MACHINE LEARNING-BASED METHODS FOR THE SEGMENTATION OF SCANNING ELECTRON MICROSCOPY IMAGES OF FINE-GRAINED SHALE SAMPLES

by

BINQIAN YIN

DISSERTATION

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in
Earth and Environmental Sciences at
The University of Texas at Arlington
December 2022

Arlington, Texas

Abstract

Machine Learning-based Methods for the Segmentation of Scanning Electron Microscopy Images of Fine-Grained Shale Samples

Binqian Yin, Ph.D.

The University of Texas at Arlington, 2019

Supervising Professor: Qinhong Hu

The segmentation of scanning electron microscopy (SEM) images is critical yet time-consuming for geological studies, as it will need to differentiate the boundaries for different mineral objects to facilitate subsequent analyses, such as porosity calculation. Recently, machine learning methods, especially convolutional neural networks (CNNs), have been explored to segment SEM images for fine-grained shale samples. However, existing methods fail to address two critical issues in the segmentation of shale rock images---insufficient labeled data and imbalanced objects. To this end, this dissertation has proposed a machine learning pipeline that consists of supervised, semi-supervised, and active learning stages to reduce manual efforts in segmenting shale rock images with limited label data and imbalanced object distribution. The supervised learning method incorporates ensemble learning with a new loss function to tackle the imbalanced object problem, achieving 9% higher mIoU than the state-of-the-art method. The semi-supervised method yields an acceptable accuracy using only 6% labeled data. This work has also proposed a novel algorithm to speed up the semi-supervised loss function in SU-Net using caching, skip zeros, and batching optimizations. Finally, this study has developed the first active learning-based segmentation pipeline that only selects a small number of images that need labels to achieve maximum accuracy. This search prepared 5000 shale rock image cuts in the experiments and divided them into training, validation, and testing datasets. The experimental results show that the methods proposed in this dissertation can clearly distinguish each object from others with boundaries by using significantly fewer labeled images than existing methods. Therefore, these methods are cost-effective to help geoscientists gain insights by building neural network models from a small dataset of SEM images.

Acknowledgments

I would like to acknowledge financial support from the Nuclear Energy University Program managed by the Office of Nuclear Energy at the U.S. Department of Energy (award number DE-NE0008797), the AAPG Grants-in-Aid program (Kenneth O. Stanley Memorial Grant in 2021), and Graduate Research Grant of Geological Society of America (2021). I am very honored to be the recipient of AAPG and GSA scholarships, and I'm extremely grateful to those who provided me with generous support.

I would also like to extend my sincerest gratitude to all those who have contributed to my research in other ways—by offering guidance, constructive feedback, or invaluable mentorship. As an early-career researcher, these forms of support mean a great deal to me, and I don't take them for granted. With the help of these grants and resources, I look forward to advancing my work and contributing meaningful insights to my field. First, I would like to express my deepest gratitude to my Ph.D. supervisor: Dr. Qinhong Hu. His invaluable advice, continuous support, and patience helped me get to this stage; his rigorous work attitude and excellence in academic research encouraged me throughout my academic research and daily life. I could only have undertaken this journey with Dr. Yingying Zhu, who generously provided knowledge and expertise when I hit the bottleneck during my research. It has also been an amazing experience working at Pacific Northwest National Laboratory during the summer of 2021. I would like to extend my sincere thanks to my mentor: Dr. Jiajia Li; thank you so much for your patient guidance and help during my internship, as the positive and creative research attitude inspires me deeply.

Thanks, of course, to all my family for their invaluable support. Thank you, my mom. Thank you for your emotional support that nothing can replace; you are always my endless driving force in my life. Thank you, my fiancé, Keren. You are the best person I have ever met. Thank you for your selfless help and support; you are my pillar of strength. I would also like to express my gratitude to my friends, who have served as a wonderful source of encouragement over the years. Even in my most challenging moments, you all always remind me that I'm capable of achieving great things.

Finally, I would like to express my heartfelt appreciation to my beloved pets: my angel turtle Daroubao, my bunny Doudou, and my kitty Lucas. I'm so grateful for the comfort and companionship they provide me with each day. Even in my darkest moments, they're always there to lift my spirits and bring light into my life.

Nov. 2022

Dedication

This dissertation is made possible by the unfaltering love and support of my parents, family members, pets, and fiancé. I owe them a debt of gratitude for sticking with me through the ups and downs of my academic journey.

Table of Contents

List of Figures

Chapter 1: Introduction

Since 2000, the breakthrough in developing shale resources has rapidly increased unconventional oil and gas production to alleviate the shortage of conventional oil and gas resources. As the most promising unconventional oil and gas resources, shale formations have brought a huge commercial economy to the petroleum industry in the past several decades, positioned as the largest source of natural gas and crude oil production in the USA (The U.S. Energy Information Administration, 2021).

Fine-grained shale is primarily composed of clay-sized sediments, organic matter, pores, as well as other minerals. Unlike conventional reservoirs, the petro-physical properties of shale rocks are unique, especially considering their submicron-range pore structure (e.g., porosity, pore size and distribution, pore shape, and pore connectivity). Nano-scale pores and throats in shale rocks are the primary factors causing the pore connectivity limitation to significantly influence fluid flow and petroleum production. The morphology of the pores also substantially influences the wettability and phase behavior of shales. The phenomenon that a substantial proportion of the pores being hosted with organic matter is ubiquitous in over-mature shales, which have different wettability from the pores in the mineral grains, increasing the complexity of the enhanced recovery process during the petroleum development (Passey et al., 2010). Thus, characterizing the pore structure of shales has a critical impact on the efficiency of shale petroleum exploration. There has been increasing attention devoted to the studies of fine-grained shale on connectivity between pores, as well as liquid flows through, and wettability and permeability of shale (Huang et al., 2020; Huang et al., 2021; Meng et al., 2022; Wang et al., 2020; Wang & Wang et al., 2022), to better understand the impact of shale characteristics. However, the methodology is still evolving because of the challenges posed by shale's distinctive structures.

Advanced techniques of digital rock analyses (Vernon-Parry, 2000) have become increasingly important for studying complex shale rock structures at micron scales. Scanning electron microscopy (SEM) techniques for shale samples provide insights to geoscientists on the characteristics of featured geological objects and their spatial variations (Loucks et al., 2009). SEM uses a focused beam of electrons to scan the surface of a sample, which can be secondary electrons (Seiler et al., 1983), backscattered electrons (Lloyd, 1987), or characteristic X-rays (Suryanarayana & Norton, 1998). Secondary electrons provide information about surface topography, backscattered electrons describe surface heterogeneity, and characteristic X-rays are used to estimate the distribution of elements. The physical properties of shale, such as pore size and distribution, mineral grain composition, and clay particles, can be obtained from SEM images through subsequent analyses (Loucks et al., 2009; Yang et al., 2016).

Image processing techniques have been widely applied for automated analyses (Azarafza et al., 2019; 2021; Mahmoodi-Eshkaftaki et al., 2020; Shivhare & Cecil, 2021). Shale SEM analyses aim to extract statistical information from raw images, for which segmentation is an indispensable step to identify mineral phases and pore structures in this process. Nevertheless, traditional segmentation is a time-consuming process that requires significant manual efforts. In shale samples, the nano-scale pores are located between and within fine-grained minerals and organic matter, and the complex pore structure causes difficulties in segmentation. To accelerate segmentation, the thresholding method (Sahoo et al., 1988) is a widely used method to create labels from grayscale images by a fixed pixel threshold. However, by using color differences of pixels, thresholding cannot apply in all scenarios, especially when two objects have a large overlap in colors.

In the past decade, advances in image segmentation driven by deep learning have attracted attention from researchers as they can automatically identify pixel-level information in images with demonstrated high accuracy. Therefore, there has been unprecedented rapid development in deep learning-based segmentation in the past several years (Minaee et al., 2021). The first step in deep learning-based image segmentation methods is to extract image features. Convolutional neural networks (CNNs) have been shown to be remarkably effective in extracting features from images. While the CNN was first proposed to classify simple images (LeCun et al., 1998), it has been rapidly developed in the past decade to solve more complicated tasks, such as image segmentation (He et al., 2016), image restoration (Deng et al., 2017), and image enhancement (Tao et al., 2017). In addition, CNNs have also been adopted in other applications, including cancer detection (Wang et al., 2019), autonomous driving (Zhu et al., 2016), and face detection (Jiang & Learned-Miller, 2017).

Notably, among all CNN architectures, the U-Net architecture (Ronneberger et al., 2015) is one of the most popular ones for segmentation, and it has achieved state-of-the-art results in natural and medical images. U-Net consists of an encoder and a decoder. The encoder takes as input an image and produces features, which are then fed into the decoder to generate a segmentation mask. Since U-Net, its variants with the U-shape architecture have been developed to improve its performance and reduce its running time. SegNet (Badrinarayanan et al., 2017) reduces the computation cost by employing the max-pooling indices of the encoder for up-sampling instead of using transposed convolution. On the other hand, wide U-Net and U-Net++ (Zhou et al., 2018) have been proposed to improve the original U-Net by combining the input with the output of transposed convolutions at each level, which requires several skip connections to convey low-level features directly from the encoder to the decoder. In addition, Residual U-Net (Zhang et al., 2018) uses the residual blocks (He et al., 2016) in the encoder and decoder parts of the U-Net architecture to improve the network's performance.

Recent research has novel machine learning methods to automate shale image segmentation. Zhang et al. (2019) used a pre-trained Inception-v3 model published by Szegedy et al. (2016) to extract features and combine the results from several machine learning algorithms. Chen et al. (2020) employed U-Net to segment some minerals and adopted thresholding to segment other minerals. Karimpouli & Tahmasebi (2019) proposed methods to augment the dataset and extend the SegNet architecture to achieve better performance than threshold-based methods. Wang & Dalton et al. (2022) compared the segmentation accuracy using U-Net, wide U-Net, and UNet++ on digital rock images. Cao et al. (2022) proposed a joint-learning framework to simultaneously tackle digital rock image segmentation and petrophysical-elastic parameters computation.

However, the above approaches do not address two critical problems that commonly exist in fine-grained shale samples---imbalanced object distribution and scarcity of labeled data. Existing approaches do not address an important problem in fine-grained shale samples with grain sizes smaller than 62.5 μm. In many shale images, minerals are highly imbalanced, indicating that some minerals take up a large proportion of the dataset while others only take up a very small proportion. General CNNs are likely to achieve sub-optimal performance by learning imbalanced labels. Karimpouli & Tahmasebi (2019) tackled this problem by augmenting images from the original dataset. Moreover, traditional augmentation techniques such as cropping, flipping, and rotation do not change the ratio between objects. U-Net uses a weighted loss function to tackle this problem, in which a pre-computed weight map for each pixel is used to balance the frequency of pixels of each class. The weight map, however, is computed by using heuristic functions before training, rendering suboptimal performance in many cases (Sudre, 2017).

While increasing the size of training data may mitigate the problem, training a much larger dataset can be costly in terms of time and resources. In our experience, labeling a medium-sized raw image pixel-wise can take 2-3 hours. However, existing approaches also rely on a large amount of training data to achieve satisfactory performance; they all follow the supervised learning approach that requires manual labels to provide the ground truth in the training stage (Russell & Norvig, 1995). In contrast to supervised learning, unsupervised learning generates models with features from raw data without labels (Bishop & Nasrabadi, 2006). Semi-supervised (or weakly supervised) learning has also been employed to tackle the challenges of obtaining labels (Zhu et al., 2009). Compared with unsupervised learning approaches, semi-supervised approaches train models using a significant amount of unlabeled data and a small amount of labeled data. Active learning techniques are often employed to select the most representative samples when using semi-supervised learning or supervised learning (Settles, 2009).

My research focuses on designing effective machine learning methods to segment objects in shale rock images automatically. I proposed a series of approaches to address the above problems. First, I investigated existing approaches for imbalanced shale rock images and pinpointed the root cause of low accuracy in overweighing the loss of highly frequent objects. To address this issue, I proposed a novel ensemble learning method to train separate neural networks for individual objects, with each neural network employing the focal loss function to focus on pixels that are difficult to identify (Lin et al., 2017). Next, I dealt with a larger dataset containing many unlabeled shale images. Since manually labeling these images are costly and error-prone, I investigated both unsupervised and semi-supervised learning using fewer training data. I found that though unsupervised approaches can distinguish objects with far distances or distinct colors, they are insufficient for close objects with similar colors. To this end, I built a semi-supervised learning model that achieves comparable accuracy with supervised learning approaches but requires significantly fewer data. Using the semi-supervised approach, I can segment images with similar attributes as the learned dataset, but if the new dataset has vastly different images, our semi-supervised model could render inaccurate results. Based on the observation, I investigated the active learning approach to discover a subset from raw images such that they could improve a model's accuracy if labeled. Overall, I derived a hybrid active learning approach that addresses the problem by employing both contrastive learning (Chuang et al., 2020) to discover diverse samples and uncertainty sampling (Zhu & Goldberg, 2009) to obtain highly uncertain samples.

The hypothesis of the dissertation is proposed below:

Using a machine learning pipeline that consists of supervised, semi-supervised, and active learning stages, we can significantly reduce manual efforts in segmenting shale images with limited label data and imbalanced object distribution.

This dissertation describes three major contributions:

- Designing the first ensemble learning-based model to segment shale images with imbalanced objects, outperforming the state-of-the-art approach by 9% mIoU (mean Intersection Over Union);
- Investigating unsupervised and semi-supervised methods for segmenting images of shale and proposed an innovative semi-supervised model that can achieve acceptable accuracy using only 6% labeled data and can be trained as fast as U-Net;
- Developing the first active learning-based segmentation pipeline that optimizes both uncertainty and diversity to select only a small quantity of images that need labels to achieve acceptable accuracy.

This dissertation is divided into six additional chapters, each covering distinct facets of my research. Chapter 2 goes into detail on the relevant literature concerning machine learning, deep learning, and image segmentation. I introduce the dataset central to my work in Chapter 3. Chapter 4 first describes my ensemble learning model. Then Chapter 5 focuses on semi-supervised learning and the benefits of minimizing label requirements. Chapter 6 discusses the innovative active learning-based approach for shale segmentation. Lastly, in Chapter 7, I synthesize this dissertation work and propose areas for future investigations.

Chapter 2: Literature Review

This chapter supplies necessary background information for the methods and reviews existing machine learning-based approaches for shale image segmentation.

2.1. Types of Machine Learning Problems

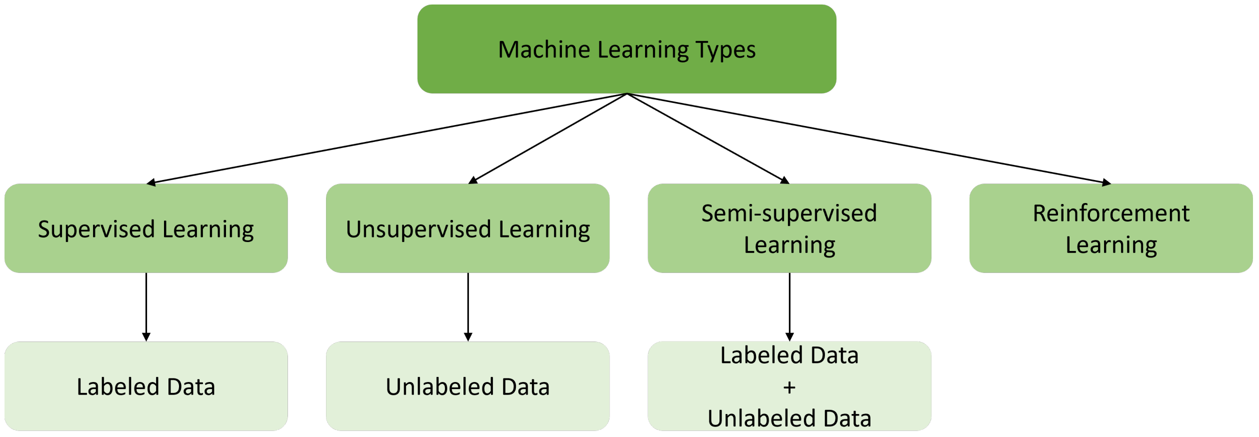

Figure 1 Categorization of typical machine learning problems

Machine Learning (Jordan & Mitchell, 2015) is an artificial intelligence approach that improves algorithm performance through experience and using data. Figure 1 presents common types of machine learning problems and highlights their differences in required datasets. The following sections describe key knowledge in supervised learning, unsupervised learning, and semi-supervised learning. Reinforcement learning is not covered in this dissertation.

2.1.1. Supervised and Unsupervised Learning

Traditionally, machine learning problems can be divided into two categories (Bishop & Nasrabadi, 2006): supervised and unsupervised learning. In supervised learning, we are given a dataset with input X and their corresponding values Y, where Y can be categorical or continuous. Then, we build a classifier for categorical data or a regressor for continuous data by learning features from the given dataset. The model that we build is supposed to predict value Y' for future unseen input X'. A representative of supervised learning is predicting what category an image belongs to by training a large dataset in which each image has a label.

In contrast to supervised learning, we are only given X without their corresponding values using unsupervised learning. As a result, we can only infer the underlying relationship among samples in the given dataset. A common example of unsupervised learning is clustering customers into groups that share common attributes to build business strategies.

2.1.2. Semi-supervised learning

Semi-supervised learning (Chapelle et al., 2009) combines supervised learning and unsupervised learning, which learns a model from a small amount of labeled data and a large amount of unlabeled data. Intuitively, we can adopt the following heuristic approach for semi-supervised learning (Triguero et al., 2013):

- First, we train a model using a supervised learning algorithm with labeled data only.
- Then, we apply this model for unlabeled data, add outputs with high confidence to the labeled data, and retrain the model using all labeled data.
- The above two steps can be repeated many times until the model yields accurate results.

Otherwise, recent progress (Yang et al., 2021) has been focused on automated semi-supervised learning approaches, including generative models, consistency regularization models, graphic-based models, and pseudo-labeling models. It is worth noting that semi-supervised learning does not always yield satisfactory performance. Certain assumptions (Zhu & Goldberg, 2009) need to hold for the training data, including:

- Smoothness assumption: data that are close to each other are more likely to share a label.
- Cluster assumption: data in the same cluster are more likely to share a label.
- Manifold assumption: data lie approximately on a manifold of much lower dimension than the input space.

In a word, if one could take leverage the knowledge on $p(x)$ gained through unlabeled data for

$p(y|x)$, where y is the label to predict, semi-supervised learning will yield an improvement.

2.2. Effective Learning Techniques

Most commonly, we train a model from scratch and use it for interference. There also exist techniques to boost performance, and we describe the most relevant ones in the following sections.

2.2.1. Ensemble Learning

Supervised learning algorithms build a single model that is the best fit for a particular problem. Ensemble learning applies multiple learning algorithms and combines their results to obtain better performance than using individual algorithms alone. The execution of ensemble learning can be divided into two phases. In the first phase, multiple models are trained to yield the best result on a given dataset, which is usually a partition on the raw dataset. In the second phase, the outputs of different models are fused based on simple heuristic methods or a new model. The most common ensemble methods include bagging (Breiman, 1996), boosting (Freund et al., 1999), and stacked generalization (Wolpert, 1992), as shown in Figure 2.

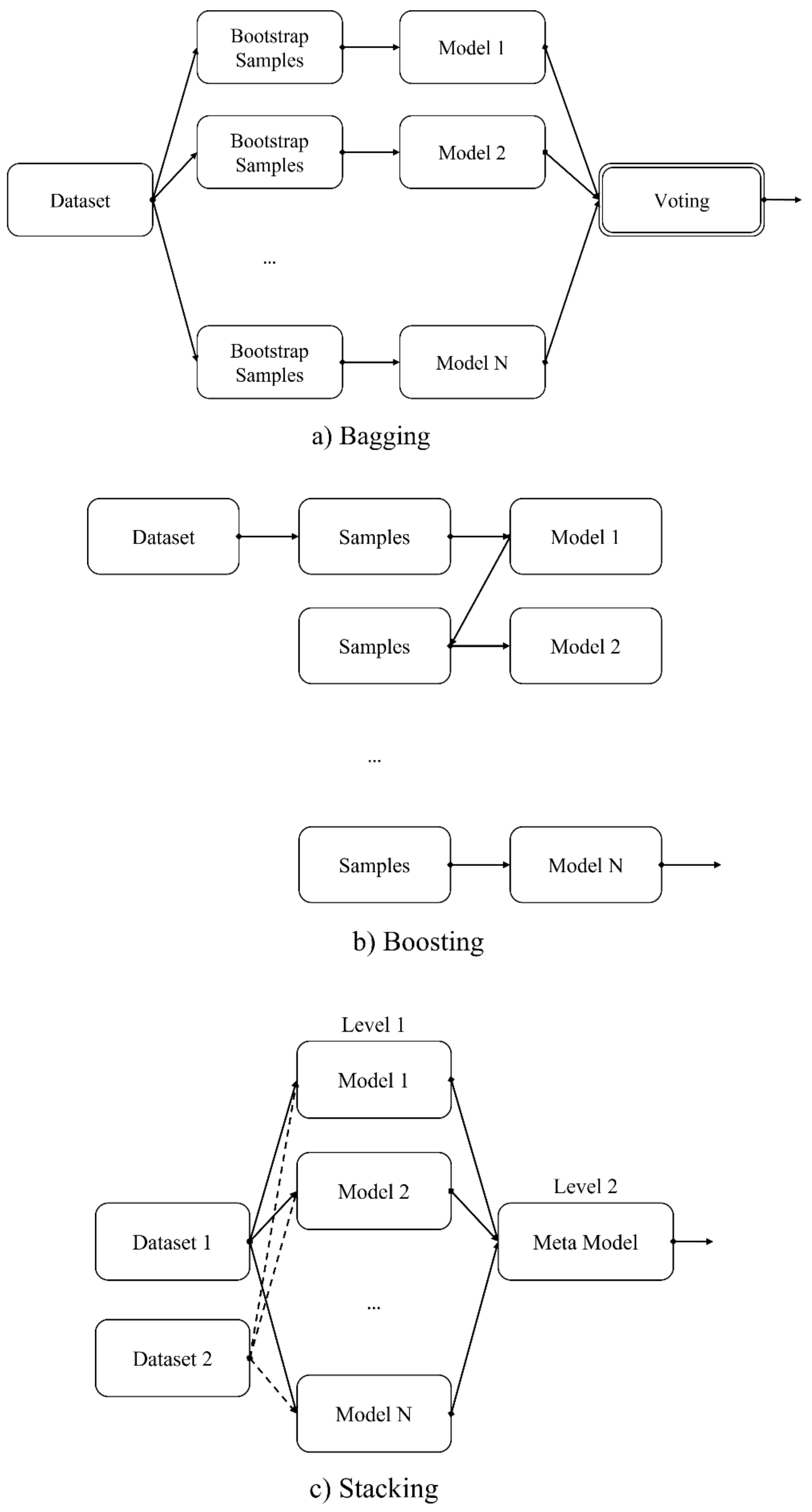

Figure 2 Diagrams of different ensemble learning methods

Bagging applies the same learning algorithm for multiple sampled training datasets to train different models. Usually, the training datasets are generated by re-sampling the raw training dataset. To combine the results of individual models, simple approaches such as majority voting or calculating averages are used.

Unlike bagging, boosting iteratively generates new training datasets according to each sample's weight. In the raw dataset, each sample has an equal weight. We can reduce the weight of correctly predicted samples in the subsequent training datasets but increase the weight of mis-predicted samples. In this way, the subsequent models focus more on mis-predicted samples because the weighted sum measures a model's accuracy among all samples.

