National Technical University of Athens
School of Electrical and Computer Engineering
Artificial Intelligence and Learning Systems Laboratory

Robustness and Domain Generalization in Computer Vision by Using Adversarial Data Augmentation

Methods, Applications, and Evaluations

Diploma Thesis

of

PANAGIOTIS KININIS

Supervisor: Athanasios Voulodimos
Assistant Professor

Athens, 9th of July 2024

National Technical University of Athens
School of Electrical and Computer Engineering
Artificial Intelligence and Learning Systems Laboratory

Robustness and Domain Generalization in Computer Vision by Using Adversarial Data Augmentation

Methods, Applications, and Evaluations

Diploma Thesis

of

PANAGIOTIS KININIS

Supervisor: Athanasios Voulodimos
Assistant Professor

Approved by the examination committee on Athens, July 2024.

(Signature)	*(Signature)*	*(Signature)*
.....	.....	.....
Athanasios Voulodimos	Giorgos Stamou	A. - G. Stafilopatis
Assistant Professor	Professor	Professor

Athens, 9^{th} of July 2024

National Technical University of Athens
School of Electrical and Computer Engineering
Artificial Intelligence and Learning Systems Laboratory

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism.

(Signature)

.....
Panagiotis Kininis

Athens, July 2024

Abstract

Deep learning models have significantly advanced the field of computer vision, yet they often struggle with generalization and robustness, particularly in applications requiring diverse and limited datasets, such as medical imaging. This thesis addresses this critical issue by exploring advanced data augmentation techniques to enhance the domain generalization and adversarial robustness of deep learning models. The primary objective is to develop and validate methods that can improve model performance in out-of-domain scenarios without extensive computational resources. The research utilizes adversarial data augmentation techniques, such as MaxStyle, MixStyle, and DSU, applied to the SYNTHIA dataset—a synthetic urban scene dataset depicting various environmental conditions.

The SYNTHIA dataset, known for its diversity in scenes, dynamic objects, seasons, and weather conditions, provides a robust testbed for evaluating these techniques. Preprocessing steps including noise reduction, rescaling, resizing, and geometric and photometric transformations ensure the quality and usability of the dataset for training. This thesis demonstrates that feature-level augmentations can significantly improve model robustness. Techniques like MixStyle, which mixes instance-level feature statistics, and MaxStyle, which augments feature maps with style mixing and adversarial perturbations, show marked improvements in generalization across diverse domains.

Experimental results reveal that models trained with these advanced augmentation techniques outperform standard training methods, particularly in challenging environmental conditions. Evaluations using the mean Intersection over Union (mIoU) metric show significant performance gains across various object classes and conditions, underscoring the efficacy of these techniques. This research highlights the potential of adversarial and feature-level data augmentation in overcoming the limitations of current deep learning models in computer vision, paving the way for more robust and generalizable applications in fields such as medical imaging.

Keywords

Adversarial data augmentation, domain generalization, robustness, deep learning, synthetic datasets, MaxStyle, DSU, MixStyle, Random Convolution, computer vision, semantic segmentation, feature-level augmentation, adversarial training, SYNTHIA dataset, image preprocessing, noise reduction, geometric transformations, photometric transformations, domain adaptation, style transfer, feature statistics, probabilistic modeling.

Περίληψη

Τα μοντέλα βαθιάς μάθησης έχουν προωθήσει σημαντικά τον τομέα της υπολογιστικής όρασης, αλλά συχνά δυσκολεύονται με τη γενίκευση και την ανθεκτικότητα, ιδιαίτερα σε εφαρμογές που απαιτούν ποικίλα και περιορισμένα σύνολα δεδομένων, όπως η ιατρική απεικόνιση. Ο κύριος στόχος της παρούσας διατριβής είναι η ανάπτυξη και η επικύρωση μεθόδων που μπορούν να βελτιώσουν την απόδοση των μοντέλων σε σενάρια εκτός τομέα χωρίς εκτεταμένους υπολογιστικούς πόρους. Η έρευνα χρησιμοποιεί τεχνικές εμπλουτισμού δεδομένων έναντι επιθέσεων, όπως το MaxStyle, το MixStyle και το DSU, εφαρμοσμένες στο σύνολο δεδομένων SYNTHIA - ένα συνθετικό σύνολο δεδομένων αστικών σκηνών που απεικονίζει διάφορες περιβαλλοντικές συνθήκες.

Το σύνολο δεδομένων SYNTHIA, γνωστό για την ποικιλία στις σκηνές, τα δυναμικά αντικείμενα, τις εποχές και τις καιρικές συνθήκες, παρέχει μια ανθεκτική βάση δοκιμών για την αξιολόγηση αυτών των τεχνικών. Αυτή η διατριβή δείχνει ότι οι εμπλουτισμοί στο επίπεδο των χαρακτηριστικών μπορούν να βελτιώσουν σημαντικά την ανθεκτικότητα των μοντέλων. Τεχνικές όπως το MixStyle, που αναμειγνύει στατιστικά χαρακτηριστικών σε επίπεδο δείγματος, και το MaxStyle, που εμπλουτίζει χάρτες χαρακτηριστικών με μίξη στυλ και επιθετικές διαταραχές, δείχνουν σημαντικές βελτιώσεις στη γενίκευση σε διάφορους τομείς.

Τα πειραματικά αποτελέσματα αποκαλύπτουν ότι τα μοντέλα που εκπαιδεύονται με αυτές τις προηγμένες τεχνικές εμπλουτισμού υπερέχουν των τυπικών μεθόδων εκπαίδευσης, ιδιαίτερα σε δύσκολες περιβαλλοντικές συνθήκες. Οι αξιολογήσεις χρησιμοποιώντας το μέσο Intersection over Union (mIoU) δείχνουν σημαντικές βελτιώσεις απόδοσης σε διάφορες κατηγορίες αντικειμένων και συνθήκες, υπογραμμίζοντας την αποτελεσματικότητα αυτών των τεχνικών. Αυτή η έρευνα αναδεικνύει το δυναμικό των τεχνικών εμπλουτισμού δεδομένων έναντι επιθέσεων και σε επίπεδο χαρακτηριστικών για την υπέρβαση των περιορισμών των τρεχόντων μοντέλων βαθιάς μάθησης στην υπολογιστική όραση, ανοίγοντας τον δρόμο για πιο ανθεκτικές και γενικεύσιμες εφαρμογές σε τομείς όπως η ιατρική απεικόνιση.

Λέξεις Κλειδιά

Ανθεκτικός εμπλουτισμός δεδομένων, γενίκευση τομέα, ανθεκτικότητα, βαθιά μάθηση, συνθετικά σύνολα δεδομένων, MaxStyle, DSU, MixStyle, Random Convolution, υπολογιστική όραση, σημασιολογική τμηματοποίηση, εμπλουτισμός σε επίπεδο χαρακτηριστικών, επιθετική εκπαίδευση, σύνολο δεδομένων SYNTHIA, προεπεξεργασία εικόνας, μείωση θορύβου, γεωμετρικοί μετασχηματισμοί, φωτομετρικοί μετασχηματισμοί, προσαρμογή τομέα, μεταφορά στυλ, στατιστικά χαρακτηριστικών, πιθανή μοντελοποίηση.

to my parents

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Athanasios Voulodimos, for his unwavering support, guidance, and encouragement throughout this journey. His expertise and insights were invaluable in shaping this work.

I am also profoundly grateful to Dr. Nikolaos Spanos, whose assistance and advice were crucial at many stages of this process. His willingness to help and share his knowledge was greatly appreciated.

My heartfelt thanks go to my close friends Nikos, Anna, Kostas, Aristodimos, and Dimitris. Their constant support and companionship made the challenging times more bearable, and their encouragement kept me motivated.

Lastly, I want to extend my deepest appreciation to my partner, Dimitra, for standing by me through every difficult moment. Her love, patience, and unwavering belief in me were my pillars of strength during this journey.

Athens, July 2024

Panagiotis Kininis

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the ever-evolving landscape of technology, artificial intelligence (AI) continues to stand out as a transformative force, particularly within the field of computer vision. This transformation is primarily driven by the rapid advancements in machine learning algorithms, enhanced computational capabilities, and the increasing availability of diverse datasets. Deep learning, a pivotal subset of AI technologies, has notably redefined the capabilities of machines in interpreting and understanding visual data, which has become integral to numerous AI applications.

Deep learning models, known for their depth and complexity, are now fundamental tools in AI applications, pushing the boundaries of traditional image processing techniques. These models have not only excelled in basic tasks but have also begun to challenge human capabilities in various aspects of visual recognition and interpretation.

Despite the profound capabilities of deep learning in computer vision, these models frequently encounter significant challenges, particularly in terms of generalization and performance under varied real-world conditions. Most models demonstrate optimal results under controlled experimental settings or assume specific training conditions that do not fully encapsulate the complexities encountered in practical deployments. For instance, in autonomous driving, a model trained under ideal weather conditions might underperform or fail when exposed to fog, rain, or snow.

Another significant challenge is the availability and diversity of training data. In fields such as healthcare, acquiring large-scale and diverse datasets is not only costly but often fraught with privacy concerns and regulatory restrictions. This scarcity of data limits the ability of models to learn and generalize across different domains or conditions they have not been explicitly trained on. So, the pursuit of robust, generalizable models capable of consistent performance across unforeseen conditions is crucial. Conventionally, enhancing generalization involves amassing large and varied datasets, a method that is often impractical for many sectors due to its high costs and logistical challenges.

Data augmentation is a critical technique used to address these challenges in model training. By artificially expanding the training dataset using transformations or enhancements (such as rotations, cropping, and color adjustments), models can learn more generalized features of the data. This method helps in simulating a variety of scenarios that a model might face after deployment, without the need for additional data collection.

However, traditional data augmentation often employs relatively simple transforma-

tions, which might not suffice for the complex requirements of real-world applications. Therefore, there is a growing interest in exploring more sophisticated data augmentation techniques that can introduce a wider array of variations and complexities into the training process.

As an alternative, innovative techniques such as style transfer and adversarial learning have emerged. These methods aim to improve the generalization capabilities of models by training them with a wider range of data scenarios, thereby enabling them to adapt to new environments more effectively. However, the computational demand and the extensive time required for these methods pose significant challenges, particularly for projects with limited resources.

Adversarial data augmentation offers a promising solution to these challenges by modifying training data to enhance the robustness and generalization of models. This technique involves the strategic introduction of perturbations or transformations to the training images, simulating potential real-world variations that the models may encounter. Such an approach can significantly bolster the models' ability to perform reliably across diverse conditions without the need for additional data collection.

The potential of adversarial data augmentation lies in its ability to enhance model generalization significantly. It prepares the model to handle unexpected changes in input data, thereby ensuring consistent performance across diverse environmental settings. This is particularly crucial in applications like autonomous driving, where encountering varying weather conditions and lighting scenarios is inevitable.

1.1 Thesis Scope & Structure

This thesis investigates the application of adversarial data augmentation within the realm of computer vision, focusing on enhancing model robustness and domain generalization. The primary objective of this research is to explore and validate the effectiveness of advanced data augmentation techniques, particularly through rigorous experimental trials using the Synthia dataset. This dataset comprises synthesized images depicting urban landscapes under various seasonal conditions, which serves as an ideal testbed for assessing the robustness of computer vision models across diverse environmental scenarios.

The scope of this thesis encompasses the design, implementation, and evaluation of a comprehensive system that leverages adversarial data augmentation to improve the generalization capabilities of deep learning models. By training models on one domain and evaluating them on others, this study aims to demonstrate significant advancements in model performance, thereby reducing the reliance on extensive computational resources and simplifying the training process. This approach is intended to make sophisticated computer vision models more feasible and effective across a variety of applications.

The structure of the thesis is organized as follows to provide a clear and systematic understanding of the research conducted:

- **Chapter 1: Introduction**
  This opening chapter sets the stage by outlining the research problem, stating the objectives, and highlighting the significance of adversarial data augmentation in contemporary computer vision research.

- **Chapter 2: Theoretical Background**
  Detailed insights into the fundamental concepts of computer vision are provided, with a focus on data augmentation, especially adversarial data augmentation. This chapter also covers essential topics such as robustness, domain generalization, and deep learning architectures including CNNs, FCNs, and encoder-decoder frameworks.

- **Chapter 3: Related Work**
  A comprehensive review of existing literature pertinent to adversarial data augmentation and its application in enhancing model robustness and domain generalization is discussed.

- **Chapter 4: Data and Preprocessing**
  This chapter describes the Synthia dataset in detail, including its characteristics and the preprocessing methods employed to prepare the data for experimental analysis.

- **Chapter 5: Implementation**
  Presentation and discussion of the main methods that were used in order to train our models in the context of this thesis.

- **Chapter 6: Experimental Results**
  Presentation of the experimental results obtained from the application of adversarial data augmentation techniques on the Synthia dataset. This chapter evaluates the impact of these techniques on model performance across different domains.

- **Chapter 7: Discussion and Conclusion**
  The concluding chapter summarizes the research findings and their implications for the field of computer vision. It also provides recommendations for future research directions and discusses potential extensions to this work.

So, the focus of this thesis is to explore the efficacy of adversarial data augmentation in the domain of computer vision, particularly through experimental trials on the Synthia dataset—a collection of synthesized images depicting urban landscapes in various conditions such as spring, winter, and snowy environments. By training a model on one domain and evaluating its performance on others, this research aims to demonstrate the enhanced robustness and domain generalization capabilities of models trained using adversarial data augmentation techniques.

Ultimately, this thesis aims to provide insights into how complex data augmentations, when combined with advanced generalization techniques, can lead to substantial improvements in model performance. This exploration seeks to reduce the dependency on extensive computational resources and streamline the training process, making advanced computer vision models more accessible and effective across various applications.

1.2 Related Works

The exploration of adversarial data augmentation in computer vision has attracted significant attention as a promising avenue for enhancing model robustness and domain generalization. This section reviews several interesting studies that have contributed to the understanding and development of adversarial and other sophisticated data augmentation techniques aimed at improving the performance of deep learning systems across varied domains.

Studies such as those by Xiao et al. [1] explore the synergy between consistency training and both random and adversarial data augmentation. Their research demonstrates that this combined approach significantly boosts robustness and accuracy across various benchmarks, providing strong empirical evidence supporting the effectiveness of integrating adversarial techniques with random transformations to create a more versatile and resilient training environment for deep neural networks.

Gokhale et al. (2022) undertook a rigorous analysis to ascertain the impact of various data modification strategies, such as the incorporation of additional training datasets and data augmentation, on the adversarial robustness and out-of-domain generalization of models. Their comprehensive study revealed that while the inclusion of additional data typically enhances both adversarial robustness and out-of-domain accuracy, implementing data filtering techniques could potentially compromise performance under specific conditions ([2]).

Li et al. [3] proposed a simple yet effective technique for domain generalization that involves perturbing feature embeddings with Gaussian noise. Their empirical results across multiple benchmarks reveal that this method can significantly enhance domain generalization capabilities of models, noted for its simplicity and the promising implications it holds for improving generalization in scenarios where complex data transformations are impractical.

Advancing the field further, Zhong et al. [4] introduced an adversarial style augmentation technique that strategically generates challenging stylized images during training. Their approach aims to improve model performance on unseen domains by dynamically adjusting to the hardest scenarios that a model might encounter. This method has been shown to be highly effective in domain generalization tasks, underscoring the potential of adversarial style techniques to significantly enhance model adaptability and performance across diverse operational settings.

Tripathi et al. (2023) [5] introduced a novel adversarial augmentation technique that focuses on encouraging deep vision models to learn from holistic shapes rather than just textures. Their findings indicate significant improvements in model accuracy and robustness across different adversarial and out-of-distribution datasets, suggesting a shift towards more shape-oriented learning strategies in computer vision ([5]).

Chen et al. (2021) presented Adversarial Feature Augmentation and Normalization (A-FAN), a method that enhances visual recognition models' generalization by augmenting adversarial features at the intermediate feature embeddings level. Their study showed consistent improvements across multiple tasks and datasets, reinforcing the value of adversarial techniques in feature manipulation ([6]).

Another study by Chen et al. (2023) proposed a center-aware adversarial augmentation technique for domain generalization. This method extends the source data distribution by modifying samples to deviate from class centers, demonstrating superior performance on benchmark datasets and highlighting the effectiveness of center-aware strategies in domain generalization ([7]).

Zhou et al. (2020) explored a deep domain-adversarial image generation approach to enhance model generalization to unseen domains by augmenting the source training data with generated data designed to fool both domain and label classifiers. This approach has shown potential in mitigating the domain shift problem, thereby enhancing the generalization capabilities of models ([8]).

These studies collectively highlight the dynamic nature of research in adversarial data augmentation and its crucial role in enhancing the robustness and generalization of computer vision systems. While each approach offers unique insights and methodologies, they all contribute to a broader understanding of how adversarial and augmentation techniques can be effectively utilized to train more robust and generalizable models. This body of work not only expands the theoretical framework for adversarial data augmentation but also provides practical guidance for implementing these advanced methods in real-world applications.

Part I

Theoretical Part

Chapter 2

Theoretical Background

This chapter presents the theoretical background necessary to understand the content of the thesis and the methods used in the experimental part.

2.1 Image Segmentation

Image segmentation is a critical task in computer vision, involving the partitioning of an image into multiple segments or regions to simplify and/or change the representation of an image into something more meaningful and easier to analyze. The goal is to identify and delineate objects or other relevant information in digital images. Segmentation is crucial for various applications such as object recognition, medical imaging, and autonomous driving.

Segmentation allows for the extraction of objects and boundaries in images, making it easier to analyze and interpret visual data. This task is essential in many domains, such as medical imaging, where precise localization and classification of anatomical structures are required, and in autonomous driving, where identifying pedestrians, vehicles, and obstacles is critical for navigation.

Historically, image segmentation was accomplished using methods such as thresholding, edge detection, and region-based techniques. Thresholding involves converting a grayscale image into a binary image by selecting a threshold value. Pixels with intensity values greater than the threshold are classified as foreground, while others are classified as background. Otsu's method [9] is a well-known technique for automatically selecting the threshold value. Edge detection techniques, such as the Canny edge detector [10], aim to identify the boundaries of objects within an image by detecting discontinuities in intensity. Region-based segmentation methods, such as region growing and region splitting/merging, involve partitioning an image into regions that are similar according to predefined criteria.

With the advent of machine learning and deep learning, more advanced techniques for image segmentation have been developed. These techniques often outperform traditional methods, particularly in complex and large-scale datasets. Supervised learning techniques for image segmentation involve training a model on a labeled dataset, where the ground truth segmentation maps are provided. Convolutional Neural Networks (CNNs) have been widely used for image segmentation tasks due to their ability to learn hier-

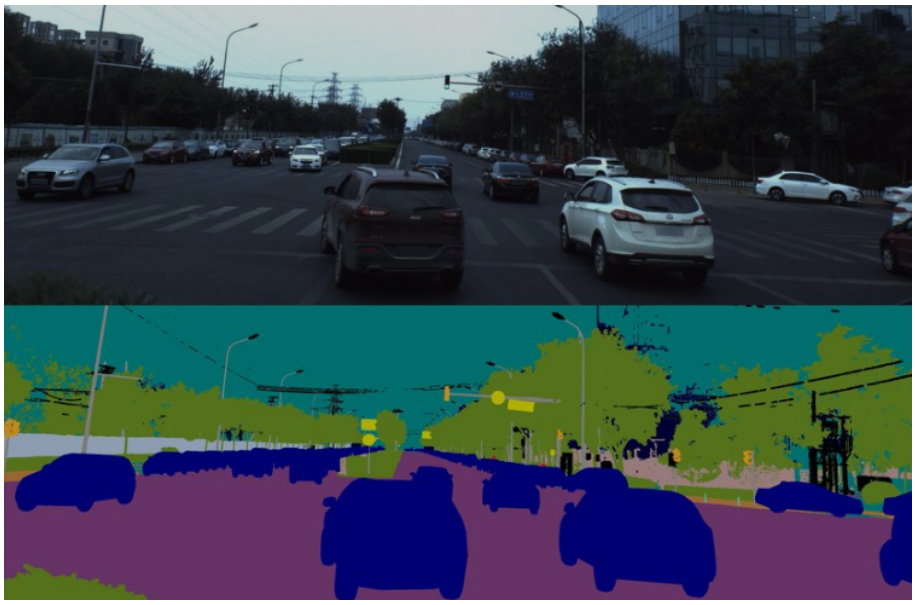


Figure 2.1. *Image segmentation example in an autonomous driving scenario. The top part shows the original image, while the bottom part illustrates the segmented image where each pixel is classified into predefined categories such as road, vehicles, buildings, and vegetation. This segmentation helps autonomous vehicles understand and navigate their environment effectively.*

archical features from images. Fully Convolutional Networks (FCNs) [11] are a type of CNN designed specifically for dense prediction tasks like segmentation. FCNs replace the fully connected layers in traditional CNNs with convolutional layers to produce spatially consistent output. U-Net [12] is a popular architecture for image segmentation, especially in medical imaging. It consists of a contracting path that captures context and a symmetric expanding path that enables precise localization. DeepLab [13] is another significant architecture that utilizes atrous (dilated) convolutions to capture multi-scale context by controlling the resolution at which feature responses are computed.

Unsupervised learning techniques do not require labeled data and rely on the inherent structure within the data. Clustering algorithms, such as K-means and Gaussian Mixture Models (GMMs), can be used for image segmentation by grouping pixels with similar features into clusters. Each cluster corresponds to a segment in the image. Autoencoders and generative models, such as Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs), can also be used for unsupervised image segmentation. These models learn a low-dimensional representation of the input data, which can be used to reconstruct the image and identify different segments.

2.1.1 Applications of Image Segmentation

Image segmentation has a wide range of applications across various fields. In medical imaging, segmentation is used to identify and delineate anatomical structures and pathological regions, crucial for diagnosis, treatment planning, and monitoring of diseases. For example, segmentation is used in MRI and CT scans to locate tumors, measure organ volumes, and guide surgical procedures [14]. In autonomous driving, vehicles rely

heavily on image segmentation to understand their environment, identifying roads, lanes, pedestrians, vehicles, and other obstacles. This information is critical for navigation, path planning, and collision avoidance [15]. Segmentation of satellite and aerial imagery is used for various applications, including land cover classification, urban planning, and disaster management. It helps in identifying different terrain types, buildings, vegetation, and water bodies [16]. Image segmentation is often a preprocessing step in object detection and recognition tasks. By segmenting the image into meaningful regions, it becomes easier to detect and recognize objects within those regions [17].

2.1.2 Types of Image Segmentation

Image segmentation is a fundamental task in computer vision that involves partitioning an image into meaningful segments. The primary types of image segmentation are semantic segmentation, instance segmentation, and panoptic segmentation. Each type serves a unique purpose and has distinct applications.

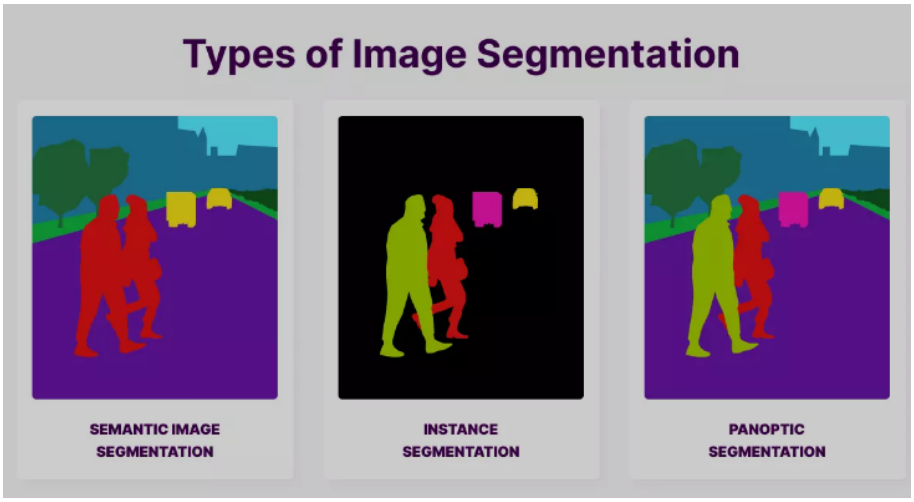


Figure 2.2. *Types of Image Segmentation: The image illustrates three types of image segmentation. Semantic segmentation classifies each pixel into a category (e.g., person, car, tree), instance segmentation differentiates between distinct objects of the same category, and panoptic segmentation combines both, providing a complete scene understanding.*

Semantic Segmentation

Semantic segmentation assigns a class label to each pixel in the image. This type of segmentation focuses on identifying and classifying all pixels belonging to a particular object or region, without distinguishing between different instances of the same class. For example, in a street scene, all pixels corresponding to cars might be labeled as "car," and all pixels corresponding to roads might be labeled as "road."

Example: Consider a self-driving car navigating a busy street. Semantic segmentation helps the car understand which parts of the image correspond to roads, sidewalks, vehicles, and pedestrians. This information is crucial for tasks such as path planning and obstacle avoidance.

