, and the combination of batch size = 2 and initial learning rate = 0.01 for the comparison. The algorithm trainings are conducted on datasets of Graphite, MoS2, and WTe2, with each lasting 40 epochs. Meanwhile, the learning rate decays at epoch 15, 25, and 30. Figure 3.1 includes evaluation plots based on different datasets of 2D materials and combinations of batch size and learning rate.

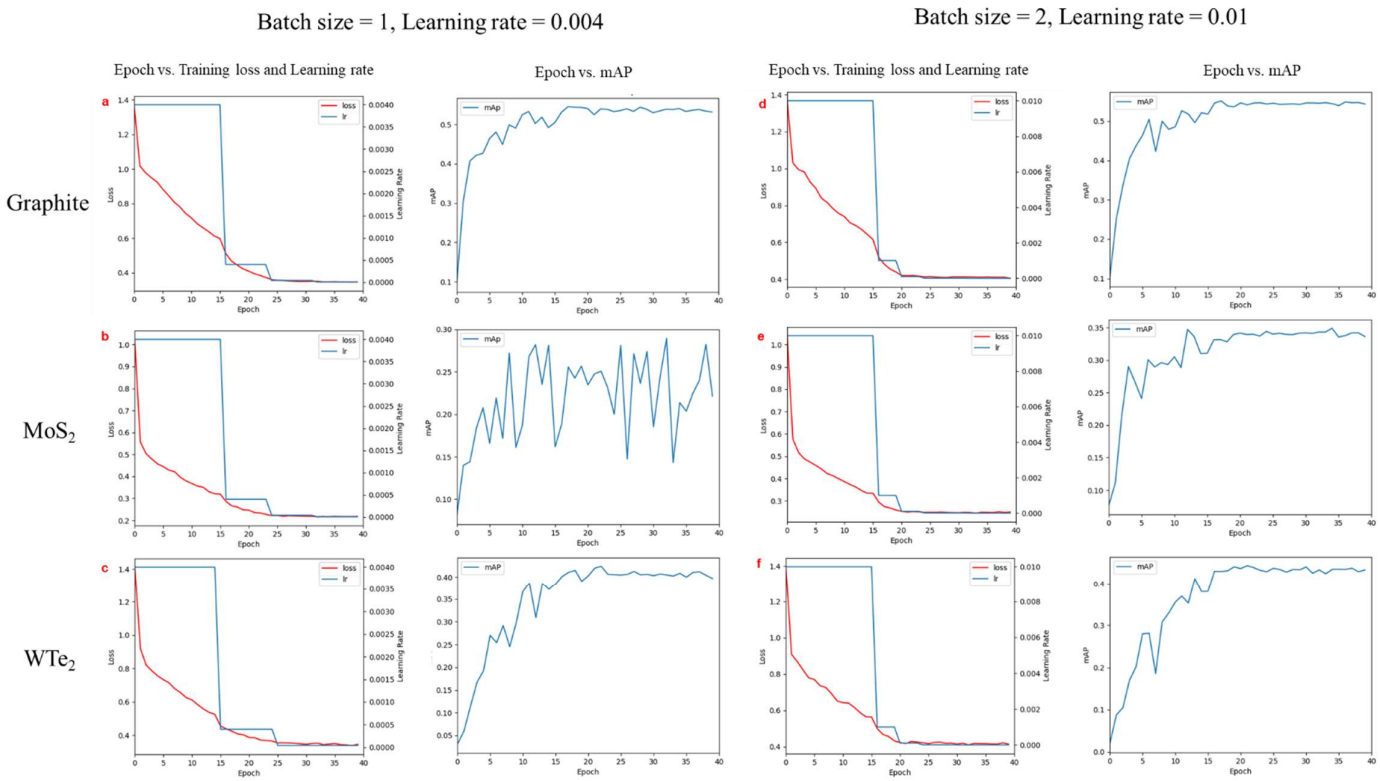


Figure 3.1: Evaluation plots of training sessions for Graphite, MoS_2, and WTe_2, showing training loss, learning rate, and mAP as functions of increasing epochs, across different batch sizes and initial learning rates. Plots (a) and (d) correspond to Graphite, (b) and (e) to MoS_2, and (c) and (f) to WTe_2.

According to the evaluation plots, we observe that the general training loss is lower with a batch size of 1 and a learning rate of 0.04 for the same dataset of 2D materials. However, the mAP values are lower compared to the case where the batch size is 2 and the learning rate is 0.01. Especially for MoS2, where the mAP plot exhibits significant fluctuations with the increasing of epochs. This indicates poor model generalization and performance on the validation dataset. Furthermore, combined with the observation that the training loss approaches 0.2, it suggests that overfitting occurs during this training process. Generally speaking, when the batch size is 1 and the learning rate is 0.004, models exhibit lower training loss, lower mAP, and larger fluctuations. Although models fit training datasets better, they perform at a lower ability on validation datasets. Conversely, with a batch size of 2 and a learning rate of 0.01, models demonstrate better generalization ability and stability throughout the

training process. Therefore, according to the objective of practical applications by trained model, the batch size of 2 is a more suitable selection in this thesis.

Furthermore, to investigate the influence of the total number of epochs, we trained the models under the same hyperparameter settings (batch size = 2, learning rate = 0.01) for up to 100 epochs. Figure 3.2 shows related evaluation plots.

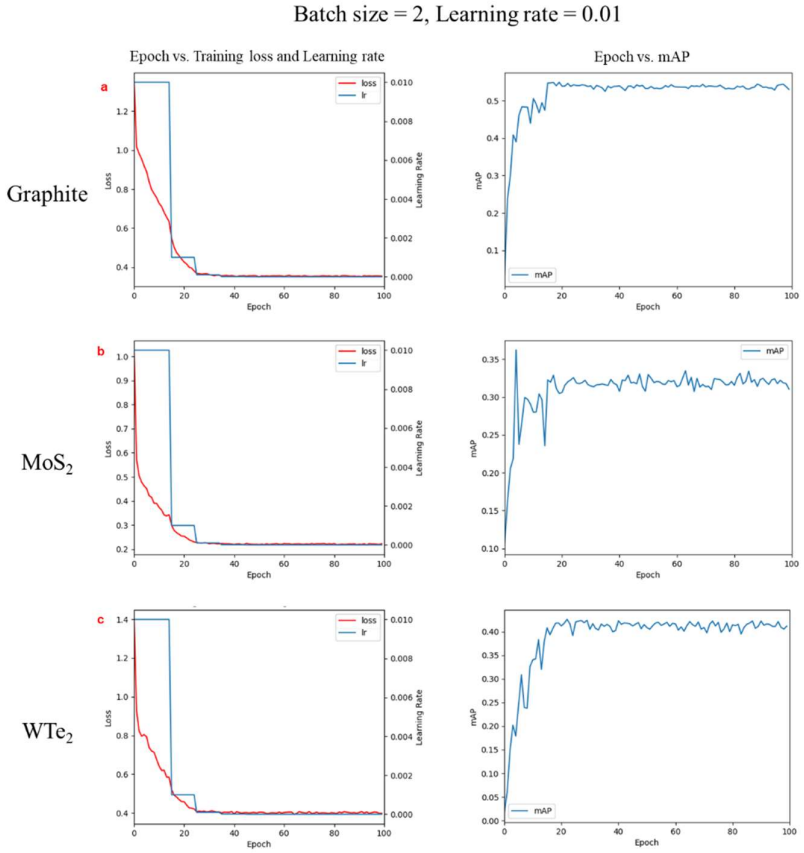


Figure 3.2: Evaluation plots of training sessions for Graphite, MoS_2, and WTe_2, showing training loss, learning rate, and mAP as functions of 100 epochs. Plot (a) is related to Graphite, (b) to MoS_2, and (c) to WTe_2.

