Evaluating 3D Image Classification Using Single RGB-D Images

Author:
Manuel Freistein
Matriculation No.: 928694

First Reviewer:
Prof. Dr. Kristian Hildebrand
Second Reviewer:
Prof. Dr. Frank Haußer

A thesis submitted to Faculty VI - Data Science -
Berliner Hochschule für Technik (BHT)
for the degree of Master of Science

29th February 2024

Abstract

This paper aims to improve conventional RGB image classification systems through the simple addition of depth information derived from the RGB images themselves. As a proof of concept, it uses the SynthNet RGB image classification pipeline, which achieved state-of-the-art results on the VisDA dataset, as a baseline and integrates monocular estimated depth channels. Specifically, it attempts to improve Acc@1 on the Topex dataset, a complex machine parts dataset defined by a significant synthetic to real domain gap. Utilizing the latest in transfer learning strategies, vision transformers and monocular depth estimation models, the study devises strategies for data augmentation, model adaptation, and the application of freezing schedules. It also conducts a thorough comparison of depth data integration techniques, compares the performance of monocular estimated versus CAD rendered depth data and provides a detailed error analysis of the best performing models. The findings demonstrate a significant enhancement in classification accuracy, culminating in a model that achieves an Acc@1 of 69.70%. The applied methods offer practical solutions to industrial challenges and blueprints for integrating depth data into cutting edge classification models.

Acknowledgments

Completing this master's thesis marks a significant milestone in a journey that began with a complete career pivot. This endeavor, daunting in its scope and scale, was made possible through the unwavering support and guidance of numerous individuals to whom I owe a debt of gratitude.

First off, I must express my profound appreciation for the opportunity afforded to me by the free education system in Germany, and particularly the BHT. This is a privilege and I am deeply grateful for access to the knowledge and learning it has provided.

I extend my thanks to Prof. Kristian Hildebrand for his supervision and mentorship throughout this project. I am equally thankful for Dennis Ritter whose assistance was indispensable. Dennis' readiness to answer my numerous inquiries about the Synthnet pipeline were crucial in overcoming several challenges.

I would also like to express my gratitude to all my friends, my family, and particularly to those people who tutored me in programming and mathematics. My deepest thanks in this regard are reserved for Felix, a dear friend without whom I would certainly not be in the position I am today. Felix, I am profoundly grateful.

Most importantly, to my incredible wife, words cannot fully capture my appreciation. You have been my rock, enduring my incessant whining with an astonishing reservoir of patience and empathy. Your encouragement to push through, when this meant enduring even more moaning, was nothing short of mythical. You are the unsung hero of this journey, and I am immeasurably grateful for your love and support.

Lastly, I dedicate a special note of gratitude to my baby boy, Ben Issam. In moments of doubt and exhaustion, it was the thought of you that reignited my resolve and inspired me to persevere. Your presence has been a source of joy and motivation, reminding me of the brighter aspects of life amidst the rigors of academic pursuit.

This thesis is not just a reflection of my efforts but a mosaic of the contributions and sacrifices of each one of you. Thank you!

Table of Contents

List of Figures

List of Tables

A Introduction

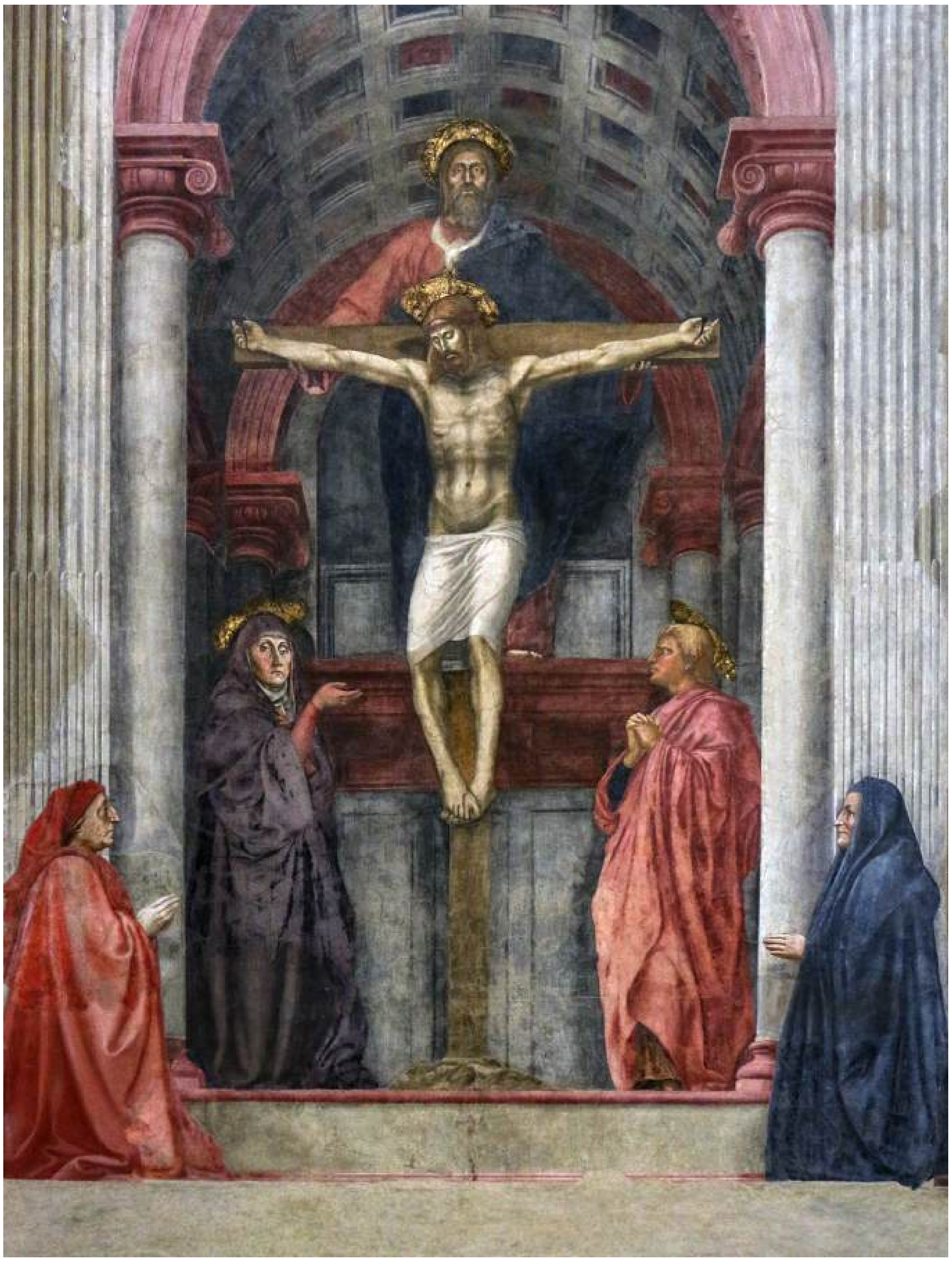


Figure A.1: Masaccio, Holy Trinity c. 1427, fresco, 667 x 317 cm, Santa Maria Novella, Florence, Italy, photograph by Fr. Lawrence Lew, O.P., online: https://www.flickr.com/photos/paullew/49876650823.

Imagine wandering into a Florentine church in the early 15th century and encountering Masaccio's "The Holy Trinity" for the first time. This fresco, a marvel of the early Renaissance, unveiled to its viewers the mesmerizing magic of linear perspective, creating an illusion of depth so persuasive that the church's architecture appeared to unfold inward, transcending the surface of the wall. Masaccio's masterpiece demonstrated the potential of a simple technique to bring a two-dimensional surface to life.

This research tries to harness a similar transformative power for use in image classification. By extending single RGB images with a monocular estimated depth (D) channel, the project aims to improve the performance of the RGB SynthNet image classification pipeline [RHH$^+$23] for the Topex dataset.[1] It is expected that this simple tweak can substantially enhance the system's ability to categorize images from a highly intricate dataset of machine components with a significant synthetic to real domain gap. In doing so, the research not only explores the capabilities of computers to perceive and learn through visual means, but also offers practical solutions to current industrial challenges, echoing the Renaissance's spirit of technical as well as visual innovation.

Image classification technology is the basis for advancements across a range of applications. Its impact today extends through various critical and economically significant sectors, revolutionizing processes in healthcare and medical imaging, automotive and autonomous vehicles, retail and e-commerce, security and surveillance, manufacturing, quality control and many more. Despite notable successes in these major areas, there remains a wealth of "long-tail" challenges, representing untapped markets and opportunities, yet to be tackled with image classification tools. This research zeroes in on one specific but critical industrial need: the precise recognition of small machine parts. Such precision is vital for efficient inventory management, quality control, and maintenance. The inclusion of D data is anticipated to not only improve the accuracy for classifying machine parts, but also to demonstrate ease of application, making it a versatile template for a host of different classification tasks and more.

Recent innovations in GPU technology, vision transformer architectures, and transfer learning strategies have markedly improved the efficacy of image classification models, particularly for projects with small datasets. Furthermore, the introduction of synthetic data generation and domain adaptation techniques has provided crucial alternatives in situations lacking real-world data. Additionally, the progress in depth data technologies, such as the integration of sophisticated LiDAR scanners in consumer electronics, heralds an era of high-quality 3D datasets that promise to further augment image classification capabilities. This thesis aims to capitalize on

[1]Synthnet achieved SOTA performance on the VisDA dataset.

these cutting-edge developments in order to develop a robust and precise RGB-D image classification system.

A.1 Task and Challenges

The core objective of this thesis is to strengthen the SynthNet pipeline [RHH+23] for the Topex dataset. The SynthNet paper applied structured transfer learning, data augmentation and domain adaptation stragegies to transformer models for the VisDA [PUK+17] and Topex datasets, achieving SOTA performance. This process involved retraining pre-trained models with a new classification head for the source domain, followed by fine-tuning and a hyperparameter search. The pipeline then utilized these optimized parameters for Unsupervised Domain Adaptation (UDA) experiments, applying techniques like Conditional Domain Adversarial Network (CDAN) and Minimum Class Confusion (MCC). The integration of D data into the Synthnet pipeline is anticipated to further improve classification accuracy, with a particular focus on the Acc@1 metric. However, incorporating D data presents several challenges that must be addressed.

The first challenge of this thesis involves expanding the Topex RGB dataset to include D data. Fortuitously, the CAD-based nature of the Topex training dataset allows for the rendering of 3D images. Recognizing that such D data generation is not possible in many instances, this thesis will primarily explore monocular depth estimation techniques as a practical alternative. The availability of various D data sources enables a comparative analysis of the quality of D data and facilitates the evaluation of the practical viability of monocular depth estimation models.

Another significant challenge in this thesis is the limited size of the Topex dataset. Topex is comprised of 3,264 CAD-rendered images and 6,146 real RGB photographs. Robust classification models typically require large, diverse datasets for effective training and generalization. Furthermore, the dataset features machine parts with subtle variances in shape, size, or texture. If that wasn't enough, the real challenge lies in bridging the domain gap between synthetic training and real test images within the Topex dataset. These features necessitate creative approaches to optimize data use. Particularly, because the industrial nature of the dataset demands exceptionally high precision from the classification models.

Lastly, the integration of D data into sophisticated RGB vision transformer models as well as optimization of such models introduces its own set of challenges. A comprehensive and detailed approach is crucial for effectively incorporating D information and ensuring seamless cross-modal interactions. The research aims to devise strategies that not only integrate but also amplify the capabilities of image classification models when utilizing RGB-D data.

A.2 Methods

To improve the performance of the Synthnet pipeline on the Topex dataset various methods and technologies were adopted (see figure A.2). Notably, the pipeline now incorporates a D data estimator at its outset, has refined the pre-processing techniques to accommodate D data, reconfigured the vision transformer model for processing both RGB and D data, and revised the fine-tuning approach. The domain adaptation process remains to be adapted to D input, but can be added at a later time.

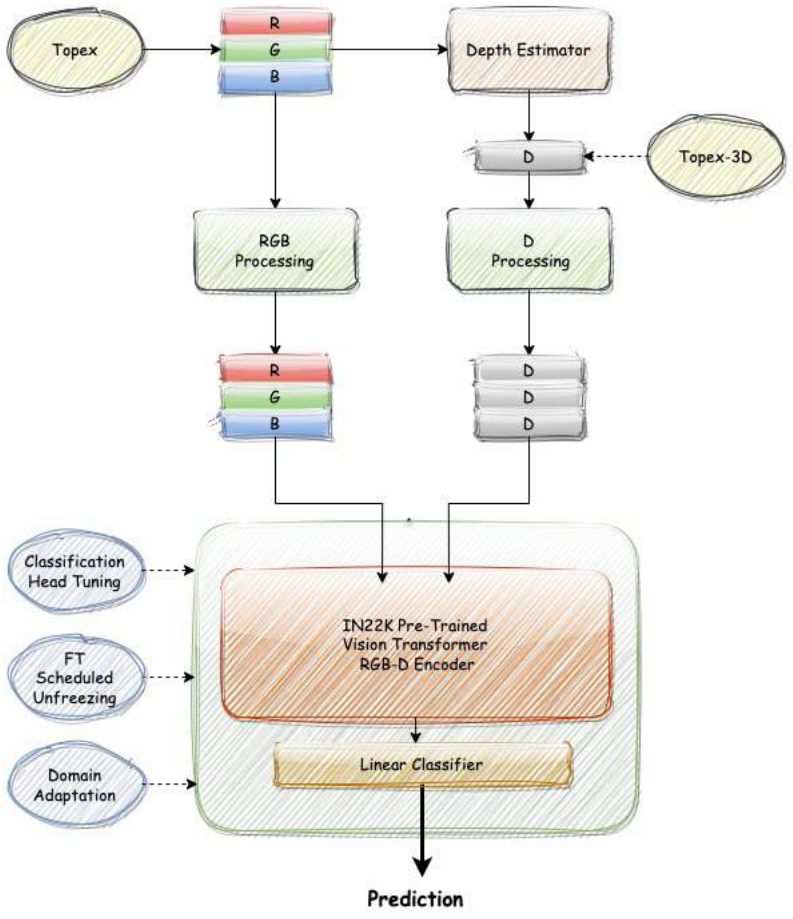


Figure A.2: The updated Synthnet Pipeline for RGB-D images.

Monocular Depth Estimation: The Topex dataset was expanded to include a range of D datasets. This expansion involved using suitable monocular depth estimation algorithms to infer D from RGB images, providing a cost-effective, quick and automatic approach for D data acquisition. Additionally, synthetic training D data rendered from CAD models was tested for comparative purposes.

Data Augmentations: Custom data augmentations and and pre-processing techniques were applied to RGB and corresponding D datasets. These operations are crucial for preparing data for the specific model and enhancing the models' ability to generalize.

Transformer Model Adaptations: A vision transformer model was adapted to incorporate the added D dimension. This involved two primary methods: early fusion, which integrates D data with RGB data before entering the transformer's feature extractor, allowing simultaneous processing of both types of data; and late fusion, which combines feature maps from RGB and D data post-feature extraction, enabling the independent processing of each data type to capture more detailed features.

Finetuning Transformer Model: Transfer Learning strategies were implemented during training, including selective and sequential freezing and unfreezing of the transformer model layers. The goal was to maintain a balance between adding complexity by integrating new D data and preserving the model's existing RGB learned features.

A.3 Hypotheses

The guiding hypothesis of this research was that integrating D data into the Synthnet image classification pipeline would significantly enhance classification accuracy, specifically Acc@1.

Effectiveness of Monocular Depth Estimation: This paper also hypothesized that monocular depth estimation would provide adequate D information for classification tasks. Despite being an inferred method, depth estimation was expected to yield results comparable to those achieved by using rendered D maps. This would demonstrate the feasibility of using single RGB images to derive reliable D information for improving vision transformer models.

Late Fusion vs. Early Fusion: Another hypothesis was that adapting a late fusion approach would prove more effective than an early fusion approach, particularly in fine-tuning on small to moderate-sized datasets. This approach was favored, because it leveraged the model's original structure and learned weights, treating the addition of D data as a variation in the information type rather than a fundamental change to the data itself. This should lead to more effective adaptation during fine-tuning.

More elaborate Freezing Schedules Improve Model Performance: Furthermore, it was hypothesized that implementing sophisticated and dynamic freezing schedules during the training of the RGB-D vision transformer models could lead to enhanced model performance. By carefully designing the freezing schedule—choosing which layers to freeze, when to freeze them, and when to unfreeze them for subsequent training—models would achieve better accuracy and generalization capabilities, particularly on out-of-distribution data (OOD).

A.4 Thesis Structure

The remainder of this work is organized as follows. Chapter 2 will delve into a review of related works, exploring image classification, vision transformers, transfer learning and depth data generation methods. Chapter 3 will describe the basic characteristics of the datasets employed, including Topex and its D extension. It will also include an exploratory data analysis. Chapter 4 will outline the methodology, covering monocular depth estimation steps, data processing, model adaptations, training techniques and implementation details. Chapter 6 will present the experiments' results and offer an evaluation of the adapted models' performances and errors. The paper will conclude with Chapter 7, summarizing the findings, reflecting on the initial hypotheses, and suggesting directions for future research.