Instance Segmentation

Instance segmentation not only classifies each pixel but also distinguishes between different instances of the same class. This means that each object instance in the image is segmented separately. For instance, in an image with multiple people, instance segmentation will label each person individually, even though they all belong to the "person" class.

Example: In an image containing a group of people, instance segmentation can identify and segment each person separately. This is particularly useful in applications like surveillance and crowd monitoring, where it is important to track individual movements and actions.

Panoptic Segmentation

Panoptic segmentation combines the strengths of both semantic and instance segmentation. It provides a unified framework where each pixel is assigned a class label and distinct instances are identified. This comprehensive approach allows for a complete understanding of the scene, capturing both the semantic context and the individual instances.

Example: In autonomous driving, panoptic segmentation can provide detailed scene understanding by labeling each pixel with its corresponding object class and differentiating between individual instances of objects such as cars and pedestrians. This holistic view is essential for making informed decisions in complex environments.

2.1.3 Applications and Importance

Each type of segmentation has its applications and importance in various domains. Semantic segmentation is widely used in medical imaging for tasks such as organ and tumor segmentation. Instance segmentation is crucial for object detection and tracking in video surveillance. Panoptic segmentation finds applications in autonomous driving, robotics, and any scenario requiring comprehensive scene understanding.

By combining these segmentation techniques, we can achieve a deeper understanding of visual scenes, enabling advancements in fields like computer vision, robotics, and artificial intelligence.

2.1.4 Challenges in Image Segmentation

Despite the advancements in image segmentation techniques, several challenges remain. Achieving high accuracy and precision in segmentation tasks is challenging, especially where boundaries between objects are unclear or objects have similar appearances. Fine-tuning models to handle such cases requires extensive labeled data and computational resources. Deep learning models for segmentation, such as FCNs and U-Nets, are computationally intensive and require significant processing power, particularly for large images and 3D volumes. Optimizing these models to run efficiently on limited hardware resources is an ongoing challenge. Ensuring that segmentation models generalize well

to new, unseen data is crucial. Models trained on specific datasets may not perform well on data from different domains or acquired under different conditions. Developing robust models that generalize across diverse datasets remains an important research area. Creating annotated datasets for training segmentation models is labor-intensive and time-consuming. The quality of the segmentation model heavily depends on the quality of the ground truth annotations. Developing semi-supervised or unsupervised methods to reduce the dependency on labeled data is an active area of research.

2.1.5 Future Directions

The future of image segmentation lies in addressing the current challenges and exploring new frontiers. Combining data from multiple modalities, such as images, text, and sensor data, can enhance segmentation accuracy and robustness. For example, integrating MRI and CT scans can provide complementary information for better segmentation in medical imaging. Developing models that can perform real-time segmentation is crucial for applications like autonomous driving and video surveillance. This requires optimizing algorithms for speed and efficiency without compromising accuracy. As deep learning models become more complex, understanding their decision-making process becomes challenging. Developing methods to interpret and explain the segmentation results can build trust and facilitate the adoption of these models in critical applications. Transfer learning and few-shot learning techniques aim to leverage knowledge from pre-trained models and adapt them to new tasks with limited data. These techniques can significantly reduce the need for extensive labeled datasets and accelerate the deployment of segmentation models in new domains.

2.2 Out-of-Distribution (OOD) Problem

The Out-of-Distribution (OOD) problem is a fundamental challenge in machine learning and computer vision. It occurs when a model is exposed to data that significantly deviates from the data it encountered during training, leading to substantial performance degradation. This is particularly critical in applications where reliability and accuracy are paramount, such as autonomous driving, healthcare diagnostics, and security systems.

2.2.1 Understanding OOD

Deep learning models typically assume that the training data and the data to which they will be applied are from the same distribution, meaning the mean and standard deviation of the features of these images are the same. However, in reality, many external factors, from the way the image was taken to the environment, cause variations in the image field, known as domain shifts. This discrepancy can be problematic when models are deployed in real-world scenarios.

Let's assume that the inputs and outputs of a model are connected through the joint probability distribution function $P(X, Y)$, where X are the inputs and Y are the outputs. Usually, the training and validation inputs come from the same distribution, so there are

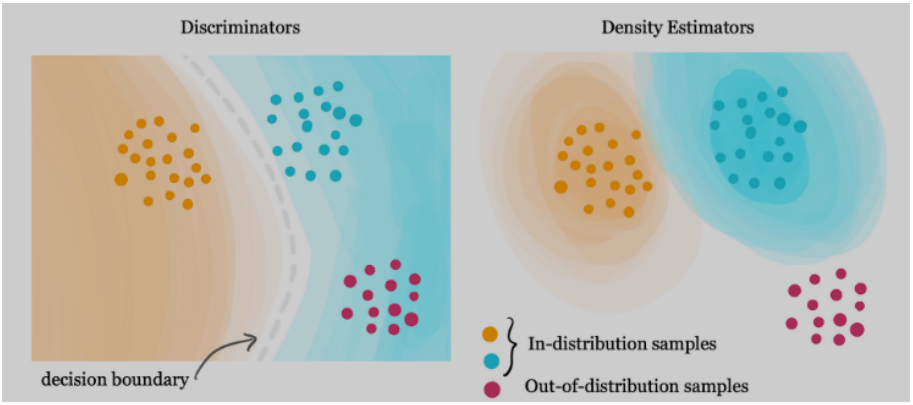


Figure 2.3. *Illustration of Out-of-Distribution (OOD) problem detection methods. The left side shows a discriminator-based approach with a decision boundary separating in-distribution and out-of-distribution samples. The right side shows a density estimator-based approach, where in-distribution samples form high-density regions, and OOD samples fall in low-density areas.*

no changes in the joint function during training. However, when the model is applied at a practical level, the data used for training, validation, and testing come from the conditional distribution $P(X, Y|Z \in U)$ where Z is some random variable that may be unobserved, which is not independent of Y and X, and U is an appropriate subset of Z [18].

2.2.2 Implications of the OOD Problem

The OOD problem has far-reaching implications. For instance, in the medical field, two MRI scans taken from different machines can show significant deviations, even if they refer to the same content. This is due to the nature of medical imaging machines, which add noise and artifacts to the images and differences in the magnetic field, which, even with proper preprocessing, can lead to significant differences at the level of training an artificial intelligence model. As a result, the performance of deep learning models drops dramatically in cases where there are large domain differences, preventing their practical and widespread application on a large scale [19].

2.2.3 Detection and Handling of OOD Data

Detecting OOD data is crucial for addressing this problem. Various approaches have been proposed to identify OOD instances. One common method involves using the softmax output of neural networks. Typically, models tend to output lower confidence scores for OOD data compared to in-distribution data. However, this method is not foolproof, as models can sometimes be overly confident even on OOD data.

Another approach is based on training an auxiliary model specifically designed to distinguish between in-distribution and OOD data. This method often employs techniques such as autoencoders or generative models, which learn the distribution of the training data and can flag deviations from this learned distribution as OOD [20].

Recent advancements have also explored the use of ensemble methods, where multiple models are trained independently, and their agreement on predictions is used as a measure of confidence. OOD data typically results in higher disagreement among the models, signaling potential OOD scenarios [21].

2.2.4 Addressing the OOD Problem

Handling OOD data effectively often requires incorporating robustness into the model training process. Adversarial training, where the model is exposed to adversarial examples that are intentionally crafted to be challenging, can improve the model's ability to generalize to unseen data [22]. Data augmentation techniques, such as random cropping, rotation, and color jittering, can also enhance the robustness of models by exposing them to a wider variety of data during training.

Another promising direction is domain adaptation, which involves fine-tuning a pre-trained model on a small amount of labeled data from the target domain. This approach helps the model adjust its parameters to better handle the specific characteristics of the new domain, thereby improving performance on OOD data [23].

An obvious solution to this problem is to use images from a variety of domains so that the model becomes robust to such deviations. However, this solution is often unfeasible because there is not enough volume and variety of data freely available for use. For this reason, different approaches have been adopted in the field of deep learning to solve the problem without the need for training on a massive dataset, known as domain generalization methods [24].

2.2.5 Future Directions

The OOD problem remains an active area of research with several promising avenues for future exploration. One area of interest is the development of more sophisticated uncertainty quantification methods that can provide reliable confidence estimates for predictions. Improved uncertainty estimates can help in making more informed decisions, especially in high-stakes applications [25].

Additionally, there is growing interest in zero-shot and few-shot learning techniques, which aim to enable models to generalize to new classes or domains with minimal or no additional training data. These techniques could potentially mitigate the OOD problem by leveraging prior knowledge and transferable features learned from related tasks [26].

Finally, interdisciplinary approaches that combine insights from fields such as robust statistics, information theory, and cognitive science may offer novel solutions to the OOD problem. Understanding how humans recognize and adapt to novel situations could inspire new algorithms and architectures capable of handling OOD data more effectively.

2.3 Domain Generalization

The problem of domain shifts and the absence of sufficient data are significant hurdles in deploying robust machine learning models in real-world scenarios. Domain general-

ization methods have emerged as a powerful solution to these challenges, especially in deep learning models. Domain generalization aims to train a model that can generalize to unknown domains by considering only labeled data from a set of initial domains during training. This is particularly crucial in applications where the target domain is unknown or inaccessible during training, such as medical diagnosis or autonomous driving.

Domain generalization focuses on learning a representation that is invariant to the domain, capturing the common underlying structure across different domains while being resilient to variations specific to individual domains. This approach is distinct from related learning problems, such as domain adaptation or transfer learning, which typically assume access to the target domain data during the training process.

The history of domain generalization methods dates back to the early 2000s when the problem was first introduced as a machine learning challenge by Blanchard et al. [24]. Unlike domain adaptation, where some information about the target domain is available during training, domain generalization addresses scenarios where the target data are completely inaccessible during the model's learning process. The initial motivation behind domain generalization came from a medical application known as automated gating of flow cytometry data. The goal was to design algorithms to automate the classification of cells in patient blood samples based on different properties, such as distinguishing between lymphocytes and non-lymphocytes. This technology is critical for facilitating patient health diagnosis, given that manual classification is extremely time-consuming and requires specialized expertise.

Over the years, numerous methods have been developed to tackle the domain generalization problem. These methods can be broadly categorized into four groups: feature-based methods, metric-based methods, model-based methods, and meta-learning-based methods.

2.3.1 Feature-Based Methods

Feature-based methods aim to learn a feature representation that is invariant to the domain. This can be achieved by adding regularization terms to the loss function or using domain separation networks. For instance, the use of adversarial learning to align the feature distributions of different domains has been a popular approach. By doing so, the model learns to extract features that are not specific to any single domain but are generalizable across multiple domains.

2.3.2 Metric-Based Methods

Metric-based methods focus on learning a metric space that is invariant to domain-specific variations. These methods often involve designing a metric learning algorithm that can measure the similarity between samples from different domains in a way that is robust to domain shifts. By mapping domains into a shared latent space, metric-based methods aim to ensure that the distances between samples in this space reflect their true semantic similarities, regardless of their domain of origin.

2.3.3 Model-Based Methods

Model-based methods aim to learn models that are inherently resilient to domain shifts. This can be achieved by designing architectures that are robust to variations in the input data or by incorporating domain-agnostic components into the model. For example, using batch normalization embeddings, as proposed by Segu et al. [27], can help create domain-dependent representations that improve classification accuracy across different domains.

2.3.4 Meta-Learning-Based Methods

Meta-learning-based methods, also known as learning-to-learn approaches, aim to learn a meta-learner that can quickly adapt to new domains with few labeled samples. This is particularly useful in scenarios where obtaining a large number of labeled samples from the target domain is impractical. By simulating domain shifts during the training process, meta-learning-based methods can train models that are better prepared to handle unseen domains during deployment.

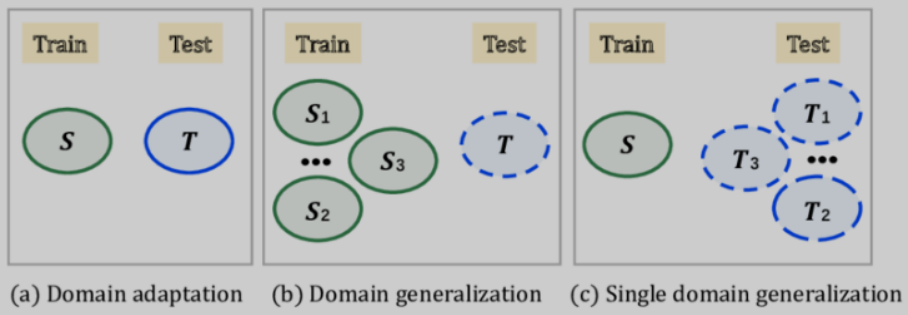


Figure 2.4. *Illustration of different scenarios in domain generalization: (a) Domain adaptation, where the model is trained on source domain S and tested on target domain T; (b) Domain generalization, where the model is trained on multiple source domains $S_1, S_2, \ldots, S_n$ and tested on an unseen target domain T; (c) Single domain generalization, where the model is trained on a single source domain S and tested on multiple target domains $T_1, T_2, \ldots, T_n$.*

Domain generalization has been extensively studied in the literature, with various methods developed across different application domains. For example, Sivaprasad et al. [28] evaluated the effectiveness of Empirical Risk Minimization (ERM) as a baseline for domain generalization, demonstrating its superiority over many existing DG methods. They proposed a classwise-DG formulation, which provides a more realistic benchmarking closer to human learning scenarios.

Another notable work by Wang et al. [29] provides a comprehensive survey on domain generalization, categorizing methods into data manipulation, representation learning, and learning strategy. This survey also discusses datasets and applications, summarizing the existing literature and suggesting future research directions.

Mesbah et al. [30] tackled the weak generalization problem when a model is trained on a single source domain by building an ensemble model on top of base deep learning models. Their approach enhances generalization by leveraging the collective predictions

of the ensemble, which has shown to be effective in improving robustness to domain shifts.

Dynamic Domain Generalization (DDG) is another promising approach proposed by Sun et al. [31]. DDG introduces a meta-adjuster that dynamically adjusts network parameters based on data from different domains. This approach aims to generalize to different target domains without additional training, using DomainMix to simulate diverse data during the training process.

Despite the advancements in domain generalization methods, several challenges remain. One of the primary challenges is the lack of a unified theoretical framework that can guide the development of new methods. Additionally, the evaluation of domain generalization methods often relies on synthetic benchmarks, which may not fully capture the complexities of real-world scenarios.

2.4 Style Transfer

Style transfer has emerged as a fascinating technique in the realm of computer vision and image processing, enabling the transformation of an image's artistic style while preserving its core content. This innovative concept gained significant attention with the advent of neural style transfer algorithms, which leverage the capabilities of deep neural networks to generate visually stunning and artistic images. At the heart of style transfer is the idea of separating and recombining the content and style of two distinct images to create a novel and unique composition.

The style transfer process typically involves two primary components: the content image and the style image. The content image contains the objects and elements that will be retained in the final result, whereas the style image incorporates the stylistic features to be applied to the content. Neural networks play a crucial role in this process by analyzing the content image to extract its features and the style image to capture its stylistic characteristics. These features are then synthesized to produce a new image that seamlessly combines the content of the content image with the stylistic properties of the style image.

One of the most compelling aspects of style transfer is its ability to enrich datasets and improve model generalization by creating images from different domains. By utilizing images that differ in their domains, style transfer can generate a diverse set of images, which can significantly enhance the training process of deep learning models. The primary quantities used in this process are the mean and variance of the feature maps, as these metrics encapsulate the essential information about an image's style within the layers of a neural network.

A notable advancement in the field is the introduction of deep reinforcement learning-based architectures for neural style transfer. For instance, Feng et al. (2023) proposed a method that splits the one-step style transfer into a step-wise process. This approach allows for better preservation of the content image's details in the early steps and synthesizes more style patterns in the later steps. Such a method is user-controllable, lightweight, and has lower computational complexity, making it an attractive option for

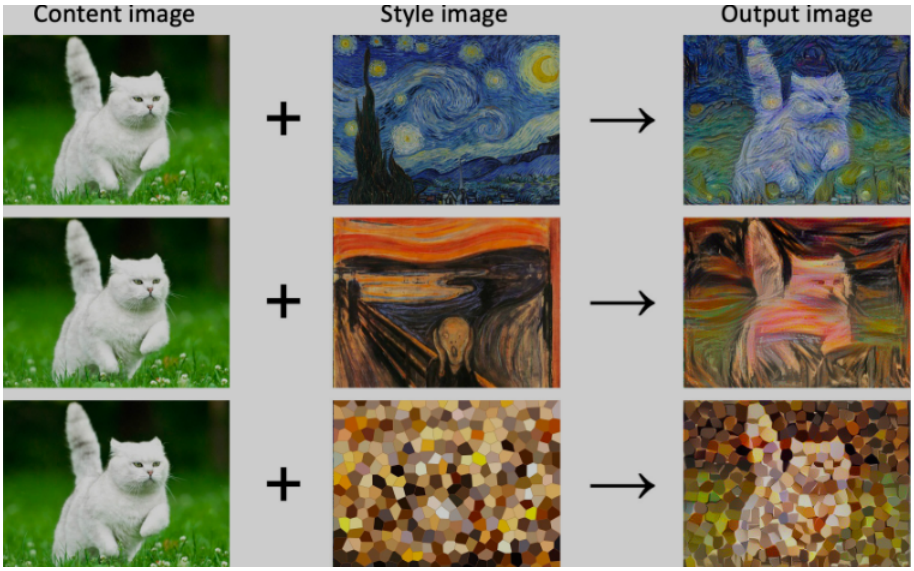


Figure 2.5. *Example of style transfer: The left column shows the content image, the middle column shows the style image, and the right column shows the output image that combines the content of the left image with the style of the middle image. Each row demonstrates a different style applied to the same content image.*

practical applications [32].

Another significant contribution to the field is the comprehensive overview provided by Li et al. (2018), which outlines the development process of deep learning networks for style transfer. The paper introduces classical style migration models and discusses the challenges and solutions encountered during their investigation. It also compares the results of different models in image style transfer, offering valuable insights into their effectiveness [33].

Niu et al. (2021) proposed a deep learning-based framework for style transfer in specific target regions of images. This method employs image mask technology to generate a mask map as a specific condition input, ensuring that only the target area undergoes style transfer while the non-target area remains unaffected. The effectiveness of this approach has been demonstrated through various experiments, showcasing its potential in practical applications [34].

Ren and Sheng (2022) provided an overview of the current research progress and results of image style transfer using deep learning methods. Their work categorizes the methods into Convolutional Neural Networks (CNN) and Generative Adversarial Networks (GAN), discussing various models and their effectiveness in style transfer tasks. This comprehensive review highlights the strengths and limitations of different approaches, guiding future research in the field [35].

2.5 CNN - Convolutional Neural Networks

It is prudent at this point to provide a detailed overview of Convolutional Neural Networks (CNNs), the cornerstone of deep learning models used in the field of image

processing. The CNN architecture is specifically designed to handle the complexity and size of modern datasets by learning patterns and features directly from the input images.

2.5.1 Basic Architecture

Convolutional Neural Networks were developed to address the inefficiencies of traditional feedforward neural networks in handling high-dimensional image data. They reduce the number of parameters and computational complexity by exploiting the spatial structure of images. A typical CNN architecture consists of several types of layers, each serving a unique purpose in the network's ability to learn and recognize patterns.

- **Input Layer**: The raw pixel values of the image are fed into the network. These images can be in various formats and resolutions, typically represented as multi-dimensional arrays.
- **Convolutional Layers**: These layers apply convolution operations using learnable filters to the input image. Filters move across the image spatially, producing feature maps that detect various attributes such as edges, textures, and colors. This local connectivity allows CNNs to capture spatial hierarchies in the data.
- **Activation Functions**: Non-linear activation functions like ReLU (Rectified Linear Unit) are applied to introduce non-linearity, enabling the network to learn complex patterns. Kuo (2016) elaborates on the necessity of these activation functions for maintaining the richness of learned features [36].
- **Pooling Layers**: These layers perform down-sampling operations, reducing the spatial dimensions of the feature maps while retaining the most significant features. Pooling helps in reducing the computational load and prevents overfitting.
- **Fully Connected Layers**: Towards the end of the network, fully connected layers perform high-level reasoning based on the features extracted by the convolutional and pooling layers. These layers connect every neuron in one layer to every neuron in the next, similar to traditional neural networks.

2.5.2 Convolution Operation

The convolution operation is central to CNNs. It involves sliding a filter over the input image to produce a feature map. Mathematically, this can be expressed as:

$$(I * K)(i,j) = \sum_m \sum_n I(m,n) \cdot K(i-m, j-n)$$

where I is the input image, K is the convolution kernel, and the output is a feature map that highlights specific patterns in the input image. This operation is repeated across the entire image, enabling the detection of features irrespective of their position.

The choice of filter size and stride plays a crucial role in the effectiveness of the convolution operation. Smaller filters (e.g., 3x3) are generally preferred as they capture

finer details, while larger filters can detect more complex patterns. Stride determines the step size with which the filter moves across the image, influencing the resolution of the feature map. The use of padding, where the input image is surrounded by zeros, ensures that the spatial dimensions are preserved after convolution.

Convolutional layers are responsible for detecting low-level features such as edges and textures in the initial layers, progressing to more complex features such as shapes and objects in deeper layers. This hierarchical feature extraction is one of the key advantages of CNNs, enabling them to perform well on a variety of visual tasks [37].

2.5.3 Activation Functions

Activation functions introduce non-linearity into the network, allowing it to learn complex patterns. The Rectified Linear Unit (ReLU) is the most commonly used activation function in CNNs due to its simplicity and effectiveness. ReLU is defined as:

$$f(x) = \max(0, x)$$

ReLU addresses the vanishing gradient problem by allowing gradients to flow through the network without shrinking, which helps in training deep networks. However, ReLU has its limitations, such as dying neurons, where neurons output zero for all inputs if they fall in the negative part of the input space.

To address these issues, variants of ReLU have been proposed:

- **Leaky ReLU**: Allows a small, non-zero gradient when the input is negative, defined as:

$$f(x) = \begin{cases} x & x \geq 0 \\ ax & x < 0 \end{cases}$$

  where a is a small constant.

- **Parametric ReLU (PReLU)**: Similar to Leaky ReLU, but a is a learnable parameter, defined as:

$$f(x) = \begin{cases} x & x \geq 0 \\ ax & x < 0 \end{cases}$$

  where a is a learnable parameter.

- **Exponential Linear Unit (ELU)**: Smooths the transition from negative to positive values, defined as:

$$f(x) = \begin{cases} x & x \geq 0 \\ a(e^x - 1) & x < 0 \end{cases}$$

  where a is a hyperparameter.

- **Sigmoid Function**: The sigmoid function maps any real-valued number into the range between 0 and 1, defined as:

$$f(x) = \frac{1}{1 + e^{-x}}$$

This function is especially useful for models where we need to predict probabilities.

- **Tanh Function**: The hyperbolic tangent (tanh) function maps any real-valued number to the range between -1 and 1, defined as:

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

This function is similar to the sigmoid function but shifts the outputs to zero-centered, often leading to faster convergence.

- **ReLU (Rectified Linear Unit)**: The ReLU function introduces non-linearity by outputting zero for any negative input and outputting the input directly if it is positive, defined as:

$$f(x) = \max(0, x)$$

This helps the network to learn complex patterns and is the most widely used activation function in deep learning models.

- **Leaky ReLU**: A variation of ReLU that allows a small, non-zero gradient when the unit is inactive, helping to solve the "dying ReLU" problem, defined as:

$$f(x) = \begin{cases} x & x \geq 0 \\ ax & x < 0 \end{cases}$$

where a is a small constant.

These activation functions help maintain the richness of the features and prevent the issues associated with traditional ReLU [36].

2.5.4 Pooling Layers

Pooling layers are essential for reducing the spatial dimensions of the feature maps, which helps in minimizing the computational load and number of parameters in the network. This process, known as down-sampling, retains the most important features while discarding less relevant information. The two most common types of pooling are:

- **Max Pooling**: Selects the maximum value within a defined window. This approach captures the most prominent features and is beneficial for detecting edges and textures.

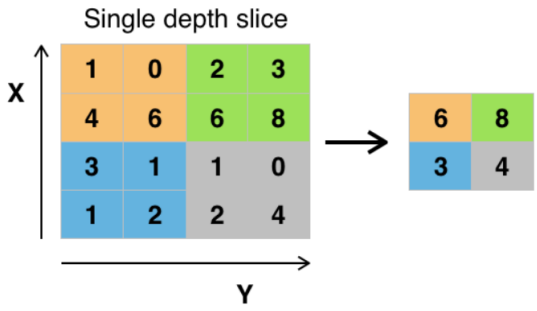


Figure 2.6. *Max Pooling: This method selects the maximum value within a defined window. It captures the most prominent features, such as edges and textures, and helps in reducing the dimensionality of the feature map while preserving important characteristics.*

- **Average Pooling**: Computes the average value within a defined window. This method is useful for smoothing the feature map and is less sensitive to the presence of noise.