Here, we find that the models rapidly converge within the first 40 epochs, reaching their performance peak during this early stage of the training process. In the following epochs, mAP values of models do not increase anymore and even show a downward trend. Training loss of models decreases to the approximate constant after 60 epochs. Therefore, we summarize that models cannot

continue optimizing beyond a certain point; they reach their saturation value in the first 40 epochs. The optical evaluations for each 2D materials under this case (batch size = 2, learning rate = 0.01) are shown below:

Batch size = 2, Learning rate = 0.01, Epoch = 100		
Name of 2D materials	Optimal training loss	Optimal mAP
Graphite	0.35	0.54
MoS_2	0.22	0.36
WTe_2	0.40	0.42

Table 3.3: Optimal values of training loss and mAP for 2D materials when batch size = 2, learning rate = 0.01.

As a result, in subsequent training attempts, there is no need to set a large number of epochs. To save a large amount of time and computational resources, we continue to use 40 epochs for each training session.

Our goal is to find an appropriate initial learning rate value for a batch size of 2. Hence, we utilized the interval search method and the early stopping strategy to rapidly identify the optimal learning rate value. Specifically, we start by setting the initial learning rate to 0.02, then print out the training loss and mAP values after each epoch, and closely monitor the changes in these values. If the values reach saturation, we immediately stop the training process and start another training loop with a lower initial learning rate. These approaches help us quickly find a suitable combination of hyperparameters based on our hardware configuration, namely batch size = 2 and learning rate = 0.012. Figure 3.3 includes evaluation plots for this case. Also, the optimal values for each 2D material are shown in Table 3.4.

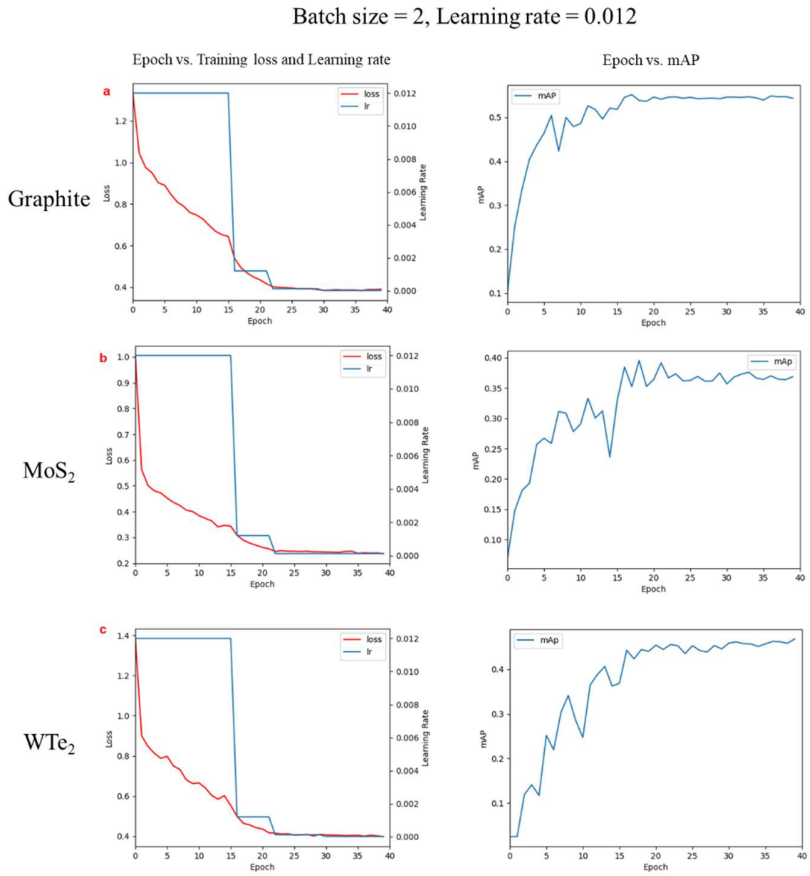


Figure 3.3: Evaluation plots of training sessions for Graphite, MoS_2, and WTe_2, showing training loss, learning rate, and mAP values when batch = 2, learning rate = 0.012. Plot (a) is related to Graphite, (b) to MoS_2, and (c) to WTe_2.

Batch size = 2, Learning rate = 0.012, Epoch = 40		
Name of 2D materials	Optimal training loss	Optimal mAP
Graphite	0.38	0.55
MoS_2	0.24	0.40
WTe_2	0.40	0.46

Table 3.4: Optimal values of training loss and mAP for 2D materials when batch size = 2, learning rate = 0.012.

The mAP values of models are sufficiently high to realize practical detections of 2D materials. Reference [11], the source of the datasets for this thesis, indicates that achieving mAP values of 0.49

for Graphene and 0.52 for WTe_2 is considered practical for 2D material detections. Consequently, our algorithm can effectively train the feasible models. The best versions of the models from the training mentioned above are utilized for predicting 2D materials images and for practical application in a laboratory environment, as presented in the results section.