B Related Work

This chapter delves into some foundational concepts and recent advancements in image classification, vision transformers, transfer learning strategies and monocular depth estimation. It also explains the applied evaluation metrics and loss function.

B.1 Classification

In Machine Learning, the overarching goal is to predict outcomes based on available data. This prediction task is categorized into two main types: classification and regression. In a classification problem, the outcome is defined by a finite set of distinct classes. The algorithm calculates probabilities indicating the likelihood of a specific unit belonging to each class. Subsequently, a classification rule is applied to assign the unit to either a single class (single-label classification) or multiple classes (multi-label classification). In single-label classification, a prevalent rule is to assign the unit to the class with the highest probability. When a classification task involves more than two classes, it is referred to as multi-class classification.

B.1.1 Evaluation Metrics

Evaluating the performance of classification models is typically done by contrasting it with others models or examining its effectiveness under various tuning parameters. Specific metrics are required to accurately measure performance. The focus of this project is on a multi-class, single-label classification problem. The effectiveness of this model will be assessed using a set of well-defined metrics, which are essential for comprehensive understanding and comparison [GBV20].

Confusion Matrix: The Confusion Matrix stands as an essential tool in the evaluation of multi-class, single-label classification models. It serves as the foundation for deriving a multitude of performance metrics. In a multi-class context, the Confusion Matrix is formatted as an N×N table, where N represents the total number of distinct classes. The rows of this matrix are the actual classes of the data, while the columns correspond to the classes as predicted by the model. The accuracy of these predictions is visually highlighted along the main diagonal of the matrix, where the predicted classes align perfectly with the actual classes [GBV20].

In addition to showcasing correct predictions (true positives and true negatives) for each class, the Confusion Matrix also categorizes the types of errors made by the model. False positives are instances where the model incorrectly identifies a unit as belonging to a certain class, whereas false negatives refer to instances where the model fails to recognize a unit's true class. These distinctions are crucial in understanding not just the overall accuracy, but also the specific strengths and weaknesses of the model in classifying each category.

Precision and Recall: In multi-class classification, precision for a given class is defined as the number of true positives (the correctly predicted instances of the class) divided by the total number of instances predicted as that class (true positives plus false positives for that class). This reflects the accuracy of positive predictions for each individual class [GBV20]. Mathematically, it is expressed as:

$$\text{Precision}_{\text{class}} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}} \tag{1}$$

Recall for a given class is the number of true positives divided by the actual total number of instances of that class (true positives plus false negatives for that class). This measures the model's ability to correctly identify all instances of each class [GBV20]. It is calculated as:

$$\text{Recall}_{\text{class}} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \tag{2}$$

Accuracy: A fundamental metric, accuracy measures the proportion of true results (both true positives and true negatives) among the total number of cases examined. It's calculated as the sum of correct predictions divided by the total number of predictions.

Accuracy-1 (Acc@1): This is a specific form of accuracy where the prediction is considered correct if the top class (i.e., the class with the highest probability) is the actual class. This metric will evaluate how often the model's highest confidence prediction is correct.

Balanced Accuracy: In cases involving datasets with large numbers of classes (such as 102 classes in the Topex dataset), standard accuracy offers a quick overview of the model's overall performance. However, this metric can mask the true effectiveness of the model, particularly in situations with class imbalance. This is where Balanced Accuracy becomes particularly relevant. It calculates the mean recall achieved across each class. This approach ensures that each class is equally represented in the metric, making it a reliable measure of model performance even when some classes are underrepresented in the dataset [GBV20]. The formula for Balanced Accuracy in a multi-class setting is:

$$\text{Balanced Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{3}$$

F1-Score: The F1-Score is calculated as the harmonic mean of precision and recall, providing a balanced measure between these two metrics. The formula is:

$$\text{F1-Score} = 2 \cdot (\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}) \tag{4}$$

This score is interpreted as a weighted average of precision and recall, with the best possible value being 1 (indicative of perfect precision and recall) and the worst being 0. A high F1-Score indicates a well-balanced classification model that achieves both high precision and high recall [GBV20].

Macro F1-Score: The F1-Score can be extended to work in the multi-class case, but it requires some modifications. There are two different Macro F1 Scores. F1_1 is particularly useful and will be used in this paper. Unlike the F1_2, which calculates the harmonic mean of a macro-averaged precision and recall, F1_1 computes the F1 Score for each class individually and then takes the arithmetic mean of these scores. This method ensures that each class contributes equally to the final score, making it more robust in the face of class imbalances. The formula for the F1_1-Score is:

$$\text{Macro F1}_1\text{-Score} = \frac{1}{n}\sum_{k=1}^{n}\text{F1}_k = \frac{1}{n}\sum_{k=1}^{n} 2 \cdot (\frac{\text{P}_k \cdot \text{R}_k}{\text{P}_k + \text{R}_k}) \tag{5}$$

Here, P_k and R_k represent the precision and recall for class $_k$ respectively. A high Macro F1 Score (between 0 and 1) suggests that the model performs well across all classes, while a low score indicates poor performance in one or more classes [OB21].

B.1.2 Cross-Entropy Loss

Cross-entropy loss, also known as log loss, plays a crucial role in evaluating multi-class classification models. In a multi-class setting, each class label is typically represented in a one-hot encoded format, where the actual class has a value of 1, and all others are 0. Cross-entropy loss measures how well the predicted probability distribution aligns with the true distribution.

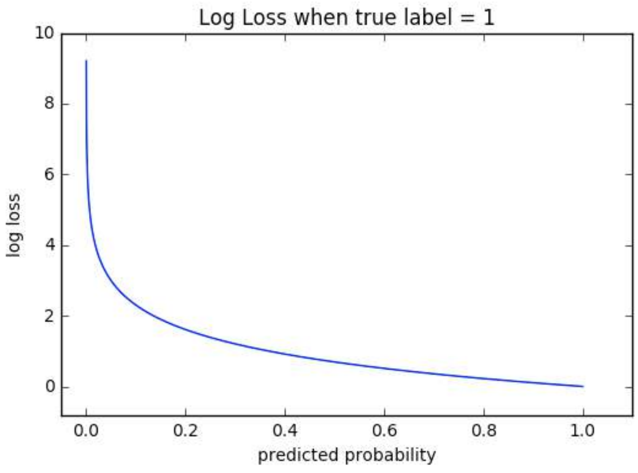


Figure B.1: Cross Entropy Loss.

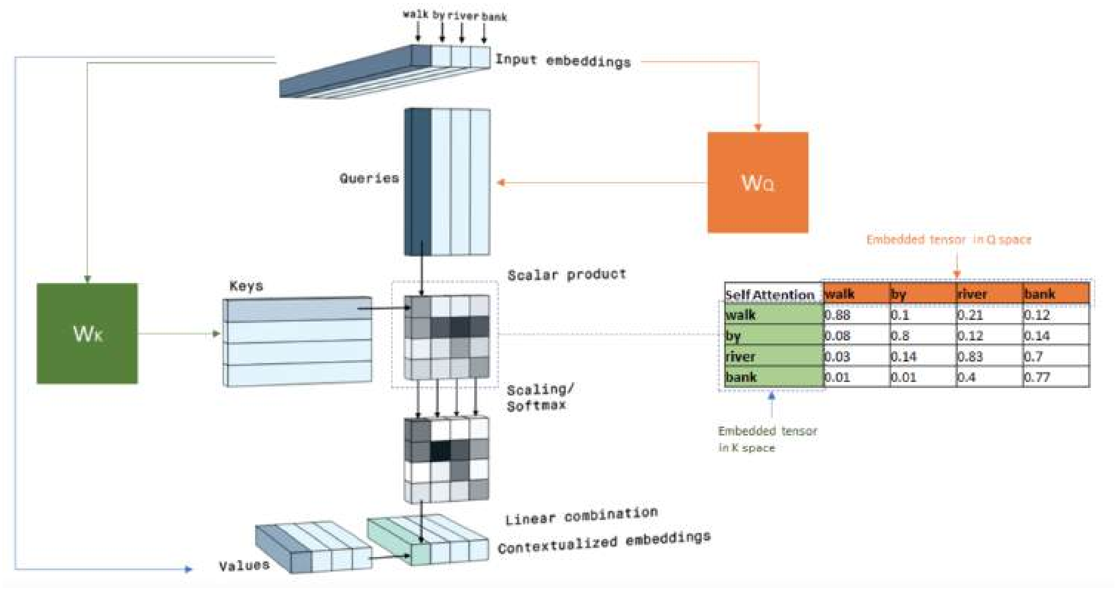
A key characteristic of cross-entropy loss is its sensitivity to the confidence of predictions: the loss increases as the predicted probability deviates from the actual class label. For instance, a prediction with low confidence (a low probability value) that is correct will result in a significantly higher loss compared to a more confident correct prediction (see figure B.1). By penalizing confident and incorrect predictions more heavily, cross-entropy loss effectively encourages the model to be both accurate and confident in its predictions, making it a powerful tool for training multi-class classification models.

B.2 Vision Transformers

Vision transformers (ViTs) are a class of models that apply the principles of the original transformer architecture, typically used in natural language processing, to the realm of computer vision. The original transformer, introduced by Vaswani et al. in 2017 [VSP+23], revolutionized the field of language processing with its ability to capture long-range dependencies between elements in a sequence, using a mechanism known as self-attention. Dosovitskiy et al. succesfully applied this concept to visual data [DBK+21].

B.2.1 Self-Attention

Self-attention is a mechanism that enables the model to weigh the importance of different parts of the input data to all other parts of the input data during processing. It acts similarly to human visual attention, where we focus on specific parts of our visual field and assign different levels of importance to different segments of the information. In neural networks, attention mechanisms dynamically highlight relevant features of the input data, enhancing the model's ability to make predictions based on those features. It helps the models to focus on the most pertinent parts of the input.



Self Attention	walk	by	river	bank
walk	0.88	0.1	0.21	0.12
by	0.08	0.8	0.12	0.14
river	0.03	0.14	0.83	0.7
bank	0.01	0.01	0.4	0.77

Figure B.2: Key, Query, Value Computation Visualized [Pel21]

Each element of the input data is represented in three different ways – as a query, a key, and a value. These representations are obtained through learned linear transformations. The mechanism computes the dot product of the query with all keys to get attention scores. These scores determine how much focus each element of the input sequence should get in relation to others. The attention scores are then passed through a softmax layer to convert them into probabilities. Finally, the softmax output is multiplied with the value and summed up. This results in an output that is a weighted sum of all values, where the weights are determined by how relevant each part of the input is to the other parts [VSP+23].

Self-attention allows the model to consider the entire input, rather than just local or neighboring elements, enabling it to capture complex, long-range relationships across the input. This is particularly powerful in tasks like image processing, where understanding the broader context is crucial. For example, Convolutional Neural Networks (CNNs), the standard for many computer vision tasks pre-transformer, are constrained by their receptive fields focusing on local or neighboring elements as well as pooling operations. This can lead to a loss of spatial information and difficulties in capturing global, long-range interactions [LTHX22].

B.2.2 Architecture

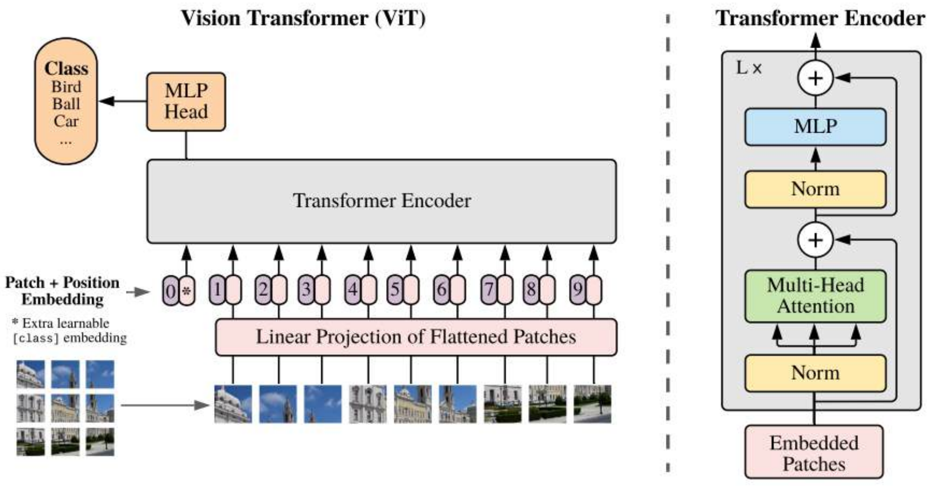


Figure B.3: ViT Architecture [DBK$^+$21]

(1) In ViTs, an input image is divided into fixed-size patches. These patches are then flattened and linearly embedded, similar to token embedding in language models.

(2) Since the transformer architecture does not inherently account for the order of the input, positional encodings are added to the patch embeddings to retain positional information.

(3) At the core of the ViT sits its self-attention mechanism, which allows the model to weigh the importance of different patches when processing each patch. This enables the model to focus on the most relevant parts of the image for a given task.

(4) The embedded patches are passed through layers of transformers. Each layer consists of multi-head self-attention mechanisms coupled with feed-forward neural networks which enable the model to simultaneously focus on various visual aspects and phenomena. In this way, the model can learn intricate and nuanced representations of the visual input data.

(5) For tasks like image classification, the output of the transformer layers is passed through a classification head, typically a linear layer, to make predictions.

The Vision Transformer model, introduced in "An Image is Worth 16x16 Words"

[DBK+21] demonstrated that a pure transformer, tailored to process sequences of image patches, can achieve remarkable performance on image classification tasks.

B.2.3 Swin Transformers

Swin Transformers, introduced by Liu et al. in 2021 [LLC+21], represent another significant advancement in the application of transformers to computer vision. They are designed to address some of the limitations of the original ViTs in processing high-resolution images and capturing local features.

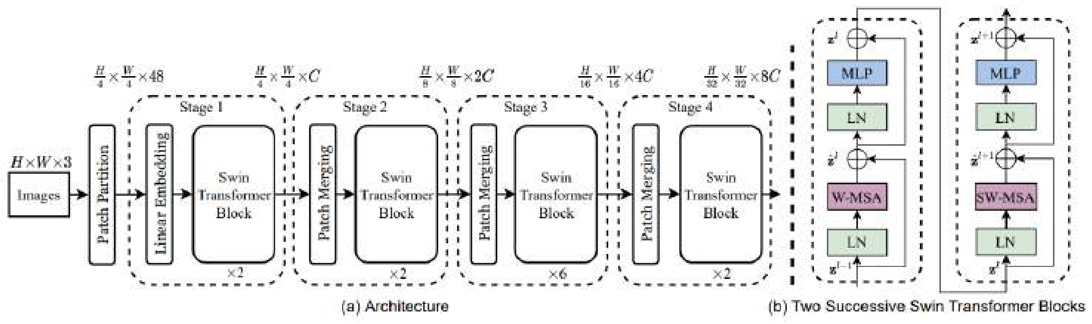


Figure B.4: Swin Architecture (tiny version) [LLC+21]

SwinV1: The first iteration, SwinV1, features a hierarchical design with shifted window-based self-attention. This structure allows the model to efficiently handle image content across different scales and adapt to varying image sizes, a limitation in standard ViTs. SwinV1 processes images using non-overlapping local windows, and these windows are shifted across layers to facilitate cross-window connections. This innovative approach reduces computational complexity, as it avoids the globally operated self-attention mechanism. As a result, SwinV1 has shown superior performance in tasks like object detection and semantic segmentation, outperforming existing models [LLC+21].

SwinV2: Building on SwinV1's success, SwinV2 enhances the model's capacity and ability to handle higher resolutions. It introduces crucial architectural modifications aimed at better scalability and efficiency. A key adaptation in SwinV2 is the implementation of a res-post-norm structure, which replaces the earlier pre-norm configuration. This change boosts training stability and allows for a more effective scaling up of the model's capacity. Additionally, SwinV2 adopts scaled cosine attention in lieu of the traditional dot product attention mechanism, further contributing to the model's enhanced scaling capacity. The introduction of a log-spaced continuous relative position bias facilitates more effective model transfer across various window resolutions. These improvements enable SwinV2 to excel in learning from larger datasets and handling higher-resolution inputs. Consequently, SwinV2 achieves remarkable improvements in various computer vision tasks, setting new standards, especially in scenarios involving high-resolution image processing and large-scale training [LHL+22].