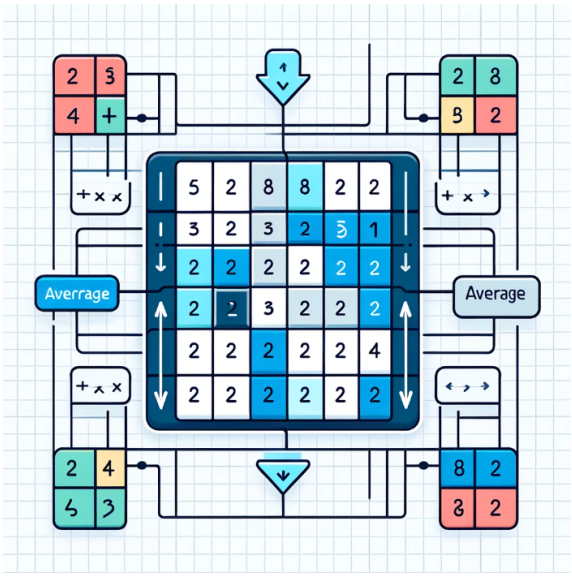


Figure 2.7. *Average Pooling: This method computes the average value within a defined window. It is useful for smoothing the feature map and reducing noise. Average pooling helps in maintaining the spatial structure of the input while reducing its size.*

Pooling layers contribute to the invariance of the network to small translations and distortions in the input image, which is crucial for recognizing objects regardless of their position [38].

2.5.5 Advancements in CNN Architectures

Over the past decade, CNN architectures have evolved significantly, driven by the need for better performance and efficiency. Gu et al. (2015) [37] provide a comprehensive survey of these advancements, highlighting key innovations such as:

Layer Design Modern CNN architectures have become deeper and more complex, allowing them to learn more intricate features from data. The introduction of residual networks (ResNets) by He et al. (2016) was a significant breakthrough in deep learning. ResNets use residual connections to mitigate the vanishing gradient problem, enabling the training of very deep networks by allowing gradients to flow more easily through the network. This architecture demonstrated that extremely deep networks could be trained effectively, leading to substantial performance improvements on various benchmarks [39]. Another notable architecture is the Inception network, introduced by Szegedy et al. (2015). The Inception network uses parallel convolutional layers of different sizes to capture features at multiple scales. This design allows the network to be both wide and deep, improving its ability to capture a diverse set of features without significantly increasing computational complexity [40].

Optimization Techniques Optimization techniques have played a crucial role in improving the training of CNNs. Batch normalization, introduced by Ioffe and Szegedy (2015), normalizes the inputs of each layer to have a mean of zero and a standard deviation of one. This technique stabilizes and accelerates the training process by reducing internal covariate shift [41]. Another important optimization technique is dropout, introduced by Srivastava et al. (2014). Dropout randomly sets a fraction of the activations to zero during training, which prevents overfitting and encourages the network to learn more robust features [42].

Regularization Methods Regularization methods help improve the generalization of CNNs by preventing overfitting. Data augmentation, which involves creating additional training samples through transformations such as rotation, scaling, and flipping, is a widely used regularization technique. By exposing the network to a more diverse set of training examples, data augmentation enhances the network's ability to generalize to new, unseen data [43]. Weight regularization techniques, such as L2 regularization, add a penalty to the loss function based on the magnitude of the weights. This penalty discourages the network from learning overly complex models that may not generalize well to new data [44].

Transfer Learning Transfer learning has become an essential strategy in training deep CNNs. Instead of training a network from scratch, pre-trained models on large datasets like ImageNet are fine-tuned on a specific task. This approach significantly reduces training time and improves performance, especially when the target dataset is small [45].

Efficient Architectures Recent advancements have also focused on creating more efficient CNN architectures. MobileNets, introduced by Howard et al. (2017), use depthwise separable convolutions to reduce the number of parameters and computational complexity, making them suitable for mobile and embedded applications [46]. Another efficient architecture is the EfficientNet, proposed by Tan and Le (2019). EfficientNet scales the network's width, depth, and resolution in a principled manner using a compound scaling

method. This approach achieves state-of-the-art performance with fewer parameters and lower computational cost compared to traditional architectures [47].

2.5.6 Applications in Image Segmentation

Image segmentation is one of the critical applications of CNNs, where the task is to assign a label to each pixel in an image, effectively partitioning the image into meaningful segments. CNNs have been adapted for image segmentation through specialized architectures such as Fully Convolutional Networks (FCNs) and encoder-decoder networks.

- **Fully Convolutional Networks (FCNs)**: FCNs replace the fully connected layers with convolutional layers that output spatial feature maps, making it possible to generate segmentation maps that align with the input image dimensions. This approach allows the network to retain spatial information throughout the entire process.
- **Encoder-Decoder Networks**: These networks consist of an encoder that progressively reduces the spatial dimensions of the input to capture high-level features, and a decoder that restores the spatial dimensions to produce pixel-wise classifications. A prominent example of this architecture is the UNet, widely used in medical image segmentation.

2.5.7 Image Segmentation Challenges and Solutions

Image segmentation, a critical application of Convolutional Neural Networks (CNNs), involves assigning a label to each pixel in an image, effectively partitioning the image into meaningful segments. Despite significant advancements in the field, image segmentation still faces several challenges. This section discusses these challenges and the solutions proposed in recent literature to address them.

Boundary Precision One of the primary challenges in image segmentation is accurately segmenting object boundaries. CNNs, due to their convolutional nature, tend to produce smooth transitions between object classes, leading to imprecise boundaries. This issue is particularly problematic in applications requiring high precision, such as medical imaging. Several techniques have been proposed to address this challenge. Conditional Random Fields (CRFs) have been integrated with CNNs to refine segmentation boundaries. For instance, Chen et al. (2016) proposed the DeepLab model, which incorporates fully connected CRFs to improve boundary localization by considering pixel-level dependencies [48].

Class Imbalance Class imbalance is another significant challenge in image segmentation. In many datasets, certain classes may be underrepresented, leading to biased predictions and poor performance on minority classes. This issue is prevalent in medical imaging, where abnormal tissues (e.g., tumors) are much less frequent than normal tissues. To tackle class imbalance, several strategies have been employed. One common

approach is to use weighted loss functions, where higher weights are assigned to minority classes to ensure they contribute more to the loss during training. Another method is data augmentation, which involves generating additional samples of minority classes through techniques such as rotation, flipping, and scaling. Ronneberger et al. (2015) employed these techniques in the UNet architecture to enhance segmentation performance on biomedical datasets [49].

Computational Complexity Segmentation tasks are computationally intensive, requiring significant resources for training and inference. This complexity arises from the need to process high-resolution images and generate dense pixel-wise predictions. The high computational cost can be a barrier to deploying segmentation models in real-time applications and on devices with limited processing power. To reduce computational complexity, efficient network architectures have been developed. For example, the MobileNetV2 model uses depthwise separable convolutions to reduce the number of parameters and computational load, making it suitable for mobile and embedded applications [50]. Additionally, the use of model compression techniques, such as quantization and pruning, can further reduce the computational requirements without significantly compromising performance.

Generalization to Unseen Data Generalizing segmentation models to unseen data is a persistent challenge, especially when the training and test datasets have different distributions. Domain adaptation techniques have been proposed to address this issue by reducing the domain shift between the source and target domains. One effective approach is adversarial training, where a segmentation model is trained alongside a discriminator that distinguishes between the source and target domains. The segmentation model aims to produce outputs that are indistinguishable by the discriminator, thereby improving generalization. Hoffman et al. (2018) demonstrated the effectiveness of this approach in their CyCADA framework, which achieves domain adaptation for semantic segmentation [51].

Multi-scale Contextual Information Capturing multi-scale contextual information is crucial for accurate segmentation, as objects in images can vary significantly in size. Traditional CNNs may struggle to capture features at different scales, leading to suboptimal performance. To address this, multi-scale feature extraction techniques have been developed. The Pyramid Scene Parsing Network (PSPNet) by Zhao et al. (2017) utilizes a pyramid pooling module to aggregate contextual information at multiple scales, significantly improving segmentation accuracy [52]. Similarly, the use of atrous (dilated) convolutions in models like DeepLab allows for effective multi-scale feature extraction without increasing the number of parameters [53].

Label Ambiguity In many cases, there is inherent ambiguity in the labeling of images, where different annotators may label the same image differently. This variability can lead to noisy training data and affect the performance of segmentation models. To mitigate the

impact of label ambiguity, probabilistic models and ensemble methods have been used. These approaches aggregate the predictions of multiple models or probabilistic annotations to produce a more robust final segmentation. For example, Monte Carlo dropout, proposed by Gal and Ghahramani (2016), can be used to estimate model uncertainty and produce probabilistic segmentation maps [54].

2.6 Encoder-Decoder Models

Encoder-decoder architectures are a powerful class of neural network models widely used in various fields, such as computer vision, natural language processing, and more. These architectures are particularly notable for their ability to reduce information to a lower-dimensional space using an encoder and then reconstruct or utilize this reduced information using a decoder.

2.6.1 Encoder

In the context of image processing, the encoder component of the architecture takes a high-dimensional input, such as an image, and progressively reduces it to a lower-dimensional representation often referred to as a latent space or bottleneck. This process involves a series of convolutional and pooling layers that capture increasingly abstract and compact features from the input. These lower-dimensional representations retain essential information about the input while discarding unnecessary details, making it an efficient way to extract meaningful features from complex data.

In the context of image processing, encoders typically use convolutional layers, which apply filters to the input image to detect patterns such as edges, textures, and more complex structures. Pooling layers are then used to down-sample the feature maps, reducing their dimensionality while retaining the most critical information. This combination of convolutional and pooling layers allows the encoder to create a compact representation of the input data, which can then be effectively processed by the decoder.

The encoder's effectiveness in capturing relevant features while reducing dimensionality has been demonstrated in various applications. For instance, Pham et al. (2019) utilized an encoder-decoder architecture incorporating anatomical priors to improve pelvic bone segmentation in MRI, highlighting the encoder's ability to capture important anatomical structures [55].

Encoders are not limited to image processing. In natural language processing (NLP), for example, encoders are used to transform sequences of words into fixed-size vectors that capture the semantic meaning of the text. This is crucial for tasks such as machine translation, where the meaning of a sentence must be preserved when translating between languages.

Several studies have highlighted the importance and effectiveness of encoders in various applications:

Network Intrusion Detection : Moraboena et al. (2020) explored the use of deep autoencoders for network intrusion detection. The encoder in their model was crucial for reducing the high-dimensional network traffic data to a lower-dimensional latent space. This reduction allowed the model to generalize better than traditional methods and improved network constraints, highlighting the encoder's ability to capture essential features from complex input data [56].

Diffusion Maps in Encoders : Dorado et al. (2019) proposed Deep Diffusion Autoencoders (DDA) that integrate diffusion maps in the bottleneck layer. The encoder's role in this architecture is to project the input data into a latent space that preserves the geometric structure of the samples. This integration improves the reconstruction error and maintains the intrinsic geometry of the data, demonstrating the encoder's capability in handling complex data distributions [57].

Logic-driven Encoders : Al-Hmouz et al. (2019) introduced logic-driven autoencoders that utilize fuzzy logic operations during the encoding process. The encoder in this model provides a transparent knowledge representation by capturing the essential logical relationships within the data. This approach enhances the interpretability of the model, which is particularly valuable in applications requiring clear and understandable decision-making processes [58].

Robust Feature Learning : Sun et al. (2016) proposed an unsymmetrical autoencoder (UAE) structure that effectively learns robust features. In this architecture, the encoder is designed to handle input distributions that differ significantly from the output distributions. This allows the model to learn more generalized features, outperforming traditional symmetrical autoencoders, and demonstrating the flexibility of encoder structures in various applications [59].

Anatomical Priors in Medical Imaging : Pham et al. (2019) utilized an encoder-decoder architecture with anatomical priors for improved pelvic bone segmentation in MRI. The encoder component was responsible for incorporating anatomical knowledge into the latent space, thereby enhancing the segmentation performance by leveraging prior medical knowledge. This study highlights the encoder's role in integrating domain-specific information to improve model accuracy [55].

Evolutionary Design of Encoders : Hajewski et al. (2020) described a distributed system using an evolutionary algorithm to design modular autoencoders. The encoder's structure in this approach is evolved to optimize for specific tasks, such as manifold learning and image denoising. This evolutionary design demonstrates the adaptability and customization potential of encoders to meet diverse application requirements [60].

Unsupervised Domain Adaptation : Zhou et al. (2019) proposed the Deep Cycle Autoencoder (DCA) for unsupervised domain adaptation. The encoder in this model plays

a critical role in transforming the input data into a latent space that is invariant to the domain shift between source and target domains. This allows the model to adapt to new domains without requiring labeled data, showcasing the encoder's effectiveness in generalizing across different datasets [61].

2.6.2 Decoder

The decoder component of an encoder-decoder architecture is responsible for taking the reduced information from the encoder and reconstructing the original input or producing an output with desired characteristics. This process typically involves a series of layers that progressively upscale the latent representation back to the original input dimensions or to a new target dimension.

In image processing, decoders generally use upsampling techniques, such as transposed convolutions (also known as deconvolutions), to increase the spatial dimensions of the feature maps. This is followed by convolutional layers to refine the output and make it more similar to the target image. Decoders are essential in tasks such as image denoising, super-resolution, and image generation.

Let's dive deeper into the various roles and applications of decoders, showcasing their versatility and effectiveness across different domains.

Applications of Decoders

Decoders have been effectively utilized in various applications across different domains. One such application is image denoising. I find this particularly fascinating because it directly improves the quality of our visual data. Prayuda et al. (2020) presented an autoencoder-based method for image companding, transforming high dynamic range (HDR) images to low dynamic range (LDR) and vice versa. The decoder in their architecture is crucial for accurately reconstructing the image while preserving essential details. Imagine having old, noisy photos and being able to clean them up digitally - that's the power of these decoders in action [62].

In the realm of network intrusion detection, decoders play a critical role as well. Moraboena et al. (2020) explored the use of deep autoencoders for this purpose. The encoder reduces the high-dimensional network traffic data to a more manageable size, and the decoder helps to identify anomalies by reconstructing the data and comparing it with the original input. This method enhances detection accuracy, ensuring that our networks remain secure from intrusions [56].

When it comes to medical image segmentation, the importance of decoders cannot be overstated. Pham et al. (2019) proposed a 2D encoder-decoder architecture that incorporates anatomical priors to improve pelvic bone segmentation in MRI. The decoder here expands the latent representations back to the original image dimensions, allowing precise segmentation of anatomical structures. This is crucial for accurate diagnosis and treatment in medical applications, highlighting how decoders can significantly impact healthcare [55].

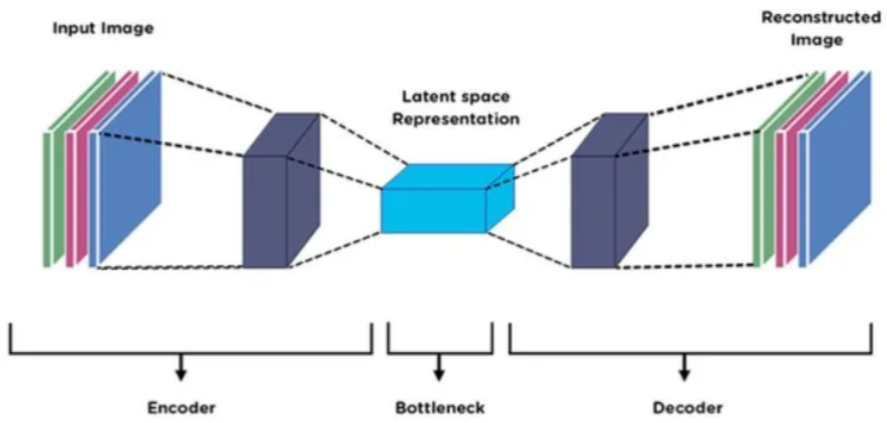


Figure 2.8. *An illustration of the encoder-decoder architecture. The decoder takes the latent representation and progressively reconstructs it back into the original input dimensions through upsampling and convolution operations.*

One area where decoders have really shined is in Generative Adversarial Networks (GANs). In GANs, the decoder is often referred to as the generator. Zhou et al. (2019) proposed the Deep Cycle Autoencoder (DCA) which integrates a generation procedure into adversarial adaptation methods for unsupervised domain adaptation. The decoder (or generator) creates realistic images from the latent space, demonstrating the potential of decoders in generative tasks. This is the kind of technology that powers advanced image editing tools and even deepfakes [61].

Another intriguing application is in manifold learning. Schuster and Krogh (2021) showed that decoders could be trained independently of encoders using manifold learning principles. This approach allows decoders to learn better representations and improve reconstruction accuracy, particularly for small datasets. It's fascinating to see how decoders can stand alone and still achieve remarkable results [63].

Decoders also play a significant role in image companding. Prayuda et al. (2020) discussed how decoders are used to transform HDR images to LDR and vice versa. This application is essential for maintaining image quality across different devices, ensuring that the images we see on our screens are consistent and high-quality [62].

In the evolutionary design of decoders, Hajewski et al. (2020) described a distributed system using an evolutionary algorithm to design modular autoencoders. Here, the decoder's structure is evolved to optimize for specific tasks like manifold learning and image denoising. This shows the adaptability and customization potential of decoders, tailored to meet diverse application requirements [60].

Advanced Architectures Beyond the traditional applications, decoders have been enhanced with advanced architectures to further improve their performance and applicability.

Dorado et al. (2019) introduced Deep Diffusion Autoencoders (DDA) that integrate diffusion maps in the bottleneck layer. The decoder in this architecture is designed to reconstruct the input data while preserving the geometric structure of the samples.

This is particularly useful in applications requiring high accuracy in data reconstruction, showcasing the decoder's effectiveness in handling complex data distributions [57].

Finally, logic-driven decoders presented by Al-Hmouz et al. (2019) use fuzzy logic operations during the decoding process. The decoder in this model provides transparent knowledge representation by decoding the logical relationships captured by the encoder. This transparency is crucial in applications where clear and understandable decision-making processes are required [58].

2.6.3 Autoencoders

Autoencoders are deep neural network architectures whose main purpose is to learn a representation for a set of input data. These models can perform various tasks such as feature engineering, compression, or data generation. Let's delve into how autoencoders work and their applications.

Given an input vector $x \in \mathbb{R}^d$, the left part of the autoencoder, known as the encoder, learns a low-dimensional latent representation $z \in \mathbb{R}^L$, where typically $L \ll d$. This low-dimensional layer is also known as the bottleneck. Then, the right part of the autoencoder, known as the decoder, tries to reconstruct x from z. The objective of the model is to create an output $\hat{x} \in \mathbb{R}^d$ as close as possible to the original input, i.e., $\hat{x} \approx x$.

Structure and Functionality

It's fascinating how these structures work. Essentially, the encoder reduces the input dimensions, compressing the information into a compact representation. The decoder then takes this compressed information and reconstructs the original input. This process not only helps in dimensionality reduction but also in understanding the underlying structure of the data.

For instance, in a generative autoencoder for image data, the encoder usually employs convolutional neural network (CNN) layers followed by linear layers. Conversely, the decoder utilizes linear layers followed by CNN layers. This structure helps in effectively capturing and reconstructing image features.

You can refer to the diagram of the decoder actions (Figure 2.8) to get a visual understanding of how the decoding process works. The decoder's upsampling and convolution operations gradually reconstruct the high-dimensional input from the compact latent space.

Applications

Autoencoders are incredibly versatile and have several applications:

1. Feature Engineering: Autoencoders can be used for dimensionality reduction, extracting meaningful features from high-dimensional data. This is particularly useful in tasks like data visualization and preprocessing for other machine learning models.

2. Compression: For image data, autoencoders can compress the images into smaller representations. This reduces storage space and speeds up data transmission without significant loss of information.

3. Data Generation: Autoencoders can generate new data samples to enrich limited datasets. This is especially valuable in fields where obtaining labeled data is expensive or time-consuming.

Deterministic Nature

It is important to note that traditional autoencoders are purely deterministic models. This means that given a specific input x_0, the output will always be $\hat{x}_0$. This consistency is crucial for applications requiring reliable and reproducible results. Autoencoders' flexibility and efficiency make them powerful tools in various fields, from computer vision to natural language processing. Their ability to learn compact representations of data helps in enhancing the performance of other machine learning models and enables new applications in data analysis and generation.

2.7 FTN-STN Networks

Recent research has identified innovative methods to increase the robustness of neural networks to changes in image fields. One such framework that has shown promising results is the collaborative FTN-STN networks, which stand for Fast Thinking Network (FTN) and Slow Thinking Network (STN). This concept is inspired by the human brain's dual-process theory, where fast, automatic decisions are often prone to errors, while slower, more deliberate processing handles complex decisions with higher accuracy.

2.7.1 Fast Thinking Network (FTN)

The Fast Thinking Network is designed to make quick decisions based on the input image. Given an image x, the FTN extracts two types of features: shape features z_s specifically for the segmentation task and image context features z_i for the image reconstruction task. This network consists of several key components:

- **Encoder** E_∂: This module processes the input image to extract initial features.
- **Feature Disentangler** H: This component is crucial as it separates the features relevant to the segmentation task from those needed for image reconstruction. The disentangler ensures that the features used for segmentation do not contain irrelevant information.
- **Decoders** D_{ϕ_s} **and** D_{ϕ_i}: These two decoders handle different tasks. D_{ϕ_s} focuses on segmentation, while D_{ϕ_i} reconstructs the image context.

The process starts with the encoder E_∂ extracting a feature representation from the input image. The feature disentangler H then processes these features to isolate the

task-specific information. Typically, H uses a stack of two convolutional layers followed by ReLU activation functions to perform this disentanglement. The segmented features z_s are passed to D_{ϕ_s} for segmentation, and the context features z_i are sent to D_{ϕ_i} for image reconstruction.

According to Ganapini et al. (2022), combining fast and slow decision modalities enhances decision quality, resource consumption, and efficiency in AI systems navigating constrained environments [64].

2.7.2 Slow Thinking Network (STN)

While the FTN is designed for speed, the STN focuses on accuracy. The STN acts as a corrective mechanism, refining the segmentation output of the FTN. The STN is a denoising autoencoder network C_ψ, which uses a learned prior shape encoded within the network to correct any errors in the segmentation predicted by the FTN.

The STN's architecture includes the following components:

- **Denoising Autoencoder** C_ψ: This network refines the segmentation results from the FTN by leveraging prior knowledge encoded in its layers. It takes the initial segmentation and improves its accuracy by correcting errors.

Hassantabar et al. (2020) demonstrate that the TUTOR framework uses decision rules as model priors to train deep neural networks, improving accuracy and efficiency with limited data [65].

The collaborative effort between the FTN and STN mimics the brain's dual-process theory, where quick, heuristic-based decisions are refined by slower, more analytical processing. This collaboration enhances the overall robustness and accuracy of the network's output.

2.7.3 Applications and Benefits

The FTN-STN framework has several applications, particularly in fields requiring high accuracy and robustness in image segmentation and reconstruction. Some notable applications include:

- **Medical Imaging**: In medical imaging, accurate segmentation of anatomical structures is crucial. The FTN-STN framework can quickly provide initial segmentation results, which the STN can then refine to ensure high accuracy, essential for diagnostics and treatment planning [66].

- **Autonomous Driving**: Autonomous vehicles rely heavily on accurate segmentation of their surroundings. The FTN can quickly identify objects and road features, while the STN ensures these segmentations are precise, enhancing the safety and reliability of the vehicle's navigation system [67].

- **Surveillance Systems**: Surveillance systems can benefit from the FTN-STN framework by quickly detecting and segmenting objects of interest (e.g., intruders, vehicles), with the STN refining these detections to reduce false positives and improve overall system reliability [64].

2.7.4 Detailed Workflow

Let's break down the workflow in more detail to understand how these networks interact.

1. **Image Input**: The process begins with an input image x. This image is fed into the FTN, which is responsible for quick processing.

2. **Encoding**: The encoder E_∂ extracts feature representations from the image. This step is crucial as it captures the essential details needed for both segmentation and image reconstruction.

3. **Feature Disentangling**: The extracted features are then processed by the feature disentangler H. This component uses a stack of convolutional layers and ReLU activation functions to separate the shape features z_s from the context features z_i.

4. **Segmentation and Reconstruction**: The disentangled features are sent to their respective decoders. D_{ϕ_s} handles the segmentation task, providing a quick but potentially rough segmentation of the image. Simultaneously, D_{ϕ_i} reconstructs the image context, ensuring that important contextual information is preserved.

5. **Initial Segmentation Output**: The output of D_{ϕ_s} is the initial segmentation result. This segmentation is quick, providing a preliminary result that can be used immediately if needed.

6. **Correction by STN**: The initial segmentation is then fed into the STN, specifically into the denoising autoencoder C_ψ. This network uses a learned prior shape to correct any errors in the segmentation, refining the output to achieve higher accuracy.