Chapter 4

Results and Applications

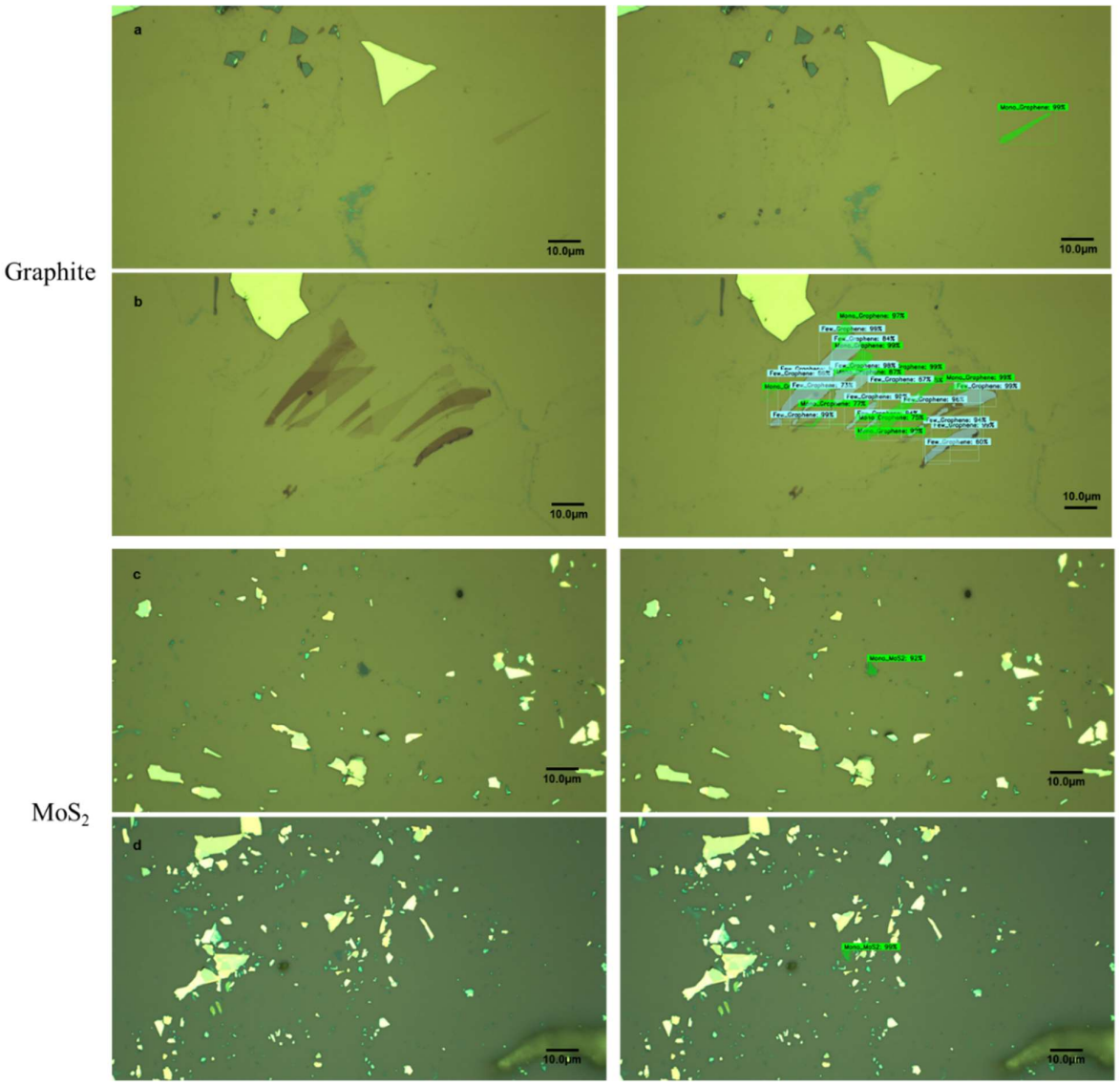
In this chapter, we utilize the trained models to identify optical images of 2D materials, discuss the performance of models, and verify the feasibility of machine learning in the promotion of 2D materials research. Additionally, as highlighted in prior chapters, a primary objective in the field of 2D materials research is to enhance user-friendliness and streamline operation. Our approaches also focus on realizing the popularization and generalization of our algorithms by integrating codes into software complemented by a simple and clear user interface. Consequently, this chapter also introduces and discusses derivatives based on our algorithms, including software and hardware enhancements in 2D materials research.

4.1 Predictions in Optical Images of 2D Materials

Applying trained models to predict images is the most basic and direct method to reflect the results. In this thesis, predictions are also based on the PyTorch framework. After loading the corresponding model of the 2D material image to be predicted, we invoke PyTorch commands to switch the Mask R-CNN to evaluation mode, which disables training-specific behaviors of certain layers. Meanwhile, to facilitate rapid image prediction, we continue to utilize the GPU to operate calculations. The prediction

made by PyTorch returns a dictionary that includes lists of object categories, probabilities, bounding boxes, and segmentations. Notably, the threshold for probability is adjustable and is set to 0.5 in this thesis. Then, predictions with a probability less than 0.5 are classified as false and are ignored.

The visualization of predicted outcomes is realized using the OpenCV library [21]. Based on values extracted from the predicted dictionary, we can precisely draw the bounding boxes and segmentations on the original image, and label the relevant category name and probability nearby using OpenCV tools. Firstly, we apply models with the highest mAP value to predict optical images from validation datasets. Figure 4.1 displays the predicted images and their corresponding original images.



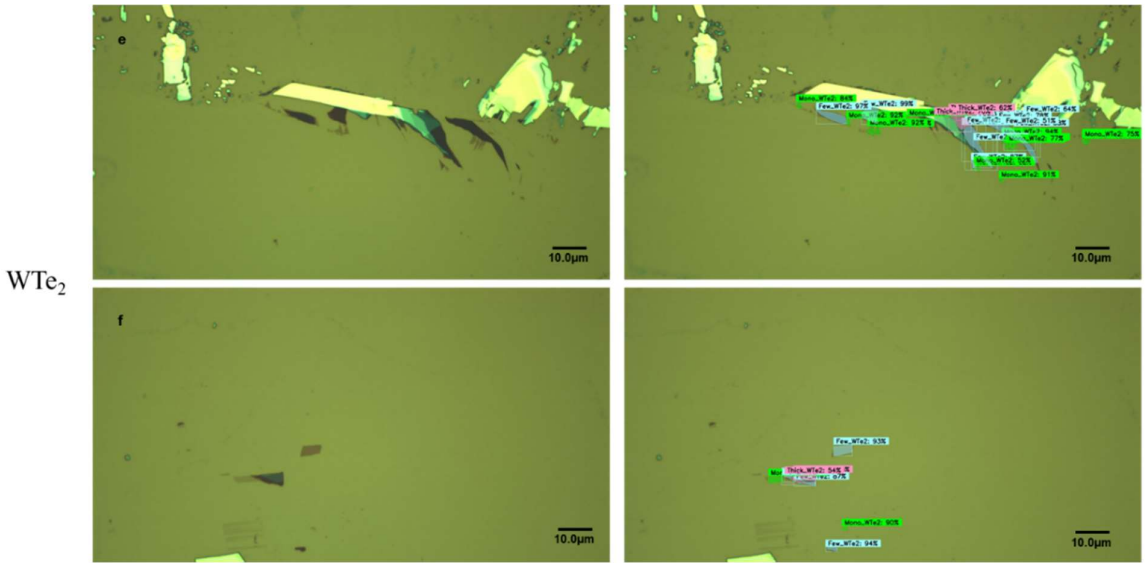


Figure 4.1: Original images (left) and predicted images (right). Images (a) (b) are related to Graphite, (c) (d) to MoS_2, and (e) (f) to WTe_2.

In the predicted images, monolayers are labeled with green bounding boxes and segmentations, few-layers with blue, and thick-layers with pink. By comparing the annotations from the validation datasets to the labels in the predicted images, we observe that all annotated objects are successfully detected by the models and displayed in the predicted images. Furthermore, we find that in regions of the original image containing different categories or a relatively large area of flakes, as shown in predicted images (b) and (e) in Figure 4.1, the models can identify the entire flake as a group of many small flakes, labeled by bounding boxes and segmentations. According to the definition of IoU discussed in Chapter 2.22, such individual small predicted areas can lead to a decrease in IoU, thereby resulting in a lower mAP value. This is a primary factor leading to the overall mAP values being around 0.5. However, the detected flakes are prepared for further research and must be manually collected and reviewed. Although models cannot identify entire segmentations using only a single label, the predicted outcomes, including accurate positions and classifications, are deemed completely acceptable in practical research scenarios.

At the end of Chapter 2.1, we discuss the distinctions between images from datasets and images from our laboratory. To further explore the generalization and feasibility of our models under various research scenarios, we focus more on the predicted outcomes of our optical images. We test two existing optical images of 2D materials in our laboratory, Graphite and MoS2, and the predicted outcomes are shown in Figure 4.2 and Figure 4.3.