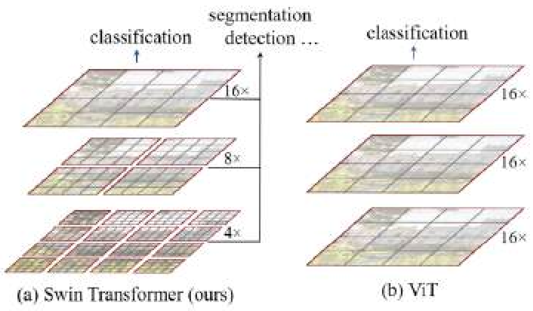


Figure B.5: Swin Windows and Hierarchical Design compared to the original ViT [LLC+21].

B.3 Transfer Learning

Deep neural networks, notably CNNs and transformers, have achieved significant success in image classification tasks. However, their performance largely depends on having access to extensive labeled datasets. In many real-world scenarios, obtaining such data can be challenging due to issues like scarcity, privacy concerns, high costs, or the need for specialized expertise in data labeling [TK23]. To address this, Transfer Learning has emerged as a corrective. It involves the process of leveraging knowledge gained in one model (the source) and applying it to improve performance on a different model (the target). Transfer Learning can occur across various dimensions, including tasks, domains, or languages, with Task Adaptation and Domain Adaptation being particularly relevant for this thesis.

B.3.1 Task Adaptation

Task Adaptation represents the most prevalent and accessible form of transfer learning. It aims to apply the knowledge acquired from a source task to enhance a target task. This process involves two main stages: pre-training and adaptation. In the pre-training stage, a model is trained on the source task, building a foundational knowledge base. During the adaptation phase, this knowledge is then applied to the target task [Rud19].

While pre-training usually is very expensive, it only needs to be performed once. Its primary objective is to develop a broad understanding that can be applied to various target tasks. Research suggests that models trained on large-scale data demonstrate strong transferability to diverse downstream tasks [JSWL22]. In contrast, the adaptation phase is often more efficient, allowing for rapid adjustments to specific target tasks using smaller datasets. The effectiveness of this phase depends on the similarity between the source and target datasets. A closely related source dataset ensures that the learned features are applicable to the target task. Conversely, significant differences between the source and target domains may lead to negative transfer, where pre-training may adversely affect the target model's performance. Studies indicate that models utilizing pre-trained weights often make similar errors in the target domain and share common characteristics in their loss landscapes. This contrasts with models initialized randomly, which display different error patterns and loss landscape features, underscoring the influence of pre-training on model behavior [PG22].

There are several ways to use a pre-trained model for task adaptation, two of which were tested for this thesis: feature extraction and fine-tuning.

Feature Extraction: Feature extraction involves utilizing a pre-trained model's weights in a fixed manner, commonly referred to as "freezing", during the training process on the target task. This means these weights are not updated or changed. The primary aim is to leverage the pre-trained model for pre-processing the data related to the target task. This approach is based on the assumption that the pre-trained model already possesses a substantial understanding of the features relevant to the target task, thereby eliminating the need for additional training on target task data. However, the classifier component of the pre-trained model, which is tailored to the source task, does not transfer seamlessly to the target task. As a result, this classifier must be retrained or replaced to suit the new task. While the general features captured by the source model are useful, the specific mapping of these features to particular classes or outcomes needs to be learned by the target task classifier [Rud19].

Fine-Tuning in Transfer Learning: Similar to feature extraction, fine-tuning requires adjusting the classifier to align with the target task. However, fine-tuning differs significantly in its approach to the pre-trained model's layers. In this process, some or all layers of the pre-trained model are "unfrozen," making their weights modifiable during training. The degree of unfreezing varies: sometimes, only the uppermost layers are unfrozen, while in other scenarios, the entire model becomes trainable. Both the pre-trained model and the newly adapted classifier undergo fine-tuning on a dataset specific to the target task. This step is critical for re-calibrating the model's weights to more closely align with the particular requirements of the target task [Rud19].

A notable concern when fine-tuning on small datasets is the potential for overfitting. In this case, the model becomes too specialized to the target training data. Careful monitoring for overfitting is essential. To combat overfitting, the fine-tuning learning rate is typically set lower than in pre-training phase. This conservative approach helps preserve the valuable knowledge the model has previously acquired, allowing for minor yet precise adjustments in the model's weights. Another challenge is the risk of catastrophic forgetting. This occurs when the updates made during fine-tuning cause the model to lose the knowledge it had acquired during pre-training [PG22]. Strategies to mitigate this include Gradual Unfreezing (GU) or discriminative fine-tuning, which involves setting different learning rates for different layers to ensure each is adjusted to an optimal extent [Rud19].

Scheduled Unfreezing: Scheduled unfreezing is an approach used in deep neural network training, particularly valuable for adapting a pre-trained model to a new task or dataset. This technique involves various methods, but the core principle is the sequential unfreezing of the model's layers during different stages of training. Such a progressive strategy facilitates a more bespoke and stable transition to new tasks or datasets.

Gradual Unfreezing (GU) is a specific form of this concept, where the layers of the model are methodically unfrozen from the top layer downwards during the training process. This approach allows for smoother adjustments to individual layers, rather than modifying the entire model at once. A primary objective of GU is to prevent catastrophic forgetting, enabling the model to integrate new tasks while retaining its existing knowledge base [LPVG23]. An alternative method proposed by Kumar et al. starts with training only the classification layer, subsequently extending to the entire model [KRJ+22]. This technique is particularly effective for managing distribution shifts and performs well with both in-distribution and distribution-shifted evaluation data. In scenarios with distribution shifts, Surgical Fine-Tuning is an effective technique for adapting a pre-trained model to the new task. This approach involves selective fine-tuning of specific layers, based on the type of distribution shift encountered. The key advantage of Surgical Fine-Tuning lies in its ability to preserve existing learned features while efficiently adapting the model to new tasks [LCT+23].

B.3.2 Domain Adaptation

A fundamental assumption in machine learning is that training and test data are independently and identically distributed (i.i.d.). However, this assumption often does not hold in real-world scenarios, where the data distribution encountered during testing can significantly differ from that seen during training. Most existing datasets are designed for evaluating a model's ability to interpolate within the training distribution rather than to extrapolate to entirely new distributions. Interpolation involves making predictions within the range of the training data, while extrapolation requires the model to make predictions for data points that fall outside the range of its training experience. This mismatch gives rise to the challenge of domain adaptation, which focuses on enabling models to adapt from one distribution (training) to another, distinct distribution (testing) [Rud19].

The variation in distributions between source and target domains can take various forms. For example, a model trained with high-resolution images might struggle with lower-quality, noisy or blurry test images. Also, training on images with minimal background clutter may inadequately prepare the model for complex, real-world backgrounds. Viewpoint differences present a further challenge. A model accustomed to one angle, like a frontal view, may falter when presented with different angles, such as side or overhead views. There are many more examples of domain shifts, including synthetic to real data.

Domain adaptation is generally studied in an unsupervised setting where a sufficient number of labelled examples in the source domain and only unlabelled examples in the target domain are assumed to be available, known as Unsupervised Domain Adaptation [Rud19]. The goal is to adapt a model from a labeled source domain to an unlabeled target domain. Traditional domain adaptation strategies often include manually adjusting or selecting samples from the source domain to better match the target domain, and modifying the set of features used by the model so they are more similar to those in the target domain. Contemporary methods in domain adaptation often integrate specialized adaptation modules within deep neural network architectures. These modules are specifically engineered to mitigate the disparities in data distributions between various domains [JSWL22].

In the Synthnet RGB image classification pipeline, two domain adaptation methods were utilized. While these methods were not replicated for the adapted RGB-D model discussed in this research paper, it is pertinent to acknowledge them, as they hold potential for future application.

Minimum Class Confusion: MCC is a loss function designed to reduce confusion across different classes. Its goal is to ensure that each example is distinctly classified into a single class, thereby avoiding ambiguity in classification. It employs a weighted softmax function that factors in the certainty of each classification. This weighting system gives lower importance to uncertain samples, thus focusing on and reinforcing more confident predictions. This strategy helps in minimizing errors arising from uncertain classifications, enhancing the overall accuracy of the model [JWLW20].

Conditional Domain Adversarial Network: CDAN is a domain adaptation model comprising a domain discriminator and a classifier. The discriminator differentiates between source and target domain data, while the classifier is trained to create representations indistinguishable by the discriminator. This process encourages the model to learn features that are common to both domains, enhancing its ability to generalize and perform accurately on the target domain data. It uses a multilinear map to combine features and classifier predictions before feeding them into the domain discriminator. This process ensures that the domain discriminator considers not only the features but also their correspondence to the predicted labels. In other words, it aligns the joint feature-label distributions between the source and target domains. This approach is particularly effective when the relationship between features and labels changes between the source and target domains [LCWJ18].

B.4 Monocular Depth Estimation

RGB-D data refers to the combination of RGB data with depth data. It provides a multimodal 4 channel representation of an image containing color, texture and shape information. The D channel in RGB-D images is a 2D greyscale image where each D pixel's value corresponds to the distance from the camera to the surface point.

Depth data is traditionally acquired using different techniques. One of these is structured light, which projects patterned light onto a scene and measures changes upon its return to calculate depth. Another is active stereoscopy which uses two viewpoints (like human eyes) to measure depth through the differences in images. Time-of-Flight measures the time taken for light to travel to the object and back and LiDAR uses laser pulses to measure distances. All are effective, but before Microsoft Kinect, which utilizes structured light or time-of-flight, was launched in November 2010, depth data collection was very expensive and cumbersome. It often required complicated custom setups and costly, unportable 3D scanners [Fir16]. Considering that many consumer electronics, including iPhone cameras, are already making LiDAR scanners a standard feature, D data will become easier to acquire in the future. Nonetheless, today most image datasets do not contain depth information. This can be amended with a technique called monocular depth estimation.

Monocular depth estimation involves predicting the depth of a scene from a single RGB image. This technology has gained substantial attention due to its practical applications. It does not require using physical depth sensors, thereby reducing the size and cost of computer vision systems' setups. The process begins with a single two-dimensional image as input. Unlike stereo vision methods that use two images from different viewpoints, monocular depth estimation relies on cues such as edges, textures, and object boundaries within single images to infer depth. Utilizing networks like CNNs or Transformers, they extract features to estimate depth at each pixel. The network is trained on numerous images with known depth maps, enabling it to generate depth maps for new images.

B.4.1 Zoe Depth

One of these estimators is ZoeDepth, which was introduced in the 2023 paper *ZoeDepth: Zero-shot Transfer by Combining Relative and Metric Depth* [BBW+23]. This method excels in its ability to adapt across various environments, addressing a significant challenge in the field of depth estimation. In essence, ZoeDepth merges two distinct depth estimation techniques: Relative and Metric Depth Estimation (RDE and MDE). MDE focuses on determining depth in absolute physical units, like meters. This is advantageous for practical applications in computer vision and robotics, including mapping, planning, navigation, object recognition, 3D reconstruction, and image editing. However, a common challenge MDE models face, especially when trained on diverse datasets, is that they often struggle when these datasets vary significantly in depth scale. One such example would be the difference in scale between indoor and outdoor images. As a result, MDE models tend to be overly specialized and lack generalizability. RDE on the other hand, addresses these scale variation issues by disregarding scale and allowing consistent depth predictions across various frames and scenes. RDE models can be trained on a wide array of scenes, enhancing their adaptability. However, they fall short for tasks requiring specific metric depth, as they provide depth estimations without a known scale and shift, rendering these predictions metrically meaningless. ZoeDepth aims to integrate the strengths of both approaches. Its zero-shot transfer capability enables the model to accurately adapt to new environments and datasets without additional training or fine-tuning while maintaining metric scale [BBW+23].

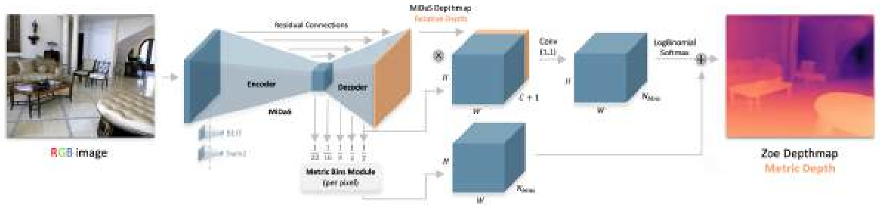


Figure B.6: ZoeDepth Architecture [BBW+23]. An RGB image is fed into transformer backbone. The bottleneck and succeeding hierarchy levels of the decoder hook into metric bin modules that compute per-pixel depth bin centers which are linearly combined to output a metric depth estimation.

ZoeDepth's functionality unfolds in two stages. Initially, it employs a MiDaS [RLH+20] encoder-decoder architecture for relative depth estimation, learning from a broad range of datasets to ensure good generalization. In the subsequent stage, the architecture is augmented with heads for metric depth estimation, which are fine-tuned on metric depth datasets. These metric heads are lightweight and domain-specific, accounting for less than 1 percent of the backbone's parameters. During inference, images are automatically directed to the relevant head using a classifier based on encoder features. This dual approach allows the model to benefit from relative depth pre-training, while accurately learning metric depth. For RDE backbone pre-training, ZoeDepth uses a mixture of 12 datasets, encompassing a variety of environments and conditions. For MDE fine tuning, ZoeDepth primarily uses two datasets: NYU Depth v2 for indoor scenes or KITTI for outdoor environments [BBW+23].

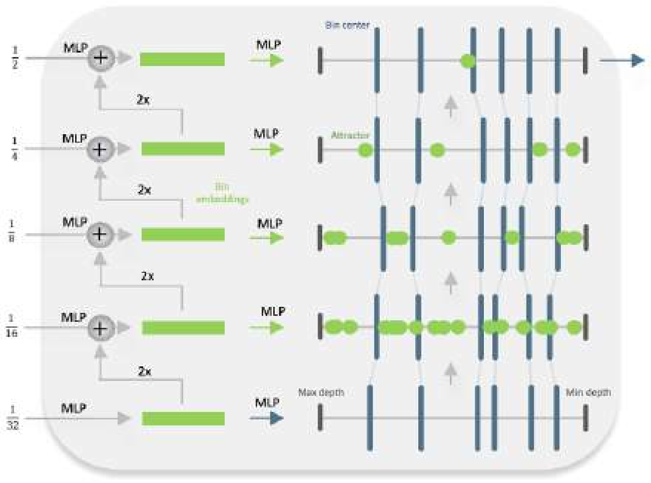


Figure B.7: Metric Bins Module [BBW+23]. Five incoming channels, corresponding to different depth hierarchies, are converted to 1-dimensional bin embeddings (green boxes). The lowest bin emebdding yields metric bin centers (blue, vertical lines), whereas the remaining embeddings provide attractors (green dots).

The metric depth estimation itself is achieved through a metric bins module, which adopts the adaptive binning principle [FBAW21]. This module takes multi-scale features from the decoder and predicts bin centers for metric depth prediction at the bottleneck stage. ZoeDepth processes inputs from five channels, each corresponding to different depth hierarchies, into 1-dimensional bin embeddings using MLPs. This is coupled with upsampling and addition operations. The lowest bin embedding determines the metric bin centers, while the others provide attractors at different hierarchy levels to adjust the bin centers. Final depth at a pixel is obtained by a linear combination of the bin centers weighted by the corresponding predicted probabilities [BBW+23].

B.5 Summary

This chapter provided an overview of the most important technologies and methods employed during the experiments in this project. It covered classification tasks, the development of vision transformers, transfer learning strategies, and the advancements in D data generation.

C Topex Dataset

This chapter provides a closer examination of the Topex dataset and its D extension. It outlines the basic characteristics of the dataset, consisting of CAD-rendered training images and a real photograph test set. It also explores some of the unique challenges Topex presents, including class imbalances, inter-class similarities, intra-class variances, the significant domain gap between synthetic training and real-world test images as well as other examples of train-test discrepancies.

C.1 Basic Characteristics

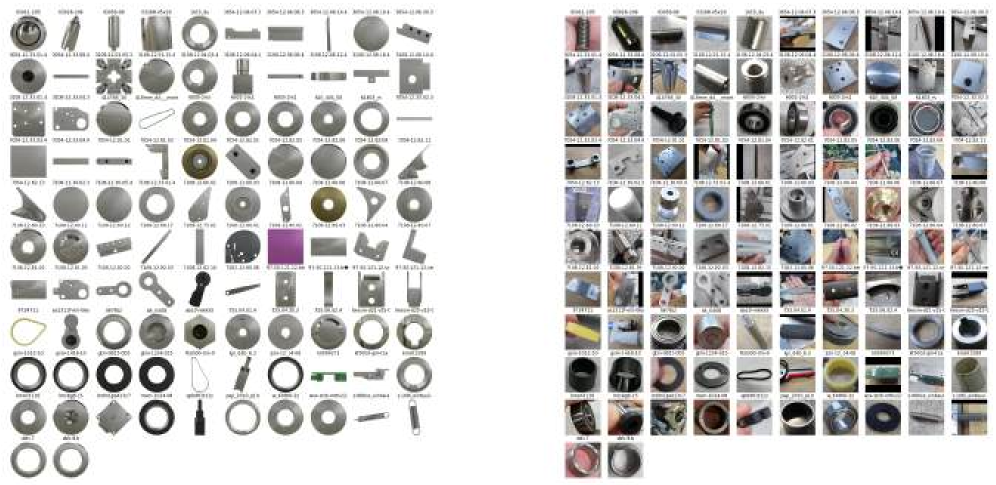


Figure C.1: Topex Train and Test Datasets Overview

The Topex dataset, created by Dennis Ritter et al. [RHH^{+}23], is a challenging image collection built to identify machine parts from photographs. There are 102 classes. Each class represents a different machine part from a labeling machine. The dataset is divided into training and testing sets.