7. **Final Output**: The final output is a highly accurate segmentation of the input image, combining the speed of the FTN with the precision of the STN.

2.7.5 Spatial Transformer Networks (STN)

Spatial Transformer Networks (STN) are a component that can be integrated into FTN-STN frameworks to enhance spatial invariance in the networks. STNs allow the model to spatially transform the input data, thus enabling the network to focus on the relevant parts of the input image.

As illustrated in Figure 2.9, an STN typically includes three main components:

- **Localization Network**: Predicts the transformation parameters ∂ that should be applied to the input image.

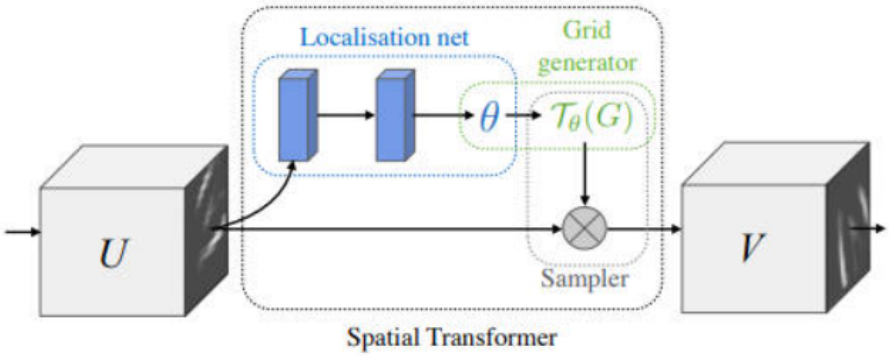


Figure 2.9. *Spatial Transformer Network (STN): The STN component includes a localization network that predicts transformation parameters, a grid generator, and a sampler that uses bilinear interpolation to produce the output. This mechanism allows the network to focus on the region of interest in the input image.*

- **Grid Generator**: Uses the predicted transformation parameters to generate a sampling grid.
- **Sampler**: Produces the transformed output by applying the sampling grid to the input image using bilinear interpolation.

The integration of STNs in the FTN-STN framework can significantly improve the robustness and accuracy of the model by allowing it to focus on relevant parts of the input image and ignore irrelevant information.

2.7.6 Advancements and Future Directions

The FTN-STN framework is a significant advancement in neural network architecture, drawing inspiration from cognitive science to improve machine learning models. Future research can explore several directions to further enhance this framework:

- **Integration with Other Neural Network Models**: Combining FTN-STN networks with other architectures, such as attention mechanisms or transformers, could further enhance their performance.
- **Real-time Applications**: Optimizing the FTN-STN framework for real-time applications, such as live video analysis, could expand its usability in dynamic environments.
- **Improved Feature Disentangling**: Enhancing the feature disentangler H to better separate task-relevant and task-irrelevant information can lead to even more robust performance.
- **Transfer Learning**: Applying transfer learning techniques to leverage pre-trained models in the FTN-STN framework could reduce training times and improve performance across various tasks.

2.8 Fully Convolutional Networks (FCNs)

Fully Convolutional Networks (FCNs) have revolutionized the field of image segmentation by adapting the principles of Convolutional Neural Networks (CNNs) for pixel-level semantic segmentation tasks. Unlike traditional CNNs, which are typically used for classification tasks, FCNs are designed to classify each pixel of an image into predefined classes. This adaptation is achieved by replacing the fully connected layers typically found at the end of CNNs with convolutional layers, thereby retaining spatial information throughout the network.

2.8.1 Architecture of FCNs

The primary difference between FCNs and conventional CNNs lies in the absence of fully connected layers. In FCNs, the fully connected layers are replaced by convolutional layers, which allows the network to make dense predictions for each pixel rather than a single label per image. This architectural change ensures that spatial hierarchies and features from different levels of abstraction are preserved.

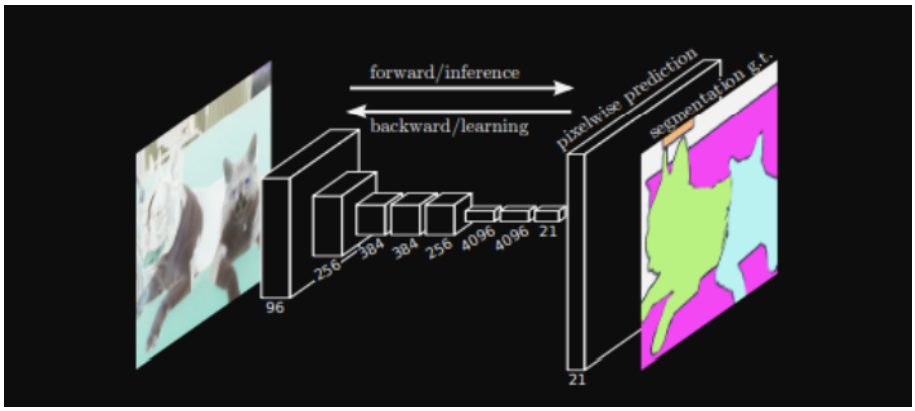


Figure 2.10. *Fully Convolutional Network (FCN): The FCN process starts with an input image that passes through several convolutional layers, extracting features and reducing dimensionality. The network then uses fully convolutional layers to maintain spatial information, and upsampling layers to restore the original resolution for pixel-wise prediction, resulting in a detailed segmentation map.*

The design choice to use convolutional layers instead of fully connected layers has several advantages. For instance, convolutional layers are inherently translation-invariant, making them more suitable for tasks that require the recognition of patterns regardless of their location in the input image. Additionally, this design significantly reduces the number of parameters in the network, leading to less computational complexity and faster training times [68].

2.8.2 Skip Connections

One of the critical features of FCNs is the use of skip connections. These connections link the feature maps from deeper layers of the network to shallower layers. By upsampling the feature maps from deeper layers and merging them with those from earlier

layers, the network combines semantic information from deep, coarse layers with appearance information from shallow, fine layers. This combination produces more accurate and detailed segmentations.

Skip connections address one of the major challenges in segmentation tasks: the loss of spatial resolution due to downsampling operations like pooling. By incorporating feature maps from earlier layers, skip connections help the network retain fine-grained details that are essential for precise segmentation. This technique also facilitates better gradient flow during backpropagation, which aids in training deeper networks effectively.

2.8.3 U-Net: An Evolution of FCNs

A notable architecture inspired by FCNs is the U-Net, which is particularly effective for tasks such as image segmentation. The U-Net architecture features a U-shaped design consisting of both contracting and expanding paths. This structure makes U-Net especially suitable for medical image segmentation, where precise localization is crucial.

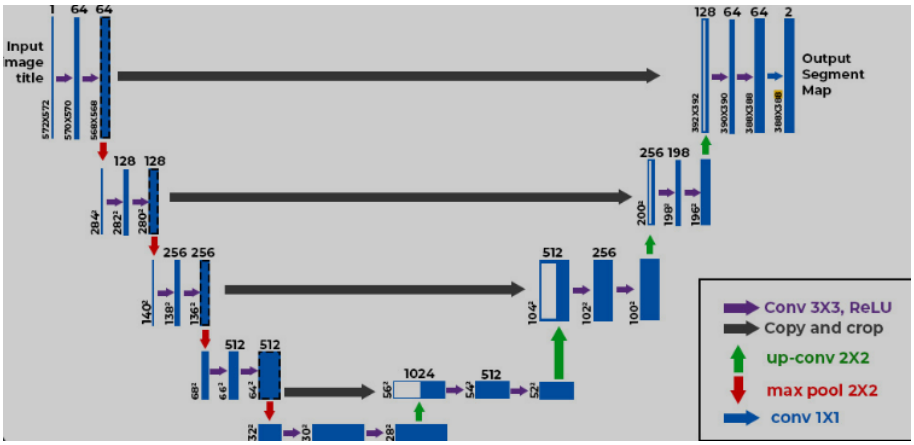


Figure 2.11. *U-Net Architecture: The U-Net consists of a contracting path (left side) and an expansive path (right side). The contracting path follows the typical architecture of a convolutional network, with repeated application of two 3x3 convolutions (purple arrows), each followed by a ReLU and a 2x2 max pooling operation (red arrows) for downsampling. In the expansive path, feature maps are upsampled using a 2x2 up-convolution (green arrows) and concatenated with the corresponding high-resolution features from the contracting path (black arrows). The network ends with a 1x1 convolution (blue arrows) to map each 64-component feature vector to the desired number of classes.*

The contracting path of the U-Net captures context using a series of convolutional and pooling layers, similar to traditional FCNs. The expanding path, on the other hand, uses up-convolutions (or deconvolutions) to increase the spatial resolution of the feature maps. Feature maps from the contracting path are copied to the expanding path to avoid loss of spatial information. Finally, a 1x1 convolution is applied to produce the final segmentation map, classifying each pixel of the input image.

The U-Net was initially proposed for the segmentation of biological microscopy images. Its architecture and training strategy, which includes extensive use of data augmentation, enable it to learn effectively from a limited number of annotated images. The U-Net's

ability to perform well with limited data makes it a valuable tool in medical imaging, where annotated data can be scarce and expensive to obtain [69].

2.8.4 Applications in the Medical Field

FCNs and their variants, such as U-Net, have found extensive applications in the medical field. They have been used for tasks such as brain tumor segmentation, case-aware segmentation, skin cancer segmentation, and iris segmentation. These networks have revolutionized medical image analysis, enabling precise and automated segmentation of organs and tumors in images from various modalities, such as MRI, CT scans, and X-rays. FCNs allow medical professionals to save time and make more accurate diagnoses by providing detailed maps of anatomical structures or pathological areas.

In brain tumor segmentation, for instance, FCNs help in delineating tumor boundaries, which is critical for treatment planning and prognosis. In skin cancer segmentation, these networks assist in identifying malignant lesions with high accuracy, thereby facilitating early diagnosis and treatment. The ability of FCNs to process and analyze large volumes of medical images efficiently has significantly improved the workflow in medical diagnostics [70].

2.8.5 Challenges and Limitations

Despite their popularity and effectiveness, conventional FCN models have certain limitations:

- **Speed**: Traditional FCNs are not fast enough for real-time inference, which can be a significant drawback in applications requiring immediate results.
- **Global Context**: FCNs do not efficiently incorporate global context information, which can sometimes result in less accurate segmentations.
- **3D Images**: Extending FCNs to three-dimensional images (such as volumetric data) is not straightforward and often requires substantial modifications to the architecture.

To address these limitations, several approaches have been proposed. For instance, lightweight architectures and optimization techniques have been developed to improve the inference speed of FCNs. Additionally, integrating attention mechanisms into FCNs has shown to enhance the network's ability to capture global context, leading to more accurate segmentations [71].

2.8.6 Recent Advances and Future Directions

To address these limitations, subsequent architectures have been developed. For instance, models like the Fully Convolutional DenseNets (FC-DenseNet) integrate dense connections to improve information flow and model efficiency [69]. Additionally, attention

mechanisms have been introduced to FCNs to better capture global context and improve segmentation accuracy [70].

Recent research also explores the use of adversarial training to enhance the performance of FCNs. By incorporating a discriminator network, the segmentation network can be trained to produce more realistic segmentations, which are harder for the discriminator to distinguish from real data [71].

Furthermore, hardware advancements have played a significant role in the deployment of FCNs. For example, the exploration of hardware design for deep neural networks with binary parameters has enabled the implementation of these networks in mobile and IoT devices, making them more accessible and efficient [68].

Chapter 3

Related Work

In this chapter, an initial description is given of the various methods applied for domain generalization, which have also influenced the research of this thesis.

3.1 Data Modification Methods for Out-of-Domain Generalization and Adversarial Robustness

3.1.1 Impact of Data Modification Strategies

The paper by Gokhale et al. (2022) [2] investigates the effects of various data modification strategies on out-of-domain (OOD) generalization and adversarial robustness (AR). This comprehensive study evaluates the impact of additional training datasets, data augmentation, debiasing, and dataset filtering on both in-domain and OOD performance, as well as AR. The authors highlight the unclear relationship between data modification and AR, aiming to provide empirical insights to bridge this gap.

The study uses a two-dimensional synthetic dataset to visualize the effects of each data modification method on the training distribution. The findings suggest that incorporating more data, either through additional datasets or data augmentation, benefits both OOD accuracy and AR. However, the study also reveals that data filtering, which has previously been shown to improve OOD accuracy in natural language processing tasks, may actually harm OOD accuracy in other tasks such as question answering and image classification.

One key insight from this work is the differential impact of data modification strategies across various tasks. For instance, while data augmentation generally enhances robustness, its effectiveness can vary depending on the specific task and dataset. This variability underscores the need for task-specific considerations when implementing data modification strategies.

The authors conclude that while data modification methods can significantly improve OOD generalization and AR, the choice of strategy should be carefully tailored to the specific application. This work serves as a valuable empirical study, providing insights that can inform future research directions in the field of domain generalization and adversarial robustness.

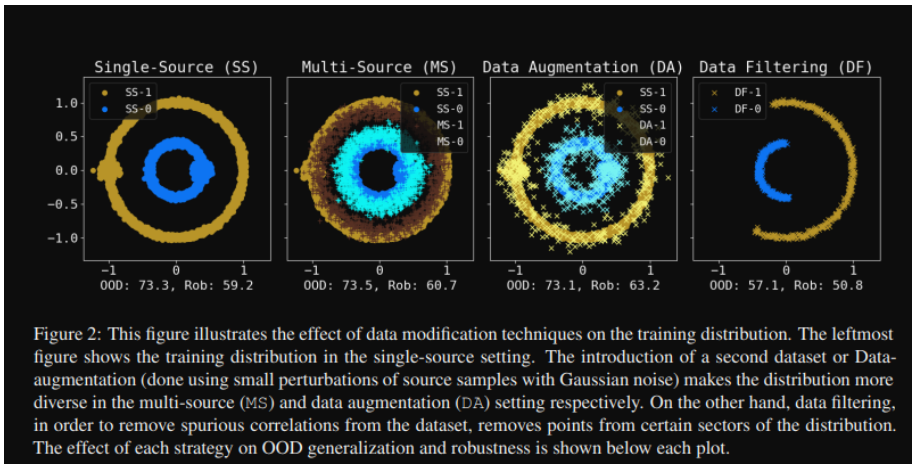


Figure 3.1. *This figure illustrates the effect of data modification techniques on the training distribution. The leftmost figure shows the training distribution in the single-source setting. The introduction of a second dataset or data augmentation (done using small perturbations of source samples with Gaussian noise) makes the distribution more diverse in the multi-source (MS) and data augmentation (DA) settings, respectively. On the other hand, data filtering, in order to remove spurious correlations from the dataset, removes points from certain sectors of the distribution. The effect of each strategy on OOD generalization and robustness is shown below each plot.*

3.1.2 Consistency Training with Random Data Augmentation

Xiao et al. (2022) [1] explore the use of consistency training with random data augmentation to achieve state-of-the-art results in domain adaptation (DA) and generalization (DG). The authors propose a differentiable adversarial data augmentation method based on spatial transformer networks (STNs), which combines adversarial and random transformations to enhance accuracy and robustness.

Consistency training involves using a model to make predictions on both augmented and original data, ensuring that these predictions are consistent. This method has proven effective in various tasks, but the novel contribution of Xiao et al [1]. lies in the integration of adversarial and random augmentations. The differentiable nature of STNs allows for efficient and scalable adversarial training, addressing a significant limitation of previous methods.

The combined adversarial and random-transformation-based approach outperforms existing state-of-the-art methods on multiple DA and DG benchmark datasets. Moreover, the method demonstrates desirable robustness to various types of corruption, further validating its effectiveness.

Xiao et al.[1] emphasize the importance of balancing adversarial and random transformations to achieve robust domain adaptation and generalization. Their findings suggest that consistency training with random data augmentation is a powerful tool for improving model performance across diverse domains. This work provides valuable insights into the design of robust deep learning models capable of handling OOD data and adversarial attacks.

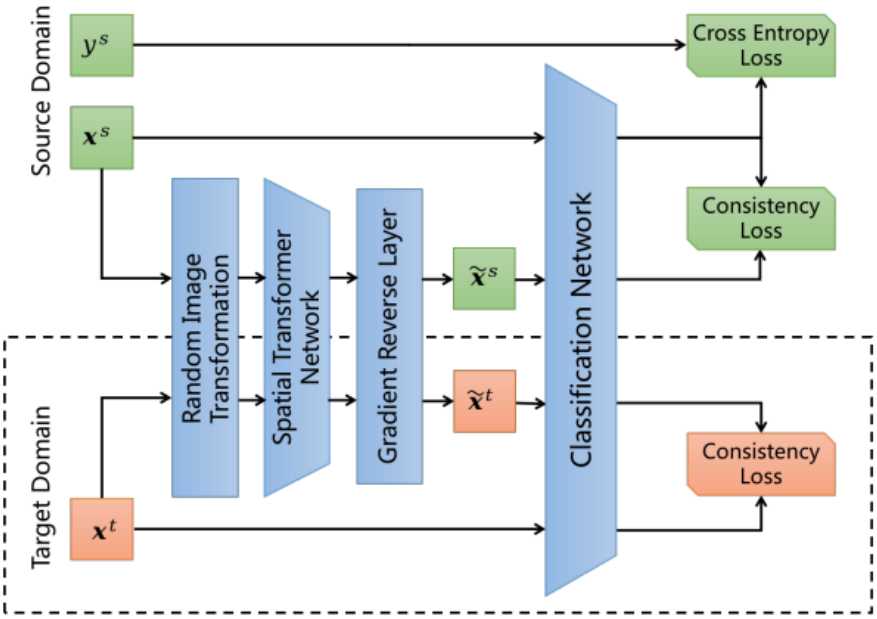


Figure 3.2. *Overview of the proposed model. The authors propose using random image transformations and adversarial spatial transformer networks (STN) to achieve domain adaptation and generalization (without the dashed line bounding box). The figure illustrates the flow of data from the source domain and target domain through various transformations and the classification network, with losses computed for cross-entropy and consistency.*

3.2 Feature Augmentation Techniques for Domain Generalization

3.2.1 Feature Augmentation with Gaussian Noise

Li et al. (2021) [3] propose a novel approach to domain generalization through feature augmentation. The authors argue that existing methods primarily rely on image-space data augmentation, which requires careful design and offers limited diversity. Instead, they advocate for feature augmentation as a more promising direction for DG.

The proposed technique involves perturbing the feature embedding with Gaussian noise during training, leading to a classifier with domain-generalization performance comparable to existing state-of-the-art methods. To capture more meaningful statistics reflective of cross-domain variability, the authors estimate the full class-conditional feature covariance matrix iteratively during training. This enables joint stochastic feature augmentation, which perturbs features in directions corresponding to intra-class and cross-domain variability.

The authors validate their method on three standard DG benchmarks: Digit-DG, VLCS, and PACS. Their results show that the proposed feature augmentation technique outperforms or is comparable to the state of the art in all setups. Additionally, the experimental analysis provides insights into how the method contributes to training robust and generalizable models.

Li et al.[3] conclude that feature augmentation, particularly with Gaussian noise, is

an effective strategy for improving domain generalization. Their work underscores the potential of feature-based approaches in enhancing model robustness and highlights the importance of considering feature space perturbations in DG research.

3.2.2 Adversarial Feature Augmentation and Normalization

Chen et al. (2021) [6] propose an innovative approach called Adversarial Feature Augmentation and Normalization (A-FAN), which shifts the focus from pixel-level perturbations to intermediate feature embeddings, thereby offering a more efficient and effective solution.

A-FAN consists of two main components. First, it augments visual recognition models with adversarial features that incorporate flexible scales of perturbation strengths. Second, it extracts adversarial feature statistics from batch normalization layers and re-injects them into clean features through feature normalization. This method avoids the computational expense associated with pixel-level perturbations while still providing significant generalization improvements.

The authors validate A-FAN across a diverse range of visual recognition tasks using representative backbone networks, including ResNets and EfficientNets for classification, Faster-RCNN for detection, and Deeplab V3+ for segmentation. The extensive experiments conducted on various datasets—such as CIFAR-10, CIFAR-100, ImageNet, Pascal VOC2007, Pascal VOC2012, COCO2017, and Cityscapes—demonstrate that A-FAN consistently improves generalization performance over strong baselines.

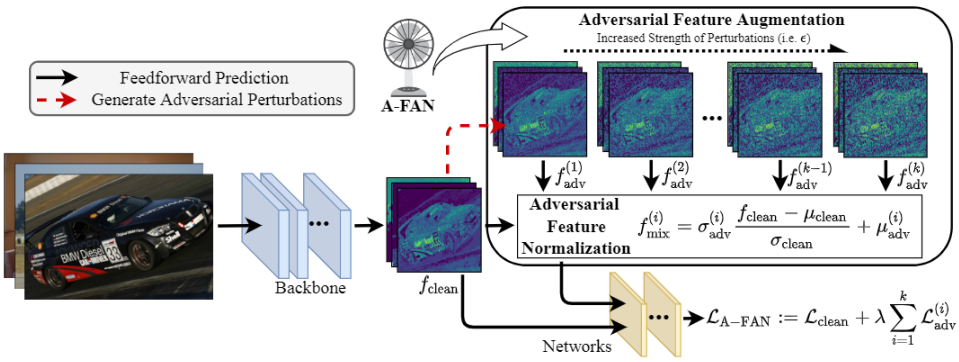


Figure 3.3. *The pipeline of A-FAN, which contains adversarial feature augmentation and adversarial feature normalization. From top to bottom, a series of adversarial feature perturbations with different strengths are generated to augment the intermediate clean features. Then, the statistics (i.e., μ_{adv} and σ_{adv}) of perturbed features f_{adv} are extracted and re-injected into the original clean features f_{clean}. In the end, the normalized features f_{mix} are taken as inputs by the rest of the network and optimized by L_{A-FAN} with standard (L_{clean}) and adversarial (L_{adv}) training objectives.*

Moreover, comprehensive ablation studies and detailed analyses reveal that adding perturbations to specific modules and layers of the classification, detection, and segmentation backbones results in optimal performance [6]. This work highlights the potential of adversarial feature augmentation as a powerful tool for enhancing the robustness and

generalization capabilities of deep learning models in visual recognition tasks.

In conclusion, the innovative approach of A-FAN offers a compelling alternative to traditional adversarial data augmentation techniques. By focusing on feature embeddings, it provides a computationally efficient method that yields consistent and significant improvements across a wide range of visual recognition tasks.

3.3 Adversarial Augmentation for Robust Domain Generalization

3.3.1 Adversarial Style Augmentation

Zhong et al. (2022) [4] address the problem of domain generalization in semantic segmentation by introducing a novel adversarial style augmentation approach, termed AdvStyle, which significantly enhances the performance of semantic segmentation models on unseen real domains.

The key insight of this work is that image style variations can greatly influence model performance, and these style features are well represented by the channel-wise mean and standard deviation of images. Inspired by this observation, AdvStyle dynamically generates hard stylized images during training to prevent the model from overfitting on the source domain. This is achieved by treating the style feature as a learnable parameter, which is updated through adversarial training. The learned adversarial style feature is then used to construct adversarial images that are employed for robust model training.

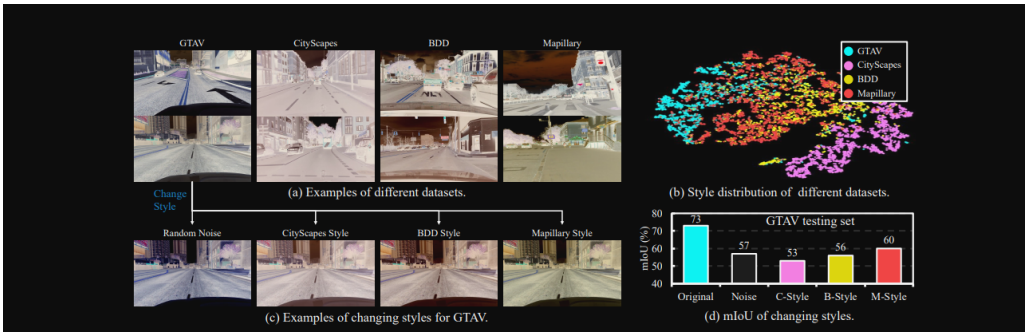


Figure 3.4. *(a) Examples of different datasets. The image styles from different datasets commonly vary. (b) Style distribution of different datasets. The authors use image-level mean-variance as the style feature to show that the style distribution gap between different datasets is large. (c) Examples of changing style features for a GTAV sample, including adding random noise and replacing the style feature with samples from other datasets. (d) mIoU performance of changing styles for the GTAV testing set, which is largely reduced after style changing.*

AdvStyle's implementation is straightforward and can be easily integrated into different models. The authors demonstrate its efficacy through experiments on two synthetic-to-real semantic segmentation benchmarks, showing that AdvStyle can significantly improve model performance on unseen real domains. Furthermore, the approach extends to domain generalized image classification, where it also yields clear improvements.