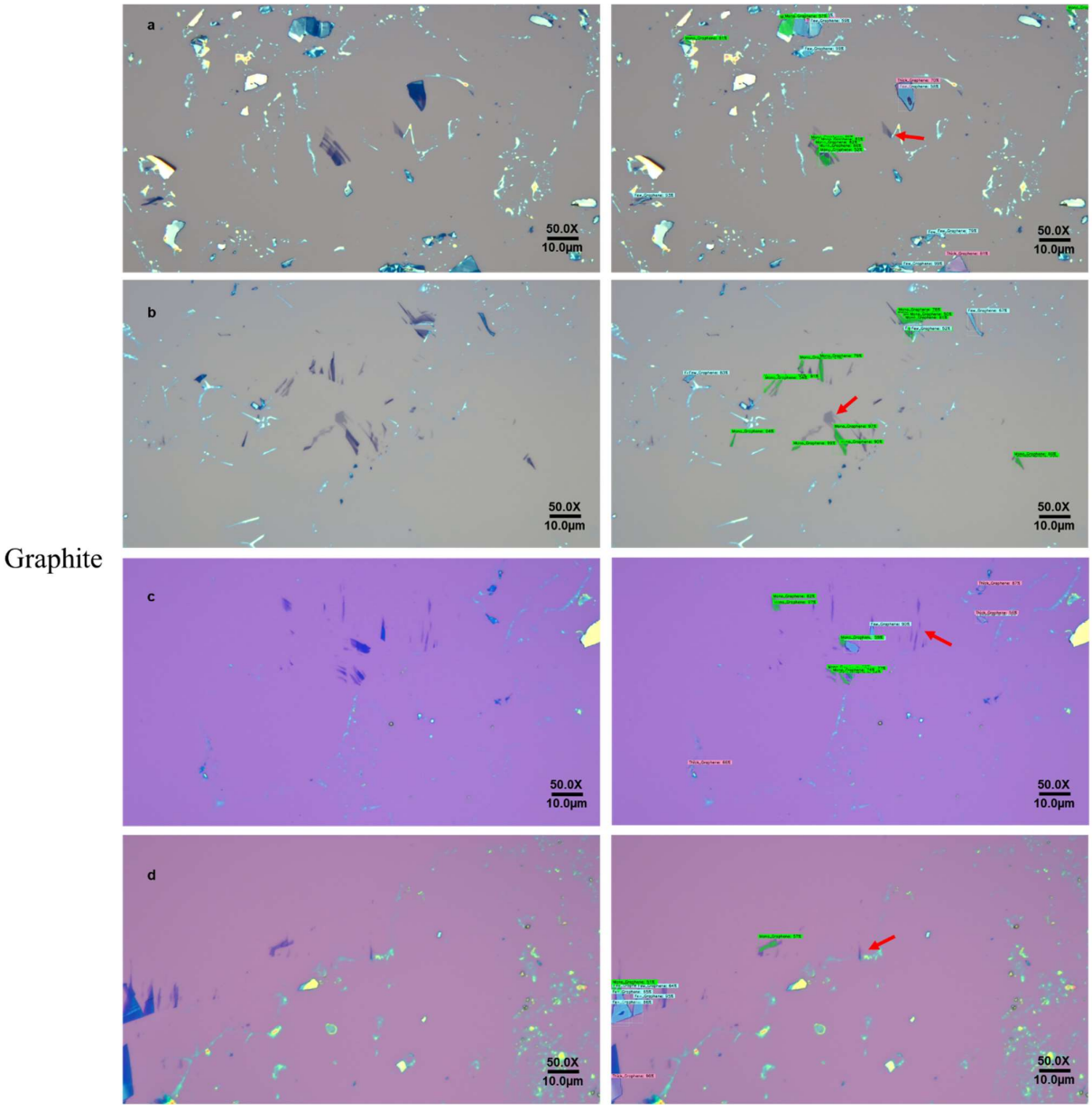


Figure 4.2: Original images (left) and predicted images (right) of Graphite, with missing monolayers indicated by red arrows. Images (a) and (b) are under high illumination intensity, while images (c) and (d) are under low illumination intensity.

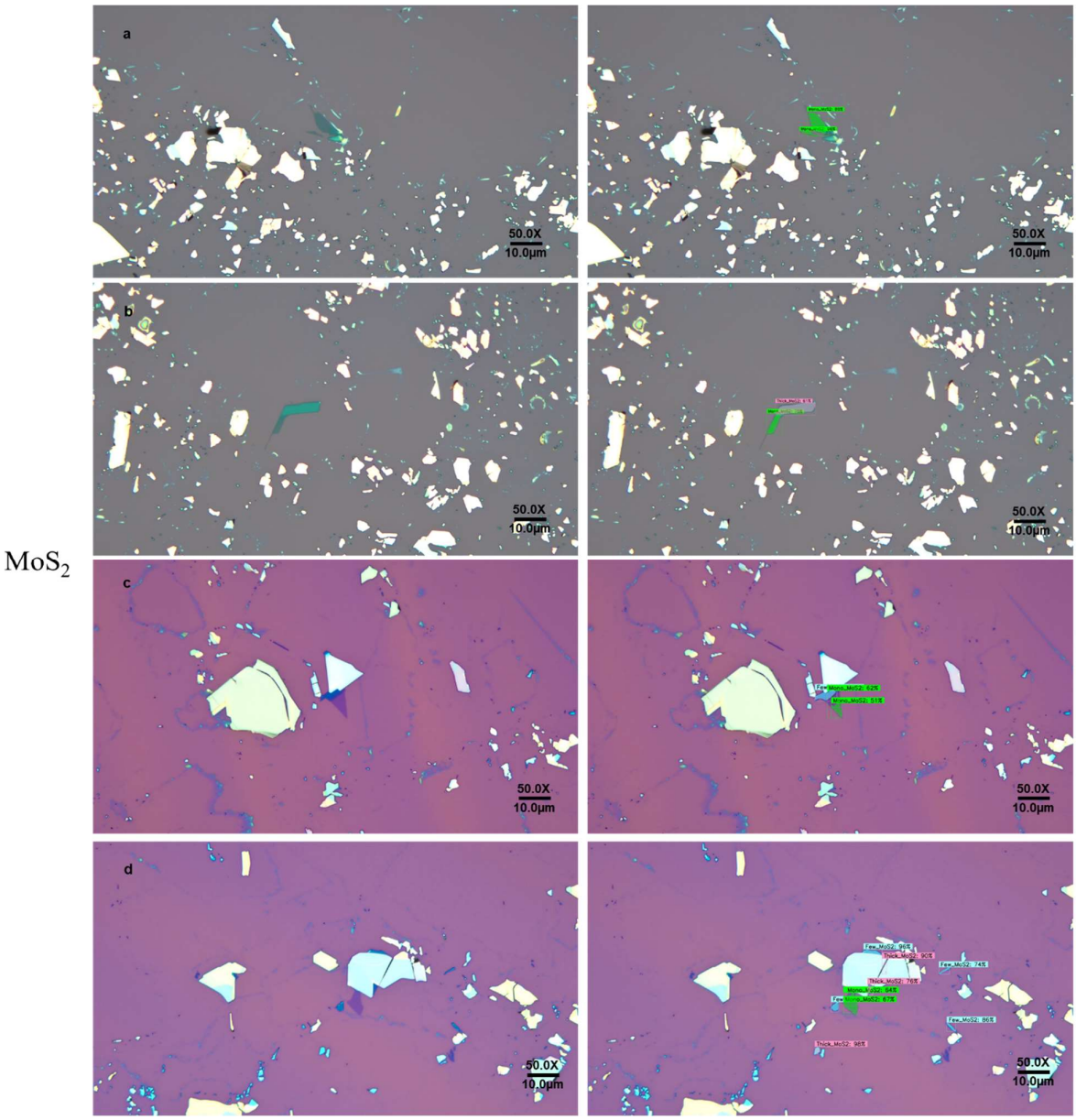


Figure 4.3: Original images (left) and predicted images (right) of MoS_2. Images (a) and (b) are under high illumination intensity, while images (c) and (d) are under low illumination intensity.