Stacked generalization (or stacking) uses multi-level training. It first trains multiple models using the raw dataset. Then, it feeds the output of the models to the second phase to train a meta-learner. Since the meta-learner learns to combine each model's prediction, it reduces the variance in the prediction. To avoid overfitting in stacked generalization, we can divide the dataset into two folds: we used the first fold to train level 1 models and gather outputs to train the meta learner (Level 2 model); then we abandoned the Level 1 models trained before and used the second fold to train a new set of Level 1 models.

2.2.2. Multi-task Learning

Multi-task learning (MTL) (Zhang et al., 2019) is an approach in machine learning which aims to improve the overall performance of multiple related tasks by considering them jointly during training. In a typical MTL setup, a shared representation or feature extractor is learned across multiple tasks with the goal of leveraging information from each individual task to benefit others.

The underlying premise of MTL is that learning multiple tasks together often leads to stronger generalization performance than learning them separately (Wang et al., 2017), as each task will contribute knowledge that can directly or indirectly help other tasks. This advantage can be particularly pronounced when tasks share common input features and output structures or when there is limited training data available for one or more tasks. MTL also allows for greater efficiency by reducing the number of parameters and computations required to train multiple tasks.

MTL has been applied in a wide variety of fields and applications, such as computer vision (Gao et al., 2020), natural language processing (Worsham & Kalita, 2020), medical imaging (Gao et al., 2020), speech recognition (Pironkov et al., 2016), and robotics (Byambatsogt et al., 2020). Different methods for MTL vary in terms of the specific architecture or training regime employed to share knowledge among tasks. Common approaches include leveraging a shared representation layer or using regularization methods to avoid overfitting to a single task. In summary, multi-task learning is a powerful paradigm in machine learning which enables knowledge sharing among tasks, leading to better generalization performance and resource efficiency.

2.2.3. Active Learning

Typically, machine learning models are trained with available samples from the dataset. Active learning uses the assumption that a model can achieve high accuracy with fewer training labels if it is allowed to choose the data from which it learns (Settles, 2009). An active learner may pose requests through an acquisition function for labeling certain through an oracle (e.g., a human expert). Active learning is well-suited to our dataset because our dataset has abundant unlabeled data may be abundant while labels are expensive to obtain.

Figure 3 describes the major steps in active learning. The most important and unique step in active learning is the query stage which selects samples that are expected to produce higher value for model training if they get labeled. Most research focuses on effective and efficient acquisition functions to decide which instances are most informative.

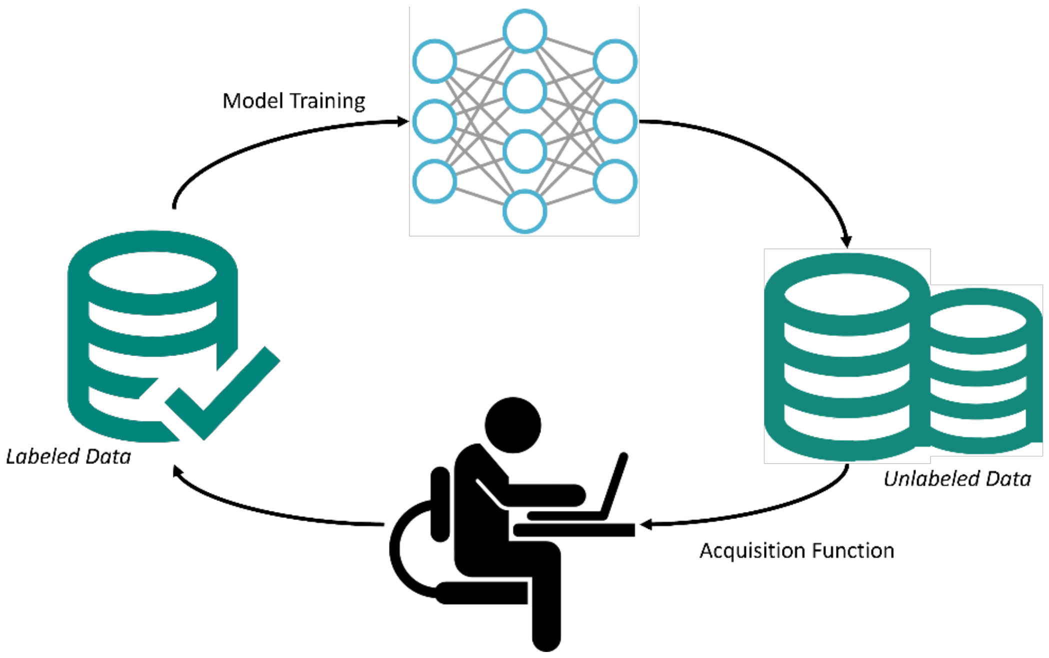

Figure 3 Steps in active learning

The acquisition function considers features of the data and the model when selecting unlabeled data. For example, uncertainty sampling (Lewis et al., 1994) allows for the selection of unlabeled data instances that the current model is uncertain about. Different methods of estimating uncertainty can be used, and the best choice often depends on the problem at hand and the specific characteristics of the dataset. By identifying instances where the model has low confidence, active learning can prioritize the labeling of those instances to improve the model in areas where it is currently weak.

Compared with uncertainty sampling, which finds out what the current model doesn’t know, diversity sampling identifies what’s missing from the model (Yang et al., 2015). Specifically, since the current model hasn't seen the entire dataset, the acquisition function tries to find a collection of samples that can well stand for the entire data distribution.

Recently, there have also been new and merged methods such as the hybrid strategy (Yang et al., 2017), expected model change (Cai et al., 2013), and loss prediction (Yoo & Kweon, 2019). In the hybrid strategy, different active learning strategies are combined. This can allow for more efficient use of labeled and unlabeled data by taking advantage of techniques such as uncertainty sampling, diversity sampling, or clustering. Expected model change prioritizes data instances that are expected to lead to the greatest change in the model if they were to be labeled and added to the training set. This method aims to maximize the impact of new data instances to improve the model more quickly. For example, the Loss Prediction technique uses the model's loss function to estimate the expected error of the model on an unlabeled instance. Data instances with a higher expected error are prioritized for labeling since they are likely to be more informative for the model. In this dissertation, I focus on investigating the uncertainty rule and the diversity rule because the goal of these approaches is still to find the most uncertain and diverse samples, but they don’t use uncertain or diverse scores directly.

2.3. Deep Learning

Deep learning is a collection of machine learning methods using artificial neural networks, which are inspired by information processing and communication of neurons in biological systems (LeCun et al., 2015). Today, deep learning has been applied to a wide range of important fields, including computer vision, image analysis, natural language processing, machine translation, bioinformatics, and climate forecasting. In the rest of this section, I will concentrate on three neural networks that are most relevant to the dissertation research.

2.3.1. Convolutional Neural Networks

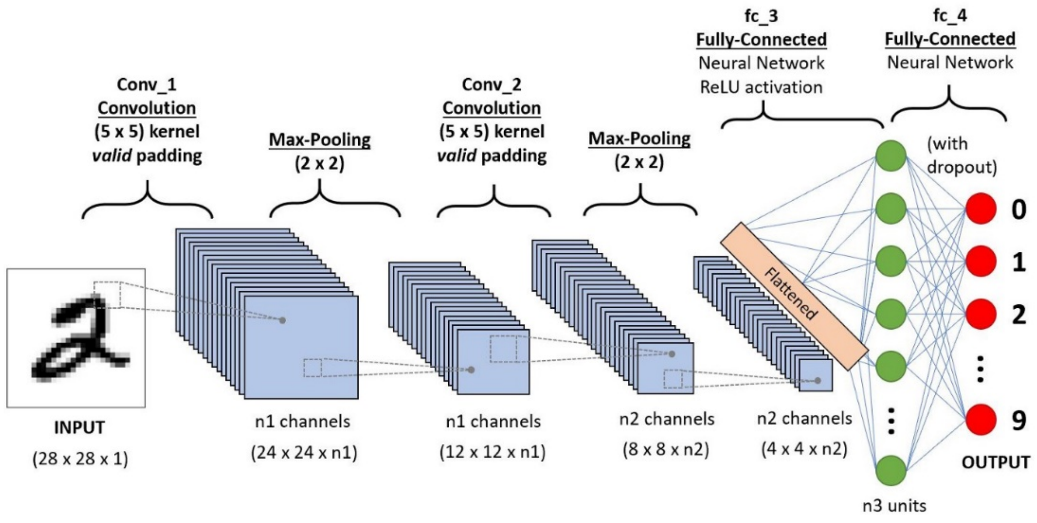

Figure 4 A CNN sequence to classify handwritten digits (Saha, 2018)

As a class of artificial neural networks, a convolution neural network comprises layers with many neurons whose values are adjusted through backward propagation algorithms (LeCun et al., 1988). CNNs are widely used for handling image tasks, such as segmentation, object detection, and classification (Krizhevsky et al., 2012; He et al., 2016; Redmon, 2016). A CNN typically contains the following components:

- Convolution layers are the building blocks to extract features from images. A convolution operation applies filters to scan the whole image by sliding, perform a dot product on each tile, and output results as new features for the next layers. By increasing the number of filters (i.e., output channels) in a convolution layer, we can increase the number of features for each pixel in the output.

- Pooling layers execute down-sampling operations to reduce the size of inputs. A local window that performs down sampling is called a kernel. For example, in a Max Pooling layer, the input will be condensed by choosing the local maximum data within each kernel.

- Up-sampling layers increase the size of inputs. One can implement an up-sampling layer with a transposed convolution operation (Noh et al., 2015) or a direct up-sampling operation (Oppenheim, 1999) by calculating output using an interpolation of neighbors.

- Concatenation layers concatenate inputs with the same shapes except for the concatenation dimension. For instance, if the concatenation dimension is the feature dimension, the output concatenates the features of all inputs.

- Activation functions are generally used to introduce non-linearity of the output of each layer. Popular activation functions include Gelu (Hendrycks & Gimpel, 2016), Relu (Dahl et al., 2013), Sigmoid (Narayan, 1997), and Tanh (Kalman, 1992).

- Loss functions measure the differences between the targets and the predicted results to prepare gradients for backward propagation. Cross-entropy loss is most widely used in image classification and segmentation (Zhang & Sabuncu, 2018).

2.3.2. Autoencoders

Autoencoder is a type of CNN that learns representative code of input data (Kramer, 1991). As shown in Figure 5, an autoencoder consists of an encoder and a decoder. The encoder maps the input into the code and the decoder that maps the code to a reconstruction of the input.

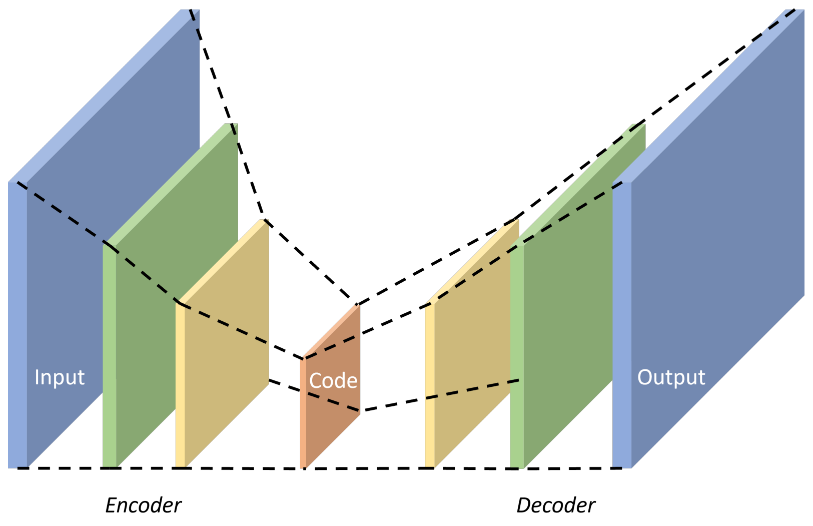

Figure 5 The architecture of the autoencoder

Once an autoencoder has been trained, we can use it to reconstruct data. However, we are unlikely to use an autoencoder to produce any new content based on the features of the input. The latent space (i.e., code) of an autoencoder is learned to encode and decode with as few losses as possible, lacking interpretable and exploitable structures. Variational autoencoders (VAEs) (Kingma & Welling, 2013) can be defined as an autoencoder whose training is regularized to avoid overfitting and ensure that the latent space can enable the generative process. Unlike autoencoders that learn how to minimize the difference between input and output, VAEs try to minimize both the reconstruction loss and a regularization term.

2.3.3 U-Net

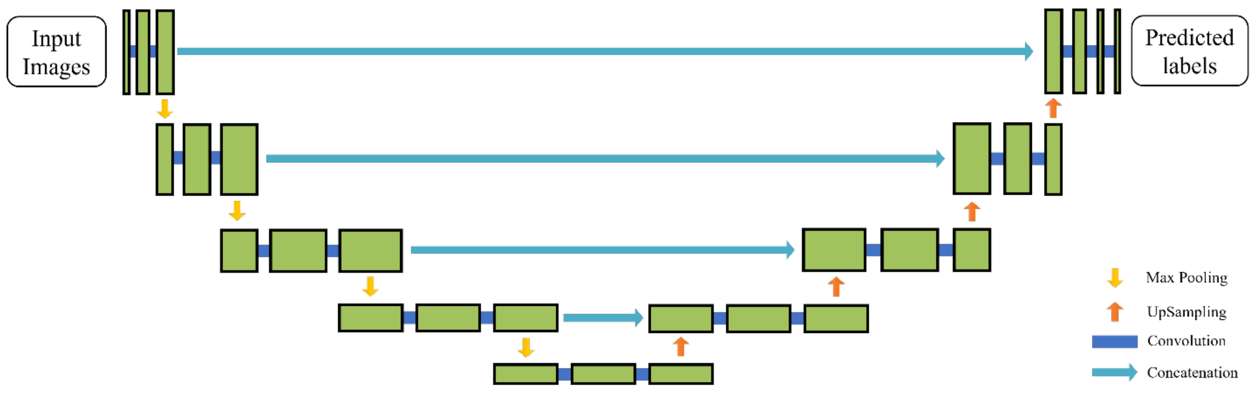

Figure 6 The architecture of U-Net

U-Net is a type of CNN designed to work with small datasets (Ronneberger et al., 2015). Figure 3 shows the architecture of the original U-Net model, which consists of a contraction path and an expansion path, making it a U-shaped structure. There are nine blocks in the U-Net architecture, and each of them is composed of two convolution layers with a filter size of 3×3. In the contraction path, five basic blocks are linked by “Pooling layers” with kernels of size at 2×2, reducing the spatial dimension of outputs. In the expansion path, “Up-sampling layers” link five basic units to increase the size of outputs. Besides, “Concatenate layers” combine the outputs from the contracting path and the outputs from up-sampling layers to yield high-resolution features. The final output of a U-Net is a probability map of each category for each pixel in the input image, and the predicted label of each pixel refers to the category with the maximum probability.

2.3.4. Generative Models

Generative models are a class of machine learning algorithms that aim to learn the underlying probability distribution of a dataset to generate new data samples from that distribution (Salakhutdinov, 2015). They are often used in tasks where a large quantity of data is desired but difficult to obtain or expensive to labels, such as in image or language modeling.

There are many different types of generative models, each with its own strengths and weaknesses. Some popular examples include generative adversarial networks (GANs) (Goodfellow et al., 2014), variational autoencoders (VAEs) (Kingma et al., 2013), and autoregressive models. Overall, generative models are a powerful tool for machine learning and data science, enabling the generation of large datasets with properties like the original distribution. They have been used in a wide range of applications, such as generating realistic images for computer graphics (Lassner et al., 2017), designing new drugs in the pharmaceutical industry (Sanchez-Lengeling & Aspuru-Guzik, 2018), and enhancing natural language generation in chatbots or virtual assistants (Lamprier et al., 2022).

2.3.4.1 Variational Autoencoder (VAE)

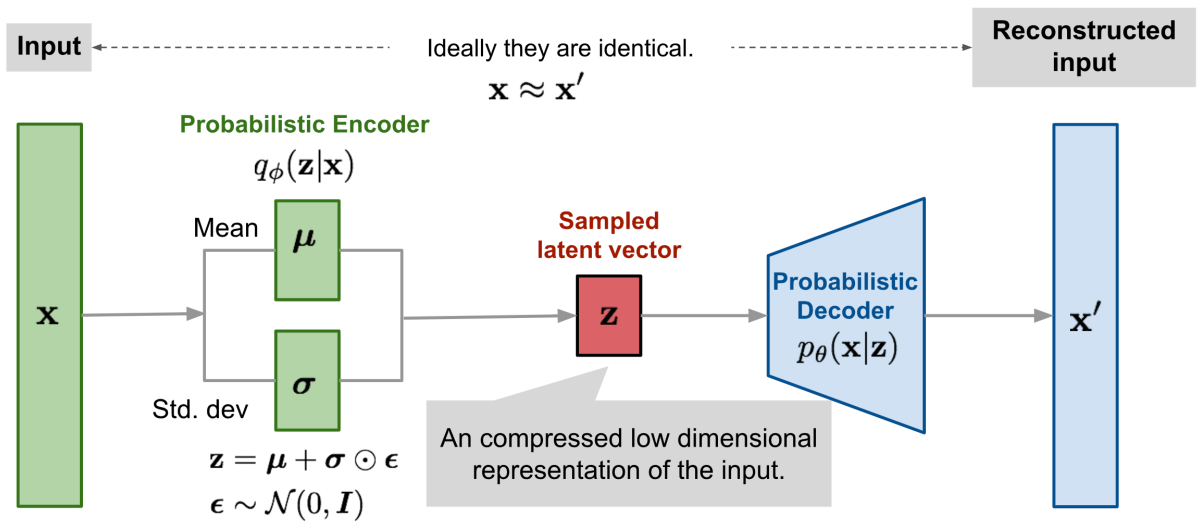
Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{x}$
Mean
$\boldsymbol{\mu}$
$\boldsymbol{\sigma}$
Std. dev
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

Figure 7 The architecture of VAE (Weng, 2018)

Variational autoencoders (VAEs) are a type of generative model which uses an encoder-decoder architecture to model the underlying probability distribution of a dataset (Kingma et al., 2013). They are particularly powerful for tasks such as image or language generation where there is a large amount of unlabeled data available. The key principle behind VAEs is that they seek to encode input data into a low-dimensional latent space, which preserves the most important features of the data while discarding noise and irrelevant details. This latent representation can then be used to generate new data samples by decoding them back into the original input space. VAEs differ from traditional autoencoders in that they impose a constraint on the latent representation - specifically that it should follow a standard normal distribution. This results in the model acting as a probabilistic encoder as well as a decoder, enabling the generation of new data samples with a high degree of variability.

Figure 7 describes the architecture of a typical VAE. Interestingly, the training process of VAE falls into the Multi-Task Learning category; this work tries to minimize both the reconstruction loss and the model distribution loss. Here we assume the output of the probabilistic encoder matches

Standard Gaussian Distribution $\epsilon \in \mathcal{N}(0,1)$. However, a classical method to compare against two distributions would require Monte Carlo sampling (Hastings, 1970), which is not differentiable. Hence, the authors of VAE proposed a so-called reparameterization trick to transform variables and enable backward propagation.

2.3.4.2. Generative Adversarial Networks (GANs)

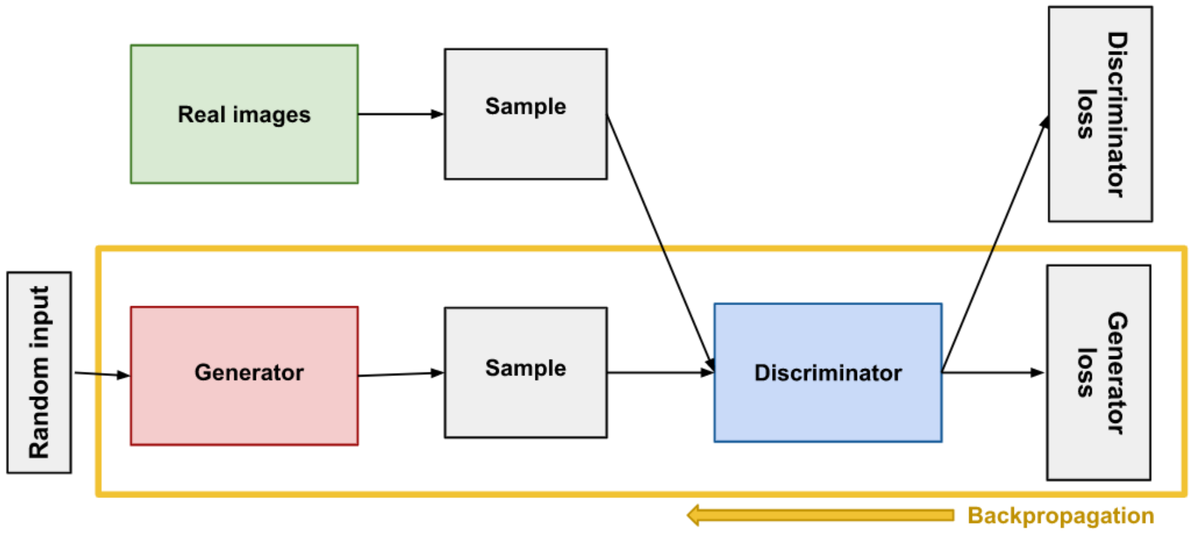

Figure 8 Training Generative Adversarial Networks (Google, 2019)

Like VAEs, Generative Adversarial Networks (GANs) can generate new data with the same statistics as the training set (Goodfellow et al., 2014). Figure 8 presents the typical structure of a GAN, which consists of two networks—a generator and a discriminator. The generator takes random inputs as seeds to generate images, while the discriminator differentiates between the generated fake images and real images. In the training process, the generator learns to increase the error rate of the discriminative network by generating fake images like real images, while the discriminator learns decision boundaries to distinguish real and fake images.

GAN applications have increased rapidly in art, science, audio, and video domains. For example, based on an analysis of portraits, an image generated by a StyleGAN looks deceptively like a photograph of a real person (Karras et al., 2019). In lots of tasks, GANs are typically superior as deep generative compared to VAEs. However, GANs are more difficult to work with and require plenty of data and tuning.

2.4. Image Segmentation

The demand for automated image segmentation is imminent in many domains, such as identifying cell boundaries in biomedical image analyses (Ronneberger et al., 2015), removing all linkages to Endnote object capturing in autonomous driving technology (Zhu et al., 2016), remote sensing image analysis for GIS data (Zhang et al., 2019), and reservoir image analyses in petroleum exploration (Knaup et al., 2019). Among all automated methods, deep neural networks have recently emerged as the dominant ones due to their unique capabilities of uncovering and learning key features from images.

There's a long line of research that uses variants of autoencoders for image segmentation. SegNet (Badrinarayanan et al., 2017), U-Net (Ronneberger et al., 2015), and V-Net (Minaee et al., 2021) belong to this category. Autoencoders are simple and efficient for pixel-wise classifications, especially on small datasets. One limitation of autoencoders is the loss of fine-grained information through the encoding processes. Therefore, some autoencoders add "skip connections" between the encoding layers and the decoding layers to prevent information loss.

The regional convolutional network (R-CNN) (He et al., 2015) and its extensions (He et al., 2017; Jiang & Learned-Miller, 2017) have shown a wide use in object detection. RCNNs rely on a backbone network to output raw proposals, then adopt Region of Interest (ROI) and a RoIPool layer to compute features from these proposals. Extensions of RCNN (Chen et al., 2022) have been used for the task of simultaneously performing object detection and semantic segmentation. Unlike autoencoders that are trained from scratch, RCNNs often adopt well-trained CNN backbones and use transfer learning to fine-tune accuracy on a new dataset (Weiss et al., 2016).