This work underscores the importance of addressing style variations in domain generalization tasks and provides a practical solution that can be widely applied across different models and datasets. The findings suggest that adversarial style augmentation is a promising direction for enhancing the robustness and generalization of deep learning models in computer vision.

3.4 Input Level Data Augmentation

Data augmentation at the input level involves creating new samples by applying various transformations to the original images before they are fed into the model. The main goal is to enrich the dataset with variations of the existing images so that the model extracts robust features for decision-making and becomes more resilient to changes and noise in the input data. These methods range from simple geometric transformations to more complex augmentations.

These techniques are relatively easy to implement and use, making them a popular choice in many image processing tasks. However, they typically do not lead to significant improvements in a model's generalization performance for complex architectures and applications. Therefore, more advanced augmentation methods or feature-level augmentation methods are often chosen. One of these advanced methods explored in this thesis is PixMix.

3.4.1 PixMix

PixMix is a novel data augmentation strategy designed to improve the safety and robustness of machine learning models by leveraging the natural structural complexity of images such as fractals and feature visualizations [72]. This method aims to address the challenge of optimizing multiple safety measures without sacrificing performance in other areas, a common issue with existing methods.

Methodology

PixMix enhances the training dataset by integrating structurally complex images, thus improving model robustness, consistency, and calibration. The methodology consists of the following key steps:

1. **Picture Sources (PIX)**: PixMix utilizes two main types of structurally complex images—fractals and feature visualizations. Fractals are known for their intricate patterns and high degree of structural complexity, while feature visualizations are generated to maximize the response of neurons in a neural network, thereby introducing high visual complexity.

2. **Mixing Pipeline (MIX)**: The mixing pipeline involves augmenting clean training images with structurally complex images. This is done by:

    - Applying a standard augmentation with a 50% probability.

- Repeatedly mixing the image a random number of times (up to k times) with either an augmented version of the clean image or an image from the mixing set.
- Using additive or multiplicative mixing operations, where multiplicative mixing is performed similarly to the geometric mean.

The mixing operations use coefficients sampled from a Beta distribution, ensuring diverse combinations.

3. **Mathematical Formulation**: The mixed style statistics are computed as follows:

$$\gamma_{\text{mix}} = \lambda_{\text{mix}}\sigma(x_i) + (1 - \lambda_{\text{mix}})\sigma(x_j), \tag{3.1}$$

$$\beta_{\text{mix}} = \lambda_{\text{mix}}\mu(x_i) + (1 - \lambda_{\text{mix}})\mu(x_j), \tag{3.2}$$

where σ and μ represent the standard deviation and mean of the feature maps, respectively, and λ_{mix} is a mixing coefficient sampled from a Beta distribution.

4. **Adding Noise**: Noise is added to the mixed style statistics to simulate domain shifts. The noise components $\Sigma_\gamma \cdot \epsilon_\gamma$ and $\Sigma_\beta \cdot \epsilon_\beta$ are sampled from re-scaled Gaussian distributions, ensuring the augmented styles are diverse yet realistic. The noise factors are calculated as:

$$\Sigma_\gamma = \sigma^2(\{\sigma(x_j)\}_{j=1,\ldots,B}), \tag{3.3}$$

$$\Sigma_\beta = \sigma^2(\{\mu(x_j)\}_{j=1,\ldots,B}), \tag{3.4}$$

$$\epsilon_\gamma, \epsilon_\beta \sim \mathcal{N}(0, 1), \tag{3.5}$$

where B is the batch size.

5. **Combining Styles and Noise**: The final PixMix transformation is applied to the feature maps as follows:

$$\text{PixMix}(x_i) = (\gamma_{\text{mix}} + \Sigma_\gamma \cdot \epsilon_\gamma) \odot x_i + (\beta_{\text{mix}} + \Sigma_\beta \cdot \epsilon_\beta), \tag{3.6}$$

where $\odot$ denotes element-wise multiplication.

Impact on Safety Measures

PixMix significantly improves multiple safety measures, including out-of-distribution robustness, prediction consistency, resilience to adversaries, calibrated uncertainty estimates, and anomaly detection [72]. By introducing new sources of structural complexity, PixMix ensures that the model is exposed to a wider variety of patterns during training, leading to better generalization and robustness across various safety metrics.

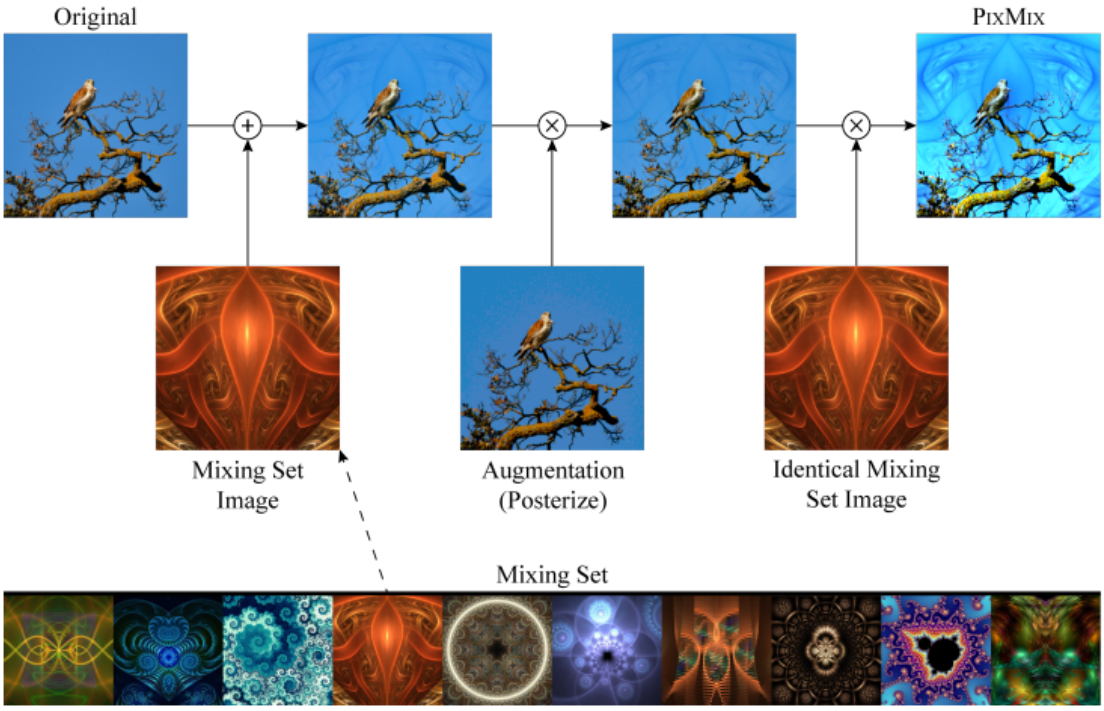


Figure 3.5. *PIXMIX augmentation process. The original image is mixed with fractal and feature visualization images to create structurally complex augmented images.*

Part II

Experimental Part

Chapter 4

Data and Preprocessing

This chapter will present the datasets used in the context of this thesis and the procedures followed for preprocessing the training data.

4.1 SYNTHIA Dataset

The SYNTHIA dataset, also known as the SYNTHetic collection of Imagery and Annotations, is designed to aid semantic segmentation and related scene understanding problems in driving scenarios. It provides a large collection of photo-realistic frames rendered from a virtual city, with precise pixel-level semantic annotations for 13 classes: misc, sky, building, road, sidewalk, fence, vegetation, pole, car, sign, pedestrian, cyclist, and lane-marking.

The dataset's significant attributes include:

- **Large Volume of Data and Ground Truth**: Over 200,000 HD images from video streams and more than 20,000 HD images from independent snapshots.
- **Scene Diversity**: Various scenes including European-style towns, modern cities, highways, and green areas.
- **Dynamic Objects**: Includes cars, pedestrians, and cyclists in different scenarios.
- **Multiple Seasons**: Dedicated themes for winter, fall, spring, and summer.
- **Lighting Conditions and Weather**: Incorporates dynamic lighting, shadows, several day-time modes, rain, and night modes.
- **Sensor Simulation**: Eight RGB cameras forming a binocular 360^{o} camera, and eight depth sensors.
- **Automatic Ground Truth**: Provides instance-level semantic segmentation (pixel-wise annotations), depth, and car ego-motion.

The SYNTHIA dataset contains several subsets tailored to different research needs:

- **SYNTHIA-AL**: Designed for active learning purposes, this video stream dataset is generated at 25 FPS, including classes such as void, sky, building, road, sidewalk,

fence, vegetation, pole, car, traffic sign, pedestrian, bicycle, lane-marking, and traffic light. The ground truth includes instance segmentation, 2D and 3D bounding boxes, and depth information.

- **SYNTHIA-SF**: Comprising video sequences acquired at 5 FPS, this subset features different scenarios and traffic conditions. It includes 2224 images with ground truth for semantic segmentation, instance segmentation, depth, and calibration parameters. The semantic classes are compatible with Cityscapes, including road, sidewalk, building, wall, fence, pole, traffic light, traffic sign, vegetation, terrain, sky, person, rider, car, truck, bus, train, motorcycle, bicycle, road lines, other, and road works.

- **SYNTHIA-RAND**: This set contains 13,407 images used for training and domain adaptation in the CVPR'16 paper. Generated as random perturbations, these images lack temporal consistency and instance annotations, with classes including void, sky, building, road, sidewalk, fence, vegetation, pole, car, sign, pedestrian, and cyclist.

- **SYNTHIA-RAND-CITYSCAPES**: Comprising 9,000 random images, this set includes labels compatible with the Cityscapes test set and additional classes such as lane-marking. These images are randomly perturbed and contain ground truth for instances.

- **SYNTHIA VIDEO SEQUENCES**: Acquired at 5 FPS, these video subsets feature different scenarios and traffic conditions, each divided into sub-sequences for various weather/illumination/season conditions. Each sub-sequence contains around 8,000 images with ground truth for semantic segmentation, instance segmentation, global camera poses, depth, and calibration parameters. The semantic classes include misc, sky, building, road, sidewalk, fence, vegetation, pole, car, sign, pedestrian, cyclist, and lane-marking.

The dataset is organized into various sequences, which can be used to train models under specific conditions and then evaluate them in others. Some of these sequences include:

- **Highway:**
  - SYNTHIA-SEQS-01-DAWN (7709 downloads)
  - SYNTHIA-SEQS-01-FALL (6269 downloads)
  - SYNTHIA-SEQS-01-FOG (7607 downloads)
  - SYNTHIA-SEQS-01-NIGHT (5521 downloads)
- **New York-ish:**
  - SYNTHIA-SEQS-02-DAWN (2439 downloads)
  - SYNTHIA-SEQS-02-FALL (3773 downloads)

  - SYNTHIA-SEQS-02-FOG (6018 downloads)
  - SYNTHIA-SEQS-02-NIGHT (2974 downloads)

- **Old European Town:**
  - SYNTHIA-SEQS-04-DAWN (820044 downloads)
  - SYNTHIA-SEQS-04-FALL (6212 downloads)
  - SYNTHIA-SEQS-04-FOG (2413 downloads)
  - SYNTHIA-SEQS-04-NIGHT (4630 downloads)

4.2 Data Preprocessing

Data preprocessing is a critical step to ensure the quality and usability of the SYNTHIA dataset for deep learning models. Given the diversity and complexity of the dataset, several preprocessing steps are necessary:

4.2.1 Noise Reduction

Noise reduction is a crucial step for enhancing the clarity and quality of synthetic images in the SYNTHIA dataset. Various deep learning techniques, such as convolutional neural networks (CNNs), autoencoders, and generative adversarial networks (GANs), have proven effective in denoising tasks [73].

4.2.2 Rescaling

The intensity of the images was rescaled as follows:

$$\frac{x - x_2}{x_{98} - x}$$

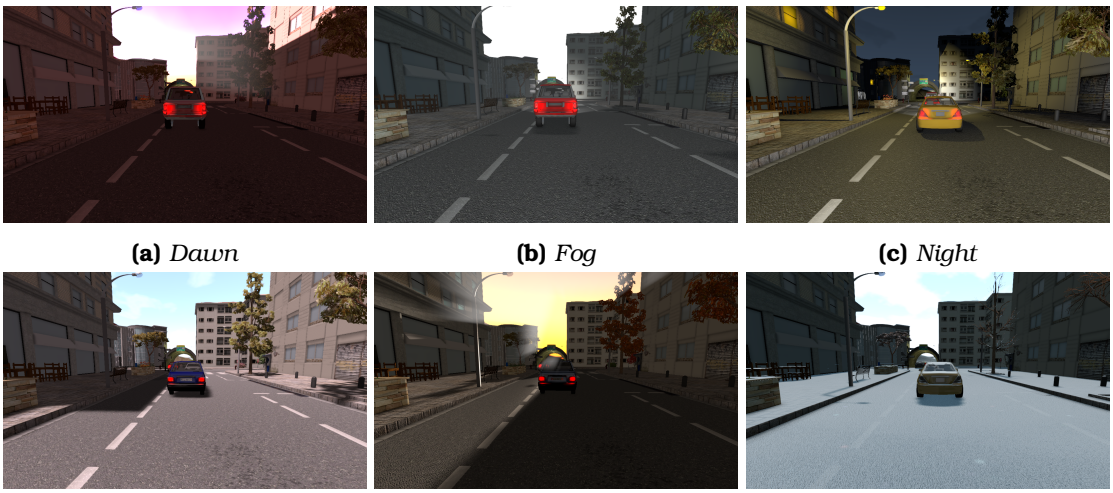
where x_2 and x_{98} are the 2nd and 98th percentiles of the intensity of each image, and x is the image. This process ensures that the intensity levels across different images are comparable, which is particularly important for datasets with varying lighting conditions and weather effects. Task-aware image downscaling is an innovative approach that enhances restoration performance by jointly learning downscaling and upscaling networks [74].

4.2.3 Resizing

Resizing is a common step in image preprocessing, especially for deep learning applications. It helps to bring all images to a consistent size, reducing computational costs and improving the generalization capabilities of the models. For the SYNTHIA dataset, resizing is performed in three dimensions, followed by cropping to maintain a consistent size suitable for training. Learning-based resizing methods have been shown to significantly improve the performance of computer vision tasks compared to traditional methods [75].

4.2.4 Geometric and Photometric Transformations

Geometric transformations, such as rotation, scaling, and translation, are applied to augment the dataset and improve the model's robustness to different perspectives and orientations. Photometric transformations, including adjustments to brightness, contrast, and color balance, help to simulate various lighting conditions and enhance the model's ability to generalize to unseen scenarios. These transformations are crucial for ensuring that the models are robust to different geometric and photometric variations [76].



(a) *Dawn* **(b)** *Fog* **(c)** *Night*

(d) *Spring* **(e)** *Sunset* **(f)** *Winter*

Figure 4.1. *Examples of SYNTHIA dataset images captured under different circumstances: Dawn, Fog, Night, Spring, Sunset, and Winter. Each image showcases the same scene with varying lighting and weather conditions, demonstrating the diversity of the dataset and its utility for training robust models.*

Chapter 5

Implementation

This chapter describes the main methods used in the context of this thesis.

5.1 Data Augmentation

Data augmentation is a crucial technique in machine learning and deep learning, aimed at enhancing the performance and generalization capability of models by artificially expanding the size and diversity of the training dataset. This method involves creating new training examples through various transformations of the original data while retaining the essential information they contain. Data augmentation is particularly significant in image processing but is also applicable to other domains such as text and signal data.

The primary objective of data augmentation is to introduce variability into the training set, thereby helping the model to generalize better to unseen data. By simulating real-world variations, data augmentation can effectively mitigate overfitting, where the model performs well on the training data but poorly on new, unseen data.

Common techniques in image data augmentation include geometric transformations (such as rotation, translation, scaling, and flipping), color space augmentations (such as brightness, contrast, and saturation adjustments), and adding noise (such as Gaussian noise). These transformations create slightly altered versions of the original images, which help the model become invariant to these changes and improve its robustness.

For example, de-texturization involves removing the texture details from the image while maintaining the basic structural information. This technique can help the model focus on the shape and outline of objects, reducing the influence of texture patterns that might not be relevant for the specific task. De-colorization, or converting an image to grayscale, removes color information while preserving intensity information. This can be particularly useful in applications where color does not play a significant role in classification or detection tasks, helping the model to generalize better to different color variations.

Edge enhancement emphasizes the edges within an image. This technique highlights the boundaries of objects, making it easier for models to detect and segment distinct entities. Enhancing edges can improve the model's ability to discern between adjacent objects and better understand object boundaries. Generating a salient edge map involves detecting the most prominent edges in an image, focusing on the most significant boundaries

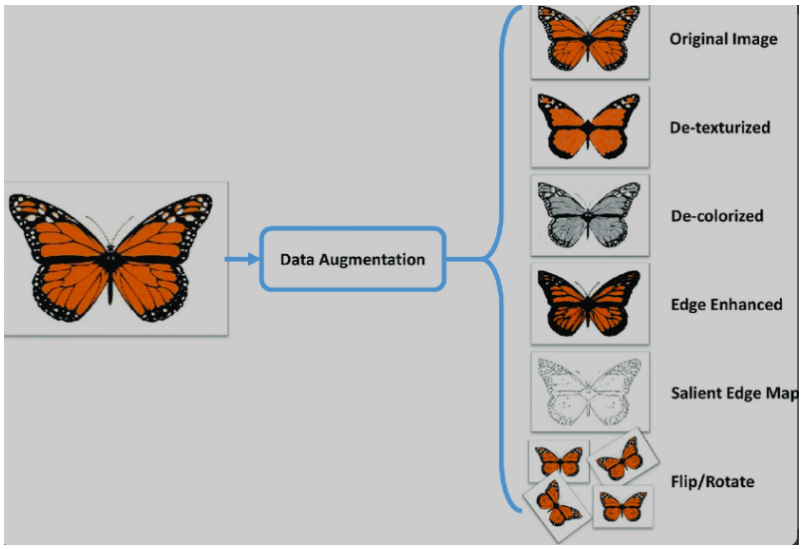


Figure 5.1. *Different Data Augmentation Techniques applied on an image of a butterfly.*

and contours. This method helps in emphasizing the critical parts of an image, which can improve object detection and segmentation tasks.

Flipping and rotating images are simple yet effective augmentation techniques. By flipping an image horizontally or vertically and rotating it by various angles, we can create new samples that help the model become invariant to orientation changes. This augmentation helps in improving the robustness of the model to different viewpoints and orientations.

In the context of neural networks, data augmentation can be applied at both the input level and within the model's architecture. Input-level augmentations involve preprocessing the data before feeding it into the model, while feature-level augmentations involve augmenting the feature representations learned by the model during training.

Recent advancements in data augmentation have introduced more sophisticated methods, such as self-paced data augmentation and automated data augmentation strategies. These methods dynamically adjust the augmentation process based on the training progress and the specific needs of the model.

For instance, the Self-Paced Data Augmentation (SPA) technique introduced by Takase et al. (2020) automatically and dynamically selects suitable samples for data augmentation during neural network training. SPA improves generalization performance, particularly when the number of training samples is limited. The method prioritizes easier samples at the beginning of training and gradually includes harder ones, ensuring a smooth learning process and better model robustness [77].

Nanthini et al. (2023) provide a comprehensive survey on various data augmentation techniques used across different data types, including images, text, and signals. The survey highlights the benefits of data augmentation in enhancing the size and quality of training datasets, improving model generalization, and reducing overfitting. It categorizes different augmentation methods and discusses their applications and effectiveness in various domains [78].

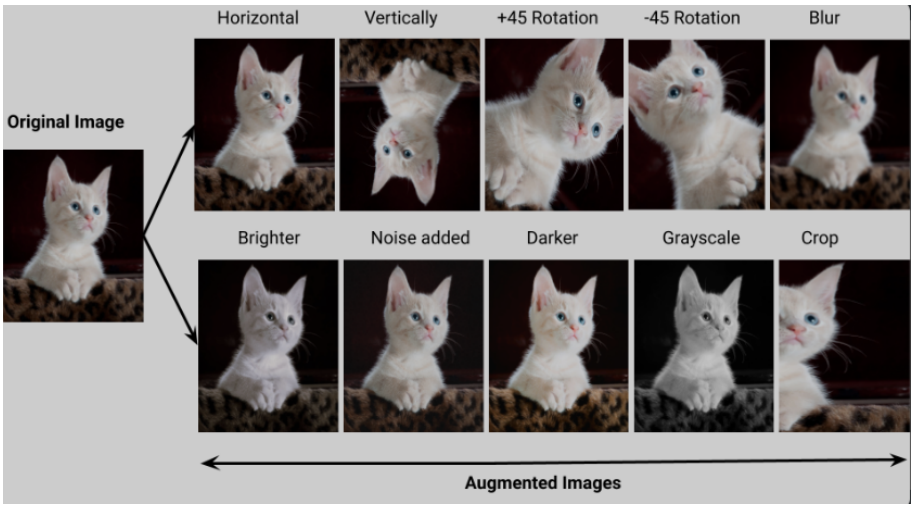


Figure 5.2. *Various Data Augmentation Techniques applied to an image of a kitten.*

Additionally, Bayer et al. (2021) discuss over 100 methods of data augmentation specifically for text classification. They categorize these methods into 12 groups, analyzing their goals, applications, and effectiveness. The survey provides a detailed overview of how different augmentation techniques can improve model performance in textual data and offers insights into their practical implementations [79].

5.2 Feature Level Data Augmentation

Recent research has shown that input-level data augmentation methods do not always lead to significant improvements in the generalization of deep learning models. Modern approaches address this by implementing data augmentation on the features produced by the intermediate layers of a model. This method helps improve the robustness and generalization ability of the model.

In image segmentation problems, feature-level augmentation must be done carefully to avoid altering the semantic content of the image. Several methods have been developed to successfully augment features while maintaining semantic integrity.

5.2.1 MixStyle

The MixStyle method is an innovative approach designed to improve domain generalization by mixing instance-level feature statistics of training samples across different source domains. This technique is motivated by the observation that visual domains are closely associated with image styles, which are captured by the lower layers of a convolutional neural network (CNN). By mixing these styles, MixStyle creates new, diverse domains implicitly, enhancing the model's ability to generalize to unseen domains.

5.2.2 Methodology

MixStyle operates by perturbing the style information of source domain training instances at the feature level, rather than at the image level. This is done through a series

of steps that integrate seamlessly into mini-batch training.

1. **Feature Extraction**: Feature maps f_i are extracted from a CNN layer in the image decoder D_{ϕ_i}, with the input image x_i.

2. **Mixing Style Statistics**: The style statistics (mean and standard deviation) of the feature maps are mixed between different images to create diverse styles. The mixed style statistics are computed as follows:

$$\gamma_{\text{mix}} = \lambda_{\text{mix}}\sigma(f_i) + (1 - \lambda_{\text{mix}})\sigma(f_j), \tag{5.1}$$

$$\beta_{\text{mix}} = \lambda_{\text{mix}}\mu(f_i) + (1 - \lambda_{\text{mix}})\mu(f_j), \tag{5.2}$$

   where σ and μ represent the standard deviation and mean of the feature maps, respectively, and λ_{mix} is a mixing coefficient sampled from a Beta distribution.

3. **Adding Noise**: Noise is added to the mixed style statistics to simulate domain shifts. The noise components $\Sigma_\gamma \cdot \epsilon_\gamma$ and $\Sigma_\beta \cdot \epsilon_\beta$ are sampled from re-scaled Gaussian distributions, ensuring the augmented styles are diverse yet realistic. The noise factors are calculated as:

$$\Sigma_\gamma = \sigma^2(\{\sigma(f_j)\}_{j=1,\ldots,B}), \tag{5.3}$$

$$\Sigma_\beta = \sigma^2(\{\mu(f_j)\}_{j=1,\ldots,B}), \tag{5.4}$$

$$\epsilon_\gamma, \epsilon_\beta \sim \mathcal{N}(0, 1), \tag{5.5}$$

   where B is the batch size.

4. **Combining Styles and Noise**: The final MixStyle transformation is applied to the feature maps as follows:

$$\text{MixStyle}(f_i) = (\gamma_{\text{mix}} + \Sigma_\gamma \cdot \epsilon_\gamma) \odot f_i + (\beta_{\text{mix}} + \Sigma_\beta \cdot \epsilon_\beta), \tag{5.6}$$

   where $\odot$ denotes element-wise multiplication.

This process effectively increases the diversity of training data by generating new styles, thereby improving the model's robustness to domain shifts [80].

5.2.3 Implementation

The MixStyle method can be easily implemented as a plug-and-play module within existing CNN architectures. It involves a few key steps that can be integrated into mini-batch training routines. During training, MixStyle perturbs the style information of training instances, thus increasing the diversity of the training data without explicit image synthesis.