Figure 4.2, predicted outcomes of graphite images under high illumination intensity and low illumination intensity, which are respectively shown in grey (image a, b) and pink (image c, d) background. As we discussed in Chapter 2.1, the generalization of models is directly related to the appearance of training datasets. Therefore, for predicting our images of graphite, the model can

identify and classify the majority of flakes but miss few required objects. In each predicted image from Figure 4.2, missing monolayers, namely graphene, are labeled by red arrows manually.

Nevertheless, the MoS_2 model demonstrates excellent performance in predicting our images. According to Figure 4.3, we observe that all the monolayers, regardless of the background color, are successfully detected with precise segmentations. The mAP of the MoS2 model is 0.4, which is the lowest among the three models of 2D materials. However, greater similarity in appearance, especially in shapes and details, between the training datasets and our images of MoS_2, enables the model to generalize better in predicting our images. Conversely, the visual differences between the training datasets and our optical images of graphite are more pronounced, leading to relatively poor performance in identifying our images. Hence, to enhance the generalization capability of our models specifically for our laboratory environment, we can compile a collection of our own optical images to create customized training datasets.

Additionally, the total time required to predict a single optical image depends on the number of objects it contains, which includes both inference time and NMS time. Based on our GPU's performance, it takes between 0.05 and 0.1 seconds to process and predict one image with a size of 3840×2160 pixels. The low time consumption indicates that our models are applicable for efficient real-time identification of 2D materials.

In summary, the models exhibit satisfactory performance in predicting images from our laboratory, making them suitable for further discussion and practical applications.

4.2 2D Materials Identification Software

The operation of image prediction by using Python codes necessitates both technical skills and strict procedures, including setup of basic parameters, loading the certain model and category index, and loading the image through its path. During this process, any mistake can lead to coding errors, which are not conducive to processing large batches of images. Moreover, considering the universality and user-friendliness of our algorithms for all 2D materials researchers, we integrate the algorithms into software equipped with a simple user interface for convenient usage. Figure 4.4 is the demonstration of the image prediction made using this software.

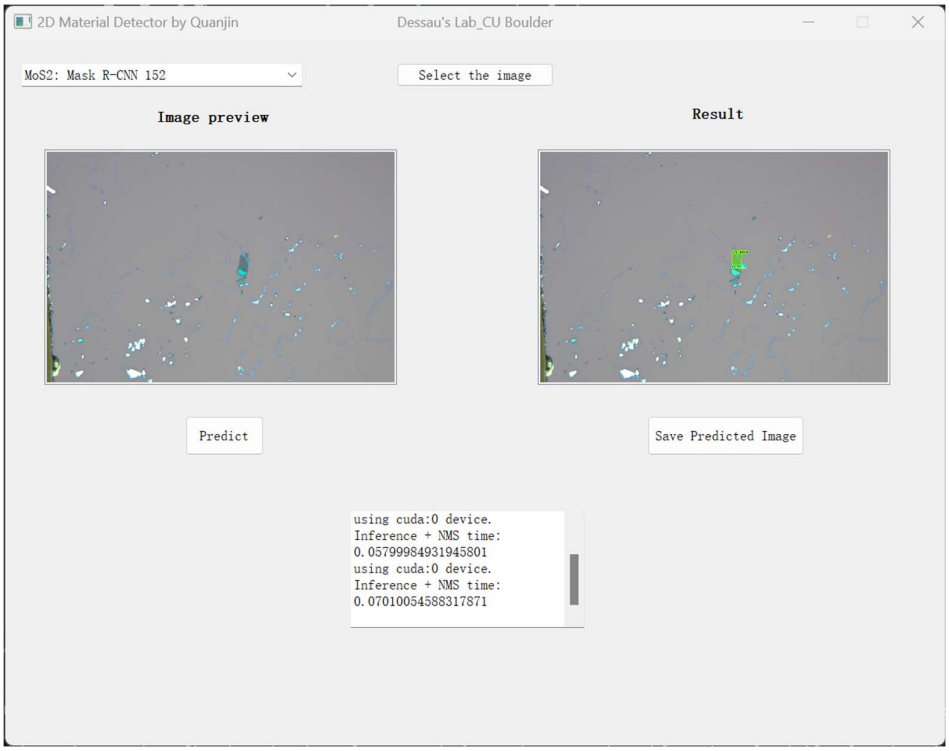


Figure 4.4: The user interface of the software for the identification of 2D materials.

The user interface design is based on the PyQt5 library [22], which enables simple and intuitive operations such as model loading, image prediction, and saving of predicted images. This software is

not only more accessible for practical 2D materials research but also lays the foundation for achieving real-time identification of 2D materials in future work.

4.3 Software for Machine-Learning-based 2D Materials Transfer System

Our objective is to standardize the methodology for identifying 2D materials with high efficiency and accuracy. Thus, the simultaneous identification of 2D materials during the research process, namely real-time identification, holds significant importance for this thesis. Motivated by this, our group develop a machine-learning-based 2D material transfer system, with automation as its core idea. This system includes digital microscopy and a motorized stage within a glove box, and is equipped with a computer for running algorithms. To integrate various functions and facilitate communication between components, we design a system software to monitor and control the entire system.

Specifically, the operation of model loading in the system software is similar to that of the identification software, but it switches the input from images to real-time videos. This adaptation means the system software covers the typical function of a microscope display by processing the video signal from the camera. The motorized stage consists of a KDC-101 motor controller and a Z825B motorized actuator from Thorlabs, capable of achieving a precision step of 0.1 μm in the X, Y, and Z directions. Additionally, the system software facilitates the operation of the motor controller with a simple click of the direction buttons, incorporating features such as the customization of motor parameters (acceleration, speed, and step size) and real-time updates of stage positions (X, Y, and Z).

Foremost, the logic to achieve real-time identification involves predicting video frames one by one. Subsequently, predicted video frames, complete with bounding boxes and segmentations, are

outputted in the display window. To enhance the collection of research images, the system software features a screenshot button that allows for the direct capture of the current display window. Simultaneously, captured images are temporarily stored and displayed as icons in the photo list, where clicking on them opens a built-in photo viewer. This photo viewer enables the operation of observing, deleting, and saving full images. Each full image is linked with its specific positions and is shown in the photo viewer. Consequently, researchers can simply input the image's positions into the motor controller to move the stage and locate the region of 2D materials in the image. In addition to these primary functions, the software also includes several auxiliary functions, such as customization of the scale bar display, a toggle switch for prediction display, and a text browser for user interactions. Figure 4.5 showcases and describes the user interface and functions of the system software's main window.