Though supervised methods are the most used in training image segmentation models, there has also been progressing in unsupervised, semi-supervised, and active learning-based approaches. Early work (Chen et al., 2005; Ng et al., 2006) on unsupervised image segmentation used clustering algorithms (Hartigan & Wong, 1979) on handcrafted features to group data points by similarity. Recently, the state-of-the-art methods no longer require handcraft features but instead automatically learn from the training process. For example, W-Net is an unsupervised architecture that connects two U-Nets together (Xia & Kulis, 2017). W-Net is jointly trained by optimizing both the segmentation and the reconstruction errors. The work of Kim et al. (2020) and Kanezaki (2018) trained a neural network by optimizing the difference between the CNN output and Superpixel's output (Achanta et al., 2012). The state-of-the-art unsupervised segmentation models rely on the following assumptions:

- Pixels of similar features should be assigned the same label.
- Spatially continuous pixels should be assigned the same label.
- The number of unique cluster labels should be large

Unfortunately, since the close pixels with similar colors can be different objects, unsupervised learning could be inferior in various circumstances. For this reason, unsupervised learning is only used as the preparation step in real applications.

In contrast, Semi-supervised (or weakly supervised) learning has also been employed to tackle the challenges of obtaining labels (Zhu et al., 2009). Compared with unsupervised learning approaches, semi-supervised approaches train models using a significant amount of unlabeled data and a small amount of labeled data. As an example, Souly et al. (2017) proposed a semi-supervised semantic segmentation method by employing GANs, where the generator network is utilized to produce additional training images. Papandreou et al. (2015) utilized the Expectation-Maximization (EM) approach in the loss function of a CNN for segmenting pictures with few labels (Green, 1990). Tang et al. (2018) created graphic-based semi-supervised methods for segmenting photos with poor annotations, including bounding boxes or scribbles. Many scientific applications have also used semi-supervised learning. For example, Zhou et al. (2019) proposed a semi-supervised learning method using the attention mechanism to segment medical images. Castillo-Navarro et al. (2021) introduced a large-scale earth observation dataset and presented a semi-supervised architecture based on multi-task training to segment images. To the best of my knowledge, there's no prior work that targets the segmentation of shale images using either semi-supervised learning or unsupervised learning, which is the main theme of this dissertation.

Recently, there has been a collection of work that adopts active learning for image segmentation. Generally, we could annotate either part of an image or the whole image to create labels. For example, Mackowiak et al. (2018) proposed a method that automatically selects regions that need labeling based on both the amount of human effort needed and the amount of information in the regions. Metabox+adopts the same idea, except that it trains a regression model for cost estimation (Colling, 2021).

This dissertation relies on annotating the entire image because minerals in shale samples are super fine-grained; annotating only part of the regions could render significant errors. Research in this direction can adopt similar approaches as active learning for image classification since we are measuring the uncertainty and diversity of entire images to build the acquisition function. Traditional approaches calculate heuristic methods, such as Markov random field smoothness and entropy terms, to evaluate candidate image samples (Li et al., 2010; Top, 2011). Since the suggestive annotation method (Yang et al., 2017) adopts a fully convolutional network (FCN) and active learning to significantly reduce manual labeling, there has been emerging research on active learning not just directly based on diversity and uncertainty. Instead, they construct comprehensive models that consolidate uncertainty and diversity. For example, VAAL learns a latent space using a VAE model and a discriminator (Sinha et al., 2019). The VAE is learned to fool the discriminator that all images have been labeled, while the discriminator is trained to better differentiate between labeled and unlabeled images. Minimax active learning optimizes VAAL by selecting samples with high entropy that are classified as unlabeled (Ebrahimi et al., 2020). However, minimax learning requires a manual entropy threshold. Task-aware VAAL tackles this issue by feeding the uncertainty metrics as a feature of the latent space (Kim et al., 2021). In contrast to existing work, our framework can consider multiple features, and we handle the task-awareness issue by adding a cold start phase.

Chapter 3: Dataset

As the raw training dataset, my research group gathered 15 gray-scale SEM images of a sample of Woodford shale (Oklahoma, USA). The SEM images were captured with a Hitachi S-4800 II instrument at 1024×688 resolution. Sandpapers with grit sizes ranging from 400, 800, 2400, and 3000 were used to treat the polished shale samples in that order. The samples were coated with platinum to increase conductivity and enhance image quality prior to analyses by Field Emission (FE)-SEM. To better display the features of the sample surface, imaging was performed at a low voltage of 1.5kv.

I marked each pixel with a corresponding object, including pores, clays, organic matter, and mineral grains. Each image took an expert about 3 hours to mark and verify labels using ImageJ (Abràmoff et al., 2004). Figure 9 shows a raw image and its labels, with clays marked as black, organic matter as red, and pores as green, while the remaining part of the picture as yellow to represent mineral grains. This study aims to segment all four objects in SEM images.

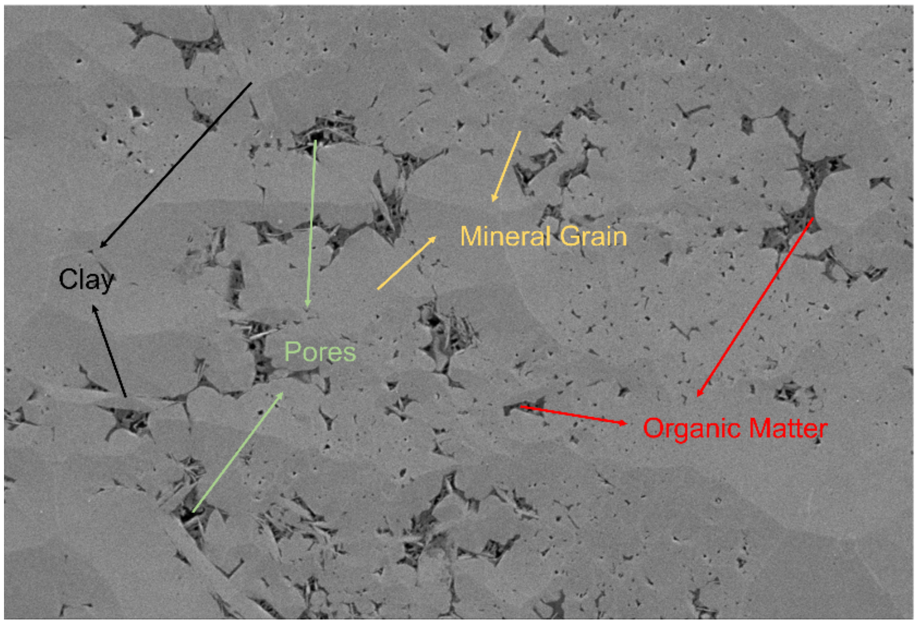

Figure 9 A shale SEM image annotated with pores, clay, organic matter, and mineral grains

The goal of my research is to segment all four objects from SEM images and then adopt the following steps to separate raw images into smaller images by:

- Creating the training dataset, the validation dataset, and the testing dataset using independent subsets of raw images. Note that each project in this dissertation uses a different number of raw images.

- Using the sliding window method to process the raw images and labels to generate sub-images with a size of 128×128 using a stride size of 32 (as shown in Figure 10, a sliding window is a rectangle with fixed width and height that "strides" across an image to produce tiny sub-images that may or may not overlap); and

- Taking out sub-images with >99% pixels belonging to a single mineral object.

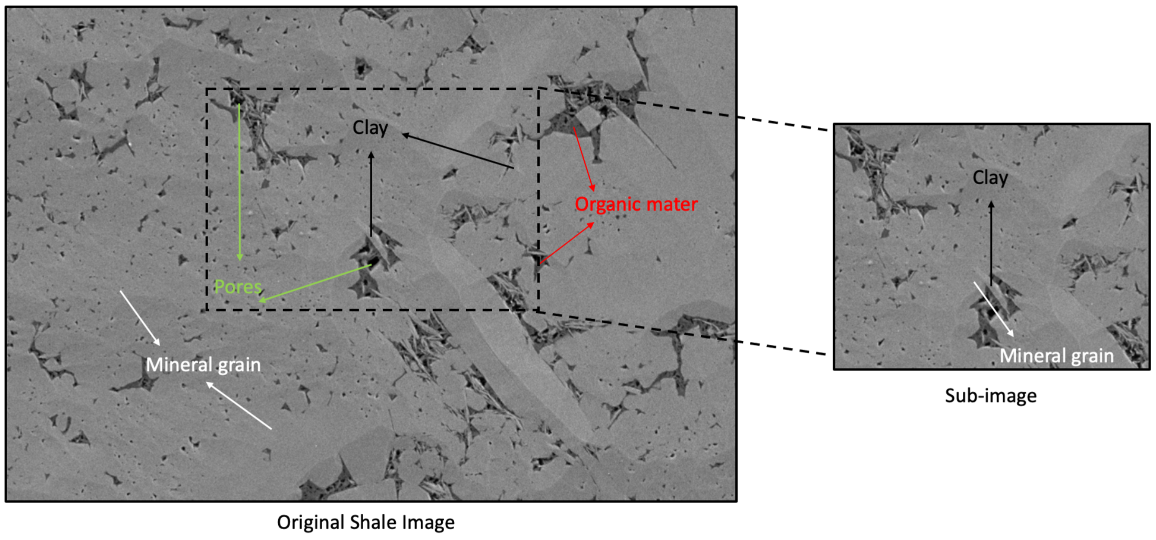

Figure 10 An example shale SEM image annotated with pores, clays, organic matter, and mineral grains

Figure 11 shows the composition of shales in the dataset, from which we can observe two interesting phenomena that pose challenges for the segmentation. First, the threshold analyses cannot easily distinguish organic matter, clays, and pores because their colors are overlapped, as the overlapped regions occupy a large portion of each mineral; hence, we believe using machine learning approaches could outperform simple threshold analyses by considering regional and contextual information. Second, pores and clays only account for a small portion of the shale compared to mineral grains and organic matter; as a result, the skewed training dataset may result in skewed outcomes and low accuracy when classifying all minerals using a single model.

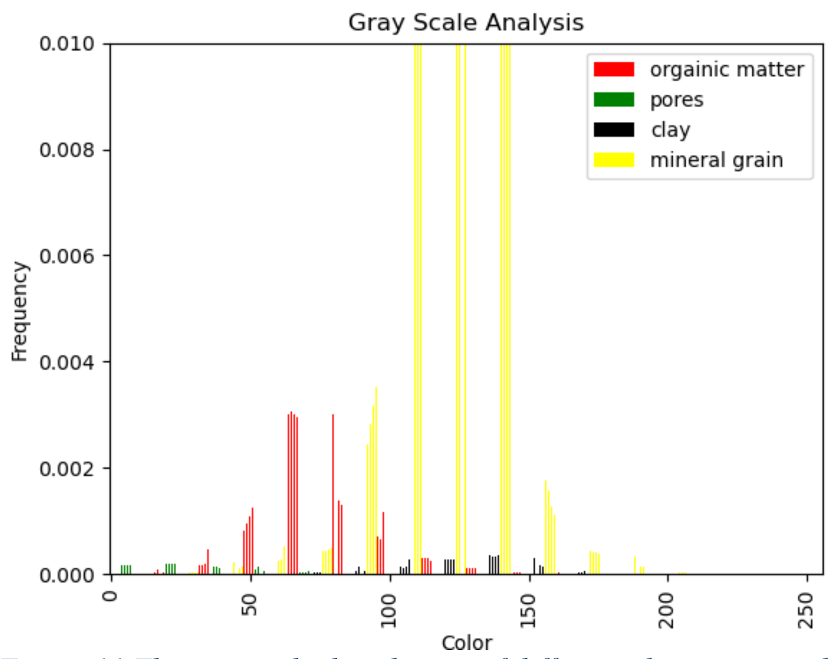

Figure 11 The grayscale distribution of different objects in our dataset; the X-axis ranges from 0 to 255

Predicting labels for each image pixel is challenging for unsupervised learning approaches as they are often established under certain conditions. A common assumption is that pixels near each other with the same color tend to be classified as the same class (Xia & Kulis, 2017; Kim et al., 2020). However, objects in shale rocks do not always follow this assumption. For example, grains of clays and minerals have similar colors and are near each other. In Figure 10's sub-image on the right, these grains cannot be differentiated just using the color and position heuristics, we must consider the fact that clays often exist within organic matters, but mineral grains are not. As a result, existing supervised methods rely on many accurate labels to obtain acceptable segmentation results.

This work revised the U-Net architecture, a popular approach for biomedical image analyses, by incorporating a loss function that addresses the imbalance issue. Furthermore, I used the ensemble learning method to train multiple models and combined the results to improve the overall performance of segmentation. I name this new approach “Paw-Net” in the following sections. The overall results show that this new method improves the mean Intersection over Union (mIoU) of mineral objects from 0.49 to 0.58, compared to the original U-Net model. This method can clearly distinguish each object from others with boundaries, even in highly unbalanced images. Therefore, this method is cost-effective to help geoscientists gain insights by building neural network models from a small dataset of SEM images

4.1. Overview

In this work, the raw dataset contains five images; each has a size of 1076×768 pixels. Four out of five raw images were used to form the training dataset, and the rest raw image was used as the testing dataset. First, I used the sliding window method to process the raw images and labels to generate 2610 sub-images with a size of 128×128 using a stride size of 32. Next, I discarded 448 sub-images whose major pixels (>99%) only contain a single mineral. Further, I took out 20% of sub-images and labels from the training dataset randomly to form the validation dataset. Finally, the training dataset has 1433 sub-images, the validation dataset has 359 sub-images, and the testing dataset has 370 sub-images.

Figure 12 illustrates the general workflow for the image segmentation task used in this work, including the following five steps:

1) Mark raw shale images to generate corresponding labels for each image.
2) Preprocess the original dataset by slicing, normalizing, and removing outliers.
3) Divide the dataset into training and testing datasets.
4) Train the U-Net and Paw-Net models.

5) Feed the testing data into the trained models and compare the prediction results of each model.

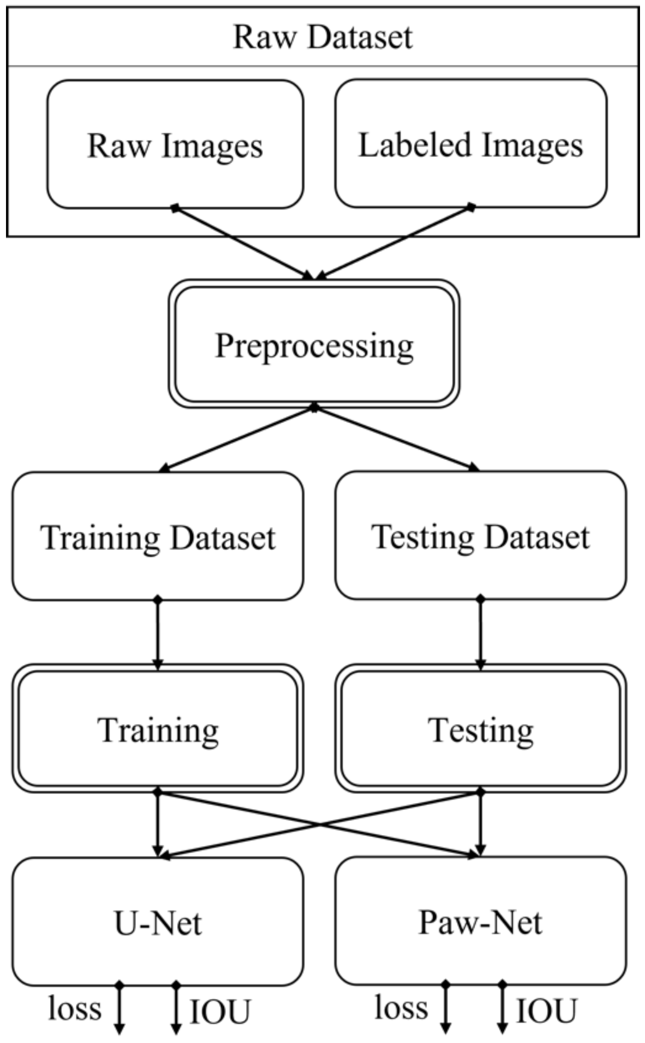

Figure 12 Workflow for shale image segmentation using deep neural networks

4.2. Model Training and Evaluation

As shown in Figure 12, I trained both U-Net and Paw-Net models with the training and evaluation approaches described in this section. I employed mini-batch training to avoid getting local minimal in the training stage. Each batch contains 32 sub-images fitting for GPU memory. I adopted the Adam function of Kingma (2014) to optimize the weight of models, using a training rate of 0.01 and a weight decay of 1×10^{-8}.

4.2.1 U-Net

For U-Net, I used the Softmax function of Bridle (1990) in the last layer to yield the probability of every mineral for each pixel.

$$p(x^i) = softmax(x^i) = \frac{e^{x^i}}{\sum_{k=1}^{\mathcal{K}} e^{x^k}} \tag{4-1}$$

In Equation 4-1, $p(x^i)$ denotes the probability of pixel x belonging to category i. The Softmax function rescales a $\mathcal{K}$-dimensional vector so that each element in the output is within the range of [0,1].

I used the Cross-entropy loss function of Shore & Johnson et al. (1981) to compute the difference between predicted results and targets.

$$L(y, \hat{y}) = -\sum_{k=1}^{\mathcal{K}} y^k \log \hat{y}^k \tag{4-2}$$

In Equation 4-2, $\hat{y}^k$ indicates the predicted output, and y^k is the ground truth. The cross-entropy loss is the sum of the loss of all $\mathcal{K}$ classes.

After the models were trained, they were used to predict the labels of images in the testing dataset. To evaluate the accuracy of models, I adopted the intersection over union (IoU) metrics (Equation 4-3).

$$IOU^i = \frac{Overlap(y^i,\ \hat{y}^i)}{Union(y^i,\ \hat{y}^i)} \tag{4-3}$$

IOU^i represents the ratio of the correctly predicted pixels that belong to category i to the union of ground truth pixels that belong to category i. Based on Equation 4-3, I derived Equation 4-4 to measure the mean IoU (mIoU) for all categories because our study aims to segment all objects in shale rock images.

$$m\,IOU = \frac{\sum_{k=0}^{\mathcal{K}} IOU^k}{\mathcal{K}} \tag{4-4}$$

4.2.2 Paw-Net

In my dataset, due to the large class imbalance being encountered, mineral objects that make up a large portion of images comprise most of the loss and thus dominate the gradient. I implemented Paw-Net for stack generalization. Unlike the traditional stack generalization methods that use output labels as the input for the second level, we feed the probability of each label as the input to the second stage. Polikar (2012) proposed that this approach could improve the stacking performance. The high-level idea of Paw-Net is that we can use “local experts” to focus on learning a single target object against others and then merge individual models to reduce variance.

Using the stack ensemble training method, Paw-Net involves two stages for Level 1 and Level 2 models, as shown in Figure 13. In the first stage, I extracted labels of each category from the raw label. Then I ingested training images and labels for each category into individual models. Each model only predicts the probability of a specific category against others to reduce the bias in learning multiple categories. In addition, I trained a raw U-Net model (Multi-Class U-Net) and combined its results with other models as the input to the Level-2 model.

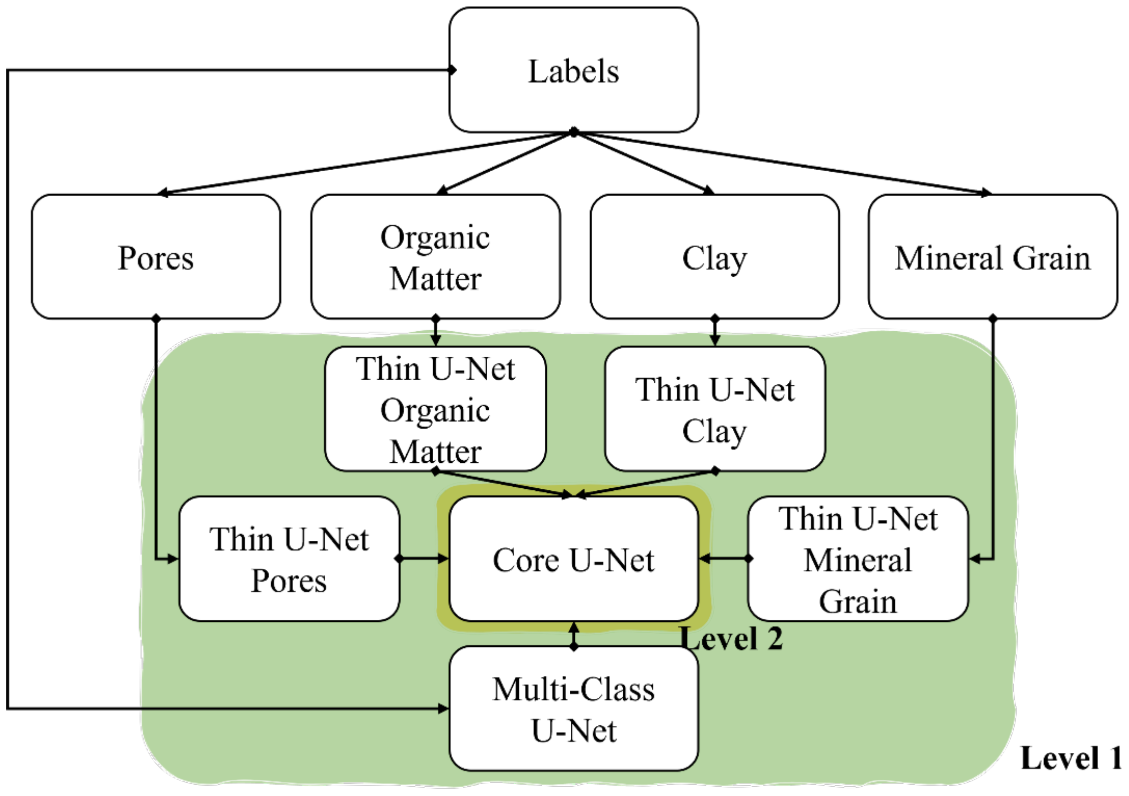

Figure 13 The architecture of Paw-Net

The models I used for individual objects are "Thin" U-Net, which have several differences compared to the original U-Net model. Firstly, since thin U-Net performs binary classification, the last layer uses the Sigmoid function of Narayan (1997) that calculates the probability of whether a pixel x belongs to a certain category (Equation 4-5).

$$p(x) = sigmoid(x) = \frac{1}{1+e^{-x}} \tag{4-5}$$

Secondly, to avoid overfitting issues, I reduced the number of channels in convolution. Also, I used bilinear up-sampling instead of transposed convolution to implement the up-sampling layer.

Thirdly, I replaced the binary cross entropy function with a new loss function called "Focal Dice loss." First proposed by Milletari et al. (2016), the Dice loss function in Equation 4-6 is very effective in scenarios where the objects of interest occupy only a very small ratio. Unlike other approaches, such as weighted cross entropy (Ronneberger et al., 2015), this work does not need to assign weights to samples of different classes to establish the balance.

$$DL(y, \hat{y}) = 1 - 2 \times \frac{\sum_{k=1}^{\mathcal{K}} y^k \times \hat{y}^k}{\sum_{k=1}^{\mathcal{K}} y^k + \hat{y}^k} \tag{4-6}$$

While Dice loss balances the importance of positive/negative pixels, it lacks the ability to differentiate between easy and hard pixels. Indeed, a pixel belonging to a rare class does not necessarily indicate it is difficult to classify. To address this problem, I combined the Dice loss with Focal loss (Lin et al., 2017) to form the Focal Dice loss function in Equation 4-7.

$$FDL(y, \hat{y}) = \left(1 - DL(y, \hat{y})\right)^{\Gamma} DL(y, \hat{y}) \tag{4-7}$$

In Equation 4-7, Γ balances the importance of positive and negative pixels to reduce the loss of easily classified pixels, which otherwise would dominate backward propagation. As a result, the Focal Dice loss function addresses both the positive/negative balance and the difficult/easy balance.