Train Set Characteristics:

The Topex training set comprises 3,264 CAD-rendered images, each at a resolution of 512x512 pixels and stored in PNG format. The uniform white background in each image highlights the objects, minimizing visual distractions. Each of the 102 classes is represented by 32 images, offering varied perspectives and orientations to encapsulate a comprehensive range of object appearances. The rendering process, conducted through a Blender-based pipeline, incorporated environment maps, lights, and virtual cameras to simulate realistic mesh orientations. Additionally, objects were textured with one of 21 selected materials, enriching the dataset with varied surface properties.

Test Set Characteristics:

The Topex test set contains 6,146 real photographs at a lower resolution of 256x256 pixels, also in PNG format. Each of the 102 classes contains 6 to 137 images. These images mimick diverse real-world situations, using different cameras under varying lighting conditions in front of different backgrounds. This diversity introduces realistic challenges such as blurriness, color variations and obstructions (e.g., fingers in the frame).

Topex 3D:

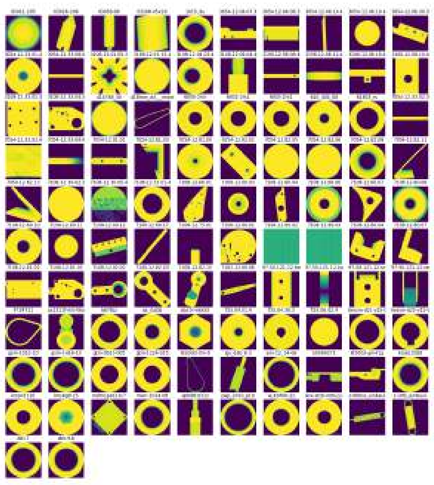


Figure C.2: Topex D Train Dataset Overview

The Topex RGB synthetic training dataset was extended by Dennis Ritter for this project to include D maps. The data was rendered through a Blender based pipeline. Two D datasets were created, one containing single channel greyscale images in PNG format, employing 0-255 normalization to facilitate integration and usability, the other, without normalization, containing EXR files to allow for more accurate depth representation.

C.2 Dataset Challenges

The dataset was designed to simulate synthetic to real data domain gap problems. It also highlights real-world complexities and common computer vision dataset issues. These include a limited dataset size, test class imbalances, inter-class similarity, intra-class variance and significant discrepancies between training and testing sets.

C.2.1 Class Imbalance

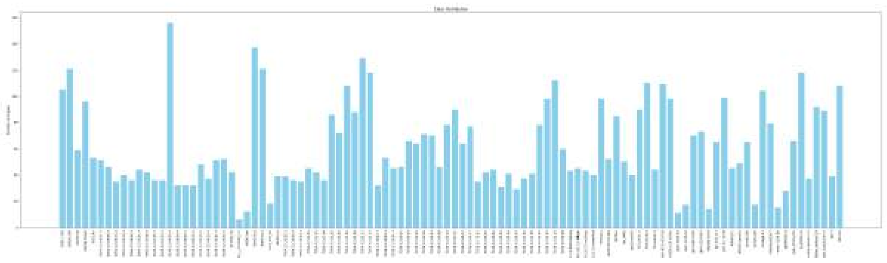


Figure C.3: Class Distribution in the Test Set

Unlike the training set, which maintains a balanced distribution of images across classes, the test set exhibits substantial variability, with some classes represented by as few as 6 images and others by as many as 137. This imbalance complicates the accurate evaluation of a model's performance. Common performance metrics like Acc@1 become unreliable in this context. For instance, if a few classes are over-represented in the test set, a model might achieve high accuracy by predominantly predicting these majority classes correctly while failing to recognize instances of underrepresented classes. This scenario can create a false impression of the model's effectiveness. To mitigate this issue, the adoption of a broader spectrum of evaluation metrics is necessary.

C.2.2 Inter-Class Similarity

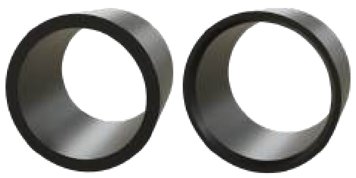


Figure C.4: Object gsm-1012-10 (left) and gsm-1416-10 (right)

A quick visual inspection of images from each class reveals that cases of extreme inter-class similarity are prevalent across the entire dataset. Many machine parts share attributes like color, texture, and shape, while differing only slightly in size (see for example figure C.4) or minor detailing (see for example figure C.5). There is a strong concern that this will result in the model learning similar feature embeddings for classes with similar-looking objects. During inference, this could cause substantial class confusion.

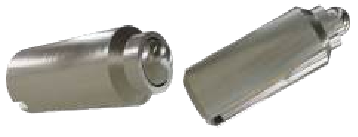


Figure C.5: Object 03011-205 (left) and 03026-206 (right)

C.2.3 Intra-Class Variance

Upon further inspection, it also becomes evident that there is a notable variation within classes. For instance, flat objects can be oriented differently (see for example figure C.6). These instances may pose challenges in recognizing images as part of the same class.

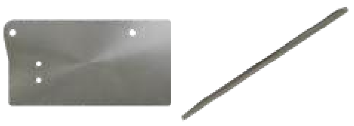


Figure C.6: Two Images of Object 7108-12.81.02

To get a quick overview of the extent of the problem the distribution of Histogram of Oriented Gradients (HOG) features per class was visualized (see graph C.7). HOG descriptors excel in identifying edge information, a key element in object distinction within images. By quantifying the frequency of gradient orientations, HOG features can effectively signify the presence of distinct object features within an image. In graph C.7, the aggregate count of HOG features for each image is categorized under its corresponding class.

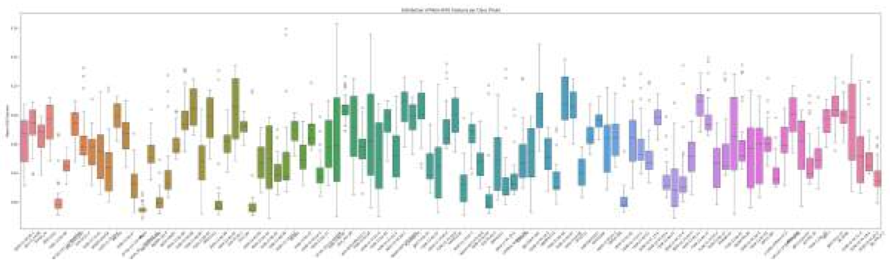


Figure C.7: Distribution of Mean HOG Features per Class (Train)

Notably, some classes display a wide interquartile range (IQR), reflecting substantial diversity in shape and form within these groups. This variation is further accentuated by the presence of numerous outliers, which may signify images whose edge content significantly diverges from the typical attributes in that class. Such a high degree of variance might present challenges for the model in comprehensively capturing and understanding the full spectrum of visual features within these classes.

C.2.4 Domain Gap

Generally it is clear, that the Topex dataset exhibits a domain gap between its synthetic training and real photograph test sets. The training set, composed of CAD-rendered images, offers a consistent and controlled visual quality with visible efforts at photorealism. In contrast, the test set, comprising real-world images, displays significantly more variation in aspects such as lighting, contrast, occlusions (like partially covered parts), natural wear and tear of objects and particularly background. The training set introduces its own form of variance, albeit synthetically. This is achieved through the manipulation of textures, lighting conditions, and object orientations. It remains to be seen whether these synthetic techniques are sufficient to bridge the domain gap effectively. In synthetic data research, there is an active discussion about the relative significance of photorealism compared to diversity. Some theories suggest that adding a number of real images to synthetic training datasets can significantly improve diversity and thereby model performance [Nik19]. While this approach was not adopted in the current project, the dataset was seemingly designed with the aim of attaining high levels of photorealism and diversity.

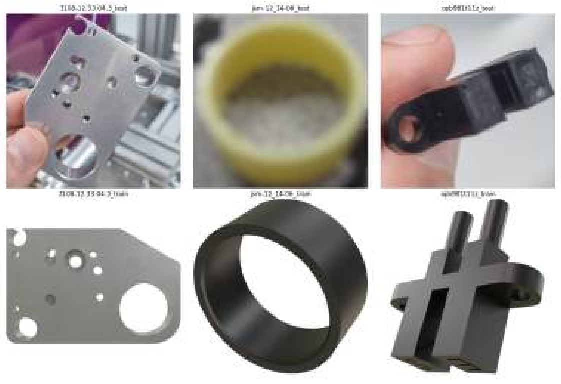


Figure C.8: Topex Train-Test Comparison

The visually most striking difference between domains is certainly the blank background in all training set images. It serves to highlight the object of interest, eliminating potential distractions and simplifying the learning process by providing clear, unambiguous representations of objects. However, it may also be inadvertently learned by the model as a feature of interest, thereby diluting the importance of the actual discriminative features. A model trained on such images might struggle when presented with complex backgrounds found in real photographs. A further danger lies in the use of data augmentation techniques, such as random cropping. When applied to images with large amounts of blank space, such techniques could unintentionally result in crops that exclude the object entirely.

Another feature of the test which was not replicated in the training set was the considerable presence of blur. A blur detection algorithm developed by S. Alireza Golestaneh and Lina J. Karam [GK17] revealed an average blurriness per class of 14% for the test dataset. The influence of such variations on model performance should not be underestimated. For instance, research conducted by Vasiljevic, Chakrabarti, and Shakhnarovich indicates that standard pre-trained CNN models often suffer substantial performance decline when dealing with blurred test images [VCS17]. They suggest integrating blurred images into the training phase to boost the model's proficiency in handling such distortions. Within the context of the Topex dataset, the lack of intentionally blurred effects during the image creation process is a significant aspect to consider. A slight downturn in performance regarding blurred image recognition might be expected.

C.2.5 Train-Test Discrepancies

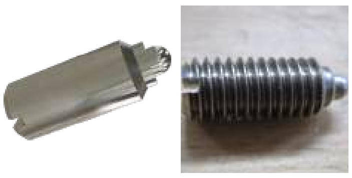


Figure C.9: Object 03026-206 Training (left) and Testing (right) Images

In examining the training and testing sets, several dataset-specific discrepancies were observed, distinct from the general synthetic-to-real differences. Notably, all threaded objects were missing their threading in the training data (refer to figure C.9). Additionally, elements such as rubber casings were often missing in the training dataset (see figure E.5). This suggests that the CAD model blueprints might not possess the necessary detail to accurately replicate many of the objects. Unsurprisingly, the addition of 21 textures during the rendering process was, on first inspection, unable to fully compensate for these deficiencies.

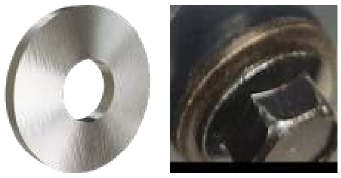


Figure C.10: Object wsx-stcb-m8x22-2.3 Training (left) and Testing (right) Images

In some cases, training images did not feature threading, while in others, an entire screw was unexpectedly included in some of the test images, thereby fundamentally changing the object in question (see figure C.10).

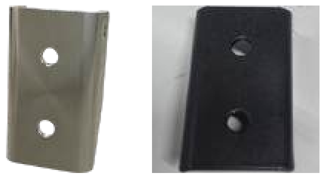


Figure C.11: Object 97-50-121-12-befestigung Training (left) and Testing (right) Images

Discrepancies in color and material also contribute to the observed differences between training and testing sets. For instance, object "97-50-121-12-befestigung" is shown in chrome in the training set, but appears in a different black metal in the testing set (see figure C.11). These instances highlight only a small portion of the discrepancies present in the dataset, which are expected to adversely affect the Acc@1 scores significantly.

C.3 Summary

This chapter explored the Topex dataset and its D extension, offering an analysis of its basic structure as well as the challenges it presents for image classification tasks. These challenges included class imbalance, inter-class similarity, intra-class variance, the domain gap and other train-test discrepancies.

D Methodology

This chapter outlines the methodology employed in this study for integrating D data into the image classification pipeline. It starts by detailing the monocular depth data estimation process using the ZoeDepth framework. The chapter then explains the various stages of data processing, including dataset splits, the use of the Transformers library's AutoImageProcessor, and the synchronous application of a selection of data augmentation techniques. The chapter further explores the application of the chosen classification model, SwinV2, and describes the implementation of two data fusion techniques, early and late fusion. The finetuning strategies adopted are discussed in detail, including classification head tuning, full model tuning, the combined approach as well as the use of alternative unfreezing schedules. General implementation details are listed at the end of the chapter.

D.1 Depth Data Generation

Given the impracticalities and high costs still associated with depth-sensing hardware, and the typical lack of available of CAD rendered D maps, monocular depth estimation was chosen as a cost-effective yet viable alternative. Specifically, the ZoeDepth model was chosen for this purpose.

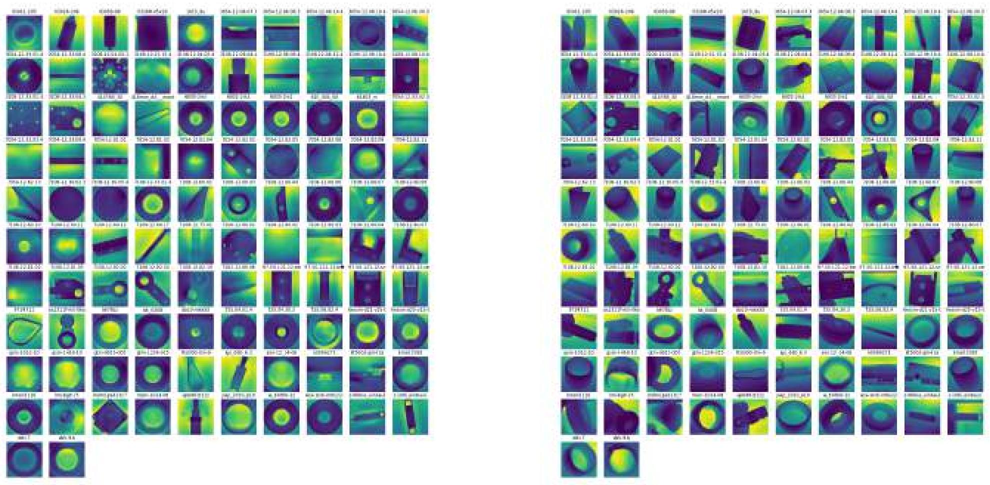


Figure D.1: Depth Maps Generated by ZoeDepth for the Topex Training and Testing Datasets

As discussed in chapter B.4.1, ZoeDepth employs advanced deep learning algorithms to produce depth maps from single RGB images. The decision to utilize ZoeDepth was driven by multiple considerations. Chief among these was its demonstrated capability of accurately estimating metric depth. Another pivotal aspect was its compatibility with conventional computing hardware, essential for the precise classification of machinery components under varied operational conditions.

Considering the Topex dataset characteristics, as well as the scope of this research project, customizing the monocular depth estimation model to this particular dataset seemed impractical. In many projects, especially those with limited resources or requiring rapid deployment, fine-tuning a model for each specific dataset is often unfeasible. Luckily, ZoeDepth was trained on an extensive number of image datasets with established depth values [2], endowing it with the flexibility to adapt to a variety of conditions. Therefore, ZoeDepth's advertised ability to perform effectively across diverse datasets without the need for extensive customization made it an ideal choice for this study's objectives.

In the first step of the project's workflow (see figure A.2), the Topex RGB images were used by ZoeDepth to infer estimated D values for each pixel. ZoeDepth's metric heads were trained either on the NYU Depth v2 dataset for indoor depth estimation or the KITTI dataset for outdoor scenes [BBW+23]. Considering this study's focus on small machine parts photographed within a 10-meter range, the NYU Depth v2 metric depth estimation head was applied to receive the most accurate D information.

After inference, the processing of D maps followed two distinct paths. The first involved normalization using OpenCV. The D data underwent min-max normalization to be scaled to a range of 0 to 255, enabling its storage in PNG format and facilitating its processing by the transformer. The D maps resolutions adhered to the original dataset's specifications of 512x1512 pixels for training images and 256x256 pixels for testing images. For visual representation in this paper, Matplotlib's default colormap was used on the normalized D maps (refer to Figure D.1). In practice, the classification models processes single-channel greyscale depth data.