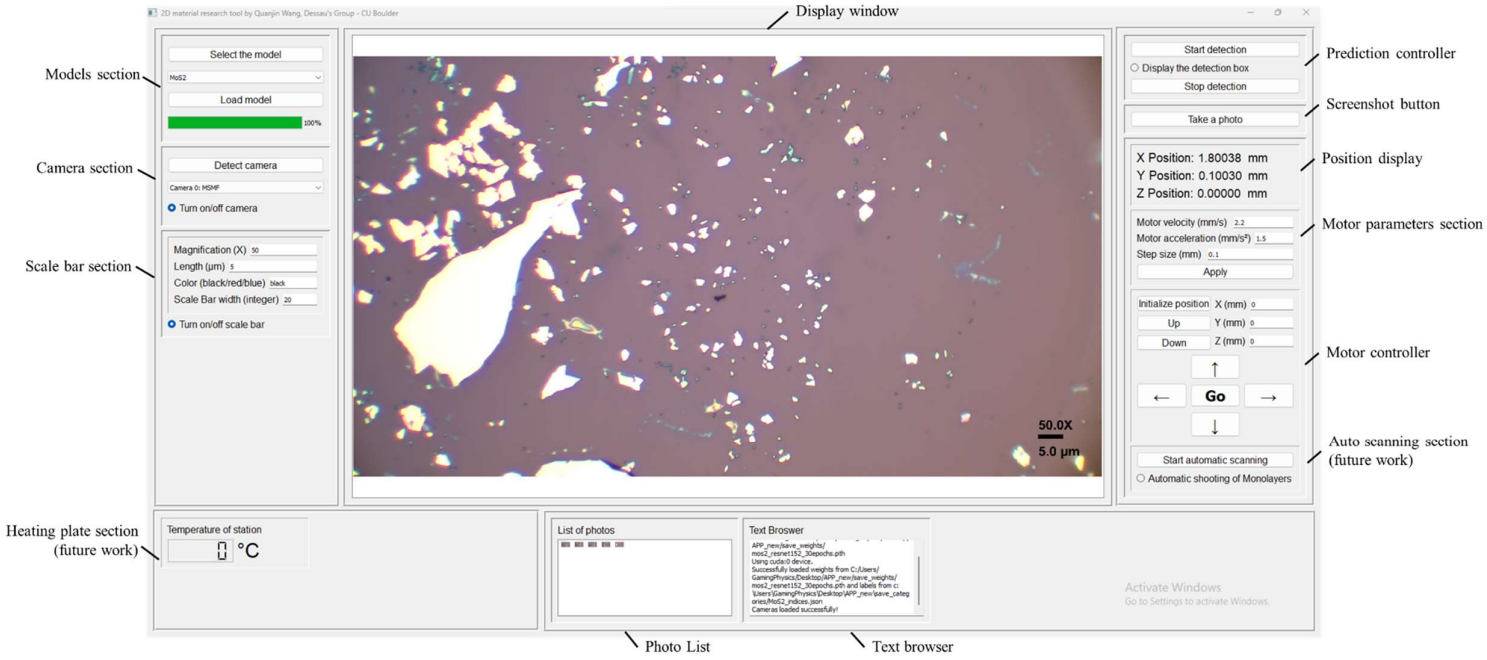


Figure 4.5: The main window's user interface of the 2D materials transfer system software.

Moreover, Figure 4.6 presents a demonstration of the built-in photo viewer and screenshots of the identification of MoS_2 flakes during practical research, which helps to further understand the functions and working principles of the system software.

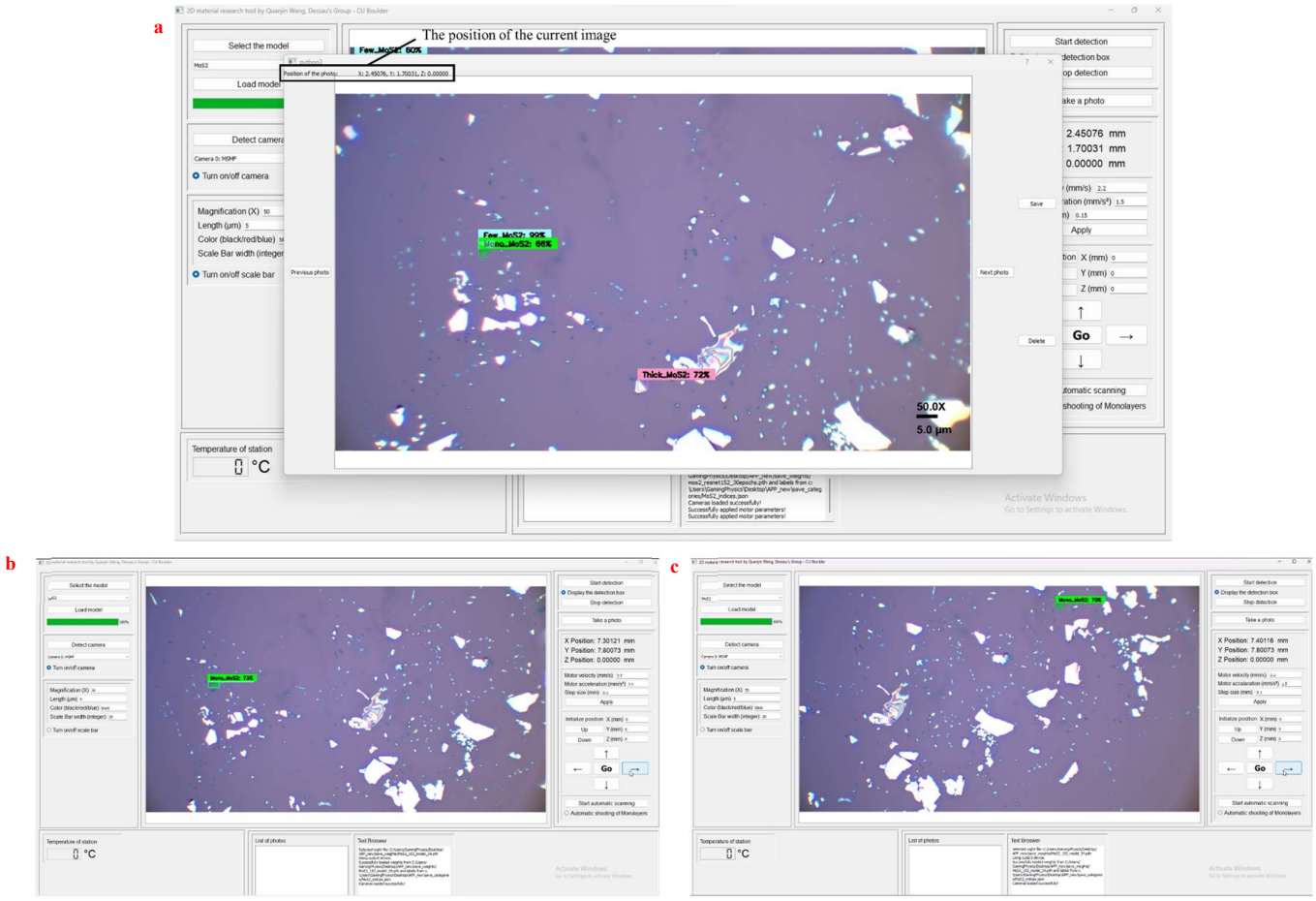


Figure 4.6: The built-in photo viewer (a) and screenshots of the identification of MoS_2 flakes (b)(c). From (b) to (c), the stage moves 0.1 mm by clicking the right button.