After training Level 1 models, the outputs of the Thin U-Net models and the raw U-Net model are fed to the second stage. Then, I trained a meta learner---another U-Net model, to output the final probability of all categories for each pixel. Using the stacking ensemble method, this dissertation can harness the capabilities of a range of well-performing “local experts” on the segmentation task to reduce the variance that existed in individual models and make predictions better than any single model.

4.3. Experiments

I ran experiments on a Linux machine equipped with an NVIDIA A100 GPU by writing the training script in PyTorch 1.10 (Paszke et al., 2019) and Python 3.8.5.

Table 1 summarizes the training and testing mIoU scores and time using different models. Paw-Net achieved better scores than U-Net on both training and testing datasets. Besides, to study the benefits of ensemble learning and the Focal Dice loss function, I implemented U-Net Focal Dice, which is a single U-Net model that classifies multiple objects using the “Focal Dice loss function” (Equation 4-7). We can see that this model outperforms the raw U-Net by 0.03 on the testing dataset while achieving 0.06 lower IoU than Paw-Net. This result proves that Paw-Net’s improvement comes from incorporating both ensemble learning and the Focal Dice loss function. Due to the increase in computing, Paw-Net takes ~5.5× longer to train and ~2× longer to test than U-Net. However, it is noteworthy that training only takes less than 3 minutes for thousands of sub-images. In addition, based on the time statistics, one can reduce the training time to around 2× by parallelizing the first stage of Paw-Net on multiple GPUs because the first-stage models can be trained independently. I also compare neural network models with LDA (Izenman, 2013), a statistical method for classification. On the one hand, LDA’s computing is relatively lightweight compared with neural network models. As a result, it has a very fast training and testing speed on our dataset. Nevertheless, on the other hand, LDA’s relatively simple equation yielded low accuracy compared with neural network models.

Table 1 Overall training and testing IoU scores of U-Net and Paw-Net

Model	*Training-IoU*	*Testing-IoU*	*Training Time*	*Testing Time*
LDA	0.25	0.22	3.42s	0.11s
U-Net	0.68	0.49	29.86s	8.61s
U-Net Focal Dice	0.71	0.52	29.80s	8.55s
Paw-Net	0.76	0.58	161.35s	15.10s

Inspired by Azarafza et al. (2021), I list hyperparameters involved in neural network models in Table 2. Hyperparameters are tunable values that control the learning process. To make our comparisons fair, we set the same hyperparameter values for all models except for the Focusing parameter, which is not applicable in the U-Net model.

Table 2 Hyperparameters of each model

Component	*Hyperparameters*	*Value*	*Meaning*
Adam optimizer	Learning rate (r)	0.001	The step size while adjusting model weights
Focal dice loss	Focusing parameter (Γ)	2.0	The down-weighted rate at easy samples
Model training	Batch size (b)	32	The number of sub-images in each iteration
	Epochs (e)	5	The number of times it works through the training dataset

Figure 14 compares the training IoU curves of U-Net and Paw-Net's Level 1 models. It can be observed that Paw-Net is better than U-Net in performing the segmentation of Pores and Clays, which are the rarest minerals in our training dataset. Note that Paw-Net has similar effects as U-Net in segmenting Mineral Grains and Organic Matter. The two objects take a much larger portion compared with Pores and Clays. Therefore, this dissertation demonstrates that the Focal Dice loss function is highly effective in training imbalanced datasets.

The training curves of Paw-Net (Level 2) and U-Net are presented in Figure 15. Because Paw-Net's input combines the outputs of the U-Net and Thin U-Net models, its training loss is reduced very fast. It can be observed that Paw-Net is consistently better than U-Net on the IoU scores. Moreover, the IoU scores of Paw-Net are relatively more stable than U-Net across iterations. During training, I performed validation at each epoch. From Figure 16, it can be seen that the validation curves follow the same trend as the training curves. It is worth noting that the IoU score of the testing dataset (0.58) is lower than the IoU score of the validation dataset (0.73) because I used a separated shale image from the training set, while sub-images in the training dataset and the validation dataset might come from the same shale image due to random dataset splitting.

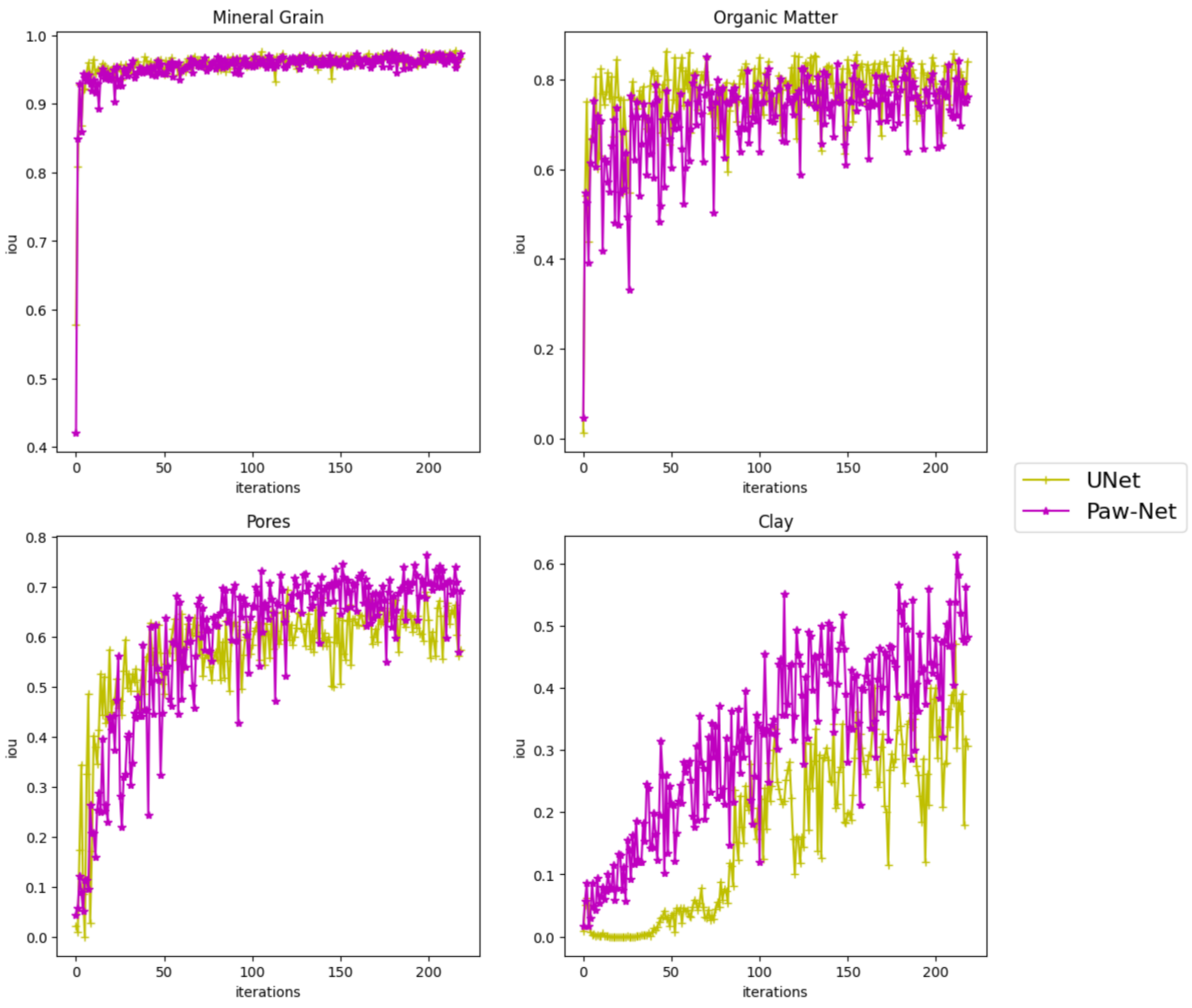

Figure 14 Comparison between training IoU curves of U-Net and Paw-Net (Level-1) for individual minerals

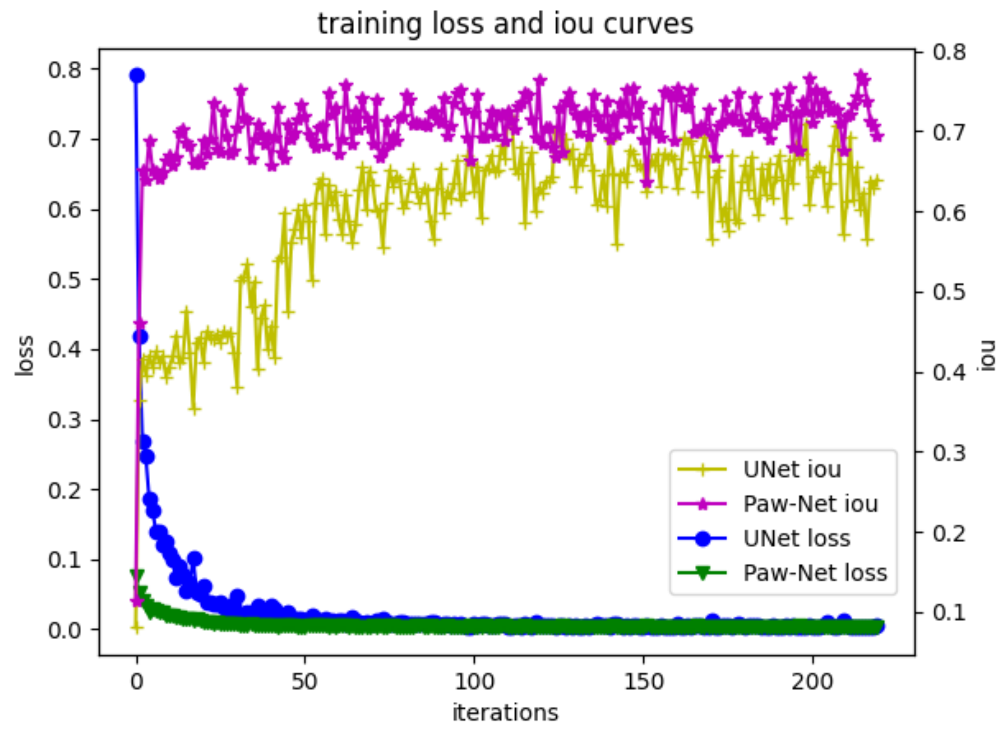

Figure 15 Training curves of Paw-Net (level-2) and U-Net

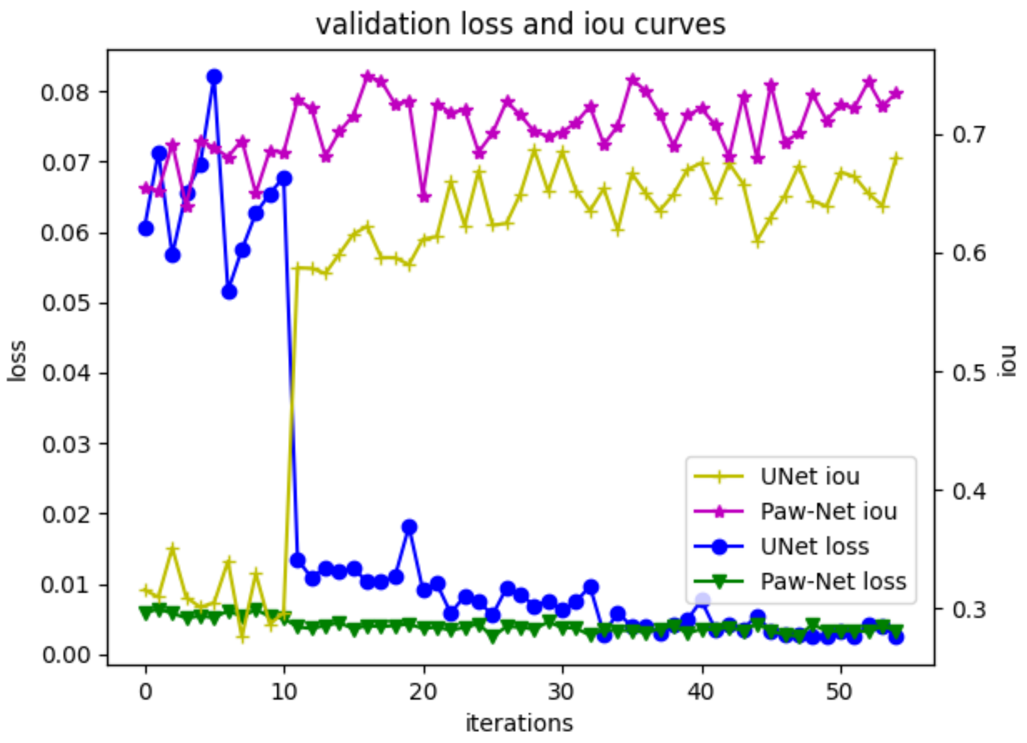

Figure 16 Validation curves of Paw-Net (level-2) and U-Net

Figure 17 shows the comparison of original SEM images, ground truth labels, predicted results from U-Net, and predicted results from Paw-Net. In the original SEM images, the large light grey barks are mineral grains, the light grey banded particles are the clay, the areas with dark grey are organic matter, and the rest black areas are pores. In the ground truth labels and prediction images, the white background stands for the mineral grains, black for clay minerals, red for organic matter, and green for pores.

According to the comparison, we can observe that the prediction results yielded by Paw-Net are more reliable than U-Net, especially for differentiating clay mineral and mineral grain objects with the same light grey color in the original images. Also, though clay minerals and pores occupy only a small part of the training dataset, Paw-Net can yield clear segmentations of these objects for result prediction.

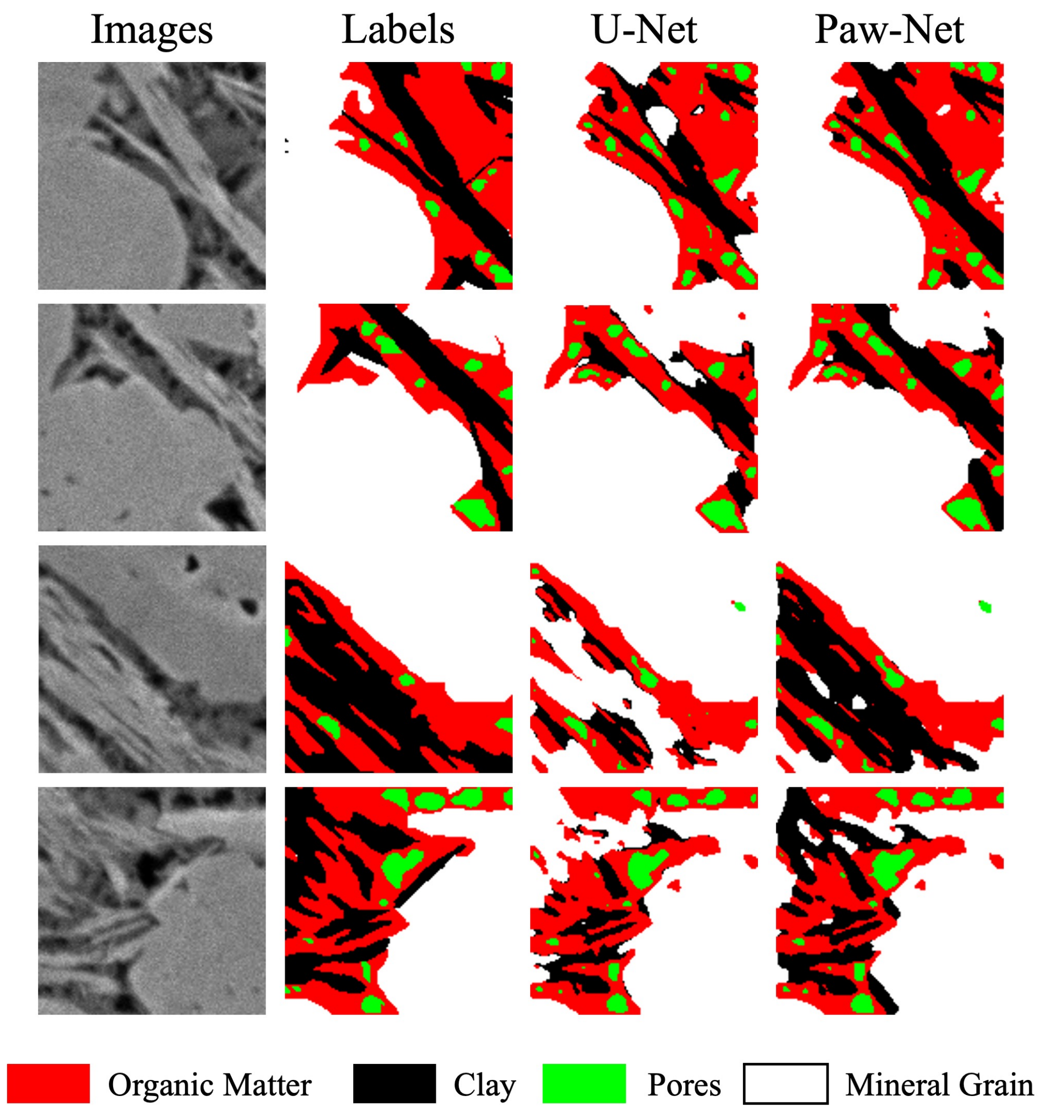

Figure 17 Comparison between segment results of Paw-Net and U-Net

4.4. Discussion

This study demonstrates that Paw-Net is valuable and could provide rich insights for geological data analyses by significantly increasing the segmentation accuracy of images with imbalance issues. Because object imbalance is a common issue in shale rock images (Karimpouli & Tahmasebi 2019; Chen et al., 2020), it is believed that Paw-Net can be used in other geology applications. In addition, the idea of using ensemble learning and focal loss can be extended to other applications beyond segmentation, such as object detection and classification.

It is noteworthy that this approach has several limitations. First, with the adoption of ensemble learning, Paw-Net’s training and inference speed can be slower than U-Net's. In Section 5, it is pointed out that Paw-Net’s training speed is 5.5× times slower than U-Net. To reduce the time, we can train the first stage models in parallel, making it only 2× slower than the U-Net model. Second, Paw-Net’s accuracy on Clay and Pores are still behind the accuracy on Mineral Grain and Organic Matters due to the lack of labels. Although incorporating additional high-quality images and labels can be helpful, it is found that even experienced geology experts can make errors in identifying these objects and need verification to correct labels. The experimental results show that Paw-Net outperforms the raw U-Net significantly on a custom dataset. By comparing the IoU scores of individual objects, the improvement comes from imbalanced objects such as clays and pores. Moreover, the segmentation results clearly show that clay objects can be more effectively separated from others using Paw-Net than U-Net.

According to the comparison, it is observed that the prediction results yielded by Paw-Net are more reliable than U-Net, especially for differentiating clay mineral and mineral grain objects with the same light grey color in the original images. Also, though clay minerals and pores occupy only a small part of the training dataset, Paw-Net can yield clear segmentations of these objects for result prediction.

Chapter 5: Semi-supervised Learning for Shale Image Segmentation

Using Paw-Net, this work is able to improve the segmentation accuracy through ensemble learning using five labeled images. In fact, the original dataset has hundreds of unlabeled images. A straightforward way to improve our model's accuracy is by injecting more labeled images. However, labeling each image is expensive. Suppose one spends three hours to label and verify an image; labeling 100 images would take 300 hours. Thus, I only labeled three extra images in the end, generating eight labeled images in the training dataset. I investigated both unsupervised and semi-supervised approaches in this work for fine-grained shale by developing a semi-supervised learning model, SU-Net, based on the U-Net architecture. I also proposed a novel algorithm to speed up the semi-supervised loss function in SU-Net using caching, skip zeros, and batching optimizations.

5.1. Overview

This work incorporated unsupervised learning to use the rest images, which assumes that image segments can be obtained by clustering the feature vectors. Like human segmentation, unsupervised learning creates a salient part of a single object instance, where each object instance tends to have large regions of similar colors or texture patterns. However, grouping spatially continuous pixels with similar colors or texture patterns into the same cluster does not always make the best result. It is also possible that different segments differ in their shapes, which is different from drafting in loss functions. Therefore, I investigated both unsupervised and semi-supervised learning in this project, with the intention of using partial labeling information to guide segmentation automatically.

There have been many heuristic approaches for unsupervised image segmentation (Kass et al., 1988; Lindeberg, 1994). However, as mentioned before, there haven't been many studies about using unsupervised neural networks for image segmentation. In this project, I adopted the W-Net architecture as the baseline (Xia & Kulis, 2017), which is the state-of-the-art unsupervised approach for image segmentation.

I used eight images in this project. Like the unsupervised project, I used the sliding window method to generate slices with a size of 128×128 using a stride size of 32. Six out of eight images were used to form the training dataset; the testing dataset has two images. For unsupervised learning, I treated all eight images as unlabeled images; for semi-supervised learning, I controlled the ratio of labeled images in experiments.

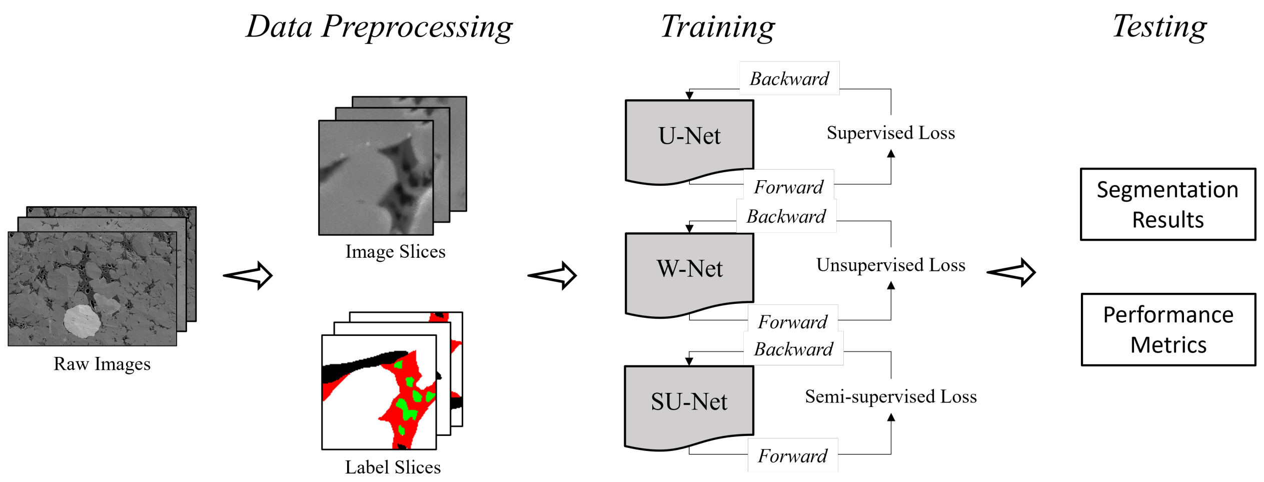

Figure 18 The workflow to train and test our models for segmentation

This study applied supervised, unsupervised, and semi-supervised approaches to train segmentation models on our custom dataset and compared their performance. I illustrate the workflow for the image segmentation tasks in Figure 18, which includes the following three stages:

- Data Preprocessing: Create corresponding labels for each raw shale image, preprocess raw images with slicing, normalization, denoising, and outlier elimination, and split the raw images into training, validation, and testing datasets.
- Training: Train models using different methods, including the fully supervised U-Net, the unsupervised W-Net, and the semi-supervised SU-Net.
- Testing: Compare the segmentation results and performance metrics yielded by each model by feeding the testing dataset into the trained models.