Alternatively, the ZoeDepth inferred D maps were not min-max normalized, thereby maintaining absolute metric depth information. The maps were save in pytorch .pt format. Despite maintaining more accurate data, depth maps that are not normalized present certain challenges due to their deviation from standard image formats. It was unclear, whether the employed models, which were pre-trained on

[2]These datasets include HRWSI, BlendedMVS, ReDWeb, DIML-Indoor, 3D Movies, MegaDepth, WSVD, TartanAir, ApolloScape, and IRS

normalized images, would be able to process these metric maps effectively. Also, ZoeDepth relies heavily on its backbone for relative depth prediction, with metric depth fine-tuning limited to heads, which constitute less than 1 percent of the total parameters [BBW+23]. Whether this configuration would suffice to enhance the Acc@1 of the Synthnet pipeline remained to be seen.

It is important to acknowledge that monocular depth estimation is an approximation technique and may produce lower quality depth maps compared to specialized depth-sensing equipment or synthetic renderings. The influence of depth image quality on classification accuracy remains an underexplored area. Existing research indicates that depth images generally enhance model performance, particularly in saliency prediction, but low-quality depth images might adversely affect outcomes [JCP22]. A comparison of ZoeDepth-generated depth maps against Topex 3D data will be undertaken during evaluation in chapter E.[3]

D.2 Data Processing

D.2.1 Data Split

For the Topex dataset, a traditional train/test split was not required, because the synthetic source and real target data are inherently different, and their segmentation into training and testing sets is an intrinsic characteristic of the dataset itself. Additionally, this study does not prioritize hyperparameter optimization, thus eliminating the need for a separate validation set for this purpose. Instead, the hyperparameters from the Synthnet pipeline were applied to the RGB-D models. This allows for a straightforward comparison between the original Synthnet model and its RGB-D variant.

However, given the time and resource constraints involved in conducting evaluations on the entire, 6,000 image, test dataset for each epoch, a smaller validation set was required. Various strategies were explored to create an efficient validation set, with the goal of increasing training speed without compromising on model performance. The term "validation set" here specifically refers to the subset of data used for calculating validation loss during the training process.

Several strategies were considered. For instance, using a *Partial Source Domain Validation Set* could have preserved the entirety of the test set for the evaluation phase. However, this method was prone to overfitting to the source domain. Another possibility was a *Mixed Source and Target Domain Validation Set*, which

[3]This comparison is limited to the training set, as Topex 3D's test set wasn't derived from CAD models and thus lacks rendered D data. For all experiments, the test D data was solely generated by ZoeDepth.

would merge images from both the training and test sets for validation purposes. This could potentially offer a more balanced evaluation of the model's ability to generalize from the source domain while minimizing the trade-off in Acc@1 on the target domain. Yet, this idea was abandoned due to unexpected challenges in achieving a balanced mix of images from both domains.

Ultimately, the strategy employed was the *Partial Target Domain Validation Set.* It involved utilizing a segment of the target domain (test set) for the purpose of validation. Here, a fixed number of images (the last five of each class, i.e., 12.05% of the total test set) were earmarked from the test set (with 6 to 137 images per class) for validation, resulting in an evenly distributed validation set. Alternatively, 15% of images from each class could have been to extracted, mirroring the distribution of the actual test set. This would have preserved more images from those classes with a low number of test images. However, a uniformly distributed validation set was preferred.

D.2.2 Data Augmentations

Image data augmentation addresses two pivotal challenges in training machine learning models. The first challenge is the scarcity of data, which may result in a model overfitting to the few available training samples. Data augmentation mitigates this by generating varied versions of existing images, thus enlarging the dataset and improving the model's generalization capabilities to unseen data. This approach was not applied here, but could be considered for future experimentations.

The second challenge pertains to the variety within each labeled category in the dataset. Each image in a dataset carries a specific class label. Data augmentation methods create modified versions of these images while retaining their original labels, enhancing the diversity within each category. This strategy grants the model a more rounded comprehension of each class [KMBB23]. This form of augmentation was employed in this project, albeit without increasing the total count of training images.

The Synthnet pipeline includes a suite of data augmentation techniques, which it calls transforms. For the purpose of conducting a comparative study, all data augmentations used by the Synthnet pipeline were replicated in this project. The augmentations cover a wide range of operations, simulating a variety of real-world variations and thereby preparing the model to handle diverse and unforeseen scenarios. They include geometric transformations like rotation, non-geometric transforms such as flipping, cropping and jitter or Multi-Images Mixing techniques such as Augmix. The ensuing section details the specific transforms employed in the experiments, their theoretical underpinnings, and their practical implications. The exact application and combination of these augmentations in the experiments are elaborated in chapter E.2.1.

Random Resized Crops: Synthnet utilizes this transform with a relative scale range of (0.7 - 1.0). For each image, the augmentation randomly selects a scale within the specified range. Alongside the scale, it also randomly selects an aspect ratio (the ratio of width to height) for the crop. Based on the randomly selected scale and aspect ratio, a crop is extracted from the original image. The location of this crop within the image is also chosen randomly. The extracted crop is then resized to a predetermined size. This variability in scale and perspective encourages the model to identify and learn scale-invariant features. Nevertheless, it's important to apply cropping judiciously as a method of data augmentation. Inappropriate use of cropping, such as highlighting blank space, can inadvertently lead to confusion for the model [KMBB23].

Random Horizontal Flip: This transform randomly flips images horizontally. "Random" in the default setting implies a 50% chance that it will or will not be flipped. This process introduces spatial variability, training the model to recognize features irrespective of their horizontal orientation.

ColorJitter: In Synthnet color jitter's parameters, brightness, contrast, saturation and hue were all set at 0.3, allowing values to randomly vary within +/- 30% of the original values for each image. The randomness ensures that each image can have different levels of brightness, contrast, saturation, and hue, making the model adaptable to various lighting and color conditions.

Random Greyscale Conversion: Here, RGB images are randomly converted to greyscale. 'Random' implies that there is a 10% probability (default setting) that the greyscale transformation is applied to any given image. This enhances the model's focus on learning from textures and shapes as opposed to relying heavily on color information.

AugMix: Augmix is an advanced image augmentation technique designed to enhance the robustness and generalization of machine learning models, particularly in image classification tasks with unforeseen data shifts [HMC$^+$20]. The specific augmentations used in AugMix typically include basic image operations like rotation, shearing, translating, or resizing the images. They also include color transformations such as changing brightness, contrast, saturation, or hue. AugMix uniquely combines these transformations by blending different augmented versions of the same image together. This blending is done using weights. AugMix also introduces randomness in the selection and application of these augmentations. This means that each time an image is processed through AugMix, it may undergo a different set of transformations, further enhancing the diversity of the training data [HMC$^+$20].

Normalization: The final tensors were normalized using the specified Synthnet standard deviation [0.5, 0.5, 0.5] and mean [0.5, 0.5, 0.5]. Alternatively, ImageNet's standard mean [0.485, 0.456, 0.406] and standard deviation [0.229, 0.224, 0.225] values were tested and did occasionally result in slightly better performance.

D.2.3 Synchronized Transforms

The specialized architectures of the RGB-D models necessitated the development of custom augmentations, rather than relying on the standard augmentations available in Pytorch. Since RGB and D images had to be processed simultaneously prior to being fed into the model, the independent application of Pytorch's random transforms led to discrepancies, such as RGB images being flipped while D images remained unchanged. These inconsistencies significantly impacted the model's performance. To overcome this issue, a bespoke solution named *Synchronized Transforms* was created. This approach involved manually replicating all required out-of-the-box PyTorch image augmentations. It ensured compatibility with Synthnet's PyTorch augmentations for RGB images while allowing corresponding RGB and D images to undergo identical augmentations (see figure D.2).

The introduction of *Synchronized Transforms* also enabled the targeted application of specific transforms, ensuring that augmentations were equally pertinent and beneficial across both RGB and D modalities. For instance, specific transformations such as color jitter, which do not apply to D data, were selectively used for RGB images only. Additionally, applying a greyscale transformation to depth images, which are inherently greyscale, was redundant and therefore omitted. A *Synchronized Augmix* technique was also implemented, drawing inspiration from the original AugMix but with modifications. This approach encompasses operations like AutoContrast, Equalize, Rotate, Solarize, Color, Contrast, Brightness, Sharpness, and an identity function, while deliberately leaving out more intricate operations such as Shear and Translate found in PyTorch's AugMix. In this study, "Synchronized AugMix" was applied to normalized D data, though its use should be reevaluated for future applications. It was not applied to metric D data because of image formatting challenges and doubts about its applicability in this context.

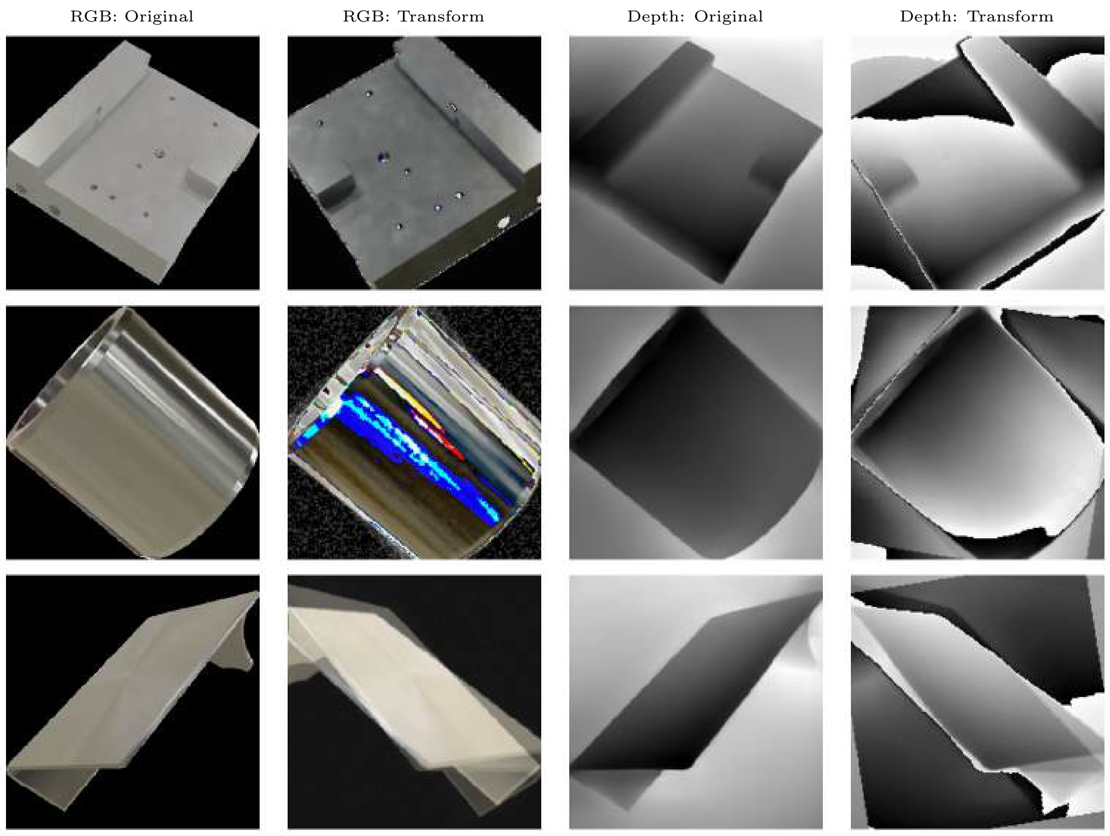


Figure D.2: Synchronized Transforms applied (Random Resized Crop, Random Horizontal Flip, Augmix, to Tensor, Normalized)

D.2.4 Transformers' AutoImageProcessor

The AutoImageProcessor from the Transformers library, designed specifically for the selected model, was used to meet only the core image preparation demands of the SwinV2 model. This processor was initialized via AutoImageProcessor. from_pretrained ("microsoft/swinv2-base-patch4-window12-192-22k"). The decision to use this particular processor, which was not used by Synthnet, stemmed from initial reservations about the applicability of several Synthnet transforms on D data as outlined in chapter D.2.2 and D.2.3. Therefore, the integration of a simple pre-processing option addressing all basic necessities offered an easy implementation and benchmark to evaluate the effectiveness of Synthnet's augmentations.

Key configurations of the processor include return_tensors set to "pt", and the activation of do_normalize, do_rescale, and do_resize. The conversion of images into PyTorch tensors aligns with the structural needs of the PyTorch framework.

Rescaling transforms the pixel value range of images from their original scale to [0, 1]. Normalization adjusts these pixel values to a standard range specifically suitable for the SwinV2 model. The processor utilizes the ImageNet standard mean and standard deviation values: image mean is set to [0.485, 0.456, 0.406], and image std is set to [0.229, 0.224, 0.225]. These values correspond to the RGB channels and are derived from a statistical analysis of the ImageNet dataset. This ensures that the image data is centered and scaled appropriately for models pre-trained on this dataset. In addition, the processor also resizes images to the mandatory dimension of 192x192 pixels. This resizing ensures that all input images conform to the uniform size requirements of the SwinV2 model, thereby facilitating consistent and effective processing.

D.3 Classification Model

D.3.1 SwinV2 Base Model

There were numerous options when choosing an appropriate classification model. The Synthnet pipeline focused on testing Vision Transformer models, specifically ViT, SwinV2, and DeiT. These models are notable for their advancements in performance, surpassing older deep learning models. The recent development of Vision Transformers, with ViT emerging in 2020 and SwinV2 in 2021, also means that their application to RGB-D data is still relatively underexplored [TK23]. Diverging from the original Synthnet pipeline's approach of testing multiple models, this project's modified version is limited to using only SwinV2. This choice was largely influenced by Swinv2's outstanding performance in prior Synthnet tests.

For this paper, a SwinV2 model was initialized utilizing the AutoModelForImageClassification. from_pretrained method from the Transformers library, specifically with the *pretrained model microsoft/swinv2-base-patch4-window12-192-22k*. The selection of this particular SwinV2 model, pre-trained on ImageNet 22K comprising a diverse range of 22,000 image classes, was determined by its use in the Synthnet study to facilitate comparative analysis. However, the ImageNet 22K backbone also consistently outperformed the ImageNet 1K backbone on the Topex dataset when tested.

For the purposes of this study, the model's number of output classes was adjusted to 102, aligning with the Topex dataset's requirements. The final classification layer was initialized randomly. Alternative model sizes (large or tiny/small), window sizes (16 or 8), or resolutions (256 or 384) could also have been chosen from the Transformers library, each offering different outcomes and impacts. Smaller window sizes, for example, may detect finer details in images but increase the computational load for the self-attention mechanism. Again, the selection was determined by the Synthnet experiments.

While the selected model provides a robust framework for image classification, there are potential challenges to consider. Firstly, the complexity and computational intensity of the model may pose issues, particularly when dealing with large datasets or requiring real-time processing. In such cases, more lightweight models could have served as a viable alternative, offering a trade-off between performance and computational efficiency.

Furthermore, the ImageNet dataset [DDS+09], which is known for its extensive and diverse collection of images, seems to align well with the requirements for a transfer learning pre-trained model. However, when considering Topex's similarity to ImageNet 22K, potential issues arise. Transfer learning tends to be more effective with data that closely resembles the training dataset [PG22]. While ImageNet 22K includes categories such as 'machine', 'bolt', and 'screw', it was not clear whether these categories would sufficiently prevent issues like 'negative transfer' in this context.

Another problem was that SwinV2 models are inherently designed for 3-channel RGB input, posing compatibility issues for 4-channel RGB-D data. Creating a new 4-channel SwinV2 model would necessitate extensive retraining. However, the availability of D data was limited as compared to the massive ImageNet RGB dataset. This study, instead of developing a new 4-channel SwinV2 model, concentrated on investigating and assessing various data fusion techniques as alternative approaches. These allow for easy processing of RGB-D data without extensive model modifications. The primary focus of this papers lay on early and late fusion strategies [FHSR+21].

D.3.2 Early Fusion

Early fusion, generally speaking, entails the blending of RGB and D data before being processed by the transformer's encoder. This approach facilitates the synergistic processing of RGB and D information, permitting the model to establish correlations between features from both data sources directly.