To sum up, the software for the machine-learning-based 2D materials transfer system is the primary achievement discussed in this thesis. The entire system, equipped with this software, successfully achieves real-time identification of 2D materials, large-scale processing of research images, and precise control of the stage via a motorized actuator. Researchers can use the system

software to rapidly and accurately locate monolayers by merely clicking buttons, eliminating the need to manually work for several hours inside a glove box. Furthermore, both the system and the software are upgradeable; the software is designed with sufficient flexibility using Python, which is universal and adaptable to add the monitoring of heating plates, among other potential required functions. Lastly, our goal is to achieve completely automatic scanning of silicon substrates. Through communication between various components and algorithms, it is theoretically possible to automatically scan and identify 2D materials without the supervision of researchers. This is going to be further discussed in the future work section of Chapter 5.

Chapter 5

Conclusion

In this thesis, we successfully apply a machine learning method related to computer vision for identifying 2D materials. We construct the Mask R-CNN-152 for model training and image predictions. Additionally, we evaluate the training loss and mAP of the trained models, achieving model optimization through fine-tuning hyperparameters and realizing relatively optimal evaluations based on specific hyperparameter settings. In the verification stage, we discover that our models exhibit excellent performance in predicting validation images, with a processing time of less than 0.1 s. We also find that our models are feasible and sufficiently acceptable for identifying 2D materials images under our laboratory conditions. Ultimately, we use our models and algorithms in the design of identification software, which facilitates simple and intuitive model usage for image predictions. According to the effective applications of models, we construct a machine-learning-based 2D materials transfer system equipped with system software. This system accomplishes real-time identification of 2D materials. Moreover, the system software enables precise control of the motorized stage, model loading, and processing of large batches of images, incorporating multiple useful and necessary functions for observation, monitoring, customization, and user interaction.

Consequently, this thesis standardizes the methodology for identifying 2D materials based on machine learning methods, embracing the principles of user-friendliness, convenience, and universal

applicability.

5.1 Future work

This thesis still has room for improvement, particularly in terms of datasets and applications. Enhancing the datasets to train models is beneficial to the specific laboratory conditions, and discussing the logic of automatic scanning in the system software is the primary goal to improve the applications. Meanwhile, this thesis also provide a potential direction for other vision architecture in the identification of 2D materials.

5.1.1 Improvement of Datasets

As we discussed in Chapter 4, the different appearance, including colors, shapes, and features, between images in training datasets and our laboratory's images of 2D materials leads to the lower generalization of models. Therefore, the most fundamental method to improve this situation is to create new training datasets using annotated optical images from our laboratory as an alternative. we can employ a transfer learning strategy to train new models based on our trained models from this thesis, which can accelerate model convergence and optimize the new models' performance. Meanwhile, the validation datasets can also be replaced by our annotated optical images, which can ensure that the evaluations of the models are more suitable and persuasive for our laboratory conditions.

However, constructing a complete dataset presents challenges. It requires not only a large number of high-quality optical images of 2D materials but also accurate and precise annotations for those

images. This process is time-consuming and requires specific skills, which need sufficient preparation before commencement.

Notably, an increasing number of databases for physics and physics-centered machine learning have been established, such as AI/ML projects at Argonne National Laboratory. Based on these open resources, we may share and acquire valuable data related to 2D materials, including images and annotations. Extensive cooperation and sharing of datasets are advantageous for training more powerful, generalizable, and universal models for the identification of 2D materials.

5.1.2 Automatic scanning of 2D materials

Realizing the automatic scanning of 2D materials is a comprehensive task, which requires cooperated calls to components and rational operation logic. The procedures of automatic scanning are systematic, with the fundamental principle of scanning the silicon substrate line by line. When the model identifies a monolayer, the software must take a photo and instruct the motor controller to move and scan the next region of the substrate. Additionally, we design an algorithm to detect the edges and corners of the substrate. After each stage motion, the software runs this algorithm to decide whether to continue moving in the same direction or move to the next line and change direction. During this process, the software must correctly identify corners to ensure the scanning loop can completely cover the substrate and be successfully terminated in the end.

However, the greatest challenge in automatic scanning is the problem of focus loss. There are many reasons for this issue, including the horizontally uneven stage and substrate, and environmental vibrations. As the motorized actuator continues to move the stage in one direction, the distance between

the objective lens and the sample can change, resulting in increasingly blurred images, which prevent the models from making accurate predictions. Therefore, addressing the focus loss issue is crucial for the continued development of auto-scanning of 2D materials.

Currently, the most anticipated solution to focus loss is based on traditional image processing algorithms. By analyzing the current image, we try to achieve dynamic adjustment of the stage's Z position to ensure that the distance between the stage and the sample remains at the focal length.

5.1.3 Vision Transformer

With the tremendous success of the Transformer architecture in the field of natural language processing (NLP), the Vision Transformer (ViT) is rapidly developing and showing impressive performance in computer vision [23]. ViT is a self-attention-based architecture, that is good at capturing global dependencies in images. This capability enables ViT to comprehend objects and their overarching information from a distance, namely global features. Consequently, ViT can identify and understand the global features and patterns of 2D materials, such as the subtle relationships between monolayers and other layers, which can significantly enhance model generalization across different research scenarios. Traditional convolutional neural networks possess built-in inductive biases, such as local connectivity and translation invariance. In contrast, ViT, not being constrained by these prior assumptions, exhibits greater flexibility and adaptability. It may outperform traditional methods when analyzing unconventional images, such as those of 2D materials. Moreover, ViT's scalability in processing large volumes of data is valuable because of the increasing amount of 2D materials datasets.

However, CNNs still dominate the field currently, as they are mature and stable. Additionally, CNNs have an excellent ability for local feature extraction, require less data, and do not consume as many computational resources as ViT does. Therefore, we cannot arbitrarily determine which model has better performance in the identification of 2D materials. A feasible and effective method is to integrate the advantages of both CNN and ViT to develop a hybrid architecture. This approach may achieve optimal performance in the identification of 2D materials, maximizing the benefits that machine learning can bring to the field of physics.

Bibliography

[1] M. Zeng, Y. Xiao, J. Liu, K. Yang, and L. Fu, “Exploring Two-Dimensional Materials toward the Next-Generation Circuits: From Monomer Design to Assembly Control,” Chemical Reviews, vol. 118, no. 13, pp. 6236–6296, Jan. 2018, doi: https://doi.org/10.1021/acs.chemrev.7b00633

[2] H. Li et al., “Rapid and Reliable Thickness Identification of Two-Dimensional Nanosheets Using Optical Microscopy,” ACS Nano, vol. 7, no. 11, pp. 10344–10353, Oct. 2013, doi: https://doi.org/10.1021/nn4047474.