The training stage follows the standard training process for neural networks, where the forward pass calculates the loss between the truth and predicted results, and the backward pass updates the model based on the derivatives of the loss. The testing stage only applies the forward pass to yield the prediction results on the testing dataset. This dissertation also reports performance metrics related to evaluating the accuracy of prediction.

To train the supervised U-Net, I used the training dataset with both images X and their corresponding pixel-wise labels Y. Then, I optimized the model by iteratively updating model weights in the training stage. The model is supposed to predict Y' for future unseen images X'.

In contrast, to train the unsupervised W-Net, I used images X without labels Y. As a result, I can only infer Y using each pixel's characteristics, such as color. For the semi-supervised SU-Net, I used images X and labels Q, where $Q \subseteq Y$.

5.2. Unsupervised Learning

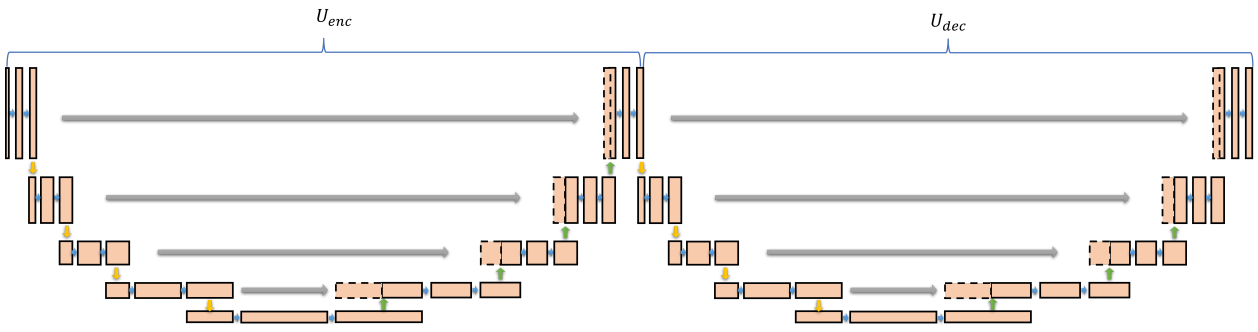

Figure 19 The architecture of W-Net

Figure 19 shows the layout of the W-Net, which is composed of two U-Nets. W-Net is jointly trained on image reconstruction and segmentation tasks (Zhang & Yang, 2021). The U-Net on the left is referred to as the encoder U-Net (U_{enc}) and is used for segmentation; the U-Net on the right is referred to as the decoder (U_{dec}) and is used for reconstruction. The loss function of W-Net is shown in Equation 5-1:

$$\mathcal{L}^{W-Net} = \mathcal{L}^{segmentation} + \mathcal{L}^{reconstruction} \tag{5-1}$$

$\mathcal{L}^{segmentation}$ is computed solely based on the output of U_{enc}, and $\mathcal{L}^{reconstruction}$ is computed through both U_{enc} and U_{dec}.

$\mathcal{L}^{reconstruction}$ measures the mean square distance between an input image and its predicted reconstruction. In Equation 5-2, I use X_i to denote the input value of a pixel belonging to an image X. The smaller the $\mathcal{L}^{reconstruction}$, the more similar the input and the reconstruction are.

$$\mathcal{L}^{reconstruction} = \frac{1}{\mathcal{I}} \sum_{i=1}^{\mathcal{I}} \left(X_i - U_{dec}\left(U_{enc}(X_i)\right) \right)^2 \qquad (5\text{-}2)$$

$\mathcal{L}^{segmentation}$ calculates normalized cut that uses cut and associativity to measure the dissimilarity between the objects that belong to different classes (Shi & Malik, 2000). A low normalized cut value often indicates that objects are well clustered. Since the original normalized cut formula assigns each pixel with its maximum possibility class using argmax, we cannot adopt it in the backward propagation process as argmax is not differentiable. Therefore, W-Net uses a "soft" version of the normalized cut, as shown in Equation 5-3, where $\widehat{Y_i^k}$ denotes the probability that the pixel Y_i belonging to class k.

$$\mathcal{L}^{segmentation} = \mathcal{K} - \sum_{k=1}^{\mathcal{K}} \frac{assoc\left(\widehat{Y^k}, \widehat{Y^k}\right)}{assoc\left(\widehat{Y^k}, Y\right)} = \mathcal{K} - \sum_{k=1}^{\mathcal{K}} \frac{\sum_{u=1}^{\mathcal{I}} \sum_{v=1}^{\mathcal{I}} w(u,v) \widehat{Y_u^k} \widehat{Y_v^k}}{\sum_{u=1}^{\mathcal{I}} \sum_{v=1}^{\mathcal{I}} w(u,v) \widehat{Y_u^k}} \qquad (5\text{-}3)$$

In Equation 5-3, $w(u, v)$ indicates the similarity between pixel u and v. Inspired by previous studies (Xia & Kulis, 2017; Tang et al., 2018; Kim et al., 2020), this work uses the distance and the color difference between pixels to represent the similarity, as shown in Equation 5-4. I call w the "weight matrix" in the rest of the dissertation.

$$w(u, v) = w_{dist} \times w_{color} = e^{\frac{-\|color\_diff(u,v)\|_2^2}{\sigma_c^2}} \times \begin{cases} e^{\frac{-\|dist(u,v)\|_2^2}{\sigma_p^2}}, & if\ \|dist(u, v)\|_2^2 \leq r \\ 0, & if\ \|dist(u, v)\|_2^2 > r \end{cases} \qquad (5\text{-}4)$$

Particularly, if $dist(u, v) > r$, where r is a relatively small radius parameter (W-Net uses $r = 5$), I set $w(u, v) = 0$. The intuition here is that with the growth of distance, it is less likely for objects with the same color to belong to the same class. I use Hamming distance to represent $dist(u, v)$. σ_c^2 and σ_p^2 are tunable terms (Norouzi et al., 2012).

5.3. Semi-supervised Learning

Though W-Net can be trained without any labels, it has two major shortcomings when being applied to our dataset: low accuracy and long computing time. As mentioned in Chapter 3, clay and mineral grains, which are often located close to each other, also have similar colors. Thus, this unique characteristic in fine-grained shale violates the assumption of $\mathcal{L}^{segmentation}$ proposed in Equation 5-4. In addition, a naïve algorithm to compute $\mathcal{L}^{segmentation}$ is expensive without any optimization, causing W-Net to be trained 33× slower than U-Net. In this section, I investigate a semi-supervised approach with limited labels to tackle the shortcomings of W-Net.

5.3.1. SU-Net

Like W-Net, SU-Net adopts the multi-task training methodology (Castillo-Navarro et al., 2021; Cao et al., 2022), in which a single model is trained to achieve multiple goals simultaneously. Multi-task training can yield better results than single-task training since the trained model can generalize better.

SU-Net differs from W-Net in a few key ways. While W-Net optimizes the image reconstruction loss and the unsupervised segmentation loss, SU-Net is designed to work with partial ground truth labels. This allows us to learn from both labeled and unlabeled data, making the model more robust to the segmentation task. In addition, this work employs a regularization parameter in SU-Net to help strike a balance between supervised loss and unsupervised loss during training. I proposed Equation 5-5 as the loss function of SU-Net to train an input image X.

$$\mathcal{L}^{SU-Net}(X) = \begin{cases} \mathcal{L}^{U-Net}(X) + \mu\mathcal{L}^{segementation}(X), & if\ X\ is\ labeled \\ \mu\mathcal{L}^{segementation}(X), & if\ X\ is\ unlabeled \end{cases} \tag{5-5}$$

Equation 5-5 consists of two parts: $\mathcal{L}_{supervised}$ uses the Cross-Entropy function to compute the supervised loss for labeled pixels, and $\mathcal{L}_{segmentation}$ adopts the normalized cut function (Equation 5-3) to compute the unsupervised segmentation loss for all images. μ is the regularization parameter that balances the supervised loss and segmentation loss. In practice, X is a batch consisting of $\mathcal{B}$ input images. I applied min-batch training to compute the average loss of B inputs at each iteration, as shown in Equation 5-6.

$$\mathcal{L}_{batch}^{SU-Net} = \frac{1}{\mathcal{B}} \sum_{b=1}^{\mathcal{B}} \mathcal{L}^{SU-Net}(X_b) \tag{5-6}$$

Unlike W-Net, which links two U-Nets together to connect two separate loss functions, SU-Net's architecture is still a single U-Net, but its weights are updated based on the combination of supervised loss and segmentation loss. Before starting training, I applied a "warm-up" phase that pre-trained each SU-Net with <1% to ensure that the data used in pre-training are a subset of the labeled data used in training. Training SU-Net without warm-up may cause performance degradation because we do not have a correspondence between predicted class indices and truth label indices for unlabeled data. More details can be found in Section 5.5.

5.3.2. Fast Normalized Cut

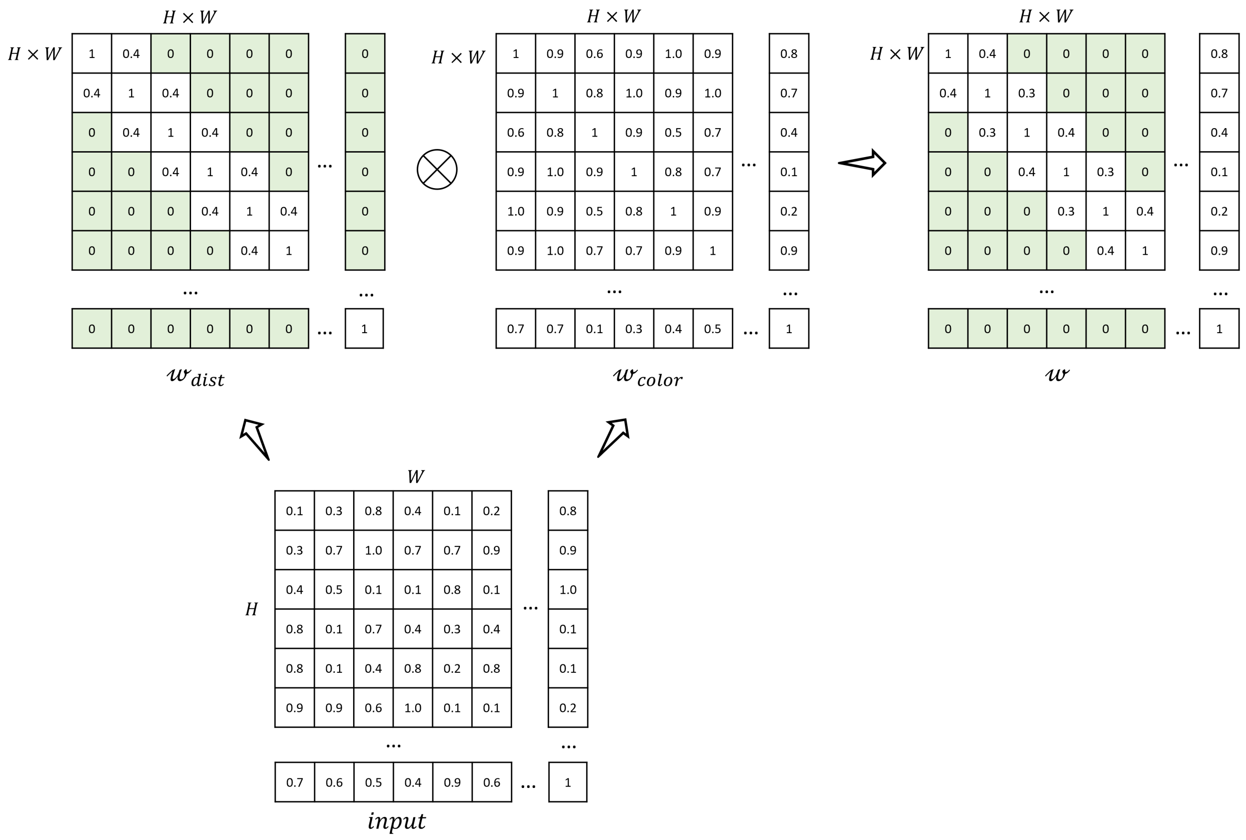

Figure 20 The key computation of $\mathcal{L}_{segmentation}$ that obtains the weight matrix w in Equation 5-3. H represents the height of the input, and W represents the width of the input. We assume $r = 1$ in this example.

It is observed that computing $\mathcal{L}^{segmentation}$ can take a long time with a naïve algorithm, and the major overhead comes from computing the weight matrix w. As illustrated in Figure 20, the native algorithm first computes the corresponding w_{dist} and w_{color} between each pixel in the input. Next, it performs element-wise multiplication on w_{dist} and w_{color} to get w. In each iteration, the computation is repeated $\mathcal{B}$ times to get the w of each input, as described in Equation 5-6. We thoroughly study the inefficiency involved in computation and make the following observations and optimizations:

Computation on $\mathcal{w}_{dist}$ is redundant. Because the distances between pixels are constant and each input image has the same shape, we can compute $\mathcal{w}_{dist}$ only once, save it in the memory, and apply it at each iteration. This optimization is called "caching." Since the distance matrix consumes up to $(H \times W)^2 \times 4$ bytes (assuming 4 bytes per pixel), it is affordable for most GPUs. Note that caching $\mathcal{w}_{color}$ is generally not possible because each input has a unique $\mathcal{w}_{color}$, caching all $\mathcal{w}_{color}$ would take $N \times (H \times W)^2 \times 4$ bytes, where N denotes the total number of images.

Computation on $\mathcal{w}_{dist} \times \mathcal{w}_{color}$ involves many zeros. Since the radius parameter $r \gg H \times W$, Equation 5-3 will involve multiplication on many zeros, wasting computation resources. To solve this problem, we rewrite Equation 5-3 as Equation 5-7 to only consider the weight between each pixel and its neighbors. Then, the size of $\mathcal{w}$ is changed from $(H \times W)^2$ to $H \times W \times (2 \times r + 1)^2$. Since $r \gg H \times W$, Equation 5-7 requires significantly less computation than Equation 5-3. To facilitate a fast transformation, we also apply padding and unfolding on the original input (Figure 21). The padding operation adds extra pixels around the edges of an image. The unfold operation fetches the neighbors of each pixel and puts them into a new matrix. I name this optimization "skip zeros."

$$\mathcal{L}^{segmentation} = \mathcal{K} - \sum_{k=1}^{\mathcal{K}} \frac{assoc\left(\widehat{Y^k}, \widehat{Y^k}\right)}{assoc(\widehat{Y^k}, Y)} = \mathcal{K} - \sum_{k=1}^{\mathcal{K}} \frac{\sum_{u=1}^{\mathcal{I}} \sum_{v=1}^{neighbors(u)} \mathcal{w}(u,v) \widehat{Y_u^k} \widehat{Y_v^k}}{\sum_{u=1}^{\mathcal{I}} \sum_{v=1}^{neighbors(u)} \mathcal{w}(u,v) \widehat{Y_u^k}} \tag{5-7}$$

Computation of $\mathcal{L}^{segmentation}$ is done serially. For Equation 5-3, computing the loss of each output in sequence and getting the average of all outputs is inefficient because the GPU resources cannot be saturated with a single input. More than that, dispatching the workload from the CPU to the GPU for each input also triggers non-trivial overhead. To further reduce the computation time, I employed the "batching" optimization by combining all inputs in a batch into a multidimensional array (i.e., tensor) with a shape of $[B, H, W]$. In this way, a single tensor operator can handle all inputs in a batch.

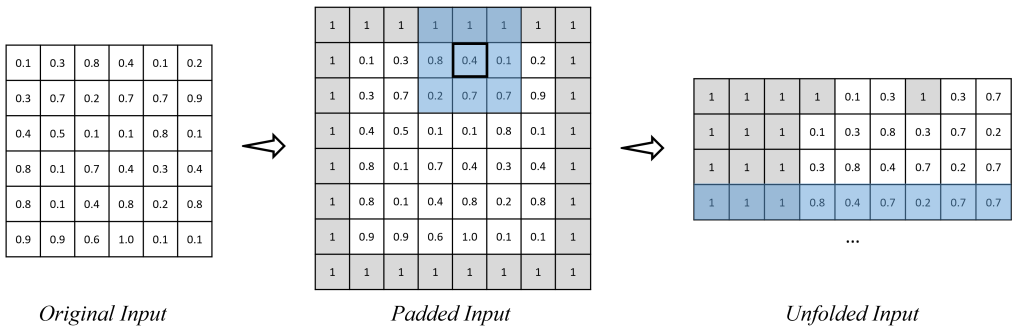

Figure 21 An illustration of the padding and unfold operations on the original input. We pad 1s to neighbors within the distance of $r = 1$ *for each pixel.*

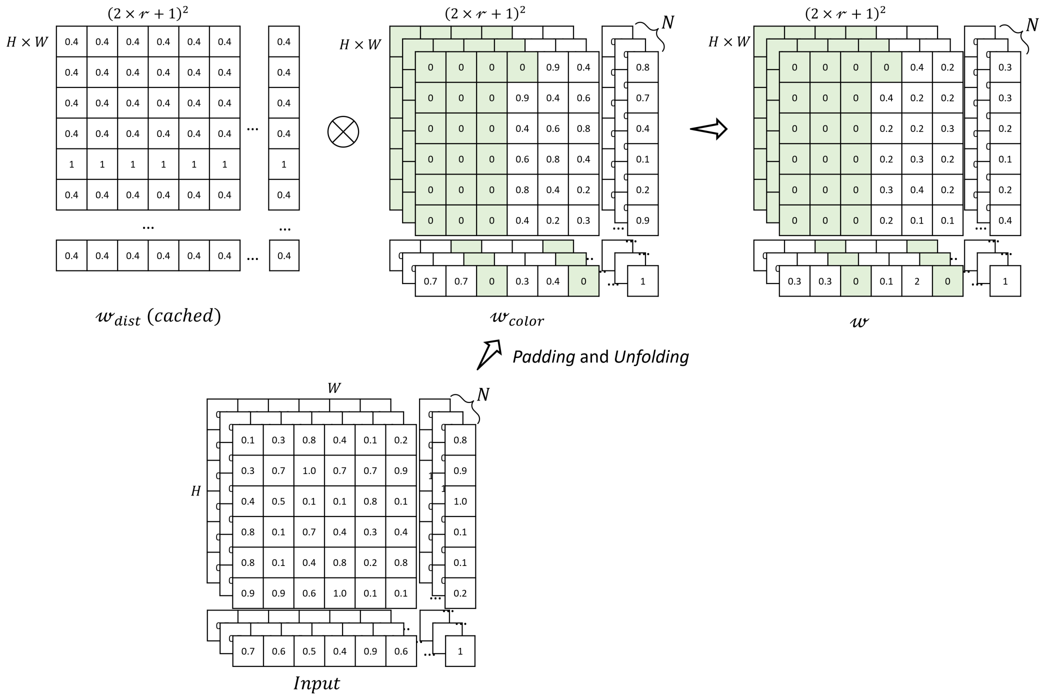

Figure 22 The optimized algorithm obtains the weight matrix w*.*

The final fast normalized cut algorithm for computing w is shown in Figure 22. I performed a series of experiments to demonstrate 10×-2207× compared to the naive algorithm in Figure 20 regarding different batch sizes ($\mathcal{B}$) and radii (r). The benefits of our algorithm come from the combination of caching, skip zeros, and batching optimizations. Note that $\hat{Y}$ and Y in Equation 5-7 are also unfolded in our algorithm to match the shape of w.

5.4. Experiments

This section presents and discusses our dataset's evaluation results of U-Net, SU-Net, and W-Net models. I ran experiments on a Linux machine equipped with a 32-GB NVIDIA A100 GPU. Our code is written in PyTorch 1.11 and Python 3.9.0.

Metrics and Training Parameters. I evaluated the accuracy of different models using two metrics---Pixel Accuracy (PA) (Zhao et al., 2018) and mean Intersection of Union (mIoU) shown in Equations 5-8 and 5-9, accordingly.

$$PA = \frac{\sum_{k=1}^{\mathcal{K}} Overlap(Y^k,\ \hat{Y}^k)}{\sum_{k=1}^{\mathcal{K}} Y^k} \tag{5-8}$$

PA measures the ratio of the total number of overlapped pixels between predicted segmentation and the ground truth and each image's total number of pixels. While PA is a standard metric to indicate the percent of correctly predicted pixels, it imposes bias towards objects with many pixels and overlooks the objects with fewer pixels. Therefore, I also used mIoU that averages over each object's extent of overlap of predicted and ground truth regions.

$$IOU^k = \frac{Overlap(Y^k,\ \hat{Y}^k)}{Union(Y^k,\ \hat{Y}^k)}$$

$$mIOU = \frac{1}{\mathcal{K}} \sum_{k=1}^{\mathcal{K}} IOU^k \tag{5-9}$$

In the training phase, I used the Adam optimizer (Kingma D. P., 2014) with a learning rate of 0.001 for all models. The batch size for U-Net and SU-Net is 16, while the batch size for W-Net is eight because it consumes a large amount of GPU memory, such that the maximum batch size a single A100 GPU could endure is eight. All models were trained with ten epochs. To train SU-Net specifically, we used $r = 2$, $\sigma_c^2 = 4$, and $\sigma_p^2 = 1$ in Equation 5-4.

5.4.1. Results Overview

Table 3 compares the accuracy and training time of each model. The PA and mIoU in the table were measured on the test dataset. When both SU-Net and U-Net consumed 100% labeled data, they yielded similar PAs, while SU-Net's mIoU was slightly better than U-Net's. Also, the time spent training U-Net, and SU-Net is close. However, it is observed that SU-Net could achieve 0.10 higher mIoU and 0.04 higher PA than U-Net when both were trained using 1/16 (~6%) labeled data. Note that U-Net (1/16) is faster than SU-Net (1/16) because U-Net (1/16) does not take unlabeled data. In comparison, W-Net's accuracies are much lower than SU-Net (1/16) even if all labeled data are used. More than that, W-Net takes 33× longer to train than SU-Net. Two reasons cause the slowness of W-Net. The most significant slowdown is caused by W-Net's naïve algorithm to compute the normalized cut loss. In addition, W-Net's architecture consists of two U-Nets, while either SU-Net or U-Net only has one.

Table 3 Accuracy and training time comparison of multiple model variants. The SU-Nets in the table were trained using $\mu = 0.01$.