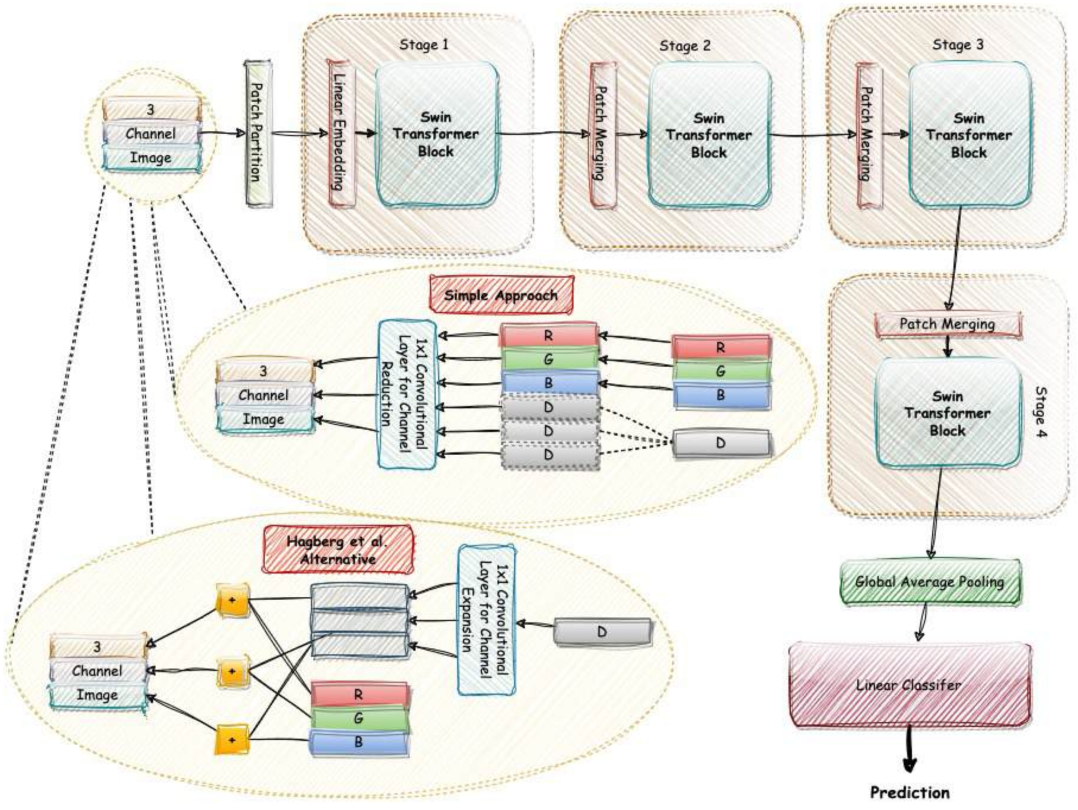


Figure D.3: Early Fusion Architecture

In the initial phase of experimentation with early fusion, a basic approach was adopted to convert the 4-channel RGB-D images into 3 channels, as illustrated in D.3. This process involved replicating the single D channel to create three identical channels. This gave additional weight to the D information. The technique can be adjusted in future experiments to align with varying requirements or hypotheses. Subsequently, the two 3-channel inputs (RGB and D) were merged into a single 3-channel output through the initial convolution layer. This layer is defined by a 1×1 kernel size and stride of 1×1, employing the Kaiming normal initialization method, and a 'fan out' and 'relu' nonlinearity mode.

This approach was designed to limit data alteration, aiming to preserve the pre-trained strengths of the SwinV2 Transformer for RGB-D data. The convolutional network in the custom model is kept simple to facilitate ease of implementation and replicability without introducing significant computational overhead. However, this method may cause the model to depend heavily on the fusion layer's ability to effectively merge the two data types. The output from this layer could be of low quality or unrecognizable to the pre-trained SwinV2 model, potentially impairing the model's feature recognition capabilities.

Several alternative approaches are available for combining RGB and D data in image classification models. For example, Hagberg et al. [HM22] propose a method where the depth channel is processed independently before being merged with RGB data. This process involves passing the single depth channel through a 1x1 convolution layer, resulting in a three-channel output akin to an RGB image. This newly formed data is then merged with the original RGB image via element-wise addition, creating an altered RGB representation.

Another, slightly more complex, approach is the 'dual-embedder' method developed by Georgios Tziafas and Hamidreza Kasaei [TK23]. In this method, each modality is processed through its respective patch embedding layer. The resulting representations are then combined using addition and L2 normalization.

D.3.3 Late Fusion

Late fusion involves the combination of RGB and D data after the each has been processed separately by the encoder. This approach differs from early fusion as it does not process the relationships and connections between RGB and D data. Nonetheless, this paper hypothesizes that late fusion is likely to be more effective, especially when fine-tuning on small to moderate-sized datasets. The rationale is that late fusion doesn't demand altering the model to adapt to a new, fused input format. Instead, it utilizes the model as it was originally trained, its original structure and learned weights. The model recognizes the addition of D data not as a fundamental change in the type of data it processes, but rather as a variation in the kind of information it needs to analyze, applying the knowledge it learned during pre-training more effectively. It can then further adapt to this new type of information during the fine-tuning stage. Late fusion also benefits from specialized, independent processing for both RGB and D data, thereby maintaining the distinct characteristics of each modality. The hypothesis that late fusion leads to better classification is supported by experiments conducted on the Washington RGB-D Objects dataset, as detailed by Georgios Tziafas and Hamidreza Kasaei [TK23]. This approach results in a more versatile and resilient models, equipped to handle diverse data types and conditions. However, the late fusion approach also has drawbacks. Generally it isn't designed to capture intermodal relationships. Furthermore, it has an increased computational demand.

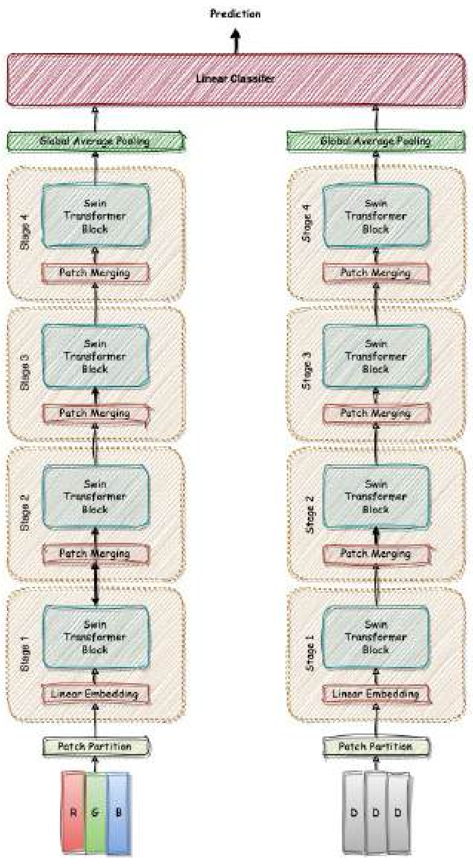


Figure D.4: Late Fusion Architecture

The late fusion model presented in this paper utilizes a Siamese Transformer framework combined with a unified Linear Classifier, as illustrated in D.3. The model consists of two parallel networks that are architecturally identical and initialized with the same pre-trained weights. Each sub-network processes the RGB and D data independently. In a manner akin to the early fusion model, the single D channel is duplicated to create three identical channels, facilitating its processing by the respective sub-network. These sub-networks are not designed to be independently fine-tuned. However, the training dynamics lead to divergent weights in each sub-network. This divergence arises, because each sub-network processes different types of data, resulting in unique gradients during backpropagation. Consequently, the sub-networks develop distinct weight profiles. This design enables the model to combine insights from both the RGB and D data for the final classification task.

To make the late fusion model operable, separate batches for RGB and D data had to be generated. The sub-networks then process the data in corresponding pairs. The outputs from the two encoders are the RGB and D respective feature maps for a single image. Hooks in the SwinV2 pooler layers capture the outputs from both sub-networks. They are concatenated before being passed to the classifier. This classifier is equipped with a linear layer adapted to handle an input feature size of 2048, twice as large as the standard classifier, and an output feature size of 102. This design is tailored to effectively accommodate and integrate the concatenated features derived from both the RGB and D data streams.

The decision to employ feature map concatenation in the late fusion model is informed by several factors. Primarily, this choice is supported by experimental evidence indicating that concatenation offers a more comprehensive representation of features from both RGB and D modalities [TK23]. The advantage of concatenation lies in its ability to preserve the integrity and fullness of features from both modalities, potentially enriching the feature set available for classification tasks.

There are several alternative fusion strategies beyond the employed approach, with key variations occurring in the post-encoder phase [STF22]. Instead of straightforward concatenation, other techniques like max pooling or averaging addition could be used to integrate multimodal data. Additionally, Roitberg et al. have suggested a Majority Voting Scheme [RPM+22]. This scheme involves determining the top-1 predicted behavior for each modality independently. The final decision is then based on the majority prediction from these unimodal classifiers. This approach maintains the independence of each modality's features and predictions and is relatively simple to implement. However, it may not fully leverage the synergistic potential between RGB and Depth data and risks introducing bias if one modality dominates the majority vote.

D.4 Finetuning

In this chapter, the methodology adopted for fine-tuning the pre-trained custom transformer models is explained. The basic training schemes are adapted from the Synthnet framework. Training is segmented into three phases: the training of a new classification head (CH), the fine-tuning of the entire model (FT), and a hybrid approach combining both strategies (CH-FT). Alternative finer-grained schedules are also explored in an effort to further enhance the performance of the hybrid approach.

D.4.1 Classification Head Tuning (CH)

The initial phase of the Synthnet fine-tuning process involves the training of a new classification head. The output of the original SwinV2 classifier is reduced from

21,841 (intended for use with the ImageNet-22k dataset) to 102 classes to align with the class count in the Topex dataset. All layers, except the classification head, are kept frozen. This approach preserves the extensive feature-extracting capabilities developed during pre-training, while adapting the model to classify the specific categories present in the Topex dataset. Classification head tuning is computationally less expensive than full training and is also expected to perform better on OOD datasets such as Topex. This effect is described in Kumar et al. [KRJ$^+$22] and can also be observed in the Synthnet results in chapter E.1. However, the decision to keep the majority of the model frozen does pose a risk of significant misalignment between the pre-trained model's knowledge and the unique characteristics of the Topex dataset.

D.4.2 Full Model Tuning (FT)

The FT stage marks a significant shift from the CH stage. Unlike CH, FT involves unfreezing and training both the feature extraction layers and the classification head of the model, allowing for a comprehensive adaptation to the Topex dataset. For this purpose, the pre-trained model is loaded and its linear classification head is replaced to accommodate the number of classes in the dataset. All model weights are unfrozen before the training process begins. Due to the results of the Synthnet model (see chapter E.1), FT is expected to perform worse than CH tuning. Nonetheless, it isn't impossible that the added D dimension requires a more thorough adaptation by the model, resulting in a better FT performance.

The hyperparameters applied during CH training are adjusted to suit requirements of a full tuning phase. The changes are detailed in chapter E.2.2. The FT adjustments to the CH data augmentations are described in chapter E.2.1.

D.4.3 Combined Approach (CH-FT)

FT typically results in enhanced accuracy with in-distribution (ID) data, where training and test sets share similar statistical properties and characteristics. However, it can lead to reduced accuracy for OOD data, where training and test sets differ significantly from one another. Recent research regarding the fine-tuning of pre-trained models for downstream tasks has shown that FT, even when conducted with extremely small learning rates, can substantially modify the features learned during pre-training. This alteration mainly pertains to ID data features, while the features related to OOD data remain largely unchanged. Consequently, this leads to the model becoming fine-tuned to ID data, making it less effective at handling OOD data. In contrast, CH tends to yield higher accuracy in scenarios involving large distribution shifts but lower accuracy on ID data [KRJ$^+$22].

The sequential CH-FT approach employed in the Synthnet pipeline and copied in this paper tries to combine the benefits of both approaches. It commences with the best-performing model from the CH stage, based on the test set's Acc@1 results, and continues fine-tuning the whole model for an additional 20 epochs. By preserving the pre-trained features with CH tuning, it ensures that the model maintains a strong baseline performance, especially on OOD data. Then, by transitioning to FT, the model further refines its decision-making ability for the specific task. Unlike straightforward FT, CH-FT alters the model's features to a lesser extent, thus better maintaining high performance OOD datasets [KRJ$^+$22].

D.4.4 Alternative Unfreezing Schedules

Expanding upon the finetuning methodologies described by Kumar et al. [KRJ$^+$22] and applied in the Synthnet pipeline, this study introduces an additional strategy that involves selectively freezing and unfreezing segments of the transformer's encoder model during CH-FT training. This dynamic strategy aims to achieve a more customized adaptation to the unique characteristics of the Topex RGB-D dataset. The method integrates concepts from gradual unfreezing, initially proposed to counteract catastrophic forgetting in transfer learning [LPVG23], and surgical fine-tuning, anticipated to yield improved results on OOD data [LCT$^+$23]. Various combinations of these techniques will be tested. This approach is pursued in the hope of being particularly advantageous for the hierarchical SwinV2 model. For this purpose, the model is segmented into several components to be frozen or unfrozen:

- **Initial Convolutional Layer**: In the early fusion model, this layer fuses the RGB and D data, combining these modalities at the first stage in the network.
- **Patch Partition and Linear Embedding Layers**: These layers segment the images into patches and generate their linear embeddings, serving as the preliminary input for the transformer encoder.
- **Encoder Stages 3-0**: The SwinV2 encoder, consisting of four stages, processes information at varying scales, beginning with smaller patches and gradually amalgamating adjacent patches in subsequent layers.
- **Global Average Pooling Layer**: Aggregates the SwinV2 transformer's encoder outputs into a single comprehensive feature map.
- **Classifier**: Situated at the end of the model, the linear classifier makes the final output predictions.

A grid search was conducted to examine diverse combinations of frozen and unfrozen layers within the model. Again, the training commences with the best-performing model from the CH stage and then fine-tunes the whole model. However, instead of all layers being unfrozen during FT, different combinations are tested. Initially, the encoder layers underwent systematic freezing and unfreezing, starting either from the later stages and moving towards the initial layers, or vice versa.[4] Subsequently, the experiment was extended to include the Initial Convolutional Layer, Patch Partition and Linear Embedding Layers, and Global Average Pooling Layer, aiming to amplify the models' learning and representational effectiveness. In the third phase, based on preliminary outcomes, surgical fine-tuning, i.e. unfreezing only single or pairwise encoder stages, was applied. Considering the domain shift in the Topex dataset and the findings of Lee et al. [LCT$^+$23], targeting the early and middle layers of the model was expected to yield the best results.

It is important to note that hyperparameters only change from CH to FT runs. Adjusting these parameters for each freezing configuration would require extensive work and is beyond the scope of this study. Another important piece of information is that for each new training phase with different freezing configurations, the minimum validation loss was reset to infinity. This ensures that the initial run in each training phase produces a minimum validation loss score, even if the model achieved a better score in a previous training phase. Although this approach may have its drawbacks, it also presents a chance to avoid local loss minima.

D.5 Implementation

This chapter delves into the environment and tools that were instrumental in the development and execution of the experiments. It includes an overview of the software stack, the versions of the primary tools and libraries used, and the use of the BHT Kubernetes cluster to facilitate efficient model training.

D.5.1 BHT Cluster

The BHT Kubernetes cluster, operated by DATEXIS, was utilized for executing experiments. Equipped with a range of Nvidia Tesla GPUs, including K80, P100, and V100, the cluster provides a robust platform for neural network training. The models primarily leveraged the Nvidia Tesla V100 GPUs for their superior computational efficiency. Data management was handled through Persistent Volume Claims, allowing for efficient storage and management of the various datasets.

[4]The Classifier remained unfrozen throughout.

D.5.2 Transformers Library

The Hugging Face Transformers library, version 4.24.0, was utilized extensively. This Python library, specifically designed for PyTorch, offers a comprehensive collection of pre-trained models and utilities for Computer Vision tasks. Its ease of use and seamless integration with deep learning frameworks like PyTorch, made it an invaluable tool for the project.

Specifically, the AutoModelForImageClassification module was employed to create image classification models using the microsoft/swinv2-base-patch4-window12-192-22k settings. Additionally, the AutoImageProcessor component was utilized in certain experiments.

D.5.3 Software Stack

The development, training, and evaluation of the models were supported by a carefully chosen set of tools. All code was executed through simple Python scripts, ensuring straightforward and effective management of the experimental workflows. Python version 3.10.9 served as the project's backbone, offering versatility and a comprehensive library ecosystem suitable for data processing and machine learning tasks. PyTorch version 2.0.1 was used for deep learning model development and dataset processing. PyTorch Lightning 2.2.0 was incorporated to ensure reproducibility, particularly with the seed_everything function.[5] Torchvision 2.1.2 was applied for handling image data and augmentations. Numpy 1.23.5 and Pandas 2.2.0 were essential for numerical computations and data manipulation. The Python Imaging Library (PIL) 1.1 was used for image processing tasks. Matplotlib 3.7.0 functioned as the primary tool for creating informative visualizations and plots, while draw.io was used for creating detailed diagrams and architectural layouts. OpenCV 4.8.1 and Seaborn were also applied for image processing and visualization tasks. Lastly, sklearn version 1.2.1 was employed for various machine learning tasks, including model evaluation, clustering, and other utilities.

D.6 Summary

This chapter presented a detailed methodology for enhancing the Synthnet image classification pipeline's accuracy by integrating D data. It outlined the process of generating D data with a monocular depth estimator. It listed the image preprocessing steps, the required adjustments to the chosen SwinV2 transformer model and explained the fine-tuning process step by step. Implementation details were specified in the last part of the chapter.