[3] Kanokwan Buapan, Ratchanok Somphonsane, Tinna Chiawchan, and H. Ramamoorthy, “Versatile, Low-Cost, and Portable 2D Material Transfer Setup with a Facile and Highly Efficient DIY Inert-Atmosphere Glove Compartment Option,” ACS Omega, vol. 6, no. 28, pp. 17952–17964, Jul. 2021, doi: https://doi.org/10.1021/acsomega.1c01582.

[4] Q. Zhao, T. Wang, Y. K. Ryu, R. Frisenda, and A. Castellanos-Gomez, “An inexpensive system for the deterministic transfer of 2D materials,” Journal of Physics: Materials, vol. 3, no. 1, p. 016001, Jan. 2020, doi: https://doi.org/10.1088/2515-7639/ab6a72

[5] G. Carleo et al., “Machine learning and the physical sciences,” Reviews of Modern Physics, vol. 91, no. 4, Dec. 2019, doi: https://doi.org/10.1103/revmodphys.91.045002.

[6] K. Suzuki, Artificial Neural Networks: Architectures and Applications. BoD – Books on Demand, 2013. Accessed: Mar. 05, 2024. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=JUufDwAAQBAJ&oi=fnd&pg=PR9&dq=Suzuki

[7] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, “Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification,” IEEE Transactions on Geoscience and Remote Sensing, vol. 55, no. 2, pp. 645–657, Feb. 2017, doi: https://doi.org/10.1109/tgrs.2016.2612821.\

[8] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017, doi: https://doi.org/10.1109/iccv.2017.322

[9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” arXiv.org, Dec. 10, 2015. https://arxiv.org/abs/1512.03385

[10] T.-Y. Lin et al., "Microsoft COCO: Common Objects in Context," Computer Vision – ECCV 2014, vol. 8693, pp. 740–755, 2014, doi: https://doi.org/10.1007/978-3-319-10602-1_48.

[11] S. Masubuchi et al., "Deep-learning-based image segmentation integrated with optical microscopy for automatically searching for two-dimensional materials," npj 2D Materials and Applications, vol. 4, no. 1, pp. 1–9, Mar. 2020, doi: https://doi.org/10.1038/s41699-020-0137-z.

[12] R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448, Dec. 2015, doi: https://doi.org/10.1109/iccv.2015.169.

[13] P. Henderson and V. Ferrari, "End-to-end training of object class detectors for mean average precision," Jul. 2016. Available: https://arxiv.org/pdf/1607.03476.pdf

[14] TorchVision maintainers and contributors, "TorchVision: PyTorch's Computer Vision library," GitHub, Nov. 01, 2016. https://github.com/pytorch/vision/tree/main

[15] NVIDIA maintainers and contributors, "CV-CUDA/python at main · CVCUDA/CV-CUDA," GitHub. https://github.com/CVCUDA/CV-CUDA/tree/main/python

[16] A. Shrestha and A. Mahmood, "Review of Deep Learning Algorithms and Architectures," IEEE Access, vol. 7, pp. 53040–53065, 2019, doi: https://doi.org/10.1109/access.2019.2912200.

[17] A. Hosna, E. Merry, J. Gyalmo, Z. Alom, Z. Aung, and M. A. Azim, "Transfer learning: a friendly introduction," Journal of Big Data, vol. 9, no. 1, Oct. 2022, doi: https://doi.org/10.1186/s40537-022-00652-w.

[18] P. Micikevicius et al., "Mixed Precision Training," arXiv:1710.03740 [cs, stat], Feb. 2018, Available: https://arxiv.org/abs/1710.03740

[19] T. Yu and H. Zhu, "Hyper-Parameter Optimization: A Review of Algorithms and Applications," arXiv:2003.05689 [cs, stat], Mar. 2020, Available: https://arxiv.org/abs/2003.05689

[20] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," arXiv:1404.5997 [cs], Apr. 2014, Available: https://arxiv.org/abs/1404.5997

[21] OpenCV maintainers and contributors, "Open Source Computer Vision Library," GitHub, Dec. 12, 2019. https://github.com/opencv/opencv

[22] Qt Group, "Python UI | Design GUI with Python | Python Bindings for Qt," https://www.qt.io/qt-for-python

[23] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," arXiv:2010.11929 [cs], Oct. 2020. https://arxiv.org/abs/2010.11929

Appendix A

Model Overview

Note: In this table, we only show examples of the basic structure of the backbone networks instead of the complete structure of layers, due to the large size of the model and limited space.

Layer Name	Parameter Shape	Number of Parameters
backbone.body		
conv1.weight	[64, 3, 7, 7]	9,408
bn1.weight	[64]	64
bn1.bias	[64]	64
layer1.0.conv1.weight	[64, 64, 1, 1]	4,096
layer1.0.downsample.0.weight	[256, 64, 1, 1]	16,384
layer1.0.downsample.1.weight	[256]	256
layer1.0.downsample.1.bias	[256]	256
...	...	...
backbone.fpn		
inner_blocks.0.weight	[256, 256, 1, 1]	65,536
inner_blocks.0.bias	[256]	256
inner_blocks.1.weight	[256, 512, 1, 1]	131,072
inner_blocks.1.bias	[256]	256
inner_blocks.2.weight	[256, 1024, 1, 1]	262,144
inner_blocks.2.bias	[256]	256
inner_blocks.3.weight	[256, 2048, 1, 1]	524,288
inner_blocks.3.bias	[256]	256
layer_blocks.0.weight	[256, 256, 3, 3]	589,824
layer_blocks.0.bias	[256]	256
layer_blocks.1.weight	[256, 256, 3, 3]	589,824
layer_blocks.1.bias	[256]	256
layer_blocks.2.weight	[256, 256, 3, 3]	589,824
layer_blocks.2.bias	[256]	256
layer_blocks.3.weight	[256, 256, 3, 3]	589,824
layer_blocks.3.bias	[256]	256

rpn.head		
conv.weight	[256, 256, 3, 3]	589,824
conv.bias	[256]	256
cls_logits.weight	[3, 256, 1, 1]	768
cls_logits.bias	[3]	3
bbox_pred.weight	[12, 256, 1, 1]	3,072
bbox_pred.bias	[12]	12
roi_heads.box_head		
fc6.weight	[1024, 12544]	12,845,056
fc6.bias	[1024]	1,024
fc7.weight	[1024, 1024]	1,048,576
fc7.bias	[1024]	1,024
roi_heads.box_predictor		
cls_score.weight	[4, 1024]	4,096
cls_score.bias	[4]	4
bbox_pred.weight	[16, 1024]	16,384
bbox_pred.bias	[16]	16
roi_heads.mask_head		
mask_fcn1.weight	[256, 256, 3, 3]	589,824
mask_fcn1.bias	[256]	256
mask_fcn2.weight	[256, 256, 3, 3]	589,824
mask_fcn2.bias	[256]	256
mask_fcn3.weight	[256, 256, 3, 3]	589,824
mask_fcn3.bias	[256]	256
mask_fcn4.weight	[256, 256, 3, 3]	589,824
mask_fcn4.bias	[256]	256
roi_heads.mask_predictor		
conv5_mask.weight	[256, 256, 2, 2]	262,144
conv5_mask.bias	[256]	256
mask_fcn_logits.weight	[4, 256, 1, 1]	1,024
mask_fcn_logits.bias	[4]	4
Total Number of Parameters		78,622,055
Number of Trainable Parameters		78,396,711