Model (ratio of labeled data)	*Pixel Accuracy*	*mIoU*	*Total Time to Train (sec)*
U-Net	0.88	0.53	81.67
U-Net (1/16)	0.83	0.43	17.82
SU-Net	0.89	0.56	87.87
SU-Net (1/16)	0.87	0.53	87.31
W-Net	0.63	0.26	2871.72

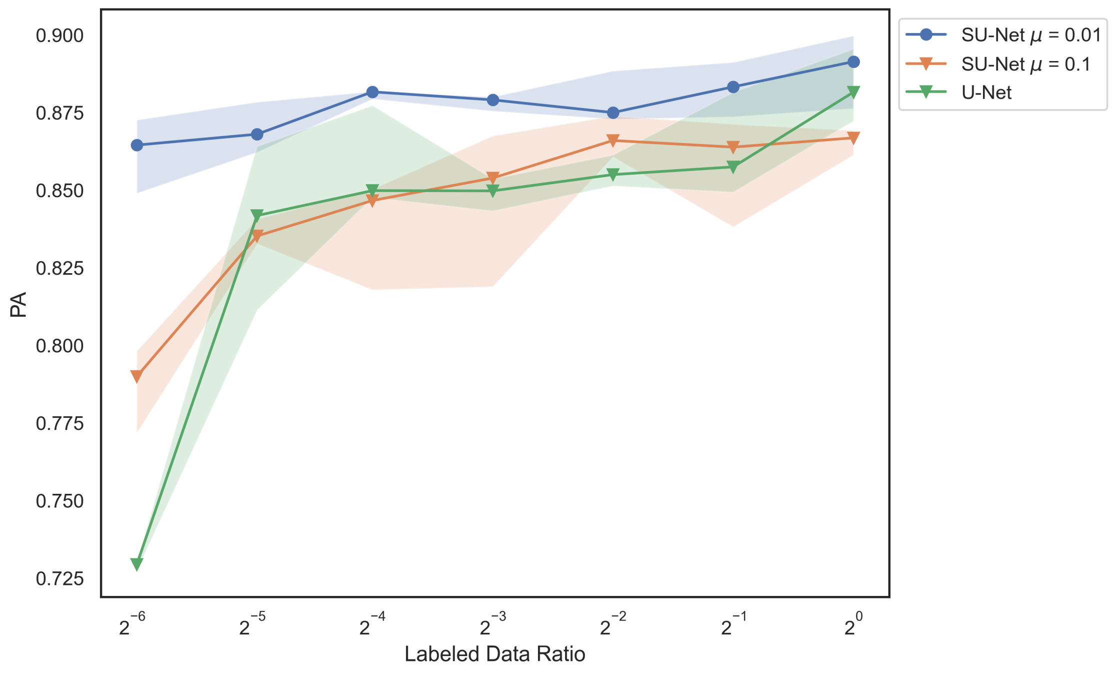

Figure 23 Illustration of how the semi-supervised factor μ affects PA under different label data ratios

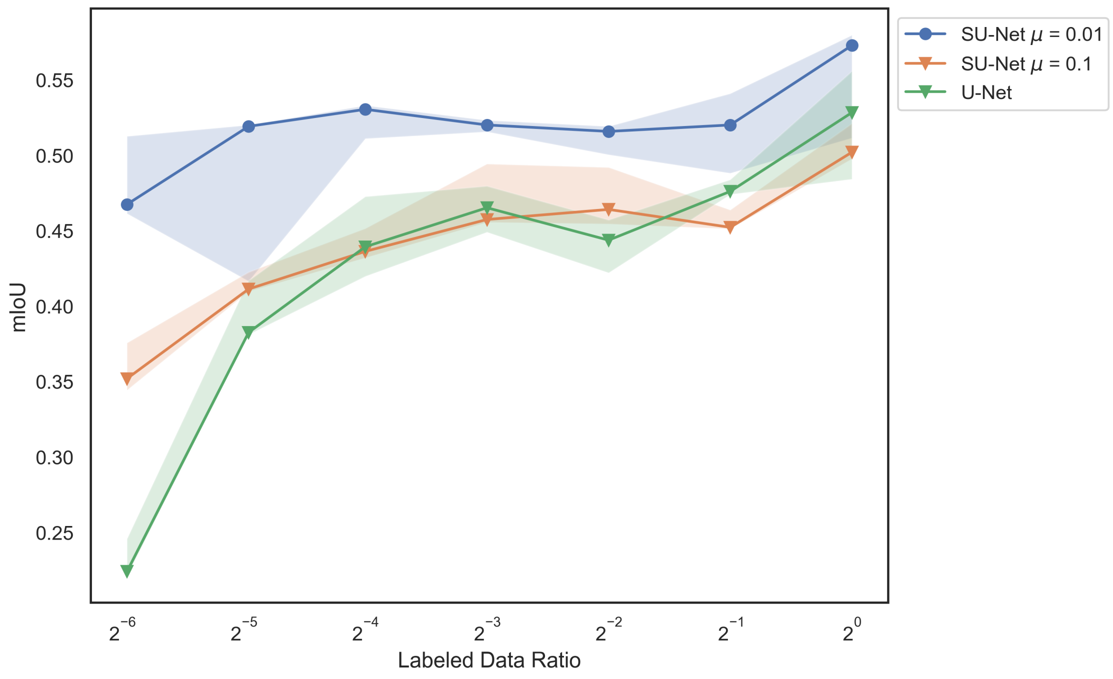

Figure 24 Illustration of how the semi-supervised factor μ affects mIoU under different label data ratios

5.4.2. Effects of the Regularization Parameter μ.

This work studied PA and mIoU on the test dataset using SU-Net trained using μ =0.1 and $\mu = 0.01$ with labeled data ratios from 2^{-6} to 1. At each ratio, I ran experiments five times and extracted the median, upper bound, and lower bound values in Figure 23 and Figure 24. it is observed that SU-Net ($\mu = 0.01$) 's accuracies are better than U-Net under all setups. SU-Net ($\mu = 0.1$) is better than U-Net when the labeled data ratio is small and becomes close to U-Net with the increase of labeled data. This phenomenon indicates that a large regularization value may cause a model to be under-fitted.

5.4.3. Training & Validation Curves Comparison.

Figure 25 and Figure 26 compare the training and validation mIoU curves of SU-Net (1/16) and U-Net (1/16) accordingly. Since U-Net and SU-Net were trained with different volumes of data, I used progress to indicate the percentage of steps. Interestingly, U-Net's training mIoU is close to SU-Net, but its validation mIoU is consistently lower than that of SU-Net. It is believed that U-Net was overfitted with the small training dataset so that it couldn't be generalized for the validation dataset.

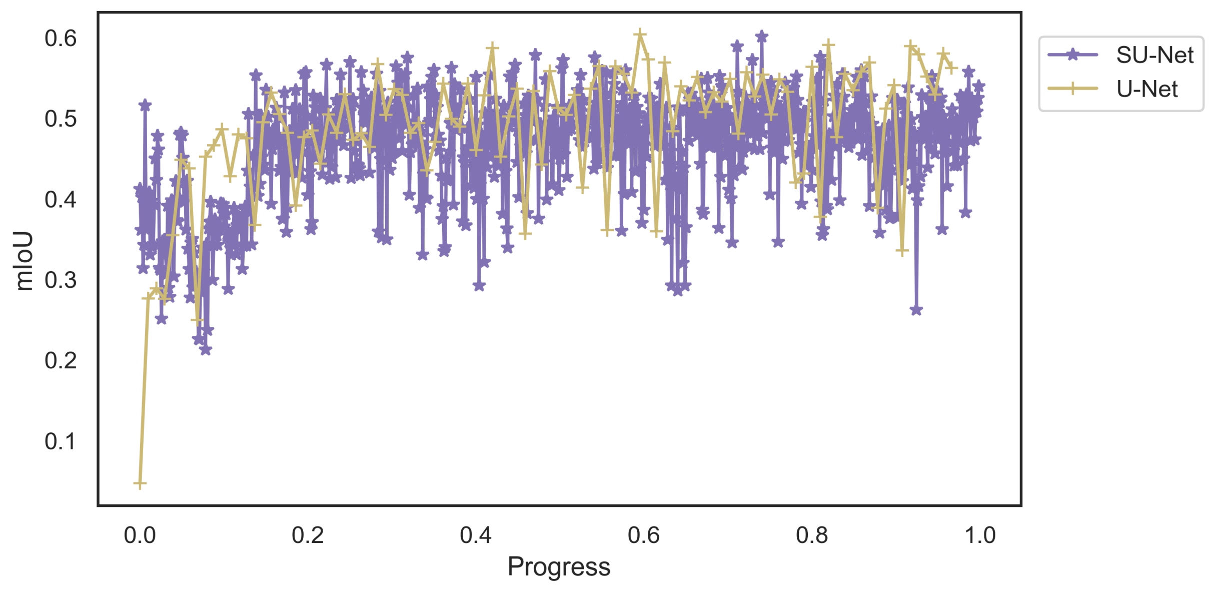

Figure 25 The change of mIoUs on the training dataset

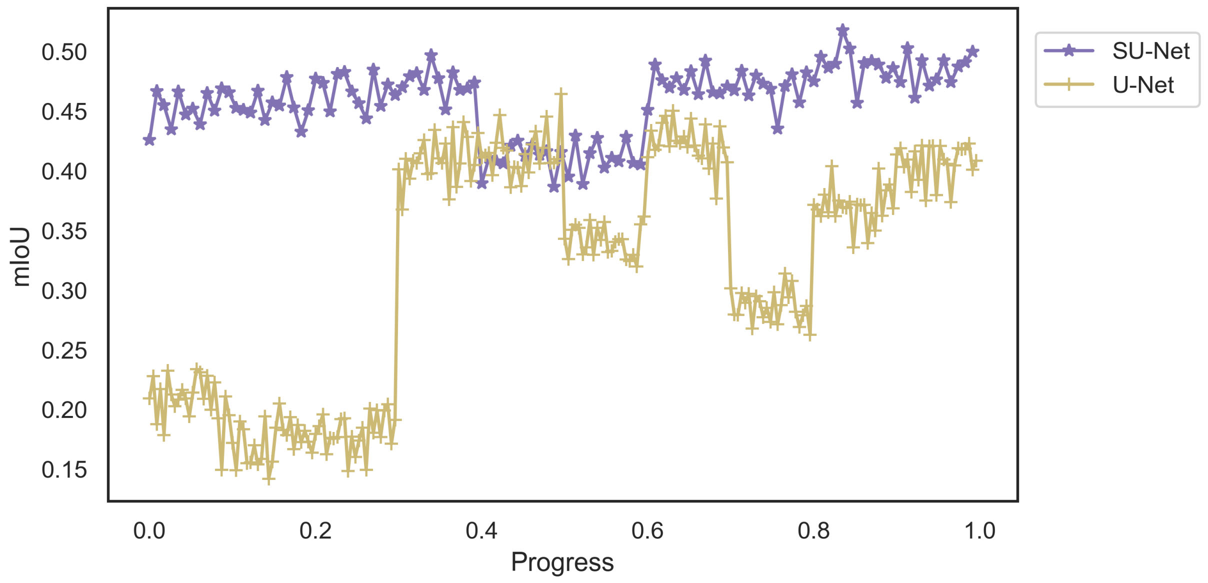

Figure 26 The change of mIoUs on the validation dataset

5.4.4. Speedup of Fast Normalized Cut

Next, I compare the speedups achieved by the fast normalized-cut segmentation algorithm with the original normalized cut algorithms used by W-Net. I measured the time of computing the normalized cut with different batch sizes and radius (r) parameters, which significantly affected the computation time. From Figure 27, we can make the following observations:

- The larger the batch size, the higher speedup fast normalized cut can bring, thanks to the batching optimization.
- The smaller the radius, the higher speedup fast normalized cut can have due to the skip zeros optimization.
- The highest speedup (2209×) is achieved with batch size = 32 and radius =1. Even with a large radius (16) and a small batch size (1), the fast normalized cut can still get greater than 10× speedup than the naïve normalized cut.

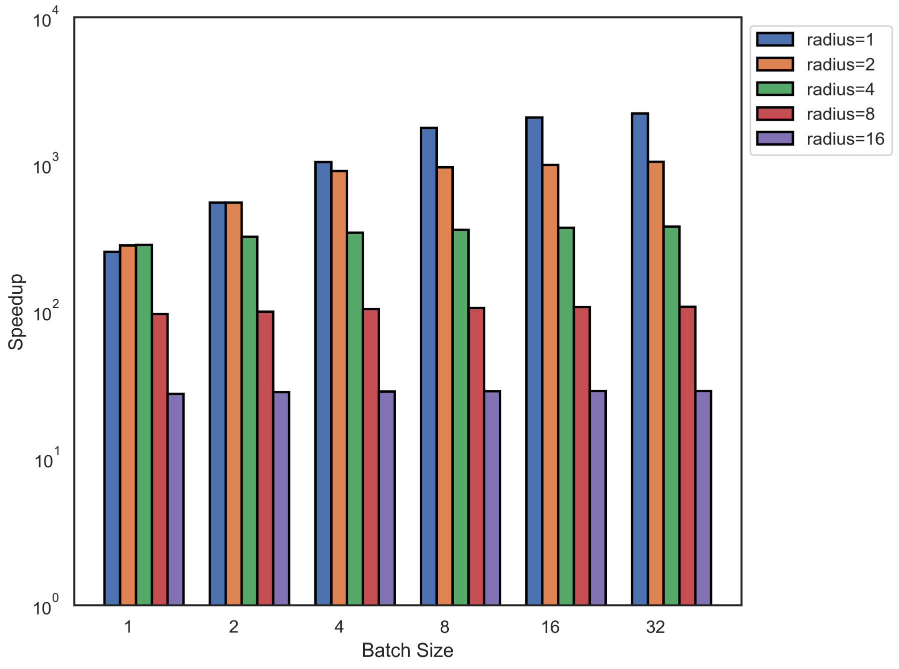

Figure 27 The speedup of fast normalized cut over the naive normalized cut with different batch sizes and radii. We used $\mathcal{K} = 4$ in all experiments, representing the number of objects in shale images.

5.4.4. Segmentation Results

I compare the segmentation results using different models in Figure 28. It is observed that U-Net (1/16) misclassifies many objects compared to SU-Net (1/16), even for objects that can be differentiated with colors (pores). In comparison, SU-Net (1/16) can classify pores accurately due to the use of a larger (unlabeled) dataset and the normalized cut loss ($\mathcal{L}^{segmentation}$). However, both SU-Net (1/16) and U-Net (1/16) cannot classify clay correctly because of lacking labeled data. SU-Net and U-Net's segmentation results are close. U-Net tends to yield bolder strikes of clays in several pictures, but it can false-positively classify some regions as clays as indicated by blue boxes. In contrast, SU-Net uses $\mathcal{L}^{segmentation}$ to regularize the final loss, preventing it from overfitting, which leads to a slightly higher accuracy than U-Net's.

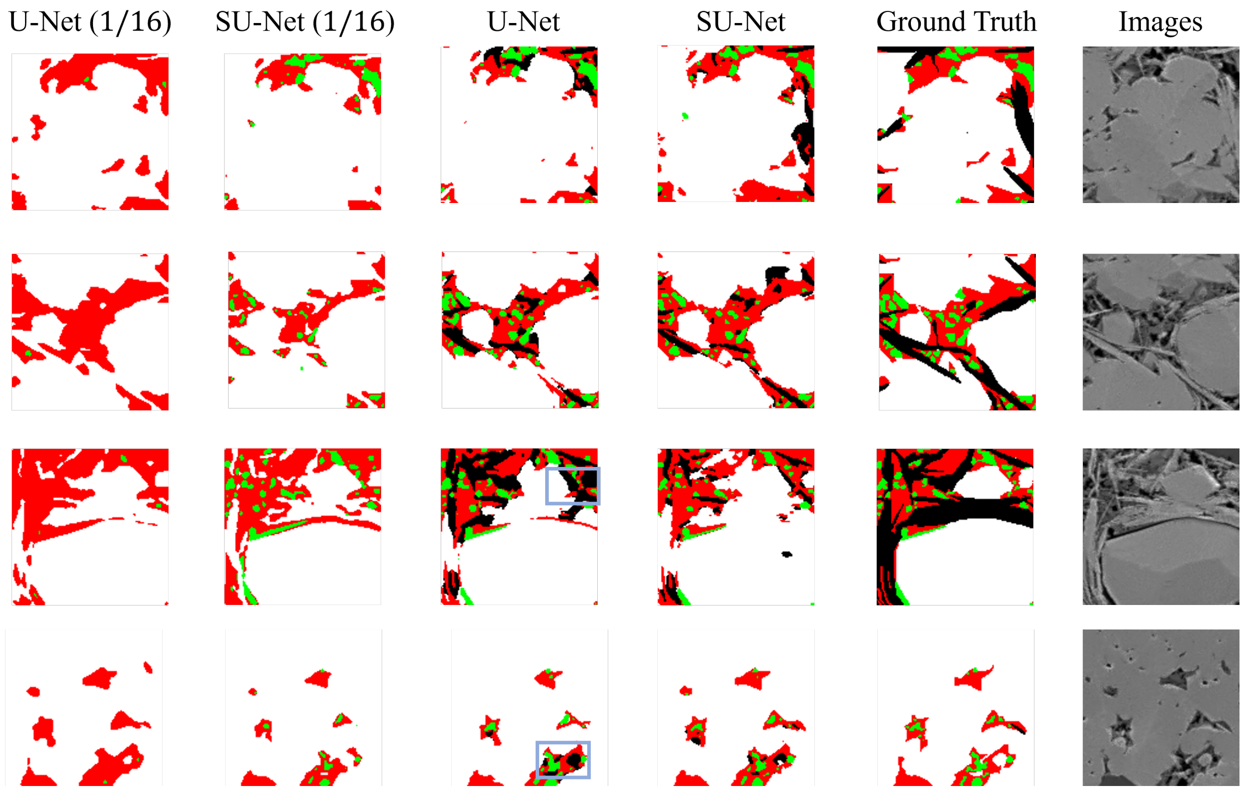

Figure 28 The segmentation results on the test dataset using different models

5.5. Discussion

Evaluating unsupervised models (e.g., W-Net) is different from evaluating supervised or semi-supervised models, of which we know the correspondence between predicted indices and true labels. For example, consider $\widehat{Y_\iota}$ as the predicted output of a pixel. If labels are sorted in the order $[pores, clays, organic\ matter, mineral\ grains]$, $\widehat{Y_\iota^1}$ refers to the probability of this pixel belonging to pores. However, since we don't use labels in unsupervised learning, such information is missing in the testing stage. In other words, we don't know which class that $\widehat{Y_\iota^1}$ refers to. To solve this problem, this work derived a heuristic approach to match predicted indices with labels. For W-Net, its output of U_{enc} is a four-dimensional probability tensor $[B, H, W, \mathcal{K}]$, where $\mathcal{K}$ denotes the total number of classes. This study first counted the frequency of different labels based on the ground truth labels of an image. Next, this work applied argmax on the output probability tensor to get each pixel's predicted index that has the maximum probability among all $\mathcal{K}$ classes, count the frequency of predicted indices, and match predicted indices with the ground truth labels based on their sorted order of frequency. For instance, if the sorted labels based on frequency are $[pores, clay, organic\ matter, mineral\ grains]$, and the sorted indices based on frequency is $[4, 3, 1, 2]$, the fourth dimension in the output tensor refers to the first class (i.e., pores). It is noteworthy that this approach could cause errors when the two labels have similar frequencies, but it is the best we could do without any label information.

Also, it is observed that some shale rock images in our dataset have noises. Supervised learning and semi-supervised learning can tolerate some noise with label information available, while these noises can cause significant accuracy degradation for unsupervised learning approaches. Hence, I denoised images using the non-local mean algorithm before training and testing (Buades et al., 2005).

Chapter 6: A Hybrid Active Learning Approach for Shale Image Segmentation

Using semi-supervised learning, this work was able to effectively achieve performance comparable to supervised learning, even with a limited number of labeled images. Interestingly, this study suggested that increasing the size of the labeled dataset didn't necessarily result in a proportional accuracy increase; after using 1/8 labeled data, additional increments of labeled data resulted in a minor accuracy improvement. This observation prompted us to take a step back and consider which images would be the most valuable to label to improve model performance. With a large number of unlabeled images available to us and with the high cost associated with obtaining labels, active learning (or human-in-the-loop learning) is the most promising approach to follow (Wu et al., 2022).

6.1. Overview

This study enhanced an active learning framework on top of the VAAL model (Sinha et al., 2019) to choose the most beneficial images for labeling. The original VAAL is used for image classification tasks only. Moreover, since VAAL only considers samples that are not well-represented (i.e., the diversity rule) and is task-agonistic, I tailored VAAL's architecture to fit the needs by considering both uncertainty and diversity measurements.

More formally, I define the active learning process in our study as follows:

Definition 1 (Active learning process). We target a segmentation problem on images with $\mathcal{K}$ classes. The images are divided into two sets, S_L and S_U, where S_L represents the labeled dataset, and S_U represents the unlabeled dataset. Initially, we train a model with $\mathcal{X}$ percent of the original data, where $\mathcal{X}$ is a small number < 2%. Next, we adopt active learning to train the model for T iterations. At each iteration, we train a segmentation model (e.g., U-Net) using S_L. Then, we apply our acquisition function (described below) to acquire a batch of B samples from S_U. The acquired samples are then labeled by a human expert, and they are removed from the S_U and added to the SL, which will serve as the training set for training a model in the next iteration.

Definition 2 (Acquisition function). In active learning, an acquisition function $\alpha(x) \rightarrow [0,1]$ refers to the rule or strategy used to calculate the probability of selecting the next sample to achieve the maximum training performance. For functions that calculate the α based on each pixel instead of an entire image, we have $\alpha(X) = \sum_{x \in X} \frac{\alpha(x)}{|X|}$ without loss of generality, where X represents the whole set of pixels x for each image.

My research aims to find an effective acquisition function that finds as minimum labeled samples as possible to achieve comparable performance when all samples have been labeled.

6.2. Uncertainty sampling

Uncertainty sampling focuses on identifying unlabeled data instances where the model is least confident in its predictions. For example, for a binary classification task, this often means selecting instances where the model assigns roughly equal probability to both possible labels. By prioritizing these uncertain instances for labeling, the model can focus on the data points where it is most likely to make mistakes or where labeling will lead to the biggest improvement in accuracy. Below I list a few representative uncertainty sampling methods (Settles, 2009) in active learning implemented in this study. In many cases, the scores used to quantify model uncertainty are obtained after applying a Softmax operator, which normalizes the probabilities so that they fall within the [0, 1] range.

Least confidence score. This strategy identifies unlabeled data instances on which the model is least confident. The basic equation (Equation 6-1) is simply the one minus the probability of the highest confidence for the label. For example, in a multi-class classification task, a model may predict that an unlabeled instance is most likely to belong to Class A, with a predicted probability of 0.6. This would yield a "least confident" score of 0.4; in general, lower least confidence scores suggest greater uncertainty and indicate that labeling these instances could be most beneficial for the model.

$$\alpha(x) = 1 - \max\left(p(\hat{y}_i|x)\right) \tag{6-1}$$

Margin score: In active learning, the "margin" score is a measure of model uncertainty based on the difference between the probabilities of the top two most likely output classes (Equation 6-2). If the model is very confident in its prediction, the margin score will be high (since there will be a large difference between the highest and second-highest probability). Lower margin scores, on the other hand, suggest greater uncertainty since the model is not clearly leaning towards one specific output class. Margin sampling in active learning is a technique that prioritizes instances with lower margin scores for labeling since these are expected to be the most informative for the model.

$$\alpha(x) = 1 - \left(\max\left(p(\hat{y}_i|x)\right) - \max_{j \neq i}\left(p\left(\hat{y}_j|x\right)\right)\right) \tag{6-2}$$

Entropy: It is designed to sample data for labeling to maximize the information gained from each selection. The intuition behind it is that selecting the most uncertain examples will lead to the most informative labels, which will help the classifier improve faster. Mathematically, entropy is a measure of uncertainty in a probability distribution. For a classification problem with $\mathcal{K}$ classes, the entropy of a predicted probability distribution p is defined as:

$$\alpha(x) = -\sum_{i=1}^{\mathcal{K}} p(\hat{y}_i|x) log\left(p(\hat{y}_i|x)\right) \tag{6-3}$$

Thus, the higher the entropy, the less certain the classifier has in its predictions for that data point. To use entropy-based uncertainty sampling, we simply select the data points with the highest entropy for labeling.

My major intuition for using multiple uncertainty metrics is that a single metric may not be informative enough to represent the model. There’s no single best choice based on different conditions. For example, it is possible that the least confidence score of a sample is high, but its margin score is relatively low; we cannot say which one performs the best in such a case.

6.2. Diversity sampling

Diversity sampling intends to find a collection of samples that can well represent the entire data distribution. Typical approaches (Monarch, 2021) include:

- Model-based outlier sampling, which identifies items that the model has not encountered before.
- Cluster-based sampling, which uses statistical methods to find a diverse group of items for labeling.
- Representative sampling, which selects items that closely match the target domain.
- Sampling for real-world diversity, which aims to reduce bias by including a wide range of real-world entities in the training data.