[5]All Synthnet RGB experiments were run with seed set to 42, which was replicated in this project.

E Evaluation

This chapter evaluates the proposed modifications for integrating D data into the Synthnet RGB image classification pipeline. The primary research objective was to investigate whether the incorporation of D data would significantly enhance classification accuracy, with a specific focus on the Acc@1 metric. The chapter begins by describing the Synthnet RGB model baseline results used for comparing the modified pipeline. The chapter details experimental variations, including exact hyperparameters, data augmentation techniques, and evaluation metrics. It proceeds to conduct a quantitative analysis structured around the three employed training schemes. An error analysis concludes the chapter, offering more detailed insights into model errors, tendencies and improvements.

E.1 SwinV2 Synthnet RGB Baseline Model

This research paper is built upon the methodologies and outcomes detailed in the Synthnet paper [RHH+23]. Synthnet trained various pre-trained vision transformer models, including SwinV2, on synthetically rendered RGB training datasets, subsequently classifying real-world images. The study focused on two datasets, VisDA and the newly introduced Topex dataset, and employed three training schemes, CH, FT, and a hybrid CH-FT approach. It also integrated domain adaptation strategies like MCC, CDAN, and their combination, CDAN-MCC, achieving SOTA performance on the VisDA dataset.

Tables E.1 and E.2 present the top results from the Synthnet SwinV2 experiments on the Topex dataset. These results will be used as the primary baseline for this study.

Model	Pre-training	train scheme	transform	lr	Acc@1
SwinV2	IN22K	CH	base	1e+1	42.34
SwinV2	IN22K	CH	base	1e-1	45.15
SwinV2	IN22K	CH	base	1e-3	51.79
SwinV2	IN22K	FT	AugMix	1e-1	1.70
SwinV2	IN22K	FT	AugMix	1e-3	26.23
SwinV2	IN22K	FT	AugMix	1e-5	25.69
SwinV2	IN22K	CH-FT	AugMix	1e-5	51.79
SwinV2	**IN22K**	**CH-FT**	**AugMix**	**1e-5**	**59.21**

Table E.1: **Synthnet Fine-Tuning Experiments:** Acc@1 in % on target domain (real images) for all source-domain-only training experiments on the RGB Topex-Printer dataset. Note that *base* transform means that random color jitter and random greyscale transforms were applied.

Model	**DA method**	**init checkpoint**	**Acc@1**
SwinV2	CDAN	IN22K	65.51
SwinV2	CDAN	CH	61.94
SwinV2	MCC	IN22K	72.86
SwinV2	MCC	CH	71.14
SwinV2	CDAN-MCC	IN22K	73.74
SwinV2	**CDAN-MCC**	**CH**	**74.86**

Table E.2: **Synthnet Domain Adaptation Experiments:** Acc@1 in % on target domain (real images) for all UDA experiments on the RGB Topex-Printer dataset. Note that *init checkpoint* describes the model checkpoint used for the UDA experiments. *CH* refers to the best-performing CH training scheme from the source-domain-only training experiments on RGB dataset and IN22K refers to the Huggingface model checkpoints.

This thesis introduces modifications to seamlessly incorporate D data into the Synthnet pipeline, while aiming to maintain the original pipeline's core features in order to facilitate a direct comparison. Key preserved aspects include the use of the same SwinV2 model as backbone, adherence to the original hyperparameters, and the fundamental training schemes of CH, FT, and CH-FT. Efforts were also made to closely mimic the original data augmentation techniques. This approach ensures that any observed performance improvements can mainly be attributed to the integration of D data.

E.2 Experimental Setup

E.2.1 Data Augmentations

Four unique data augmentation strategies were utilized during the experiments. They are outlined here to provide essential context for interpreting the results tables in the quantitative analysis chapter E.3, where they are referred to in the *transform* columns:

Base: This strategy replicates the augmentation techniques from the Synthnet study, encompassing Random Resized Crop (192, 192), Random Horizontal Flip, Random ColorJitter[6], Random Grayscale Conversion, and Normalization[7]. During the CH tuning phase, all mentioned techniques were utilized on the training data. For test and validation sets, only Normalization[8] and Resizing were implemented.

[6]Random ColorJitter was exclusively applied to the RGB data.

[7]Normalization entailed using a mean and standard deviation of 0.5 for each channel. It was not applied to metric depth channels.

[8]Normalization was not applied to metric depth data.

Auto: Tailored to meet the SwinV2 model's basic requirements, the AutoImage-Processor from the Transformers' library employs only Normalization[9], Rescaling, and Resizing (192, 192). This method was applied solely during CH training with normalized depth data.

AugMix: For the FT phase, the **Base** strategy was adapted, substituting Random ColorJitter and Random Grayscale Conversion for a custom AugMix function[10].

Metric: When processing metric depth data in the FT phase, Random Resized Crop and Horizontal Flip were applied. Normalization was applied solely on the RGB channels.

A gridsearch methodology was implemented to evaluate and optimize the combination of augmentation strategies for the various tuning phases.[11]

E.2.2 Hyperparameters

Different hyperparameters were employed for various stages of fine-tuning, adapted directly from the Synthnet pipeline. The learning rates chosen for CH-FT training were based on the best Acc@1 results achieved in the CH and FT training phases. No hyperparameter optimization was conducted beyond these predefined settings.

For CH Training:

Learning Rates:	1e+1, 1e-1, 1e-3
Epochs:	20
Batch Size:	32
Optimizer:	SGD with momentum (0.9) and no weight decay
Loss Function:	Cross Entropy Loss

For FT Training:

[9]Adopts the ImageNet standard mean [0.485, 0.456, 0.406] and standard deviation [0.229, 0.224, 0.225]

[10]This custom AugMix function was developed to replicate Pytorch's implementation used in the Synthnet experiments, but excluded operations like Shear and Translate.

[11]It is important to mention the influence of data augmentation techniques on testing and validation results. For instance, utilizing the *Auto* strategy for both CH training and testing phases resulted in notably higher performance compared to mixing *Auto* for training with *Base* for testing. Furthermore, when CH and FT training phases employed different strategies during CH-FT training, the optimal results were observed when the test dataset was augmented using the same method applied during the FT training phase. Consequently, for CH-FT models, the augmentation techniques for test and validation data always match those used in the FT training phase.

Learning Rates:	1e-1, 1e-3, 1e-5
Epochs:	20
Batch Size:	32
Optimizer:	AdamW with a weight decay of 0.01
Scheduler:	Cosine schedule with a warmup (20% of epochs = 2)
Loss Function:	Cross Entropy Loss

For CH-FT Training:

Learning Rate CH:	1e-3
Learning Rate FT:	1e-5
Epochs:	20
Batch Size:	32
Optimizer CH:	SGD with momentum (0.9) and no weight decay
Scheduler CH:	None
Optimizer FT:	AdamW with a weight decay of 0.01
Scheduler FT:	Cosine schedule with a warmup (20% of epochs = 2)
Loss Function:	Cross Entropy Loss

E.2.3 Evaluation Metrics

The experiments' training evaluation includes a model checkpointing technique, adopted from the Synthnet pipeline. At the end of each epoch, the model was assessed based on its performance against the validation set. The state of the model that performs best, as determined by its cross-entropy loss on the validation set, was saved. The training regimen concluded after the designated number of epochs. The best performing model from the entire training period, regardless of whether it was attained during the first or the last epoch, was chosen. Afterwards, three different metrics were employed to quantitatively evaluate the performance of the selected models. All of the selected models final scores were measured using the complete test set. These metrics were crucial for understanding the models' strengths and weaknesses, particularly in the context of the imbalanced Topex test set.

Top-1 Accuracy: The primary metric for assessing model effectiveness in precise image classification, Acc@1 is particularly relevant for the Topex dataset. In scenarios requiring exact identification of machine parts, the differentiation between false positives and negatives is immaterial. Its value is in providing immediate, straightforward insights into model performance, facilitating quick comparison and ranking. This metric's use in the original Synthnet study also guarantees consistency and comparability across evaluations.

Balanced Accuracy and Macro F1 Score: In the context of Topex's imbalanced test set, Balanced Accuracy and Macro F1 Score are vital for providing a more comprehensive assessment of the model's performance. They ensure that each class contributes equally to the result, regardless of its frequency in the dataset. This helps in identifying if the model is biased towards more frequent classes or if it is equally effective across all classes.

Together, these metrics offer a well-rounded evaluation of the models' overall performances.

E.3 Quantitative Analysis

This section is dedicated to providing a quantitative analysis of the modified RGB-D classification models. It includes a comprehensive presentation of results and initial discussions of the findings. The chapter is divided into three parts, detailing the results of the three major training schemes, CH, FT and CH-FT. The focus of the analysis will revolve around several key aspects:

- **Comparison with the Original Synthnet Model:** A pivotal part of the analysis is comparing the performances of the updated models to the original Synthnet model. These comparisons aim to highlight the improvements or changes in performance brought about by the updates.
- **Early Fusion vs. Late Fusion:** The impact of different fusion techniques on model performance is also examined. Specifically, the analysis will determine whether early fusion or late fusion yields better results in integrating RGB and D information.
- **Freezing Schedules and Processing Techniques:** The study explores the effectiveness of various freezing schedules, training schemes and data augmentation techniques. It seeks to identify the most effective strategies for training the RGB-D models.
- **Normalized vs. Metric Depth:** Another crucial aspect of the paper entails assessing whether model performance improves when utilizing normalized vs. metric ZoeDepth estimated D data.
- **Rendered vs. Estimated Depth:** Lastly, the differences in performance between rendered and estimated D data are examined.

E.3.1 Classification Head Tuning

Early Fusion:

Model	Pre-training	train scheme	transform	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH	Base	1e+1	39.55	40.83	36.14
SwinV2	IN22K	CH	Base	1e-1	37.24	37.95	33.96
SwinV2	**IN22K**	**CH**	**Base**	**1e-3**	**52.81**	**53.35**	**51.02**
SwinV2	IN22K	CH	Auto	1e+1	39.02	39.92	36.58
SwinV2	IN22K	CH	Auto	1e-1	39.26	40.87	36.28
SwinV2	IN22K	CH	Auto	1e-3	50.85	51.42	48.81

Table E.3: **CH Training-Early Fusion-Normalized ZoeDepth** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and ZoeDepth normalized D data. Metrics are in %.

The early fusion model CH results (Table E.3) highlight the significant impact of learning rates and data augmentation techniques on all performance metrics. When considering augmentations, *Base* slightly outperforms *Auto* at the optimal learning rate of 1e-3. Notably, a learning rate of 1e-3 consistently delivers the highest performance across all metrics for both augmentation setups. An Acc@1 of 52.81% modestly exceeds Synthnet's peak performance of 51.79%. This enhancement, albeit slight, underscores the beneficial impact of integrating depth data, even with very simple early fusion techniques.

In contrast, higher learning rates such as 1e+1 and 1e-1 are associated with a significant drop in performance for the early fusion model. This is in stark contrast to the Synthnet results, where the model demonstrated better resilience at these higher learning rates, achieving Acc@1 scores of 42.34% and 45.15%, respectively. This discrepancy suggests that the early fusion model requires a slower approach to learning due to the increased complexity of the added D data. High learning rates in this context are likely to cause overfitting and a heightened sensitivity to noise.

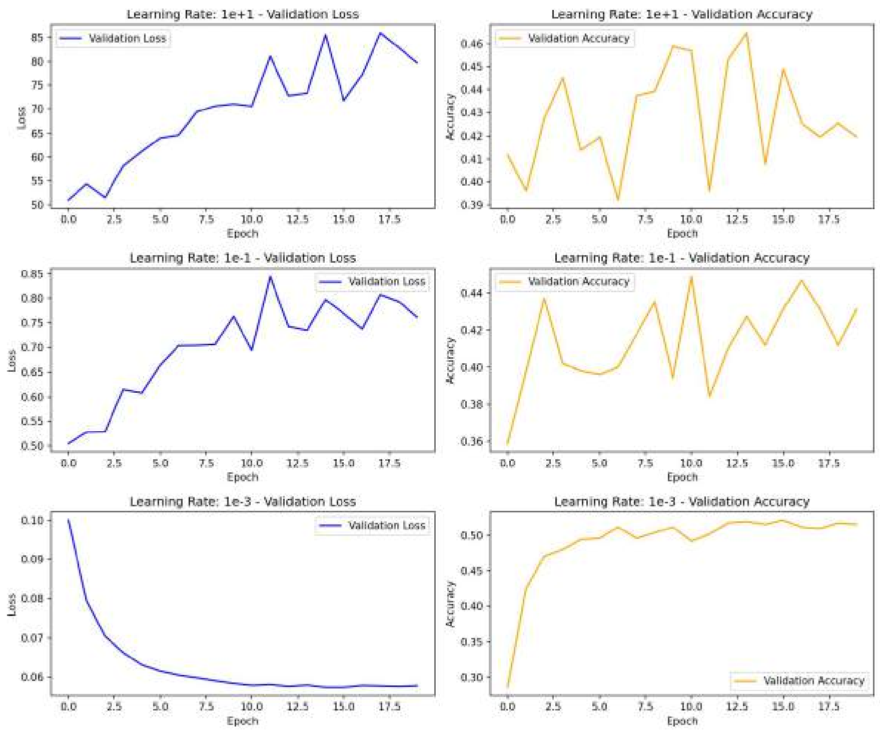


Figure E.1: **Training Progress Compared.** Early Fusion Model with Base Transform Training Progress for Learning Rates 1e+1, 1e-1 and 1e-3

It is interesting to observe the number of epochs required to achieve optimal cross entropy loss under varying learning rates for both the *Auto* and *Base* augmentation strategies, as illustrated in figure E.1. For the higher learning rates of 1e+1 and 1e-1, the best performance consistently occurred at the very first epoch, indicating rapid initial weight adjustments capturing key features but failing to generalize. On the other hand, at the lower learning rate of 1e-3, the peak performance was generally achieved much later, around epochs 15 to 20. This pattern again points to the need for a more measured and stable learning process when handling the more complex RGB-D dataset in the early fusion model.

Model	Pre-training	train scheme	transform	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH	Base	1e+1	36.22	36.05	33.50
SwinV2	IN22K	CH	Base	1e-1	40.21	38.12	35.22
SwinV2	**IN22K**	**CH**	**Base**	**1e-3**	**51.76**	**51.59**	**48.87**

Table E.4: **CH Training-Early Fusion-Metric ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and ZoeDepth metric D data. Metrics are in %.

Despite the addition of metric D data, the models did not significantly improve their performances (see E.4). The best Acc@1 score achieved with metric D data was even marginally lower than the 52.81% with normalized D data. Balanced Accuracy and F1-Macro scores also showed similar trends. While metric D data offers a more accurate representation of spatial information, its impact on the model's performance in this context was limited. It seems that the model could not fully leverage the additional detail, possibly due to the model's pre-trained architecture being more attuned to normalized data.

Late Fusion:

Model	**Pre-training**	**train scheme**	**transform**	**lr**	**Acc@1**	**Balanced Acc**	**F1-Macro**
SwinV2	IN22K	CH	Base	1e+1	46.39	46.04	41.44
SwinV2	IN22K	CH	Base	1e-1	46.13	45.47	41.97
SwinV2	IN22K	CH	Base	1e-3	56.02	57.53	53.73
SwinV2	IN22K	CH	Auto	1e+1	45.40	45.22	41.72
SwinV2	IN22K	CH	Auto	1e-1	40.09	43.47	38.86
SwinV2	**IN22K**	**CH**	**Auto**	**1e-3**	**59.99**	**60.65**	**57.58**

Table E.5: **CH Training-Late Fusion-Normalized ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and ZoeDepth normalized D data. Metrics are in %.

The late fusion approach demonstrated notable performance enhancements as compared to early fusion, particularly when employing a learning rate of 1e-3 (see table E.5). Using the *Base* augmentation setup, the SwinV2 model achieved an Acc@1 of 56.02%. This result not only signifies a 4.23% increase over the Synthnet model's best performance of 51.79% but also marks a 3.21% improvement from the early fusion model's peak performance of 52.81%. This pronounced improvement highlights the effectiveness of combining RGB and D information at a later stage in the model's architecture.

Further enhancement were observed with the implementation of *Auto* augmentations. The model's accuracy improved to 59.99%, representing an 8.2% increase from the Synthnet baseline. In scenarios where RGB and D data are processed separately, it seems that complex image augmentations, especially those conceived for RGB data, might be detrimental. It appears that sticking to basic augmentations works better. Future experiments could investigate bespoke two-pronged augmentation techniques.

The top-performing late fusion model not only excelled in Acc@1 but also achieved a Balanced Accuracy of 60.65% and an F1-Macro Score of 57.58%. This balanced performance across diverse classes demonstrates the model's robustness and consistency.