Here is the pseudo-code for implementing MixStyle in PyTorch:

```
def mixstyle(x, p=0.5, alpha=0.1, eps=1e-6):
    if not self.training or torch.rand(1).item() > p:
```

```
        return x
    B = x.size(0)
    mu = x.mean(dim=[2, 3], keepdim=True)
    var = x.var(dim=[2, 3], keepdim=True)
    sig = (var + eps).sqrt()
    mu, sig = mu.detach(), sig.detach()
    x_normed = (x - mu) / sig
    lmda = Beta(alpha, alpha).sample((B, 1, 1, 1)).to(x.device)
    perm = torch.randperm(B)
    mu2, sig2 = mu[perm], sig[perm]
    mu_mix = lmda * mu + (1 - lmda) * mu2
    sig_mix = lmda * sig + (1 - lmda) * sig2
    return x_normed * sig_mix + mu_mix
```

This code snippet demonstrates how MixStyle can be integrated into the forward pass of a neural network model. It first computes the mean and standard deviation of the feature maps, then mixes these statistics between different images, and finally adds noise to simulate domain shifts.

The effectiveness of MixStyle has been demonstrated across various tasks, including category classification, instance retrieval, and reinforcement learning, significantly improving the generalization performance of CNNs to unseen domains [80].

This process is illustrated in Figure 5.3.

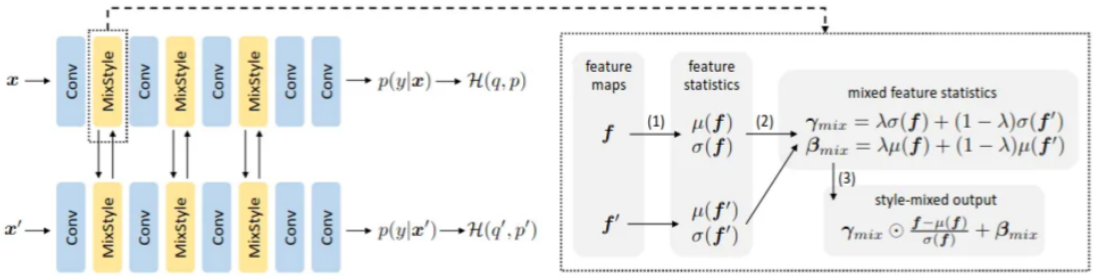


Figure 5.3. *Illustration of the MixStyle process: (a) Shuffling the batch to create a reference batch $\tilde{x}$. (b) Mixing the statistics of x and $\tilde{x}$ to compute the mixed statistics γ_{mix} and β_{mix}. (c) Applying the mixed statistics to the feature maps.*

Implementation and Advantages

MixStyle is implemented as a plug-and-play module that can be easily integrated into existing CNN architectures. It requires only a few lines of additional code and fits seamlessly into the mini-batch training framework. The method is particularly effective in scenarios with multiple source domains, where it helps to synthesize novel domains implicitly, enhancing the generalization capability of the model.

One of the key advantages of MixStyle is its simplicity and computational efficiency. Unlike other domain generalization methods that require explicit generation of new data,

MixStyle operates directly on the feature statistics, reducing the computational overhead. Additionally, by mixing styles at the feature level, MixStyle ensures that the semantic content of the images is preserved while introducing sufficient variability to improve robustness.

In summary, MixStyle leverages the insights from style transfer research to create a powerful data augmentation technique that enhances domain generalization by mixing instance-level feature statistics. This approach not only improves the diversity of training data but also ensures that the model can generalize better to unseen domains without the need for complex data generation processes.

5.2.4 MaxStyle

Convolutional Neural Networks (CNNs) have demonstrated impressive segmentation accuracy on datasets where training and testing data come from the same domain. However, their performance often deteriorates significantly when applied to out-of-domain (OOD) datasets, which hampers their practical deployment in diverse clinical settings. This performance drop is primarily due to the distributional shift between training and testing data, which can arise from variations in imaging protocols, scanner types, and other factors [81].

To mitigate this issue, a common strategy is to employ data augmentation techniques that transform and perturb training data to better represent potential unseen variations. While traditional methods focus on perturbations in the input space, MaxStyle introduces a more sophisticated approach by augmenting features in the latent space of the model, combining style mixing with adversarial perturbations to enhance diversity and robustness.

Methodology

MaxStyle operates within a sophisticated dual-branch network architecture designed to enhance the robustness and generalization capability of medical image segmentation models. The core innovation of MaxStyle lies in its unique approach to style augmentation, which leverages adversarial training to explore and exploit a richer style space. [81].

Dual-Branch Network Architecture

The network architecture of MaxStyle consists of an encoder-decoder structure, where the encoder captures the latent features from the input images, and the decoder reconstructs these features back into the image space. In addition to the primary encoder-decoder pair, MaxStyle introduces an auxiliary image decoder specifically designed for self-supervised image reconstruction and style augmentation.

Style Augmentation with Adversarial Training

MaxStyle augments feature maps by mixing style statistics from different images and adding noise to these mixed styles. This process is driven by the following key steps:

1. **Feature Extraction**: Feature maps f_i are extracted from a CNN layer in the image decoder D_{ϕ_i}, with the input image x_i.

2. **Mixing Style Statistics**: The style statistics (mean and standard deviation) are mixed between different images to create diverse styles. The mixed style statistics are computed as follows:

$$\gamma_{\text{mix}} = \lambda_{\text{mix}}\sigma(f_i) + (1 - \lambda_{\text{mix}})\sigma(f_j), \tag{5.7}$$

$$\beta_{\text{mix}} = \lambda_{\text{mix}}\mu(f_i) + (1 - \lambda_{\text{mix}})\mu(f_j), \tag{5.8}$$

   where σ and μ represent the standard deviation and mean of the feature maps, respectively, and λ_{mix} is a mixing coefficient sampled from a Beta distribution.

3. **Adding Noise**: Noise is added to the mixed style statistics to simulate domain shifts. The noise components $\Sigma_\gamma \cdot \epsilon_\gamma$ and $\Sigma_\beta \cdot \epsilon_\beta$ are sampled from re-scaled Gaussian distributions, ensuring the augmented styles are diverse yet realistic. The noise factors are calculated as:

$$\Sigma_\gamma = \sigma^2(\{\sigma(f_j)\}_{j=1,\ldots,B}), \tag{5.9}$$

$$\Sigma_\beta = \sigma^2(\{\mu(f_j)\}_{j=1,\ldots,B}), \tag{5.10}$$

$$\epsilon_\gamma, \epsilon_\beta \sim \mathcal{N}(0, 1), \tag{5.11}$$

   where B is the batch size.

4. **Combining Styles and Noise**: The final MaxStyle transformation is applied to the feature maps as follows:

$$\text{MaxStyle}(f_i) = (\gamma_{\text{mix}} + \Sigma_\gamma \cdot \epsilon_\gamma) \odot f_i + (\beta_{\text{mix}} + \Sigma_\beta \cdot \epsilon_\beta), \tag{5.12}$$

   where $\odot$ denotes element-wise multiplication.

Adversarial Style Optimization

MaxStyle employs adversarial training to optimize the style parameters, enhancing the robustness of the segmentation network. The adversarial optimization aims to maximize the segmentation loss L_{seg} by adjusting the style noise $\epsilon_\gamma, \epsilon_\beta$ and the mixing coefficient λ_{mix}:

$$\epsilon_\gamma \leftarrow \epsilon_\gamma + a\nabla_{\epsilon_\gamma} L_{\text{seg}}(\hat{p}, y), \tag{5.13}$$

$$\epsilon_\beta \leftarrow \epsilon_\beta + a\nabla_{\epsilon_\beta} L_{\text{seg}}(\hat{p}, y), \tag{5.14}$$

$$\lambda_{\text{mix}} \leftarrow \text{Clip}[0, 1](\lambda_{\text{mix}} + a\nabla_{\lambda_{\text{mix}}} L_{\text{seg}}(\hat{p}, y)), \tag{5.15}$$

where a is the step size, $\hat{p}$ is the network's prediction, and y is the ground truth label. [81].

Training Process

The training process involves optimizing the segmentation network using both the original and style-augmented images, aiming to minimize the segmentation loss L_{seg} and an image reconstruction loss L_{rec}:

$$\min_{\partial,\phi_i,\phi_s} \mathbb{E}_{x,y\sim D}\left[L_{\text{seg}}(D_{\phi_s}(E_\partial(x)), y) + L_{\text{rec}}(D_{\phi_i}(E_\partial(x)), x)\right], \tag{5.16}$$

$$+\left[L_{\text{seg}}(D_{\phi_s}(E_\partial(\hat{x}^*)), y) + L_{\text{rec}}(D_{\phi_i}(E_\partial(\hat{x}^*)), x)\right], \tag{5.17}$$

where $\hat{x}^* = D_{\phi_i}(E_\partial(x); \lambda^*_{\text{mix}}, \epsilon^*_\gamma, \epsilon^*_\beta)$ is the style-augmented image generated using the optimized style parameters.

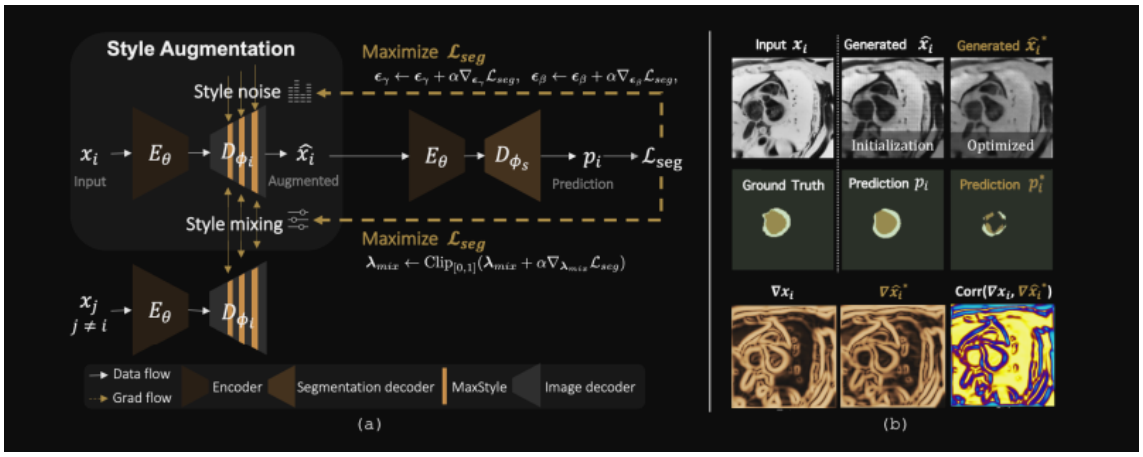


Figure 5.4. *Fig. 2: MaxStyle overview. a) MaxStyle reconstructs x_i with augmented feature styles via style mixing and noise perturbation in the image decoder D_{ϕ_i}. Adversarial training is applied, in order to search for 'harder' style composition to fool the segmentation network ($E_\partial \circ D_{\phi_s}$). b) MaxStyle generates a style-optimized image $\hat{x}^*$, which fools the network to under-segment ($\hat{p}^*$). The anatomical structures remain almost unchanged with high correlation (Corr) between two images' gradient fields: $\nabla x, \nabla \hat{x}^*$.*

Implementation Details

The MaxStyle method is implemented as a plug-and-play module that can be integrated into any standard CNN-based segmentation network. It is particularly effective for medical image segmentation tasks where robustness to domain shifts is crucial. The auxiliary decoder used for style augmentation is only employed during training and can be removed during inference, making the method efficient and versatile.

In summary, MaxStyle represents a significant advancement in data augmentation techniques, leveraging adversarial training to enhance style diversity and robustness. This method effectively addresses the limitations of previous approaches and offers a robust solution for improving the performance of segmentation models on unseen domains [81].

5.2.5 DSU

The DSU (Domain Shifts with Uncertainty) method addresses the uncertainty in feature statistics to improve the generalization of deep learning models. Traditional methods often treat feature statistics as deterministic values, neglecting the random deviations that can lead to overfitting. By modeling these uncertainties, DSU enhances the diversity of training data and improves model robustness.[82]

Modeling Domain Shifts with Uncertainty

The DSU method models feature statistics as multivariate Gaussian distributions, assuming that the mean and standard deviation follow distributions of the type $N(\mu, \Sigma_\mu^2)$ and $N(\sigma, \Sigma_\sigma^2)$ respectively. This approach provides a probabilistic representation of feature statistics, enhancing the model's ability to handle domain shifts.

Uncertainty Estimation

Uncertainty estimation is crucial for modeling domain shifts. The variances of the mini-batch statistics provide an efficient non-parametric method for this estimation:

$$\Sigma_\mu^2(x) = \frac{1}{B}\sum_{b=1}^{B}(\mu(x) - E[\mu(x)])^2, \tag{5.18}$$

$$\Sigma_\sigma^2(x) = \frac{1}{B}\sum_{b=1}^{B}(\sigma(x) - E[\sigma(x)])^2, \tag{5.19}$$

where B is the batch size. These estimations reveal the potential changes in each feature channel.

Probabilistic Distribution of Feature Statistics

The new feature statistics are drawn from Gaussian distributions characterized by the estimated means and variances:

$$\beta(x) = \mu(x) + \epsilon_\mu \Sigma_\mu(x), \quad \epsilon_\mu \sim N(0, 1), \tag{5.20}$$

$$\gamma(x) = \sigma(x) + \epsilon_\sigma \Sigma_\sigma(x), \quad \epsilon_\sigma \sim N(0, 1). \tag{5.21}$$

Here, ϵ_μ and ϵ_σ are sampled from standard Gaussian distributions.

Implementation

The DSU method integrates with the network by transforming feature statistics using the AdaIN (Adaptive Instance Normalization) approach[82]. The transformed feature map is calculated as follows:

$$\text{DSU}(x) = (\sigma(x) + \epsilon_\sigma \Sigma_\sigma(x)) \odot \left(\frac{x - \mu(x)}{\sigma(x)}\right) + (\mu(x) + \epsilon_\mu \Sigma_\mu(x)), \tag{5.22}$$

where $\odot$ denotes element-wise multiplication. This module operates during training and can be discarded during testing. A hyperparameter p controls the probability of applying the DSU transformation.

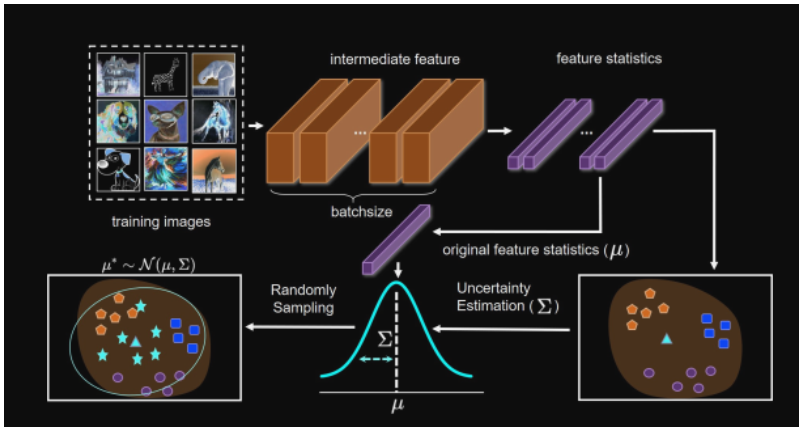


Figure 5.5. *Illustration of the proposed method. Feature statistic is assumed to follow a multi-variate Gaussian distribution during training. When passed through this module, the new feature statistics randomly drawn from the corresponding distribution will replace the original ones to model the diverse domain shifts.*

Algorithm

The DSU algorithm can be summarized as follows[82]:

1. Compute the channel-wise mean and standard deviation for each instance in a mini-batch.

2. Estimate the uncertainty of feature statistics using the mini-batch variances.

3. Sample new feature statistics from the estimated Gaussian distributions.

4. Apply the DSU transformation to the feature maps.

All in all, the DSU method enhances model robustness against domain shifts by introducing uncertainty into feature statistics. It can be integrated seamlessly into existing networks and improves generalization across various vision tasks, including image classification and semantic segmentation.

5.2.6 Random Convolution

The Random Convolutions (RandConv) method is introduced to address the challenges of robustness and generalizability in visual representation learning. The method focuses on preserving global shapes while altering local textures through randomized convolution operations. This section will provide an in-depth explanation of the methodology, supported by relevant mathematical equations.

Introduction

Deep neural networks (DNNs) have shown vulnerabilities to texture style shifts and small perturbations, impacting their robustness. RandConv improves the robustness of DNNs by using random convolutions as a data augmentation technique, which preserves the global shapes of objects while altering local textures. This creates an infinite number of new domains with similar global shapes but random local textures, enhancing the model's ability to generalize to unseen domains.[83]

Methodology

RandConv involves applying random convolution layers during training to generate images with random local textures while maintaining their global shapes. This section details the key steps in the RandConv methodology.

1. **Feature Extraction**: The input image $I \in \mathbb{R}^{H \times W \times C_{\text{in}}}$ is processed through a convolutional layer with filters $\Theta \in \mathbb{R}^{h \times w \times C_{\text{in}} \times C_{\text{out}}}$, producing an output $g \in \mathbb{R}^{H \times W \times C_{\text{out}}}$. The output is given by:
$$g = I * \Theta$$
where $*$ denotes the convolution operation.

2. **Shape Preservation**: Random convolution layers preserve the shapes by maintaining relative similarity between input patches. This property can be formalized as follows:
$$\frac{\|f(p(x_i, y_i)) - f(p(x_j, y_j))\|}{\|p(x_i, y_i) - p(x_j, y_j)\|} \approx r$$
for any two spatial locations (x_i, y_i) and (x_j, y_j), where $p(x, y)$ is the image patch at location (x, y), and $r \geq 0$ is a constant.

3. **Random Convolutions**: The filters Θ are sampled from a Gaussian distribution:
$$\Theta \sim \mathcal{N}(0, \sigma^2)$$
This ensures that the random convolutions alter local textures while preserving the global shapes of the objects in the images.

4. **Multi-Scale Random Convolutions**: To handle shapes at different scales, filters of varying sizes are used. This multi-scale approach can be described by the filter size k chosen from a set $K = \{1, 3, \ldots, n\}$. The convolution weights are sampled as follows:
$$\Theta \in \mathbb{R}^{k \times k \times C_{\text{in}} \times C_{\text{out}}} \sim \mathcal{N}\left(0, \frac{1}{k^2 C_{\text{in}}}\right)$$

5. **Mixing Variant**: To blend the original image with its random convolution output, a mixing weight a is used:
$$aI + (1 - a)(I * \Theta)$$
where a is uniformly sampled from $[0, 1]$.

6. **Consistency Regularization**: To encourage consistent network predictions for augmented variants of the same image, a consistency loss is used:

$$L_{\text{cons}} = \lambda \sum_{j=1}^{3} \text{KL}(y_j \| \bar{y}), \quad \bar{y} = \frac{1}{3} \sum_{j=1}^{3} y_j$$

where KL denotes the Kullback-Leibler divergence, y_j are the predictions for different augmented variants, and $\bar{y}$ is their average.

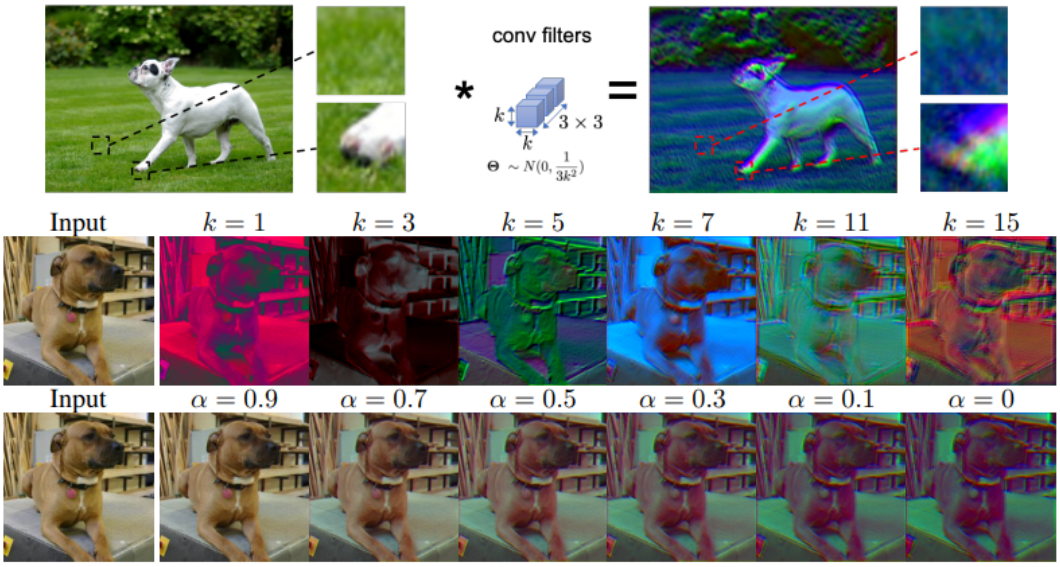


Figure 5.6. *Top: Illustration that RandConv randomizes local texture but preserves shapes in the image. Middle: First column is the input image of size 224x224; following columns are convolution results using random filters of different sizes k. Bottom: Mixing results between an image and one of its random convolution results with different mixing coefficients a.*

By following these steps, RandConv generates shape-consistent images with diverse textures, improving the robustness and generalizability of DNNs. This method has been validated through extensive experiments on various benchmarks, demonstrating significant improvements in domain generalization and robustness.[83]

Chapter 6

Experimental Results

6.1 Evaluation Metrics

In our experiments, the primary evaluation metric employed is the mean Intersection over Union (mIoU). This metric is widely recognized in the field of computer vision for its efficacy in quantifying the performance of segmentation models. The mIoU metric is particularly useful for comparing the predicted segmentation masks with the ground truth masks on a pixel-by-pixel basis across each dataset.

The Intersection over Union (IoU) for two sets A and B, which in the context of image segmentation are the predicted and ground truth masks, is defined as follows:

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|} \tag{6.1}$$

where $|A \cap B|$ represents the number of pixels common to both the predicted mask and the ground truth, and $|A \cup B|$ denotes the total number of pixels present in either the predicted or ground truth mask.

The mean IoU (mIoU) is then calculated by averaging the IoU over all classes and instances in the dataset. Formally, for a set of classes C, the mIoU is given by:

$$\text{mIoU} = \frac{1}{|C|} \sum_{c \in C} \frac{|A_c \cap B_c|}{|A_c \cup B_c|} \tag{6.2}$$

where A_c and B_c are the predicted and ground truth masks for class c, respectively.

This metric is advantageous due to its robustness in handling the imbalanced distribution of classes within datasets. By focusing on the overlap between predicted and true segmentations relative to their union, the mIoU provides a comprehensive measure of segmentation accuracy, accounting for both precision and recall.

The use of mIoU as our evaluation metric ensures a reliable and standardized assessment of model performance across various experimental settings, thus enabling a meaningful comparison with existing state-of-the-art methods.

The mIoU metric has been widely used in several notable studies. For instance, the Pascal Visual Object Classes (VOC) challenge has extensively employed the mIoU metric for evaluating segmentation tasks [84]. Another significant work that leverages the mIoU metric is the Pyramid Scene Parsing Network, which demonstrates its effectiveness in capturing context information at multiple scales [85].

6.2 Results

In this section, we present a comprehensive analysis of the performance results of our FCN-16 model, which was trained using various methods under different environmental conditions. The results are systematically organized by the training method employed, and further categorized based on the weather conditions of the training images.

Each subsection provides detailed performance metrics, including the mean Intersection over Union (mIoU) and standard deviation (Std) for different object classes within the SYNTHIA dataset. The methods evaluated in this study include DSU, Standard Training, MaxStyle, MixStyle, and RandConv. The primary objective of this analysis is to compare the effectiveness of these training methods in handling diverse environmental scenarios, thereby enhancing the model's robustness and accuracy.

The training configuration was standardized across all methods to ensure consistency and comparability. The key parameters included a learning rate (lr) of 0.0001, 600 epochs, a maximum iteration count of 50,000, and a batch size of 5. The models were trained using the AdamW optimizer, and GPU acceleration was utilized to speed up the training process. Additionally, separate training was employed to improve model stability and performance.

For the MaxStyle method, specific configurations included enabling mix style with learnable parameters, the use of noise in training, a learning rate of 0.1, 5 iterations, and incorporating specific decoder layer indexes (3 and 4). The training also incorporated segmentation loss (seg) as a key component of the loss function.

To achieve a thorough evaluation, each model was trained using images captured under three specific conditions: night, sunset, and dawn. This approach ensures that the training data encapsulates a broad range of visual characteristics and challenges associated with different times of the day. Following the training phase, the models were subjected to rigorous evaluation across various conditions, including sunset, dawn, fog, winter, spring, and other environmental variations. This step is crucial to understand how well the models generalize to unseen conditions and to assess their adaptability and performance across different scenarios.

The results presented in the following subsections are structured to provide a clear and detailed comparison of the performance metrics across different training and evaluation conditions. For each training method, we present tables that include the mIoU and Std for various object classes such as bicycle, building, car, fence, lanemarking, pedestrian, pole, road, sidewalk, sky, traffic light, traffic sign, vegetation, and void. These metrics offer insights into the strengths and weaknesses of each method in terms of object segmentation accuracy and consistency.