Among the above four approaches, cluster-based sampling and model-based outlier sampling are the most used. Cluster-based sampling aims to select a diverse set of data instances to label to cover the entire input space. This approach starts by clustering the unlabeled data into groups (e.g., K-means; Hartigan & Wong, 1979) and then selecting data instances from different clusters to label to maximize the information gained from labeling. Cluster-based sampling can be useful to avoid sampling bias or focus on areas of the input space.

Model-based outlier sampling measures the distance between the model's predictions and a given data point to identify outliers or instances that do not fit the model's expectations. By labeling these "outlier" instances, the model can improve its performance in these difficult areas. As a variant, there also exists contrastive learning-based approaches that train a contrastive model to identify whether an input has been labeled or not (Margatina et al., 2021).

Model-based outlier sampling has the advantage of pinpointing items that are completely unknown to the model, which allows for targeted labeling and learning. However, the disadvantage is that this method may not catch items that are merely uncertain rather than outright unknown.

Cluster-based sampling, on the other hand, can capture a broader range of items by identifying natural trends in the data. However, the disadvantage is that it is not specific to a particular model, which means it may pick up items that are not as informative or useful for the model's learning.

6.3. VAAL

This work improves VAAL (Sinha et al., 2019), which uses VAE to learn a low dimensional latent vector from labeled and unlabeled data and uses the discriminator to differentiate between them. As described in Figure 29, VAAL employs the model-based outlier acquisition function to find samples sufficiently different in the latent space learned from VAE. It starts by training a task learner using labeled data. Next, both labeled data and unlabeled data are fed into the VAE, whose loss function considers both the KL divergence and the reconstruction loss. After the VAE has been trained, I extract the latent space from the VAE and input it into the discriminator. Unlabeled image samples have a label of 0, while labeled image samples have a label of 1.

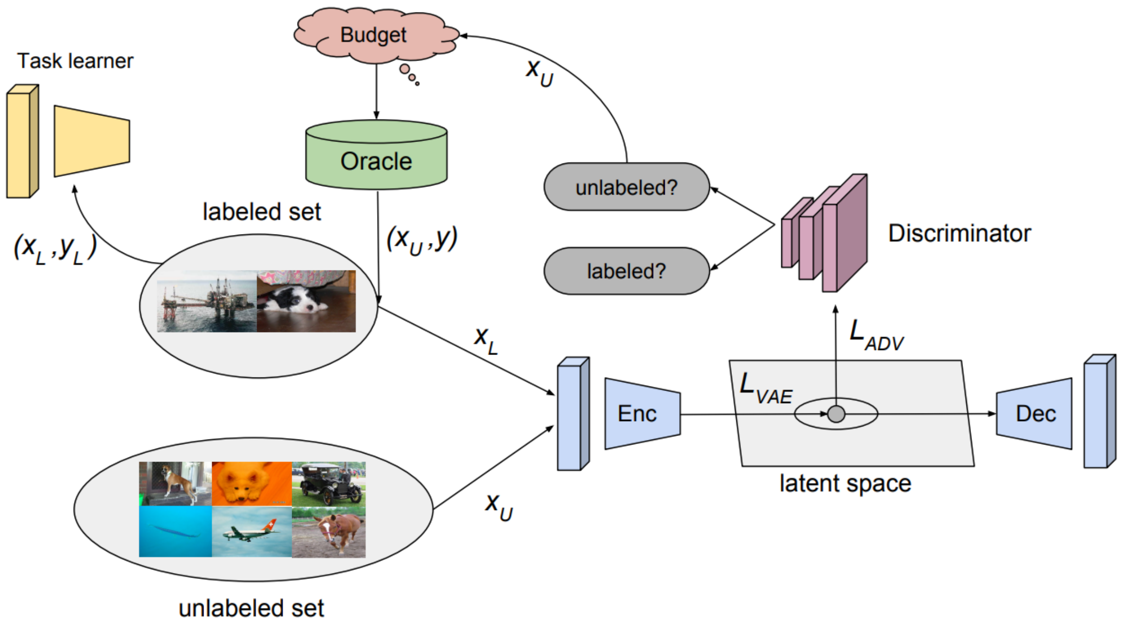

Figure 29 The workflow of the VAAL model (Sinha et al., 2019)

The loss function of the original VAAL is described in Equations 6-4 to 6-8.

$$\mathcal{L}_{VAAL} = \mathcal{L}_{unsupervised} + \mathcal{L}_{adversarial} + \mathcal{L}_{discriminator} \tag{6-4}$$

$$\mathcal{L}_{unsupervised} = \mathcal{L}_{reconstruction} + \mathcal{L}_{latent} \tag{6-5}$$

$$\mathcal{L}_{latent} = \mathcal{KL}(\widehat{p_z}|p_z) \tag{6-6}$$

$$\mathcal{L}_{adversarial} = -logD(\widehat{p_z}) \tag{6-7}$$

$$\mathcal{L}_{discriminator} = -log(D(\widehat{p_{z_l}})) - log(1 - D(\widehat{p_{z_u}})) \tag{6-8}$$

VAAL uses a β-variational VAE (Higgins et al., 2016) in which the KL divergence calculates the difference between the predicted distribution ($\widehat{p_z}$) and a unit Gaussian prior (p_z), which has a mean value equal to zero and a variance value equal to one. We can view VAAL as yet another MTL training since it optimizes multiple targets. $L_{adversarial}$ is a special form of cross-entropy loss as it fools the discriminator by assuming that all labels are one. In contrast, the discriminator is trained to differentiate between labeled ($\widehat{p_{z_l}}$) and unlabeled images ($\widehat{p_{z_u}}$). After the training steps of VAAL are done, we can extract only the encoder part of VAAL for inference. Therefore, we can define the acquisition function $\alpha(x)$ as the output of the last layer of the encoder.

6.4. Uncertainty-Aware VAAL

Existing active learning approaches are either task-aware or task-agonistic. Task agonistic approaches (Sener & Savarese, 2017; Mac Aodha et al., 2014) are equipped with acquisition functions that only consider the features of the input, while task-aware approaches (Tran et al., 2019; Wang & Ye, 2015; Kim et al., 2021) consider both the input and the performance of the current task model. In other words, task-aware models optimize the acquisition function using uncertainty scores, while task-agonistic models do not. One advantage of task-aware models is that we can let the model learn difficult samples other than different samples. However, task-aware models typically rely on the output of the task model $p(y|x)$, which is inaccurate at the early training stage.

VAAL is task-agonistic, meaning that it does not take the task model (i.e., segmentation model)'s performance into account. To make VAAL task-aware, our core idea is to extend the input features of the discriminator to let it learn both input distribution and model distribution. Assuming our task model can be trained accurately based on a certain amount of training data at each training step, essentially, the output of the discriminator indicates whether the sample can be trained well and has distinct features from the labeled dataset.

My approach (U-VAAL) differs from the approach proposed by Kim et al. (2021) from two perspectives. First, I used uncertainty metrics such as the least confidence score, but they used the output of a special "ranker" that is trained to compare the expected rank of uncertainty and that of the actual, which suffers from instability at the start of training. Second, my input to the discriminator can be a latent vector concatenated with an arbitrary number of uncertainty features, as illustrated in Equation 6-8, while they only extended the latent vector with a single feature.

$$\mathcal{L}_{discriminator} = -log(D(\widehat{p_{z_l}}, U_1(x_l), \dots, U_n(x_l))) - log(1 - D(\widehat{p_{z_u}}, U_1(x_u), \dots, U_n(x_u))) \quad (6\text{-}8)$$

Figure 30 and Figure 31 intuitively demonstrate the difference between VAAL and U-VAAL. Circles with different colors represent samples belonging to different categories. The dashed line/curve represents the decision boundary of the corresponding method. It can be seen that U-VAAL can choose samples with both high uncertainty and diversity to consider not only the feature distribution but also the difficulty of the samples.

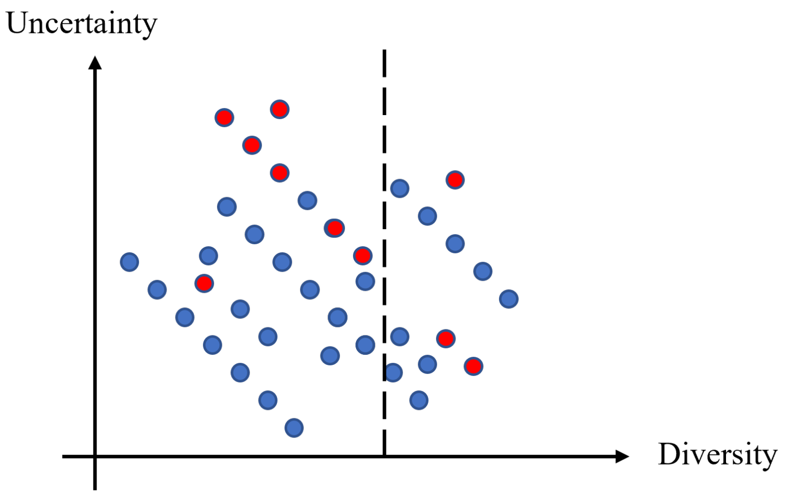

Figure 30 An illustration of how VAAL chooses samples only based on diversity; highly diverse samples will be chosen for the right part of the dashed line.

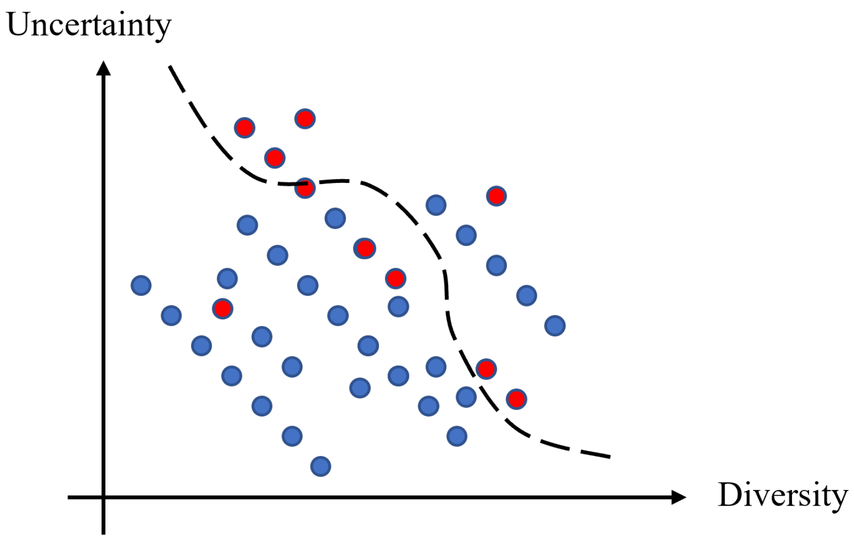

Figure 31 An illustration of how U-VAAL chooses samples based on diversity and uncertainty. Samples on the right part of the curve will be chosen.

6.5. Model Training and Inference

Figure 32 describes the U-VAAL framework. The task model is a segmentation model (e.g., U-Net), just like the unsupervised learning and the supervised learning approach. Our aim is to train the task model with as little data as possible using active learning. Based on Theorem 1, train the task model from scratch for *T* training iterations. At the end of each iteration, I apply a sampling procedure that can be further separated into two sub-steps. First, I train the VAE model by minimizing

$\mathcal{L}_{unsupervised}$ and $\mathcal{L}_{adversarial}$. After the loss of the VAE model is backpropagated, I start optimizing the discriminator by minimizing $\mathcal{L}_{discriminator}$. Unlike VAAL, I not only use the latent vector from VAE but also concatenate it with one or more uncertainty scores. In addition, I introduce a "cold start" phase: only when the accuracy of the task model is higher than a certain threshold $\mathcal{T}$, I start to incorporate uncertainty scores. At the very beginning of the training process, it can be observed that the output of the task model was unstable because of the limited training data. Thus, the "cold start" phase, together with the combination of multiple uncertainty scores, can help us mitigate the problem. In this way, the VAE model is trained to fool the discriminator model into believing that all the samples are from the labeled data, while the discriminator is trained to differentiate labeled from unlabeled samples.

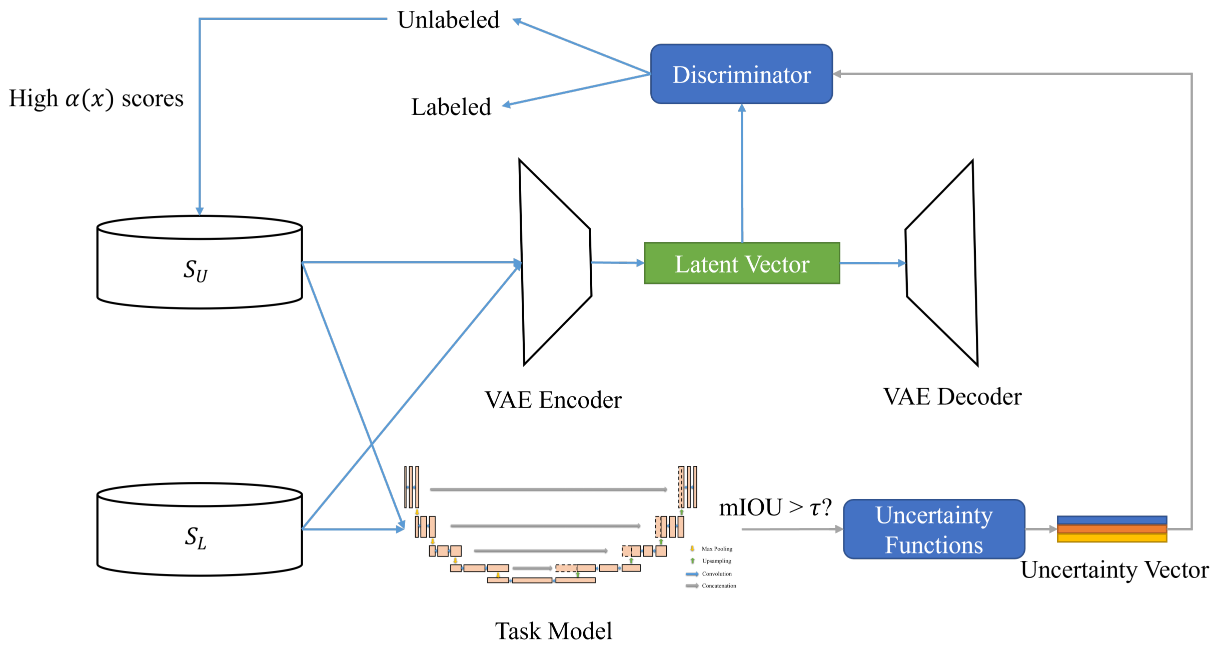

Figure 32 Our U-VAAL Framework. The uncertainty vector is used when the task model's mIoU > τ

6.6. Experiments

This section presents and discusses the SEM dataset's evaluation results of U-VAAL against other methods. I ran experiments on a Linux machine equipped with a 32-GB NVIDIA A100 GPU. My code is written in PyTorch 1.14 and Python 3.9.9. Compared with Project 1 and Project 2, I increased the number of labeled images to 12 and sliced them into 5158 sub-images. The training dataset contains 4366 sub-images, and the testing dataset contains 792 sub-images.

This dissertation designs experiments based on Theorem 1 by simulating the "labeling" process. I don't send our labels to human experts for labeling; instead, I just switch the state of an image from S_U to S_L as if it were labeled. I incrementally fetch 1% of labeled images from S_U to S_L based on the value of $\alpha(x)$ until 10% labels have been added for every three epochs.

In Table 4, We can compare the mIoU achieved by each model on the validation dataset. It is observed that U-VAAL's mIoU with 10% labeled data is higher than the other methods, including the least confidence score, margin score, entropy, and VAAL, which handle either uncertainty or diversity but not both. I didn't compare it against Kmeans since it consumes GPU memory larger than the amount that A100 is equipped with. It can be seen that VAAL and U-VAAL are better than basic methods, and U-VAAL achieves the best performance among all.

Table 4 Performance comparison between U-VAAL and existing methods

Model	*mIoU*
Least confidence score	0.59
Margin score	0.58
Entropy	0.54
VAAL	0.62
U-VAAL	0.69

Figure 33 shows the validation mIoU curve of each method across multiple iterations. Overall, our U-VAAL outperforms the rest of the models at any time. And U-VAAL is noticeably better than the original VAAL model. Interestingly, U-VAAL and VAAL show more significant advantages at the early stage of training (when the S_L is limited) than later when S_L contains sufficient data.

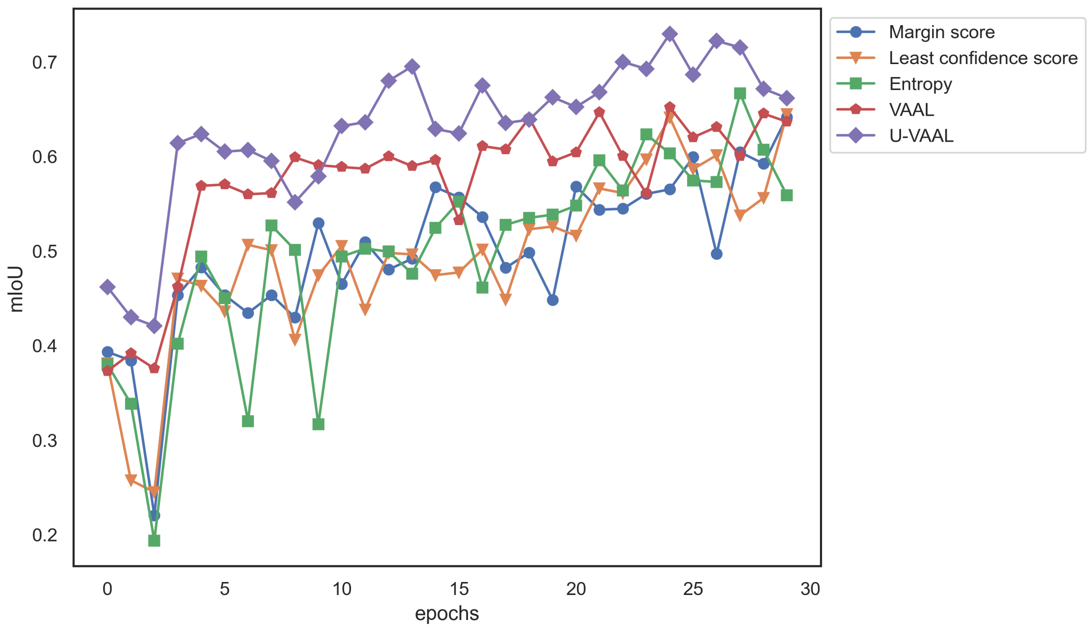

Figure 33 The mIoU curves of each model on the validation dataset

This dissertation applies multiple experiments to evaluate the impact of each variant in the method. First, I study the effect of using the "uncertainty vector" instead of a single uncertainty score and show corresponding results in Figure 34. It can be seen that U-VAAL (hybrid) achieves higher mIoU than the others, which uses a single uncertainty score and achieves similar performance.

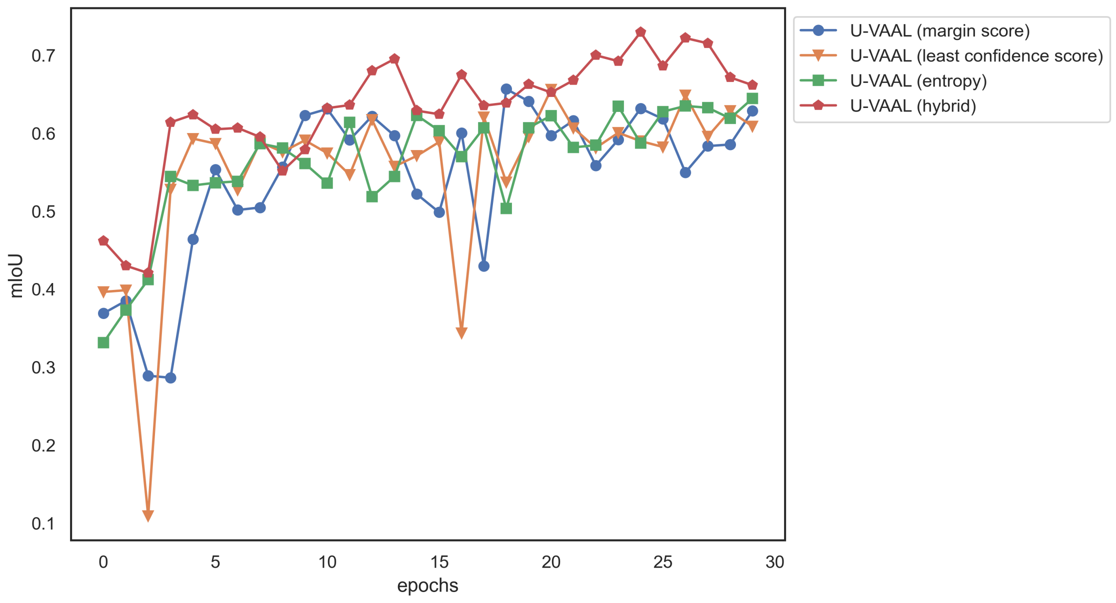

Figure 34 The mIoU curves of U-VAAL models using different uncertainty scores on the validation dataset

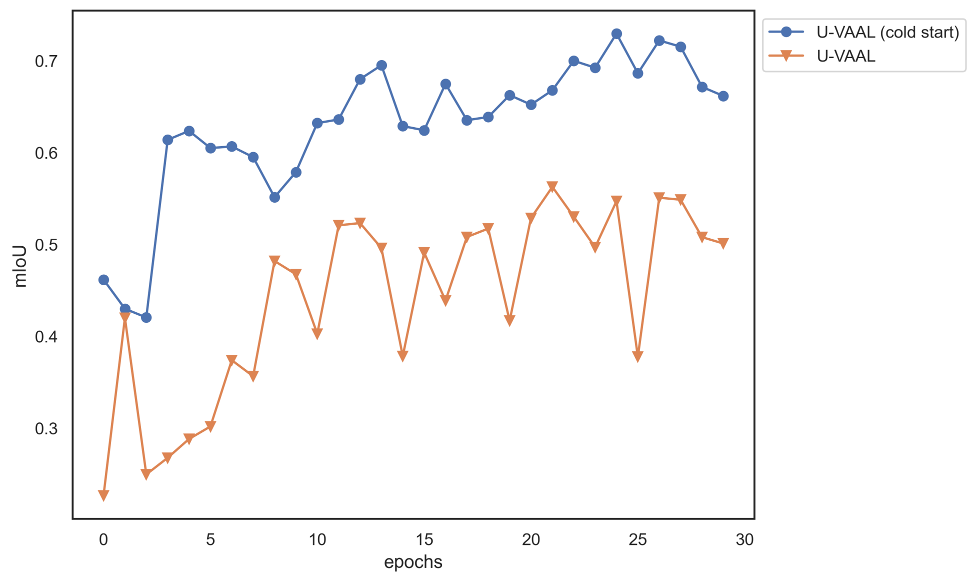

Figure 35 The mIoU curves of U-VAAL models w/ and w/o the cold start phase on the validation dataset

Next, I also study U-VAAL's performance w/ or w/o the cold start phase. Note that the results of U-VAAL shown in previous figures and tables were obtained with "cold start" enabled. From Figure 35, we can notice that both approaches achieved better performance with the increase of labeled data. However, U-VAAL without a cold start rendered relatively low and unstable mIoU scores (~0.3), compared with U-VAAL with a cold start (~0.5). When a cold start is enabled, samples are only acquired using the diversity score (VAAL). Thus, U-VAAL (cold start) only considers uncertainty scores after the sixth epoch, when the task model is more stable. This phenomenon demonstrates that our "cold-start" method can compensate for the problems that exist in task-aware models.