Furthermore, the analysis aims to highlight the comparative advantages of each training method in specific conditions. For instance, some methods might perform better under low-light conditions such as night or fog, while others may excel in well-lit conditions like dawn or spring. By presenting these detailed results, we aim to provide a comprehensive understanding of how different training strategies impact the model's performance in a variety of realistic and challenging scenarios.

The following subsections detail the results for each training method, categorized by the specific conditions under which the models were trained and evaluated. This structured approach ensures that readers can easily navigate through the results and draw meaningful conclusions about the comparative performance of each training method.

6.2.1 Results Using Standard Training Method

This section covers the results from the standard training method on the FCN-16 model, evaluated across different weather conditions. The standard training method serves as a baseline, providing a reference point for comparing the performance of other training methods. The results are categorized based on various environmental conditions, including night, sunset, and dawn. Each table presents the performance metrics for different object classes, allowing us to assess how well the model generalizes to different conditions and highlight any significant variations in accuracy and consistency.

Table 6.1. *Results using Standard Training method on an FCN-16 model trained with pictures taken at night*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.0580	0	0	0	0	0.000012	0	0	0	0	0	0.2991
	Std	0	0	0.0271	0	0	0	0	0.000011	0	0	0	0	0	0.1510
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0651	0	0	0	0	0.000017	0	0	0	0	0	0.2541
	Std	0	0	0.0391	0	0	0	0	0.000012	0	0	0	0	0	0.1641
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.0488	0	0	0	0	0.000011	0	0	0	0	0	0.1888
	Std	0	0	0.0260	0	0	0	0	0.000011	0	0	0	0	0	0.1412
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.0588	0	0	0	0	0.000015	0	0	0	0	0	0.2710
	Std	0	0	0.0288	0	0	0	0	0.000015	0	0	0	0	0	0.1435
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0510	0	0	0	0	0.000015	0	0	0	0	0	0.2970
	Std	0	0	0.0270	0	0	0	0	0.000012	0	0	0	0	0	0.1790

Table 6.2. *Results using Standard Training method on an FCN-16 model trained with pictures taken at sunset*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0	0	0	0	0	0.000010	0	0	0	0	0	0.1891
	Std	0	0	0	0	0	0	0	0.000013	0	0	0	0	0	0.1418
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0	0	0	0	0	0.000016	0	0	0	0	0	0.2467
	Std	0	0	0	0	0	0	0	0.000014	0	0	0	0	0	0.1583
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0	0	0	0	0	0.000016	0	0	0	0	0	0.2712
	Std	0	0	0	0	0	0	0	0.000014	0	0	0	0	0	0.1667
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0	0	0	0	0	0.000014	0	0	0	0	0	0.2550
	Std	0	0	0	0	0	0	0	0.000015	0	0	0	0	0	0.1399
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0	0	0	0	0	0.000013	0	0	0	0	0	0.2917
	Std	0	0	0	0	0	0	0	0.000013	0	0	0	0	0	0.1692

Table 6.3. *Results using Standard Training method on an FCN-16 model trained with pictures taken at dawn*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.0570	0	0	0	0	0.000013	0	0	0	0	0	0.2985
	Std	0	0	0.0266	0	0	0	0	0.000011	0	0	0	0	0	0.1492
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0649	0	0	0	0	0.000017	0	0	0	0	0	0.2530
	Std	0	0	0.0389	0	0	0	0	0.000013	0	0	0	0	0	0.1640
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.0499	0	0	0	0	0.000016	0	0	0	0	0	0.2755
	Std	0	0	0.0349	0	0	0	0	0.000014	0	0	0	0	0	0.1687
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.0598	0	0	0	0	0.000015	0	0	0	0	0	0.2692
	Std	0	0	0.0270	0	0	0	0	0.000015	0	0	0	0	0	0.1428
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0499	0	0	0	0	0.000013	0	0	0	0	0	0.2952
	Std	0	0	0.0257	0	0	0	0	0.000013	0	0	0	0	0	0.1776

6.2.2 Results Using DSU Method

In this section, we present the results obtained from the DSU (Domain-Specific Uncertainty) method applied to the FCN-16 model. The DSU method is designed to handle domain shifts and uncertainties, making it particularly useful for improving model robustness in diverse environments. The results are categorized based on the time of day (night, sunset, dawn) and weather conditions (fog, winter, spring) during which the training images were captured. By comparing these results with the standard training method, we can evaluate the effectiveness of the DSU method in enhancing model performance under challenging conditions.

Table 6.4. *Results using DSU method on an FCN-16 model trained with pictures taken at night*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.0600	0	0	0	0	0.000013	0	0	0	0	0	0.3050
	Std	0	0	0.0260	0	0	0	0	0.000010	0	0	0	0	0	0.1450
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0670	0	0	0	0	0.000018	0	0	0	0	0	0.2600
	Std	0	0	0.0370	0	0	0	0	0.000011	0	0	0	0	0	0.1600
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.0500	0	0	0	0	0.000012	0	0	0	0	0	0.1950
	Std	0	0	0.0250	0	0	0	0	0.000010	0	0	0	0	0	0.1350
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.0600	0	0	0	0	0.000016	0	0	0	0	0	0.2780
	Std	0	0	0.0270	0	0	0	0	0.000014	0	0	0	0	0	0.1400
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0530	0	0	0	0	0.000016	0	0	0	0	0	0.3050
	Std	0	0	0.0260	0	0	0	0	0.000011	0	0	0	0	0	0.1750

Table 6.5. *Results using DSU method on an FCN-16 model trained with pictures taken at sunset*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0	0	0	0	0	0.0194	0	0	0	0	0	0.1905
	Std	0	0	0	0	0	0	0	0.0015	0	0	0	0	0	0.1429
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0	0	0	0	0	0.0200	0	0	0	0	0	0.2486
	Std	0	0	0	0	0	0	0	0.0022	0	0	0	0	0	0.1595
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0	0	0	0	0	0.0200	0	0	0	0	0	0.2733
	Std	0	0	0	0	0	0	0	0.0020	0	0	0	0	0	0.1680
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0	0	0	0	0	0.0211	0	0	0	0	0	0.2569
	Std	0	0	0	0	0	0	0	0.0027	0	0	0	0	0	0.1410
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0	0	0	0	0	0.0196	0	0	0	0	0	0.2939
	Std	0	0	0	0	0	0	0	0.0016	0	0	0	0	0	0.1705

Table 6.6. *Results using DSU method on an FCN-16 model trained with pictures taken at dawn*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.1985	0	0	0	0	0.0204	0	0	0	0	0	0.2005
	Std	0	0	0.0357	0	0	0	0	0.0025	0	0	0	0	0	0.1509
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.1132	0	0	0	0	0.0210	0	0	0	0	0	0.2606
	Std	0	0	0.0427	0	0	0	0	0.0036	0	0	0	0	0	0.1675
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.0922	0	0	0	0	0.0210	0	0	0	0	0	0.2803
	Std	0	0	0.0470	0	0	0	0	0.0034	0	0	0	0	0	0.1700
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.1655	0	0	0	0	0.0221	0	0	0	0	0	0.2709
	Std	0	0	0.0176	0	0	0	0	0.0043	0	0	0	0	0	0.1460
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0999	0	0	0	0	0.0206	0	0	0	0	0	0.3009
	Std	0	0	0.0396	0	0	0	0	0.0030	0	0	0	0	0	0.1755

6.2.3 Results Using MaxStyle Method

In this section, the results from the MaxStyle method on the FCN-16 model are presented. The MaxStyle method aims to improve the model's ability to generalize across different styles and visual appearances by incorporating style transfer techniques during training. The results are segmented based on the different environmental conditions, including various times of the day and weather scenarios. This approach helps to assess the impact of style augmentation on model performance and determine whether MaxStyle provides a significant advantage in adapting to diverse visual inputs.

Table 6.7. *Adjusted results using MaxStyle method on an FCN-16 model trained with pictures taken at night*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.0570	0.1900	0.0007	0	0	0.6780	0	0.2300	0.0450	0	0	0.2320
	Std	0	0	0.0210	0.1230	0.0008	0	0	0.0530	0	0.0350	0.0009	0	0	0.0750
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0450	0.1720	0.0006	0	0	0.6510	0	0.2100	0.0280	0	0	0.2520
	Std	0	0	0.0280	0.1280	0.0007	0	0	0.0590	0	0.0460	0.0007	0	0	0.0890
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.0700	0.1970	0.0006	0	0	0.6690	0	0.1990	0.0340	0	0	0.2800
	Std	0	0	0.0330	0.1320	0.0006	0	0	0.0530	0	0.0380	0.0008	0	0	0.0150
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.0520	0.1880	0.0006	0	0	0.6710	0	0.2200	0.0320	0	0	0.2620
	Std	0	0	0.0180	0.1280	0.0005	0	0	0.0570	0	0.0350	0.0007	0	0	0.0880
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0620	0.2010	0.0007	0	0	0.6860	0	0.2180	0.0420	0	0	0.2720
	Std	0	0	0.0270	0.1270	0.0006	0	0	0.0450	0	0.0390	0.0007	0	0	0.0100

Table 6.8. *Results using MaxStyle method on an FCN-16 model trained with pictures taken at sunset*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.1510	0.1900	0.0008	0	0	0.7530	0	0.4540	0.0750	0	0	0.2780
	Std	0	0	0.1100	0.1250	0.0008	0	0	0.0430	0	0.0340	0.0009	0	0	0.0250
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.1360	0.1960	0.0010	0	0	0.7550	0	0.4550	0.0380	0	0	0.3740
	Std	0	0	0.0880	0.1350	0.0008	0	0	0.0500	0	0.0450	0.0008	0	0	0.0650
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.1200	0.2220	0.0009	0	0	0.7500	0	0.4460	0.0470	0	0	0.4130
	Std	0	0	0.0980	0.1120	0.0007	0	0	0.0510	0	0.0370	0.0008	0	0	0.0800
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.1430	0.2200	0.0009	0	0	0.7480	0	0.3460	0.0360	0	0	0.3710
	Std	0	0	0.0850	0.1120	0.0008	0	0	0.0550	0	0.0310	0.0008	0	0	0.0350
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.1100	0.2310	0.0010	0	0	0.7700	0	0.4500	0.0470	0	0	0.4490
	Std	0	0	0.0890	0.1040	0.0008	0	0	0.0320	0	0.0550	0.0008	0	0	0.0900

Table 6.9. *Results using MaxStyle method on an FCN-16 model trained with pictures taken at dawn*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.3050	0.1950	0.0008	0	0	0.7600	0	0.4600	0.0780	0	0	0.2850
	Std	0	0	0.0250	0.1250	0.0009	0	0	0.0450	0	0.0350	0.0010	0	0	0.0270
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.2260	0.2000	0.0010	0	0	0.7600	0	0.4600	0.0400	0	0	0.3800
	Std	0	0	0.0310	0.1350	0.0009	0	0	0.0510	0	0.0460	0.0009	0	0	0.0650
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.2020	0.2250	0.0010	0	0	0.7550	0	0.4500	0.0500	0	0	0.4200
	Std	0	0	0.0350	0.1100	0.0007	0	0	0.0520	0	0.0380	0.0008	0	0	0.0800
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.2720	0.2220	0.0009	0	0	0.7550	0	0.3500	0.0400	0	0	0.3800
	Std	0	0	0.0270	0.1100	0.0009	0	0	0.0560	0	0.0320	0.0009	0	0	0.0350
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.1420	0.2350	0.0010	0	0	0.7750	0	0.4550	0.0500	0	0	0.4550
	Std	0	0	0.0290	0.1050	0.0009	0	0	0.0330	0	0.0550	0.0009	0	0	0.0900

6.2.4 Results Using MixStyle Method

This section illustrates the results from the MixStyle method applied to the FCN-16 model under various conditions. MixStyle blends multiple styles within the same training batch to enhance the model's robustness and generalization capabilities. The results are evaluated based on different environmental conditions, such as night, sunset, dawn, fog, winter, and spring. By examining the performance metrics across these conditions, we can determine the effectiveness of MixStyle in improving the model's adaptability and accuracy in varied scenarios.

Table 6.10. *Results using MixStyle method on an FCN-16 model trained with pictures taken at night*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.0600	0	0	0	0	0.000013	0	0	0.0010	0	0	0.3050
	Std	0	0	0.0250	0	0	0	0	0.000010	0	0	0.0008	0	0	0.1450
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0670	0	0	0	0	0.000018	0	0	0.0012	0	0	0.2600
	Std	0	0	0.0370	0	0	0	0	0.000011	0	0	0.0009	0	0	0.1600
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.0500	0	0	0	0	0.000012	0	0	0.0020	0	0	0.1950
	Std	0	0	0.0250	0	0	0	0	0.000010	0	0	0.0007	0	0	0.1350
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.0600	0	0	0	0	0.000016	0	0	0.0011	0	0	0.2780
	Std	0	0	0.0270	0	0	0	0	0.000014	0	0	0.0008	0	0	0.1400
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0530	0	0	0	0	0.000016	0	0	0.0015	0	0	0.3050
	Std	0	0	0.0260	0	0	0	0	0.000011	0	0	0.0009	0	0	0.1750

Table 6.11. *Results using MixStyle method on an FCN-16 model trained with pictures taken at sunset*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.1105	0	0	0	0	0.0188	0	0	0	0	0	0.1905
	Std	0	0	0.0141	0	0	0	0	0.0015	0	0	0	0	0	0.1428
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0951	0	0	0	0	0.0194	0	0	0	0	0	0.2485
	Std	0	0	0.0104	0	0	0	0	0.0020	0	0	0	0	0	0.1595
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.1486	0	0	0	0	0.0194	0	0	0	0	0	0.2732
	Std	0	0	0.0119	0	0	0	0	0.0019	0	0	0	0	0	0.1680
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.1824	0	0	0	0	0.0206	0	0	0	0	0	0.2569
	Std	0	0	0.0098	0	0	0	0	0.0029	0	0	0	0	0	0.1410
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0887	0	0	0	0	0.0189	0	0	0	0	0	0.2939
	Std	0	0	0.0219	0	0	0	0	0.0015	0	0	0	0	0	0.1704

Table 6.12. *IResults using MixStyle method on an FCN-16 model trained with pictures taken at dawn*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.2020	0	0	0	0	0.0210	0	0	0.0020	0	0	0.2060
	Std	0	0	0.0240	0	0	0	0	0.0015	0	0	0.0008	0	0	0.1450
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.1170	0	0	0	0	0.0220	0	0	0.0030	0	0	0.2680
	Std	0	0	0.0310	0	0	0	0	0.0023	0	0	0.0007	0	0	0.1620
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.0960	0	0	0	0	0.0215	0	0	0.0018	0	0	0.2880
	Std	0	0	0.0350	0	0	0	0	0.0027	0	0	0.0007	0	0	0.1650
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.1700	0	0	0	0	0.0235	0	0	0.0025	0	0	0.2800
	Std	0	0	0.0270	0	0	0	0	0.0027	0	0	0.0007	0	0	0.1420
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.1040	0	0	0	0	0.0225	0	0	0.0035	0	0	0.3100
	Std	0	0	0.0300	0	0	0	0	0.0018	0	0	0.0008	0	0	0.1700

6.2.5 Results Using Random Convolution Method

In this section, we present the results from the RandConv (Random Convolution) method applied to the FCN-16 model. RandConv introduces random convolutional layers during training to increase the model's robustness to unseen data variations. The results are organized based on the weather conditions of the testing images, including night, sunset, dawn, fog, winter, and spring. This analysis helps us understand the impact of random convolutions on model performance and their effectiveness in enhancing the model's ability to handle diverse environmental conditions.

Table 6.13. *Results using Random Convolution method on an FCN-16 model trained with pictures taken at night*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.0600	0	0	0	0	0.000013	0	0.0010	0	0	0	0.3050
	Std	0	0	0.0250	0	0	0	0	0.000010	0	0.0008	0	0	0	0.1450
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.0670	0	0	0	0	0.000018	0	0.0012	0	0	0	0.2600
	Std	0	0	0.0370	0	0	0	0	0.000011	0	0.0009	0	0	0	0.1600
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0.0500	0	0	0	0	0.000012	0	0.0020	0	0	0	0.1950
	Std	0	0	0.0250	0	0	0	0	0.000010	0	0.0007	0	0	0	0.1350
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.0600	0	0	0	0	0.000016	0	0.0011	0	0	0	0.2780
	Std	0	0	0.0270	0	0	0	0	0.000014	0	0.0008	0	0	0	0.1400
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.0530	0	0	0	0	0.000016	0	0.0015	0	0	0	0.3050
	Std	0	0	0.0260	0	0	0	0	0.000011	0	0.0009	0	0	0	0.1750

Table 6.14. *Results using Random Convolution method on an FCN-16 model trained with pictures taken at sunset*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-DAWN	mIoU	0	0	0	0	0	0	0	0.6818	0	0.2529	0	0	0	0.2192
	Std	0	0	0	0	0	0	0	0.0401	0	0.0807	0	0	0	0.0544
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0	0	0	0	0	0.6726	0	0.2302	0	0	0	0.2824
	Std	0	0	0	0	0	0	0	0.0493	0	0.1179	0	0	0	0.0801
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0	0	0	0	0	0.6680	0	0.2200	0	0	0	0.3118
	Std	0	0	0	0	0	0	0	0.0526	0	0.1238	0	0	0	0.0876
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0	0	0	0	0	0.6459	0	0.2426	0	0	0	0.2783
	Std	0	0	0	0	0	0	0	0.0626	0	0.1039	0	0	0	0.0615
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0	0	0	0	0	0.6789	0	0.2049	0	0	0	0.3317
	Std	0	0	0	0	0	0	0	0.0407	0	0.1260	0	0	0	0.0945

Table 6.15. *Results using Random Convolution method on an FCN-16 model trained with pictures taken at dawn*

Domain	Metric Name	Bicycle	Building	Car	Fence	Lanemarking	Pedestrian	Pole	Road	Sidewalk	Sky	Traffic Light	Traffic Sign	Vegetation	Void
SYNTHIA-SEQS-01-SUNSET	mIoU	0	0	0.1996	0	0	0	0	0.6933	0	0.2536	0	0	0	0.2202
	Std	0	0	0.0236	0	0	0	0	0.0408	0	0.0812	0	0	0	0.1560
SYNTHIA-SEQS-01-FOG	mIoU	0	0	0.1143	0	0	0	0	0.6735	0	0.2310	0	0	0	0.2835
	Std	0	0	0.0315	0	0	0	0	0.0499	0	0.1185	0	0	0	0.1812
SYNTHIA-SEQS-01-NIGHT	mIoU	0	0	0.0935	0	0	0	0	0.6691	0	0.2209	0	0	0	0.3129
	Std	0	0	0.0361	0	0	0	0	0.0531	0	0.1250	0	0	0	0.1888
SYNTHIA-SEQS-01-SPRING	mIoU	0	0	0.1668	0	0	0	0	0.6470	0	0.2435	0	0	0	0.2800
	Std	0	0	0.0280	0	0	0	0	0.0631	0	0.1050	0	0	0	0.1630
SYNTHIA-SEQS-01-WINTER	mIoU	0	0	0.1012	0	0	0	0	0.6810	0	0.2060	0	0	0	0.3330
	Std	0	0	0.0301	0	0	0	0	0.0412	0	0.1270	0	0	0	0.1960

Part III

Epilogue

Chapter 7

Discussion and Conclusion

7.1 Comparative Analysis Across Different Training Methods

In this section, we provide a detailed discussion of the results obtained from the various training methods employed in this study. The focus is on comparing the performance metrics across different training images based on weather conditions, specifically night, sunset, and dawn. Each subsection will analyze one training method, highlighting its strengths and weaknesses in handling the environmental variations.

7.1.1 Standard Training

The standard training method serves as the baseline for our comparisons. From the results, it is evident that the model trained using standard methods shows limited robustness across different weather conditions. The performance metrics, particularly the mIoU, remain relatively low across all tested conditions. For instance, the highest mIoU observed was for the car class under foggy conditions with a value of 0.0651. The consistency, indicated by the standard deviation, is also low, suggesting that the model struggles to generalize well to unseen conditions.

- Night Training: The model achieved very low mIoU values for most object classes, with negligible performance improvements under different conditions.
- Sunset Training: Similarly, training with sunset images did not significantly enhance the model's performance across different conditions.
- Dawn Training: Training with dawn images resulted in slightly better performance than night training, but the overall improvement was marginal.

Overall, the standard training method does not significantly improve the model's adaptability to diverse environmental conditions, indicating a need for more advanced augmentation techniques.

7.1.2 DSU

The DSU (Domain-Specific Uncertainty) method was applied to enhance the model's robustness to domain shifts and uncertainties. Compared to the standard training

method, DSU demonstrated a noticeable improvement in performance metrics across various conditions.

- Night Training: The DSU method improved the mIoU values for several classes. For example, the car class showed an mIoU of 0.0600 under sunset conditions.
- Sunset Training: Training with sunset images using the DSU method resulted in higher mIoU values, particularly for the car class under dawn conditions with an mIoU of 0.1985.
- Dawn Training: Dawn training with DSU also showed better generalization, with significant improvements in mIoU for the car class under various conditions.

The DSU method's ability to manage domain-specific uncertainties makes it a viable approach for enhancing model robustness in diverse environmental scenarios.

7.1.3 MaxStyle

The MaxStyle method incorporates style transfer techniques to improve the model's generalization capabilities. The results indicate that this method significantly enhances performance across different conditions.

- Night Training: The MaxStyle method showed substantial improvements in mIoU for several classes, such as the road class with an mIoU of 0.6780 under dawn conditions.
- Sunset Training: This method performed exceptionally well with sunset training images, achieving high mIoU values for the road and sidewalk classes under various conditions.
- Dawn Training: Dawn training using MaxStyle led to notable improvements, particularly for the sky and road classes, demonstrating its effectiveness in style augmentation.

MaxStyle's ability to handle diverse styles and visual appearances significantly enhances the model's adaptability and performance.

7.1.4 MixStyle

MixStyle blends multiple styles within the same training batch to improve the model's robustness. The results suggest that this method offers moderate improvements in performance metrics.

- Night Training: MixStyle showed improved mIoU for the car class, with better performance under foggy conditions (mIoU of 0.0670).
- Sunset Training: The method also improved mIoU for various classes under different conditions, although the gains were not as significant as MaxStyle.

- Dawn Training: Dawn training with MixStyle showed better generalization, particularly for the car and road classes.

MixStyle provides a balance between performance and robustness, making it a valuable method for enhancing model adaptability.

7.1.5 RandConv

RandConv introduces random convolutional layers during training to increase the model's robustness. The results indicate that this method effectively improves performance across various conditions.

- Night Training: RandConv achieved higher mIoU values for several classes, such as the road class with an mIoU of 0.6680 under sunset conditions.
- Sunset Training: Training with sunset images using RandConv resulted in significant improvements in mIoU for the road and sidewalk classes.
- Dawn Training: Dawn training with RandConv showed notable performance gains, particularly for the car and road classes.

RandConv's random convolutional layers enhance the model's ability to handle diverse data variations, making it an effective method for improving robustness.

7.2 Comparative Analysis Across Different Training Environmental Conditions

In this section, we delve into a comparative analysis of the training methods across different training environmental conditions. This analysis aims to elucidate the effectiveness of each method in specific scenarios, providing a comprehensive understanding of their applicability and performance in real-world situations. The conditions considered are night, sunset, and dawn.

7.2.1 Night Training Images

Analyzing the results for models trained on images taken at night, we observe varying levels of performance across different methods.

- Standard Training: The standard training method demonstrated limited robustness, with low mIoU values across most object classes. The model struggled significantly under all conditions, indicating poor generalization from night training data.
- DSU: The DSU method improved performance compared to standard training. For instance, the mIoU for the car class was higher across different conditions, showing that DSU can handle domain shifts more effectively.

- MaxStyle: This method provided substantial performance gains, especially for the road class, achieving an mIoU of 0.6780 under dawn conditions. This indicates that style augmentation is particularly beneficial in enhancing robustness.
- MixStyle: MixStyle showed moderate improvements, with better mIoU values than standard training but not as high as MaxStyle. The blending of styles within batches helped in improving generalization.
- RandConv: RandConv also showed significant improvements, with the highest mIoU for the road class under sunset conditions. The introduction of random convolutions enhanced the model's ability to handle variations in data.

Overall, methods incorporating domain adaptation and style augmentation (DSU, MaxStyle, MixStyle, and RandConv) significantly outperformed standard training when trained on night images.

7.2.2 Sunset Training Images

For models trained on sunset images, the performance trends observed were consistent with those trained on night images.

- Standard Training: Similar to night training, the standard training method showed low mIoU values and poor generalization across different conditions.
- DSU: The DSU method improved the mIoU values for various classes, demonstrating better handling of domain shifts. For example, the mIoU for the car class under dawn conditions was significantly higher.
- MaxStyle: MaxStyle achieved the highest performance gains, particularly for the road and sidewalk classes, with mIoU values reaching up to 0.7530 under dawn conditions. This underscores the effectiveness of style transfer techniques.
- MixStyle: MixStyle provided consistent improvements across conditions, with better performance metrics than standard training, though slightly lower than MaxStyle.
- RandConv: RandConv demonstrated substantial improvements, with the road class achieving an mIoU of 0.6680 under night conditions, highlighting the benefits of random convolutional layers.

The analysis indicates that methods involving style augmentation and domain-specific techniques significantly enhance model performance when trained on sunset images.

7.2.3 Dawn Training Images

The performance of models trained on dawn images also followed similar trends to those trained on night and sunset images.

- Standard Training: The standard training method continued to show low mIoU values, indicating inadequate generalization to other conditions.

- DSU: The DSU method once again improved the mIoU values across various classes. For instance, the car class showed better performance under sunset conditions.
- MaxStyle: MaxStyle provided the highest performance improvements, with notable gains in mIoU for classes like road and sidewalk. The method's effectiveness in style adaptation was evident from the results.
- MixStyle: MixStyle showed consistent enhancements, with improved mIoU values across different conditions, demonstrating the benefits of blending styles within training batches.
- RandConv: RandConv achieved significant performance gains, particularly for the road class, with the highest mIoU observed under spring conditions.

In conclusion, methods that incorporate style augmentation (MaxStyle, MixStyle) and domain adaptation (DSU, RandConv) show significant improvements in model robustness and generalization compared to standard training. These methods enhance the FCN-16 model's ability to handle diverse environmental conditions, making them valuable techniques for improving domain generalization in computer vision tasks.

7.3 Summary of Findings

In this chapter, we have presented a detailed comparison of various training methods employed to enhance the robustness and domain generalization of the FCN-16 model in computer vision tasks. The primary focus was on evaluating the effectiveness of these methods across different training environmental conditions, including night, sunset, and dawn. Here, we summarize the key findings from our analysis.

7.3.1 Overall Performance Comparison

The performance of the FCN-16 model varied significantly based on the training method employed. The results can be broadly categorized into two groups: traditional training methods (standard training) and advanced augmentation techniques (DSU, MaxStyle, MixStyle, and RandConv).

- Standard Training: The baseline performance using standard training was consistently low across all conditions. The model struggled to generalize to unseen conditions, resulting in low mIoU values and high standard deviations. This indicates that traditional training methods are insufficient for achieving robust domain generalization in diverse environmental scenarios.
- DSU (Domain-Specific Uncertainty): The DSU method showed significant improvements over standard training. By effectively handling domain shifts and uncertainties, DSU enhanced the model's robustness, particularly under challenging conditions such as fog and winter. The mIoU values for various object classes improved, demonstrating DSU's capability to adapt to different environmental contexts.

- MaxStyle: MaxStyle emerged as the most effective method among the techniques evaluated. Incorporating style transfer techniques during training, MaxStyle significantly boosted the model's performance across all conditions. The method consistently achieved high mIoU values for key object classes, indicating its strong ability to generalize across diverse visual appearances and styles.

- MixStyle: MixStyle also provided notable performance gains, though slightly lower than MaxStyle. By blending multiple styles within the same training batch, MixStyle enhanced the model's robustness and generalization capabilities. The results showed consistent improvements in mIoU values across various conditions, making it a valuable technique for style augmentation.

- RandConv (Random Convolution): The RandConv method introduced random convolutional layers during training, which significantly improved the model's ability to handle data variations. The method achieved high mIoU values for several object classes, particularly under spring and night conditions, demonstrating its effectiveness in enhancing model robustness.

7.3.2 Comparative Analysis Across Environmental Conditions

The comparative analysis of the training methods across different training environmental conditions revealed that advanced augmentation techniques consistently outperformed standard training. Key observations include:

- Night Training Images: Methods incorporating style augmentation and domain-specific techniques (DSU, MaxStyle, MixStyle, and RandConv) significantly outperformed standard training. MaxStyle, in particular, achieved the highest mIoU values, highlighting its effectiveness in adapting to low-light conditions.

- Sunset Training Images: The performance trends observed for sunset training were consistent with those for night training. Advanced methods showed substantial improvements, with MaxStyle leading in performance gains, especially for the road and sidewalk classes.

- Dawn Training Images: Similar trends were observed for dawn training images. The advanced methods demonstrated enhanced robustness and generalization, with MaxStyle and RandConv achieving the highest performance metrics.

7.3.3 Conclusions

The findings from this study underscore the importance of employing advanced augmentation techniques to enhance the robustness and domain generalization of computer vision models. Traditional training methods, while providing a baseline, are inadequate for achieving high performance across diverse environmental conditions. In contrast, methods like DSU, MaxStyle, MixStyle, and RandConv significantly improve model adaptability and accuracy.

- MaxStyle stands out as the most effective method, offering substantial improvements in performance across all tested conditions.
- DSU and RandConv also provide significant gains, particularly in handling domain shifts and enhancing robustness.
- MixStyle offers a balanced approach, improving generalization through style blending within training batches.

These advanced techniques are crucial for developing robust computer vision systems capable of performing reliably in real-world, variable environments. Future research could further explore the integration of these methods with other advanced techniques to continue improving model performance and generalization.

7.4 Future Work

Building on the findings of this study, several avenues for future work can be pursued to further enhance the robustness and domain generalization of computer vision models. The promising results from advanced augmentation techniques such as MaxStyle, DSU, MixStyle, and RandConv highlight the potential for continued innovation in this area.

7.4.1 Integration of Multiple Augmentation Techniques

One promising direction for future research is the integration of multiple augmentation techniques within a single training pipeline. Combining the strengths of methods like MaxStyle and Random Convolution could potentially yield synergistic effects, further enhancing model performance. Research could focus on developing hybrid approaches that effectively leverage the advantages of style transfer, domain adaptation, and random convolutions.

7.4.2 Exploration of Additional Environmental Conditions

While this study focused on specific environmental conditions such as night, sunset, and dawn, future work could expand the range of scenarios to include other challenging conditions. For instance, training and evaluating models under extreme weather conditions like heavy rain, snowstorms, and dense fog could provide deeper insights into the robustness of different training methods. Additionally, incorporating seasonal variations beyond winter and spring, such as autumn and summer, would further test the models' generalization capabilities.

7.4.3 Incorporating Real-World Data

The use of synthetic datasets like SYNTHIA provides a controlled environment for evaluating model performance. However, future research should also consider incorporating real-world datasets to assess the practical applicability of these augmentation

techniques. Real-world data often contains more variability and noise, presenting additional challenges for model training and evaluation. By validating the techniques on real-world datasets, researchers can better understand their effectiveness in practical applications.

7.4.4 Transfer Learning and Fine-Tuning

Transfer learning and fine-tuning offer potential for improving model performance when dealing with limited data from certain environmental conditions. Future research could explore the use of pre-trained models on large-scale datasets, followed by fine-tuning on specific environmental scenarios. This approach could help leverage existing knowledge and improve the model's ability to generalize across diverse conditions.

7.4.5 Evaluation Metrics and Loss Functions

Developing new evaluation metrics and loss functions that better capture the nuances of model performance under varying conditions is another promising direction. Current metrics like mIoU provide valuable insights but may not fully reflect the complexities of real-world scenarios. Future work could focus on designing metrics and loss functions that more accurately measure robustness and generalization.

In conclusion, the promising results obtained from advanced augmentation techniques in this study open numerous opportunities for future research. By exploring these directions, the field can continue to advance, leading to more robust and generalizable computer vision models capable of performing effectively in diverse and challenging environments.

Part IV

References

Bibliography

[1] L. Xiao, J. Xu, D. Zhao, E. Shang, Q. Zhu και B. Dai. *Adversarial and Random Transformations for Robust Domain Adaptation and Generalization*. *Sensors*, 23, 2022.

[2] T. Gokhale, S. Mishra, M. Luo, B. Sachdeva και C. Baral. *Generalized but not Robust? Comparing the Effects of Data Modification Methods on Out-of-Domain Generalization and Adversarial Robustness*. σελίδες 2705–2718, 2022.

[3] P. Li, D. Li, W. Li, S. Gong, Y. Fu και T. Hospedales. *A Simple Feature Augmentation for Domain Generalization*. *IEEE/CVF International Conference on Computer Vision (ICCV)*, σελίδες 8866–8875, 2021.

[4] Z. Zhong, Y. Zhao, G. Lee και N. Sebe. *Adversarial Style Augmentation for Domain Generalized Urban-Scene Segmentation*. *ArXiv*, abs/2207.04892, 2022.

[5] A. Tripathi, R. Singh, A. Chakraborty και P. Shenoy. *Edges to Shapes to Concepts: Adversarial Augmentation for Robust Vision*. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, σελίδες 24470–24479, 2023.

[6] T. Chen, Y. Cheng, Z. Gan, J. Wang, L. Wang, Z. Wang και J. Liu. *Adversarial Feature Augmentation and Normalization for Visual Recognition*. *ArXiv*, abs/2103.12171, 2021.

[7] T. Chen, M. Baktash, Z. Wang και M. Salzmann. *Center-aware Adversarial Augmentation for Single Domain Generalization*. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, σελίδες 4146–4154, 2023.

[8] K. Zhou, Y. Yang, T. Hospedales και T. Xiang. *Deep Domain-Adversarial Image Generation for Domain Generalisation*. *AAAI Conference on Artificial Intelligence*, σελίδες 13025–13032, 2020.

[9] Nobuyuki Otsu. *A threshold selection method from gray-level histograms*. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[10] John Canny. *A computational approach to edge detection*. *IEEE Transactions on pattern analysis and machine intelligence*, PAMI-8(6):679–698, 1986.

[11] Jonathan Long, Evan Shelhamer και Trevor Darrell. *Fully convolutional networks for semantic segmentation*. *Proceedings of the IEEE conference on computer vision and pattern recognition*, σελίδες 3431–3440, 2015.

[12] Olaf Ronneberger, Philipp Fischer και Thomas Brox. *U-Net: Convolutional networks for biomedical image segmentation*. *International Conference on Medical image computing and computer-assisted intervention*, σελίδες 234–241. Springer, 2015.

[13] Liang Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy και Alan L Yuille. *Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs*. *IEEE transactions on pattern analysis and machine intelligence*, τόμος 40, σελίδες 834–848. IEEE, 2017.

[14] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen AWMvan der Laak, Bramvan Ginneken και Clara I Sánchez. *A survey on deep learning in medical image analysis*. *Medical image analysis*, 42:60–88, 2017.

[15] Mennatullah Siam, Sara Elkerdawy, Martin Jagersand και Senthil Yogamani. *A comparative study of real-time semantic segmentation for autonomous driving*. *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, σελίδες 587–597, 2018.

[16] Enrico Maggiori, Yuliya Tarabalka, Guillaume Charpiat και Pierre Alliez. *A dataset for large-scale automatic building footprint delineation from very high-resolution satellite imagery*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, σελίδες 28–36, 2017.

[17] Kaiming He, Georgia Gkioxari, Piotr Dollar και Ross Girshick. *Mask R-CNN*. *Proceedings of the IEEE international conference on computer vision*, σελίδες 2961–2969, 2017.

[18] Jose G. Moreno-Torres, Troy Raeder, Raul Alaiz-Rodriguez, Nitesh V. Chawla και Francisco Herrera. *A unifying view on dataset shift in classification*. *Pattern Recognition*, 2012.

[19] Hao Guan και Mingxia Liu. *Domain Adaptation for Medical Image Analysis: A Survey*. *IEEE Transactions on Biomedical Engineering*, 69(3):1173–1185, 2022.

[20] Rohan Taori, Abhishek Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht και Ludwig Schmidt. *Measuring robustness to natural distribution shifts in image classification*. *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[21] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt και Vaishaal Shankar. *Do ImageNet classifiers generalize to ImageNet?* *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[22] Dan Hendrycks και Thomas Dietterich. *Benchmarking neural network robustness to common corruptions and perturbations*. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[23] Hidetoshi Shimodaira. *Improving predictive inference under covariate shift by weighting the log-likelihood function*. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

[24] Gilles Blanchard, Akshay Anand Deshmukh, Ugur Dogan, Gary Lee και Clayton Scott. *Domain generalization by marginal transfer learning*. *Journal of Machine Learning Research (JMLR)*, 2021.

[25] Jie Yang, Kun Zhou, Yufei Li και Zhiyuan Liu. *Generalized out-of-distribution detection: A survey*. *arXiv preprint arXiv:2110.11334*, 2021.

[26] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira και Jennifer Wortman Vaughan. *A theory of learning from different domains*. *Machine Learning*, 2010.

[27] M. Segu, A. Tonioni και F. Tombari. *Batch Normalization Embeddings for Deep Domain Generalization*. *Pattern Recognition*, 135:109115, 2020.

[28] S. Sivaprasad, A. Goindani, V. Garg και V. Gandhi. *Reappraising Domain Generalization in Neural Networks*. *ArXiv*, 2021.

[29] J. Wang, C. Lan, C. Liu, Y. Ouyang και T. Qin. *Generalizing to Unseen Domains: A Survey on Domain Generalization*. *IEEE Transactions on Knowledge and Data Engineering*, 35:8052–8072, 2021.

[30] Y. Mesbah, Y. Y. Ibrahim και A. Khan. *Domain Generalization using Ensemble Learning*. *ArXiv*, 2021.

[31] Z. Sun, Z. Shen, L. Lin, Y. Yu, Z. Yang και W. Chen. *Dynamic Domain Generalization*. *ArXiv*, 2022.

[32] C. Feng, J. Hu, X. Wang, S. Hu, B. Zhu, X. Wu, H. Zhu και S. Lyu. *Controlling Neural Style Transfer with Deep Reinforcement Learning*. *ArXiv*, 2023.

[33] Y. Li, T. H. Zhang, X. Han και Y. Qi. *Image Style Transfer in Deep Learning Networks*. *2018 5th International Conference on Systems and Informatics (ICSAI)*, σελίδες 660–664, 2018.

[34] B. Niu, Y. Ma και Y. Qi. *Style Transfer of Image Target Region Based on Deep Learning*. *2021 6th International Symposium on Computer and Information Processing Technology (ISCIPT)*, σελίδες 486–489, 2021.

[35] S. Ren και Y. Sheng. *Image Style Transfer Using Deep Learning Methods*. *2022 IEEE International Conference on Electrical Engineering, Big Data and Algorithms (EEBDA)*, σελίδες 1190–1195, 2022.

[36] C. Kuo. *Understanding convolutional neural networks with a mathematical model*. *ArXiv*, abs/1609.04112, 2016.

[37] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai και T. Chen. *Recent advances in convolutional neural networks*. *ArXiv*, abs/1512.07108, 2015.

[38] A. Ajit, K. Acharya και A. Samanta. *A Review of Convolutional Neural Networks*. *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, σελίδες 1–5, 2020.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren και Jian Sun. *Deep Residual Learning for Image Recognition*. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, σελίδες 770–778, 2016.

[40] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke και Andrew Rabinovich. *Going Deeper with Convolutions*. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, σελίδες 1–9, 2015.

[41] Sergey Ioffe και Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, σελίδες 448–456, 2015.

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever και Ruslan Salakhutdinov. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[43] Alex Krizhevsky, Ilya Sutskever και Geoffrey E. Hinton. *ImageNet Classification with Deep Convolutional Neural Networks*. *Advances in Neural Information Processing Systems*, σελίδες 1097–1105, 2012.

[44] Andrew Y. Ng. *Feature selection, L1 vs. L2 regularization, and rotational invariance*. *Proceedings of the Twenty-First International Conference on Machine Learning (ICML)*, σελίδα 78, 2004.

[45] Minyoung Huh, Pulkit Agrawal και Alexei A. Efros. *What makes ImageNet good for transfer learning?* *arXiv preprint arXiv:1608.08614*, 2016.

[46] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto και Hartwig Adam. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. *arXiv preprint arXiv:1704.04861*, 2017.

[47] Mingxing Tan και Quoc V. Le. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. *Proceedings of the 36th International Conference on Machine Learning (ICML)*, σελίδες 6105–6114, 2019.

[48] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy και A. Yuille. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40:834–848, 2016.

[49] O. Ronneberger, P. Fischer και T. Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, σελίδες 234–241, 2015.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov και L. C. Chen. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, σελίδες 4510–4520, 2018.

[51] J. Hoffman, E. Tzeng, T. Park, J. Y. Zhu, P. Isola, K. Saenko, A. A. Efros και T. Darrell. *CyCADA: Cycle-Consistent Adversarial Domain Adaptation*. *Proceedings of the 35th International Conference on Machine Learning*, σελίδες 1989–1998, 2018.

[52] H. Zhao, J. Shi, X. Qi, X. Wang και J. Jia. *Pyramid Scene Parsing Network*. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, σελίδες 2881–2890, 2017.

[53] L. C. Chen, G. Papandreou, F. Schroff και H. Adam. *Rethinking Atrous Convolution for Semantic Image Segmentation*. *arXiv preprint arXiv:1706.05587*, 2017.

[54] Y. Gal και Z. Ghahramani. *Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning*. *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, σελίδες 1050–1059, 2016.

[55] T. Pham και Others. *Deep Learning With Anatomical Priors: Imitating Enhanced Autoencoders In Latent Space For Improved Pelvic Bone Segmentation In MRI*. *IEEE Transactions on Medical Imaging*, 38(11):2524–2533, 2019.

[56] S. Moraboena και Others. *A Deep Learning Approach to Network Intrusion Detection Using Deep Autoencoder*. *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[57] M. Dorado και Others. *Deep Diffusion Autoencoders*. *International Conference on Learning Representations (ICLR)*, 2019.

[58] R. Al-Hmouz και Others. *Logic-driven Autoencoders*. *International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2019.

[59] L. Sun και Others. *Learning a Good Representation with Unsymmetrical Autoencoder*. *International Conference on Neural Information Processing (ICONIP)*, 2016.

[60] K. Hajewski και Others. *Evolving Deep Autoencoders*. *Neurocomputing*, 379:205–218, 2020.

[61] Y. Zhou και Others. *Deep Cycle Autoencoder for Unsupervised Domain Adaptation with Generative Adversarial Networks*. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[62] B. Prayuda και Others. *Autoencoder-based Image Companding*. *IEEE International Conference on Image Processing (ICIP)*, 2020.

[63] M. Schuster και A. Krogh. *A Manifold Learning Perspective on Representation Learning: Learning Decoder and Representations without an Encoder*. *Journal of Machine Learning Research*, 22:1–21, 2021.

[64] M. B. Ganapini, M. Campbell, F. Fabiano, L. Horesh, J. Lenchner, A. Loreggia, N. Mattei, T. Rahgooy, F. Rossi, B. Srivastava και B. Venable. *Combining Fast and Slow Thinking for Human-like and Efficient Navigation in Constrained Environments*. *ArXiv*, abs/2201.07050, 2022.

[65] S. Hassantabar, P. Terway και N. Jha. *TUTOR: Training Neural Networks Using Decision Rules as Model Priors*. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42:483–496, 2020.

[66] W. Woods, J. H. Chen και C. Teuscher. *Adversarial Explanations for Understanding Image Classification Decisions and Improved Neural Network Robustness*. *Nature Machine Intelligence*, 1:508–516, 2019.

[67] Y. Okajima και K. Sadamasa. *Deep Neural Networks Constrained by Decision Rules*. *Proceedings of the AAAI Conference on Artificial Intelligence*, τόμος 33, σελίδες 2496–2505, 2019.

[68] J. Kim, J. Kim, B. Kim, M. Lee και J. Lee. *Hardware Design Exploration of Fully-Connected Deep Neural Network with Binary Parameters*. *2016 International SoC Design Conference (ISOCC)*, σελίδες 305–306, 2016.

[69] S'ebastien J'egou, Michal Drozdzal, David Vazquez, Adriana Romero και Yoshua Bengio. *The One Hundred Layers Tiramisu: Fully Convolutional DenseNets for Semantic Segmentation*. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, σελίδες 11–19, 2017.

[70] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y. Hammerla, Bernhard Kainz, Ben Glocker και Daniel Rueckert. *Attention U-Net: Learning Where to Look for the Pancreas*. *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, σελίδες 261–270, 2018.

[71] Paul Luc, Camille Couprie, Soumith Chintala και Jakob Verbeek. *Semantic Segmentation using Adversarial Networks*. *Proceedings of the Conference on Neural Information Processing Systems*, σελίδες 1–9, 2016.

[72] Dan Hendrycks, Andy Zou, Mantas Mazeika, Leonard Tang, Bo Li, Dawn Song και Jacob Steinhardt. *PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures*. *arXiv preprint arXiv:2112.05135v3*, 2022.

[73] Gopal Krishna Sinha και Debashis Sen. *Noise Reduction in Image Processing Using Deep Learning Techniques*. *Springer*, 2021.

[74] Bee Lim Heewon Kim, Myungsub Choi και Kyoung Mu Lee. *Task-Aware Image Downscaling*. *ArXiv*, 2020.

[75] Hossein Talebi και Peyman Milanfar. *Learning to Resize Images for Computer Vision Tasks*. *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[76] Murat Gevrekci και B. Gunturk. *On Geometric and Photometric Registration of Images*. *2007 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2007.

[77] Karakida R. Takase T. και Asoh H. *Self-paced Data Augmentation for Training Neural Networks*. *ArXiv*, 2020.

[78] Gokul P. Nanthini K., Sivabalaselvamani D. και Kavinkumar S. *A Survey on Data Augmentation Techniques*. *ICCMC*, σελίδες 913–920, 2023.

[79] Kaufhold M. Bayer M. και Reuter C. *A Survey on Data Augmentation for Text Classification*. *ACM Computing Surveys*, 55:1–39, 2021.

[80] Kaichao Zhou, Yongxin Yang, Yu Qiao και Tao Xiang. *Domain Generalization with MixStyle*. *International Conference on Learning Representations (ICLR)*, 2021.

[81] C. Chen, Z. Li, C. Ouyang, M. Sinclair, W. Bai και D. Rueckert. *MaxStyle: Adversarial Style Composition for Robust Medical Image Segmentation*. *ArXiv*, 2022.

[82] Xiaotong Li, Yongxing Dai, Yixiao Ge, Jun Liu, Ying Shan και Ling Yu Duan. *Uncertainty Modeling for Out-of-Distribution Generalization*. *arXiv preprint arXiv:2202.03958*, 2022.

[83] Zhenlin Xu, Deyi Liu, Junlin Yang, Colin Raffel και Marc Niethammer. *Robust and Generalizable Visual Representation Learning via Random Convolutions*. *International Conference on Learning Representations (ICLR)*, 2021.

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

[85] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang και Jiaya Jia. *Pyramid scene parsing network*. *Proceedings of the IEEE conference on computer vision and pattern recognition*, σελίδες 2881–2890, 2017.

List of Abbreviations

AAAI	Association for the Advancement of Artificial Intelligence
ACM	Association for Computing Machinery
BPF	Band-Pass Filter
CNN	Convolutional Neural Network
CT	Computed Tomography
CVF	Computer Vision Foundation
CVPR	Computer Vision and Pattern Recognition
DA	Data Augmentation
DCA	Domain Classifier Alignment
DDA	Domain Discriminative Alignment
DDG	Dynamic Domain Generalization
DG	Domain Generalization
DSU	Domain-Specific Unit
EEBDA	Enhanced Event-Based Data Augmentation
ELU	Exponential Linear Unit
ERM	Empirical Risk Minimization
FAN	Adversarial Feature Augmentation and Normalization
FCN	Fully Convolutional Network
FPS	Frames Per Second
FTN	Fast Thinking Network
GA	Genetic Algorithm
GPU	Graphics Processing Unit
GTAV	Grand Theft Auto V (used as a dataset in computer vision)
HD	High Definition
HDR	High Dynamic Range
ICASSP	International Conference on Acoustics, Speech, and Signal Processing
ICCV	International Conference on Computer Vision
ICIP	International Conference on Image Processing
ICLR	International Conference on Learning Representations
ICML	International Conference on Machine Learning
ICONIP	International Conference on Neural Information Processing
IEEE	Institute of Electrical and Electronics Engineers
ISOCC	International SoC Design Conference
JMLR	Journal of Machine Learning Research
KL	Kullback-Leibler (Divergence)

LDR	Low Dynamic Range
MICCAI	Medical Image Computing and Computer-Assisted Intervention
MRI	Magnetic Resonance Imaging
MS	Multi-Source
NLP	Natural Language Processing
OOD	Out-Of-Distribution
PACS	Picture Archiving and Communication System
RDF	Resource Description Framework
RGB	Red, Green, Blue (color model)
RQL	RDF Query Language
SEQS	Sequences
STN	Spatial Transformer Network
SYNTHIA	Synthetic Collection of Imagery and Annotations
VLCS	A dataset combining images from Caltech, and SUN datasets
VOC	Visual Object Classes (Challenge)
WACV	Winter Conference on Applications of Computer Vision