Chapter 7: Conclusions

The main problem being tackled in this dissertation is to train effective machine learning models for image segmentation with as little "cost" as possible in terms of the time to label images, the amount of data needed to train models, and the execution time for both model training and inference.

Project 1 (Chapter 4) addresses the challenge of unbalanced datasets, which can limit accuracy when certain objects infrequently appear in images. The novel neural network architecture, Paw-Net, overcomes this issue by using an ensemble-learning framework to reduce variance. Combining probability maps from multiple neural networks, along with the Focal Dice loss function, helps to balance negative and positive pixels, as well as difficult and easy pixels. Experiments show that Paw-Net outperforms existing methods for image segmentation with imbalance issues.

Project 2 (Chapter 5) deals with the challenge of scarce labeled data, which can restrict the effectiveness of model training. This research explores both unsupervised and semi-supervised learning models for shale image segmentation, resulting in two main contributions: the semi-supervised neural network model (SU-Net) and a fast normalized cut algorithm. SU-Net achieves acceptable accuracy with fewer labeled data requirements than the original U-Net, due in part to the use of unlabeled data as well as the normalized cut loss function. The fast normalized cut algorithm, which employs batch, skip zeros, and caching optimizations, reduces computation time significantly. In addition, the study found that adjusting the regularization parameter could further improve accuracy. Future work should explore dynamic parameter tuning and alternative semi-supervised architectures, such as the mean teacher model (Tarvainen & Valpola et al., 2017).

While the proposed semi-supervised learning method effectively reduces the number of images that require labeling, the selection of labeled images in Project 2 relied on random sampling. Consequently, **Project 3 (Chapter 6)** explores more effective sampling methods in order to find the most informative samples for labeling -- those that promote model accuracy without the need for additional data. The U-VAAL framework addresses this challenge by training a variational autoencoder (VAE) to fool a discriminator that distinguishes labeled and unlabeled datasets. The incorporation of uncertainty metrics as input for the discriminator further enables the accurate selection of labeled data. This approach proves more accurate than other active learning methods, given a limited amount of data.

Additional work is needed to measure the "human cost" of labeling images, as the assumption of linear growth in cost with the number of images may not always hold. For example, labeling an image with two objects may be easier than labeling an image with one hundred objects.

This research is also interested in generative models that can create realistic geological images in order to reduce the time and expense of photographing and sampling shale. Although existing generative models such as DALLE (Ramesh et al., 2022) and STABLE diffusion (Rombach et al., 2022) generate art images, similar work has not been conducted for high-resolution medical or geological images.

References

Abràmoff, M. D., Magalhães, P. J., & Ram, S. J. (2004). Image processing with ImageJ. Biophotonics International, 11(7), 36-42.

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., & Süsstrunk, S. (2012). SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11), 2274-2282.

Azarafza, M., Ghazifard, A., Akgün, H., & Asghari-Kaljahi, E. (2019). Development of a 2D and 3D computational algorithm for discontinuity structural geometry identification by artificial intelligence based on image processing techniques. *Bulletin of Engineering Geology and the Environment, 78*(5), 3371-3383.

Azarafza, M., Koçkar, M. K., & Faramarzi, L. (2021). Spacing and block volume estimation in discontinuous rock masses using image processing technique: a case study. *Environmental Earth Sciences, 80*(14), 1-13.

Badrinarayanan, V., Kendall, A., & Cipolla, R. (2017). Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12), 2481-2495.

Bishop, C. M., & Nasrabadi., N. M. (2006). *Pattern Recognition and Machine Learning* (Vol. 4, No. 4, p. 738). New York: Springer.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2), 123-140.

Bridle, J.S. (1990). Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing* (pp. 227-236). Springer, Berlin, Heidelberg.

Buades, A., Coll, B., & Morel, J. M. (2005). In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (Vol. 2, pp. 60-65).

Byambatsogt, G., Choimaa, L., & Koutaki, G. (2020). Guitar Chord Sensing and Recognition Using Multi-Task Learning and Physical Data Augmentation with Robotics. *Sensors ,* *20*(21), 6077.

Cai, W., Zhang, Y., & Zhou, J. (2013). Maximizing expected model change for active learning in regression. *2013 IEEE 13th International Conference on Data Mining (pp.* 51-60).

Cao, D., Ji, S., Cui, R., & Liu, Q. (2022). Multi-task learning for digital rock segmentation and characteristic parameters computation. *Journal of Petroleum Science and Engineering, 208*, 109202.

Castillo-Navarro, J., Saux, B. L., Boulch, A., Audebert, N., & Lefèvre, S. (2021). Semi-Supervised Semantic Segmentation in Earth Observation: The MiniFrance suite, dataset analysis and multi-task network study. *Machine Learning*, 111(9), 3125-3160.

Chapelle, O., Scholkopf, B., & Zien, A. (2009). Semi-supervised learning. *IEEE Transactions on Neural Networks*, 20(3), 542-542.

Chen, L.-C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., & Adam, H. (2022). Masklab: Instance segmentation by refining object detection with semantic and direction features. *Proceedings of The IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 4013-4022).

Chen, Y., Ze Wang, J., & Krovetz, R. (2005). Clue: Cluster-based retrieval of images by unsupervised learning. *IEEE Transactions on Image Processing, 14*(8), 1187-1201.

Chen, Z., Liu, X., Yang, J., Little, E., & Zhou, Y. (2020). Deep learning-based method for SEM image segmentation in mineral characterization, an example from Duvernay Shale samples in Western Canada Sedimentary Basin. *Computers & Geosciences*, 138, 104450.

Chuang, C.-Y., Robinson, J., Lin, Y.-C., Torralba, A., & Jegelka, S. (2020). Debiased contrastive learning. *Advances in Neural Information Processing Systems , 30*, 8765-8775.

Colling, P. L.-K. (2021). MetaBox+: A new Region Based Active Learning Method for Semantic Segmentation using Priority Maps. *arXiv preprint arXiv:2010.01884.*

Dahl, G. E., Sainath, T. N., & Hinton, G. E. (2013). Improving deep neural networks for LVCSR using rectified linear units and dropout. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing,* (pp. 8609-8613).

Deng, X., Zhang, Y., Yang, S., Tan, P., Chang, L., Ye, Y., & Wang, H. (2017). Joint hand detection and rotation estimation using CNN. *IEEE Transactions on Image Processing, 27*(4), 1888-1900.

Ebrahimi, S., Gan, W., Chen, D., Biamby, G., Salahi, K., Laielli, M., Zhu, S., & Darrell, T. (2020). Minimax active learning. *arXiv preprint arXiv:2012.10467* .

Freund, Y., Schapire, R., & Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 771-780.

Gao, F., Yoon, H., Wu, T., & Chu, X. (2020). A feature transfer enabled multi-task deep learning model on medical imaging. *Expert Systems with Applications, 143*, 112957.

Gao, Y., Bai, H., Jie, Z., Ma, J., Jia, K., & Liu, W. (2020). Mtl-nas: Task-agnostic neural architecture search towards general-purpose multi-task learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11543-11552.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. & Bengio, Y. (2020). Generative adversarial nets. *Communications of the ACM*, 63(11), 139-144.

Google. (2019, 5 14). *Generative Adversarial Networks.* Retrieved 11 20, 2022, from Google Developers: https://developers.google.com/machine-learning/gan/generator

Green, P. J. (1990). On use of the EM algorithm for penalized likelihood estimation. *Journal of the Royal Statistical Society: Series B (Methodological), 52*(3), 443-452.

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series c (Applied Statistics), 28*(1), 100-108.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1), 97–109.

He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask r-cnn. *Proceedings of the IEEE International Conference on Computer Vision*, (pp. 2961-2969).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

He, S. R., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*.

Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., & Lerchner, A. (2016). beta-vae: Learning basic visual concepts with a constrained variational framework. *The International Conference on Learning Representations*. Retrieved 11 20, 2022 from https://openreview.net/forum?id=Sy2fzU9gl.

Huang, H. R. (2021). Revisiting movable fluid space in tight fine-grained reservoirs: a case study from Shahejie shale in the Bohai Bay Basin, NE China. *Journal of Petroleum Science and Engineering, 207*, 109170.

Huang, H., Li, R., Jiang, Z., Li, J., & Chen, L. (2020). Investigation of variation in shale gas adsorption capacity with burial depth: insights from the adsorption potential theory. *Journal of Natural Gas Science and Engineering, 73*, 103043.

Izenman, A. J. (2013). Linear discriminant analysis. *In Modern multivariate statistical techniques (pp. 237-280).* Springer, New York, NY.

Jiang, H., & Learned-Miller, E. (2017). Face detection with the faster R-CNN. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)* (pp. 650-657).

Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 255-260.

Kalman, B. L. (1992). Why tanh: choosing a sigmoidal function. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks* (Vol. 4, pp. 578-581).

Kanezaki, A. (2018). Unsupervised image segmentation by backpropagation. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1543-1547.

Karimpouli, S., & Tahmasebi, P. (2019). Segmentation of digital rock images using deep convolutional autoencoder networks. *Computers & Geosciences*, 142-150.

Karras, T., Laine, S., & Aila, T. (2019). A style-based generator architecture for generative adversarial networks. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 4401-4410).

Kass, M., Witkin, A., & Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, 321-331.

Kim, K., Park, D., Kim, K. I., & Chun, S. Y. (2021). Task-aware variational adversarial active learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8166-8175.

Kim, W., Kanezaki, A., & Tanaka, M. (2020). Unsupervised learning of image segmentation based on differentiable feature clustering. *IEEE Transactions on Image Processing*, 29, 8055-8068.

Kingma, D. P. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint*.

Knaup, A., Jernigen, J., Curtis, M., Sholeen, J., John Borer, I., Sondergeld, C., & Rai, C. (2019). Unconventional Reservoir Microstructural Analysis Using SEM and Machine Learning. *SPE/AAPG/SEG Unconventional Resources Technology Conference.* OnePetro.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37(2), 233-243.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2021). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems 25 (2012): 1097-1105.*

Lamprier, S., Scialom, T., Chaffin, A., Claveau, V., Kijak, E., Staiano, J., & Piwowarski, B. (2022). Generative Cooperative Networks for Natural Language Generation. *arXiv preprint arXiv:2201.12320*.

Lassner, C., Pons-Moll, G., & Gehler, P. V. (2017). A generative model of people in clothing. *Proceedings of the IEEE International Conference on Computer Vision*, 853-862.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature , 521*(7553), 436-444.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278-2324.

LeCun, Y., Touresky, D., Hinton, G., & Sejnowski., T. (1988). A theoretical framework for back-propagation. *Proceedings of the 1988 Connectionist Models Summer School* (Vol. 1, pp. 21-28).

Lewis, D. D., & Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. *Machine Learning Proceedings*, 148-156.

Li, J., Bioucas-Dias, J. M., & Plaza, A. (2010). Semisupervised hyperspectral image segmentation using multinomial logistic regression with active learning. *IEEE Transactions on*

Geoscience and Remote Sensing, 48(11), 4085-4098.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal loss for dense object detection. *Proceedings of the IEEE International Conference on Computer Vision.*

Lindeberg, T. (1994). Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of Applied Statistics*, 225-270.

Lloyd, G. E. (1987). Atomic number and crystallographic contrast images with the SEM: a review of backscattered electron techniques. *Mineralogical Magazine, 51*(359), 3-19.

Loucks, R. G., Reed, R. M., Ruppel, S. C., & Jarvie, D. M. (2009). Morphology, genesis, and distribution of nanometer-scale pores in siliceous mudstones of the Mississippian Barnett Shale. *Journal of Sedimentary Research*, 848-861.

Mac Aodha, O., Campbell, N. D., Kautz, J., & Brostow, G. J. (2014). Hierarchical subquery evaluation for active learning on a graph. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 564-571.

Mackowiak, Radek, P. L., Ghori, O., Diego, F., Lange, O., & Rothe, C. (2018). CEREALS-Cost-Effective REgion-based Active Learning for Semantic Segmentation. *arXiv preprint arXiv:1810.09726*.

Mahmoodi-Eshkaftaki, M., Haghighi, A., & Houshyar, E. (2020). Land suitability evaluation using image processing based on determination of soil texture–structure and soil features. *Soil Use and Management, 36*(3), 482-493.

Margatina, K., Vernikos, G., Barrault, L., & Aletras, N. (2021). Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Meng, Z., Liu, Y., Jiao, X., Ma, L., Zhou, D., Li, H., Cao, Q., Zhao, M. & Yang, Y. (2022). Petrological and organic geochemical characteristics of the Permian Lucaogou Formation in the Jimsar Sag, Junggar Basin, NW China: Implications on the relationship between hydrocarbon accumulation and volcanic-hydrothermal activities. *Journal of Petroleum Science and Engineering, 210*, 110078.

Milletari, F., Navab, N., & Ahmadi, S.-A. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. *Fourth International Conference on 3D Vision (3DV)* (pp. 565-571).

Minaee, S., Boykov, Y. Y., Porikli, F., Plaza, A. J., Kehtarnavaz, N., & Terzopoulos., D. (2021). Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7), 3523-3542.

Monarch, R. M. (2021). *Human-in-the-Loop Machine Learning: Active Learning and Annotation For Human-Centered A.* Simon and Schuster.

Narayan, S. (1997). The generalized sigmoid activation function: Competitive supervised learning. *Information Sciences*, 99.1-2 (1997): 69-82.

Ng, H. P., Ong, S. H., Foong, K. W., Goh, P.-S., & Nowinski, W. L. (2006). Medical image segmentation using k-means clustering and improved watershed algorithm. *2006 IEEE Southwest Symposium on Image Analysis and Interpretation*, 61-65.

Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. *Proceedings of the IEEE International Conference on Computer Vision* (pp. 1520-1528).

Norouzi, M., Fleet, D. J., & Salakhutdinov, R. R. (2012). Hamming distance metric learning. Advances in Neural Information Processing Systems, 25.

Oppenheim, A. V. (1999). *Discrete-time signal processing.* India: Pearson Education.

Papandreou, G., Chen, L.-C., Murphy, K. P., & Yuille, A. L. (2015). Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. *Proceedings of the IEEE International Conference on Computer Vision*, 1742-1750.

Passey, Q. R., Bohacs, K. M., Lee Esch, W., Klimentidis, R., & Sinha, S. (2010). From oil-prone source rock to gas-producing shale reservoir–geologic and petrophysical characterization of unconventional shale-gas reservoirs. *International Oil and Gas Conference and Exhibition in China.* OnePetro.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. & Desmaison, A. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32

Pironkov, G., Dupont, S., & Dutoit, T. (2016). Multi-task learning for speech recognition: an overview. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (ESANN).* Retreived 11 20, 2022, from https://www.researchgate.net/profile/Gueorgui-Pironkov/publication/301789330_Multi-Task_Learning_for_Speech_Recognition_An_Overview/links/5728745a08ae262228b6bbde/Multi-Task-Learning-for-Speech-Recognition-An-Overview.pdf.

Polikar, R. (2012). In *Ensemble machine learning (pp. 1-34).* Springer, Boston, MA.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 779-788).

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684-10695.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical Image Computing and Computer-assisted Intervention* (pp. 234-241). Cham: Springer.

Russell, S., & Norvig, P. (1995). Artificial intelligence: a modern approach, Englewood Cliffs, N.J., Prentice Hall.

Saha, S. (2018, 15 12). *A Comprehensive Guide to Convolutional Neural Networks.* Retrieved 11 20, 2022, from towardsdatascience.com: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Sahoo, P. K., Soltani, S. A., & Wong, A. K. (1988). A survey of thresholding techniques. *Computer Vision, Graphics, and Image Processing, 41*(2), 233-260.

Salakhutdinov, R. (2015). Learning deep generative models. *Annual Review of Statistics and Its Application*, 361-385.

Sanchez-Lengeling, B., & Aspuru-Guzik, A. (2018). Inverse molecular design using machine learning: Generative models for matter engineering. *Science, 361*(2018), 360-365.

Seiler, H. (1983). Secondary electron emission in the scanning electron microscope. *Journal of Applied Physics, 54*(11), R1-R8.

Sener, O., & Savarese, S. (2017). Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Settles, B. (2009). Active learning literature survey. Retrieved 11 20, 2022, from https://burrsettles.com/pub/settles.activelearning.pdf.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*(8), 888-905.

Shivhare, S., & Cecil, K. (2021). Automatic soil classification by using Gabor wavelet & support vector machine in digital image processing. *Third International Conference on Inventive Research in Computing Applications (ICIRCA)*, 1738-1743.

Shore, J., & Johnson, R. (1981). Properties of cross-entropy minimization. *IEEE Transactions on Information Theory, 27*(4), 472-482.

Sinha, S., Ebrahimi, S., & Darrell, T. (2019). Variational adversarial active learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5972-5981.

Souly, N., Spampinato, C., & Shah, M. (2017). Semi supervised semantic segmentation using generative adversarial network. *Proceedings of the IEEE International Conference on Computer Vision*, (pp. 5688-5696).

Sudre, C. H. (2017). Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. *Deep Learning in Medical Image Analysis snd Multimodal Learning for Clinical Decision Support*, 240-248.

Suryanarayana, C., & Norton, M. G. (1998). X-rays and Diffraction. *In X-ray Diffraction*, 3-19.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *Proceedings of The IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2818-2826).

Tang, M., Djelouah, A., Perazzi, F., Boykov, Y., & Schroers, C. (2018). Normalized cut loss for weakly-supervised cnn segmentation. *Proceedings of The IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 1818-1827).

Tao, L., Zhu, C., Xiang, G., Li, Y., Jia, H., & Xie, X. (2017). LLCNN: A convolutional neural network for low-light image enhancement. *IEEE Visual Communications and Image Processing (VCIP)*, 1-4.

Tarvainen, A., & Valpola, H. (2017). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in Neural Information Processing Systems*, 30.

The U.S. Energy Information Administration. (2021, Dec 15). *Natural Gas Explained.* Retrieved Dec 24, 2021.

Top, A. G. (2011). Active learning for interactive 3D image segmentation. *International*

Conference on Medical Image Computing and Computer-Assisted Intervention, 603-610.

Tran, T., Do, T.-T., Reid, I., & Carneiro, G. (2019). Bayesian generative active deep learning. *International Conference on Machine Learning*, 6295-6304.

Triguero, I., García, S., & Herrera, F. (2013). Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information Systems*, 245-284.

Vernon-Parry, K. D. (2000). Scanning electron microscopy: an introduction. *III-Vs Review , 13*(4), 40-44.

Wang, F., Han, H., Shan, S., & Chen, X. (2017). Deep multi-task learning for joint prediction of heterogeneous face attributes. *IEEE International Conference on Automatic Face & Gesture Recognition*, 173-179.

Wang, H., Dalton, L., Fan, M., Guo, R., McClure, J., Crandall, D., & Chen, C. (2022). Deep-learning-based workflow for boundary and small target segmentation in digital rock images using UNet++ and IK-EBM. *Journal of Petroleum Science and Engineering, 215*, 110596.

Wang, W., Pang, X., Chen, Z., Chen, D., Ma, X., Zhu, W., Zheng, T., Wu, K., Zhang, K. & Ma, K. (2020). Improved methods for determining effective sandstone reservoirs and evaluating hydrocarbon enrichment in petroliferous basins. *Applied Energy, 261*, 114457.

Wang, X., Wang, M., Li, J., Shao, H., Deng, Z., & Wu, Y. (2022). Thermal maturity: The controlling factor of wettability, pore structure, and oil content in the lacustrine Qingshankou shale, Songliao Basin-. *Journal of Petroleum Science and Engineering*, 110618.

Wang, Z., & Ye, J. (2015). Querying discriminative and representative samples for batch mode active learning. *ACM Transactions on Knowledge Discovery from Data (TKDD), 9*(3), 1-23.

Wang, Z., Li, M., Wang, H., Jiang, H., Yao, Y., Zhang, H., & Xin, J. (2019). Breast cancer detection using extreme learning machine based on feature fusion with CNN deep features. *IEEE Access,, 7*, 105146-105158.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, 1-40.

Weng, L. (2018, 8 12). *From Autoencoder to Beta-VAE.* Retrieved 10 30, 2022, from https://lilianweng.github.io/posts/2018-08-12-vae/.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 241-259.

Worsham, J., & Kalita, J. (2020). Multi-task learning for natural language processing in the 2020s: where are we going? *Pattern Recognition Letters , 136* , 120-126.

Wu, X., Xiao, L., Sun, Y., Zhang, J., Ma, T., & He, L. (2022). A survey of human-in-the-loop for machine learning. *Future Generation Computer Systems, 135,* 364-381

Xia, X., & Kulis, B. (2017). W-net: A deep model for fully unsupervised image segmentation. *arXiv preprint* arXiv:1711.08506.

Yang, L., Zhang, Y., Chen, J., Zhang, S., & Chen, D. Z. (2017). Suggestive annotation: A deep active learning framework for biomedical image segmentation. *In International Conference on Medical Image Computing and Computer-Assisted Interventio*, 399-407.

Yang, R., He, S., Yi, J., & Hu, Q. (2016). Nano-scale pore structure and fractal dimension of organic-rich Wufeng-Longmaxi shale from Jiaoshiba area, Sichuan Basin: Investigations using FE-SEM, gas adsorption and helium pycnometry. *Marine and Petroleum Geology*, 27-45.

Yang, X., Song, Z., King, I., & Xu, Z. (2021). A survey on deep semi-supervised learning. *arXiv preprint arXiv:2103.00550.*

Yang, Y., Ma, Z., Nie, F., Chang, X., & Hauptmann, A. G. (2015). Multi-class active learning by uncertainty sampling with diversity maximization. *International Journal of Computer*

Vision, 113(2), 113-127.

Yoo, D., & Kweon, I. S. (2019). Learning loss for active learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 93-102.

Zhang, W., Tang, P., & Zhao, L. (2019). Remote sensing image scene classification using CNN-CapsNet. *Remote Sensing*, 494.

Zhang, Y., & Yang, Q. (2021). A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering,* 34(12), 5586-5609.

Zhang, Y., Li, M., Han, S., Ren, Q., & Shi, J. (2019). Intelligent identification for rock-mineral microscopic images using ensemble machine learning algorithms. *Sensors*, 19(18), 3914.

Zhang, Z., & Sabuncu, M. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in Neural Information Processing Systems*, 31.

Zhang, Z., Liu, Q., & Wang, Y. (2018). Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters, 15*(5), 749-753.

Zhao, W., Zhang, H., Yan, Y., Fu, Y., & Wang, H. (2018). A semantic segmentation algorithm using FCN with combination of BSLIC. *Applied Sciences, 8*(4), 500.

Zhou, Y., He, X., Huang, L., Liu, L., Zhu, F., Cui, S., & Shao, L. (2019). Collaborative learning of semi-supervised segmentation and classification for medical images. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2079-2088.

Zhou, Z., Siddiquee, M. M., Tajbakhsh, N., & Liang, J. (2018). Unet++: A nested u-net architecture for medical image segmentation. *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, 3-11.

Zhu, J., Wang, H., Tsou, B. K., & Ma, M. (2009). Active learning with sampling by uncertainty and density for data annotations. *IEEE Transactions on Audio, Speech, and Language Processing, 18*(6), 1323-1331.

Zhu, X., & Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 1-130.

Zhu, Y., Zhang, C., Zhou, D., Wang, X., Bai, X., & Liu, W. (2016). Traffic sign detection and recognition using fully convolutional network guided proposals. *Neurocomputing*, (pp. 758-766).