Late fusion models generally outperformed both the early fusion and Synthnet RGB models across almost all learning rates, except for a single unique case.[12] This performance pattern underscores the models' proficiencies in independently managing and adapting to RGB and D data respectively. This approach seems to reduce the task complexity for the model. By separately addressing the unique features of each data type, the model is less burdened by the increase in data complexity.

While these findings highlight the late fusion model's architectural advantages, the subtle differences in performance across the different CH experiments also underscore the importance of carefully selecting the appropriate combination of fusion technique, transformations and learning rates.

Model	**Pre-training**	**train scheme**	**transform**	**lr**	**Acc@1**	**Balanced Acc**	**F1-Macro**
SwinV2	IN22K	CH	Base	1e+1	46.71	47.02	43.51
SwinV2	IN22K	CH	Base	1e-1	50.86	50.10	47.13
SwinV2	**IN22K**	**CH**	**Base**	**1e-3**	**59.23**	**61.22**	**57.03**

Table E.6: **CH Training-Late Fusion-Metric ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and ZoeDepth metric D data. Metrics are in %.

Examining the effects of metric D data in table E.6, it seems that improvements over normalized D are negligible. The transition to metric D data resulted in a slight uptick in Balanced Accuracy, rising to 61.22% as compared to the 60.65% achieved with normalized D. However, the Acc@1 score decreased slightly from 59.99% to 59.23%. This implies that while metric depth data may hold potential for improving model performance in some scenarios, its impact within the late fusion CH training setup is limited. It is uncertain if different hyperparameters and data augmentation methods could have yielded better results.

Model	**Pre-training**	**train scheme**	**transform**	**lr**	**Acc@1**	**Balanced Acc**	**F1-Macro**
SwinV2	IN22K	CH	Base	1e-3	56.90	57.27	54.00
SwinV2	**IN22K**	**CH**	**Auto**	**1e-3**	**62.06**	**63.28**	**60.07**

Table E.7: **CH Training-Late Fusion-Normalized Rendered Depth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and rendered normalized D data. Metrics are in %.

[12] *Auto* transform at 1e-1.

Model	Pre-training	train scheme	transform	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH	Base	1e-3	52.20	53.44	49.11

Table E.8: **CH Training-Late Fusion-Metric Rendered Depth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and rendered metric D data. Metrics are in %.

Interestingly, when comparing these results to those achieved with normalized and metric rendered depth data (as seen in tables E.7 and E.8), a significant difference in performance is observed. The normalized rendered depth using the AutoImageProcessor achieved an Acc@1 of 62.06%, surpassing both the normalized and metric ZoeDepth data configurations. This suggests that directly rendered depth information may provide more precise or comprehensive depth cues that the model can utilize more efficiently without the need for significant adaptation. Furthermore, it appears that normalized rendered depth is more compatible with the pre-trained model optimized for normalized data, as well as with the normalized ZoeDepth validation and testing sets, when compared to the rendered metric D data with the ZoeDepth metric D validation and testing sets.

E.3.2 Full Tuning

Early Fusion:

Model	Pre-training	train scheme	transform	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	FT	AugMix	1e-1	0.73	0.98	0.01
SwinV2	IN22K	FT	AugMix	1e-3	0.72	0.98	0.01
SwinV2	**IN22K**	**FT**	**AugMix**	**1e-5**	**52.33**	**51.87**	**49.33**

Table E.9: **FT Training-Early Fusion-Normalized ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and ZoeDepth normalized D data. Metrics are in %.

In the FT phase with early fusion, a significant variance in model performance for different learning rates was observed (see table E.9). Higher learning rates, such as 1e-1 and 1e-3, led to drastically low performance with an Acc@1 barely surpassing 1%. A lower learning rate of 1e-5 resulted in a substantial improvement in performance, with the model achieving Acc@1 of 52.33%. This is probably due to a mismatch between the learning rate and the complexity of the task at hand.

The Synthnet RGB model was able to handle a lower learning rate of 1e-3 with its drop off occurring only at a learning rate of 1e-1. However, the early fusion models' peak FT performance improved the best Synthnet FT result by a remarkable 26.64%. This significant leap not only outperforms the best results from Synthnet's FT and CH training, but also raises questions about skepticism regarding the efficacy of FT training on OOD test data as outlined in [KRJ+22]. This considerable improvement in Acc@1 along with Balanced Accuracy and F1 score suggests that FT training can indeed be effective in enhancing the model's generalization and performance, especially on more complex datasets.

Model	Pre-training	train scheme	transform	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	FT	Metric	1e-1	0.10	0.98	0.00
SwinV2	IN22K	FT	Metric	1e-3	0.47	0.98	0.01
SwinV2	**IN22K**	**FT**	**Metric**	**1e-5**	**52.57**	**50.73**	**48.07**

Table E.10: **FT Training-Early Fusion-Metric ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and ZoeDepth metric D data. Metrics are in %.

The differences in results between models using normalized or metric D data were small. While the top performance with metric D data achieved a marginally higher Acc@1 at 52.57% (see table E.10), both Balanced Accuracy and F1-Macro scores were slightly lower at 50.73% and 48.07%. While metric depth data was expected to provide a more detailed representation, its actual impact on model efficacy appears to be limited. It is fair to assume, that the specific characteristics of metric D data are not being fully exploited by the early fusion model.

Late Fusion:

Model	Pre-training	train scheme	transform	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	FT	AugMix	1e-1	0.52	0.98	0.01
SwinV2	IN22K	FT	AugMix	1e-3	1.27	0.98	0.02
SwinV2	**IN22K**	**FT**	**AugMix**	**1e-5**	**57.84**	**58.91**	**56.38**

Table E.11: **FT Training-Late Fusion-Normalized ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and ZoeDepth normalized D data. Metrics are in %.

The late fusion approach in FT phase exhibited a pattern similar to that observed in the early fusion model. The models encountered significant challenges at higher learning rates of 1e-1 and 1e-3, with Acc@1 scores barely surpassing the 1% mark (see table E.11). However, a marked improvement in performance over both the Synthnet and early fusion models' FT phases was achieved when the learning rate was adjusted to 1e-5, culminating in an Acc@1 of 57.84%. Comparing the results of the best late fusion FT model to the highest-performing Synthnet model, which achieved 59.21% during the CH-FT phase, the difference is remarkably small.

Model	**Pre-training**	**train scheme**	**transform**	**lr**	**Acc@1**	**Balanced Acc**	**F1-Macro**
SwinV2	IN22K	FT	Metric	1e-1	1.25	1.16	0.17
SwinV2	IN22K	FT	Metric	1e-3	0.55	0.89	0.07
SwinV2	**IN22K**	**FT**	**Metric**	**1e-5**	**62.33**	**63.48**	**59.91**

Table E.12: **FT Training-Late Fusion-Metric ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and ZoeDepth metric D data. Metrics are in %.

The late fusion technique, when augmented with metric depth data, showcased an even greater improvement in FT performance. It surpassed even the best Synthnet CH-FT model by over 3% (see table E.12). The improvements across all three performance metrics underscore the importance of metric depth data in the late fusion setup and warrant further exploration into different freezing schedules and even lower learning rates to fully optimize FT performance on OOD test data. Moreover, testing the impact of different data augmentations could lead to further boosts.

E.3.3 CH-FT

Early Fusion:

Model	Pre-training	train scheme	CH transform	FT freezing schedule	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[C3210]	1e-5	54.47	55.02	52.03
SwinV2	IN22K	CH-FT	Base	[CI+3210]	1e-5	54.60	55.22	52.39
SwinV2	IN22K	CH-FT	Base	[C3][C2][C1][C0]	1e-5	53.71	54.54	51.89
SwinV2	IN22K	CH-FT	Base	[CI+3][CI+2][CI+1][CI+0]	1e-5	54.44	55.25	52.45
SwinV2	IN22K	CH-FT	Base	[C3][C32][C321][C3210]	1e-5	55.52	**56.49**	**53.51**
SwinV2	**IN22K**	**CH-FT**	**Base**	**[CI+3][CI+32][CI+321][CI+3210]**	**1e-5**	**55.6**	56.28	53.47
SwinV2	IN22K	CH-FT	Base	[C3]	1e-5	51.87	52.17	49.95
SwinV2	IN22K	CH-FT	Base	[C2]	1e-5	54.41	55.27	52.34
SwinV2	IN22K	CH-FT	Base	[C1]	1e-5	52.77	53.27	50.94
SwinV2	IN22K	CH-FT	Base	[C0]	1e-5	52.96	53.20	51.15
SwinV2	IN22K	CH-FT	Base	[C20]	1e-5	54.59	55.46	52.60
SwinV2	IN22K	CH-FT	Auto	[C3]	1e-5	52.28	52.34	50.40
SwinV2	IN22K	CH-FT	Auto	[C2]	1e-5	53.53	54.03	51.39
SwinV2	IN22K	CH-FT	Auto	[C1]	1e-5	51.59	51.99	50.04
SwinV2	IN22K	CH-FT	Auto	[C0]	1e-5	52.93	52.99	51.08
SwinV2	IN22K	CH-FT	Auto	[C20]	1e-5	53.68	54.31	51.49
SwinV2	IN22K	CH-FT	Auto	[C3210]	1e-5	53.76	54.19	51.5
SwinV2	IN22K	CH-FT	Auto	[CI+3210]	1e-5	53.86	54.21	51.45
SwinV2	IN22K	CH-FT	Auto	[C3][C2][C1][C0]	1e-5	53.76	54.34	51.56
SwinV2	IN22K	CH-FT	Auto	[CI+3][CI+2][CI+1][CI+0]	1e-5	54.70	55.40	52.66
SwinV2	IN22K	CH-FT	Auto	[C3][C32][C321][C3210]	1e-5	54.13	54.75	52.03
SwinV2	IN22K	CH-FT	Auto	[CI+3][CI+32][CI+321][CI+3210]	1e-5	54.20	54.98	51.93

Table E.13: **CH-FT Training-Early Fusion-Normalized ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH-FT training was applied to the model with source-domain training data and ZoeDepth normalized D data. The AugMix data augmentation method was applied during FT training. Metrics are in %.

The early fusion CH-FT experiments revealed that freezing schedules that gradually unfroze layers such as [CI+3] [CI+32] [CI+321] [CI+3210] [13] generally resulted in better performances (see table E.13). A case in point was the top-performing early fusion model, which achieved an Acc@1 of 55.60% using the *Base* transform at a learning rate of 1e-5. However, this approach did not fully exploit the potential of the added D dimension. The highest performance fell over 3% short of Synthnet's best CH-FT performance of 59.71%. It seems that combining RGB and D data before feature extraction might not effectively harness the unique features of each data type, leading to suboptimal results on the Topex dataset.

[13]Each bracket represents a separate training run. The bracket content specifies the unfrozen layers. "C" stands for Classifier, "I" for Initial Convolutional Layer, "+" for Global Average Pooling, Patch Partition, and Linear Embedding layers, and "0-3" for the four stages of the SwinV2 Transformer Model

Model	Pre-training	train scheme	CH transform	FT freezing schedule	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[CI+3210]	1e-5	54.38	54.69	51.90
SwinV2	IN22K	CH-FT	Base	[CI+3][CI+32][CI+321][CI+3210]	1e-5	57.96	58.11	55.52
SwinV2	IN22K	CH-FT	Auto	[CI+3210]	1e-5	57.89	57.92	55.84
SwinV2	**IN22K**	**CH-FT**	**Auto**	**[CI+3][CI+32][CI+321][CI+3210]**	**1e-5**	**59.50**	**59.03**	**56.79**

Table E.14: **Hagberg et al. Approach.** CH-FT Training-Early Fusion-Normalized Depth Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH-FT training was applied to the model with source-domain training data and ZoeDepth normalized D data. The AugMix data augmentation method was applied during FT training. Metrics are in %.

Further investigation into more intricate early fusion techniques, such as those proposed by Hagberg et al. [HM22], brought some improvement (see table E.14). This approach, elaborated in D.3.2, marginally enhanced performance, aligning it closer to the original Synthnet Model. However, the modest increase from 59.21% to 59.50% does not strongly support the overall efficacy of the early fusion approach.

Model	Pre-training	train scheme	CH transform	FT freezing schedule	lr	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[C3210]	1e-5	56.92	57.88	54.74
SwinV2	IN22K	CH-FT	Base	[C+3210]	1e-5	56.64	57.36	54.46
SwinV2	**IN22K**	**CH-FT**	**Auto**	**[C3][C32][C321][C3210]**	**1e-5**	**59.26**	**59.91**	**56.93**
SwinV2	IN22K	CH-FT	Auto	[C+3][C+32][C+321][C+3210]	1e-5	57.68	57.47	54.66

Table E.15: **CH-FT Training-Early Fusion-Metric ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric depth data. CH training was applied to the model with source-domain training data and ZoeDepth metric depth data. The Metric data augmentation method was applied during FT training. Metrics are in %.

The inclusion of metric D data during the full CH-FT training phase reveals a notable enhancement in model performance from 55.6% to 59.26% Acc@1 (see table E.15). It even slightly outperforms the best Synthnet RGB CH-FT model. This suggests that when the entire model undergoes comprehensive readjustment during CH-FT training, the more detailed spatial representations provided by metric D data begin to significantly contribute to the model's accuracy.

These early fusion experiments underscore the limitations of the simplistic linear transformation strategy employed. While intuitively, the addition of D data seems beneficial, the actual advantages largely depend on the specific implementation technique and the nature of the dataset. In the case of Topex, a straightforward early fusion approach is not the most effective. The minor improvements achieved with more advanced fusion techniques indicate a need for continued research and development. On another note, gradual unfreezing schedules and metric depth data led to considerable improvements.

Late Fusion:

Model	Pre-training	train scheme	CH transform	FT freezing schedule	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[C3210]	59.65	60.57	57.29
SwinV2	IN22K	CH-FT	Base	[C+3210]	60.58	61.46	58.09
SwinV2	IN22K	CH-FT	Base	[C3][C2][C1][C0]	60.48	61.61	58.25
SwinV2	IN22K	CH-FT	Base	[C3+][C2+][C1+][C0+]	61.00	62.06	58.62
SwinV2	IN22K	CH-FT	Base	[C3][C32][C321][C3210]	60.92	62.05	58.30
SwinV2	IN22K	CH-FT	Base	[C+3][C+32][C+321][C+3210]	61.8	63.36	59.46
SwinV2	IN22K	CH-FT	Base	[C3]	55.71	57.09	53.56
SwinV2	IN22K	CH-FT	Base	[C2]	60.09	61.36	57.88
SwinV2	IN22K	CH-FT	Base	[C1]	61.73	62.81	59.21
SwinV2	IN22K	CH-FT	Base	[C0]	59.88	60.87	57.31
SwinV2	IN22K	CH-FT	Base	[C21]	60.71	61.63	58.48
SwinV2	IN22K	CH-FT	Auto	[C3]	57.97	58.48	55.78
SwinV2	IN22K	CH-FT	Auto	[C2]	61.23	62.08	59.52
SwinV2	IN22K	CH-FT	Auto	[C1]	61.05	62.14	59.34
SwinV2	IN22K	CH-FT	Auto	[C0]	59.21	59.93	57.13
SwinV2	IN22K	CH-FT	Auto	[C21]	62.97	63.90	61.17
SwinV2	IN22K	CH-FT	Auto	[C+21]	61.57	62.58	60.04
SwinV2	IN22K	CH-FT	Auto	[C2][C1]	62.33	63.03	60.20
SwinV2	IN22K	CH-FT	Auto	[C3210]	59.37	60.39	57.91
SwinV2	IN22K	CH-FT	Auto	[C+3210]	59.40	60.45	57.91
SwinV2	IN22K	CH-FT	Auto	[C3][C2][C1][C0]	63.20	64.11	61.52
SwinV2	IN22K	CH-FT	Auto	[C3+][C2+][C1+][C0+]	63.60	63.99	61.45
SwinV2	IN22K	CH-FT	Auto	[C3][C32][C321][C3210]	64.16	**65.73**	**62.71**
SwinV2	**IN22K**	**CH-FT**	**Auto**	**[C+3][C+32][C+321][C+3210]**	**64.71**	65.47	62.60

Table E.16: **CH-FT Training-Late Fusion-Normalized ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and ZoeDepth normalized D data. The AugMix data augmentation method was applied during FT training. Metrics are in %.

The late fusion model generally showed enhanced performance for the CH-FT phase, with certain configurations notably surpassing their early fusion counterparts. For example, a model that employed the *Auto* augmentation and a gradual unfreezing schedule ([C+3] [C+32] [C+321] [C+3210]) at a learning rate of 1e-5 achieved an accuracy of 64.71% (see table E.16). This performance, a 5.5% increase over the best performing Synthnet model, highlights the advantages of processing RGB and D data streams separately before integration, thereby capturing the unique characteristics of each data type more effectively. Achieving a balanced accuracy of 65.47% and an F1 score of 62.60% further indicates the robustness of this approach, even with a challenging dataset like Topex.

The implementation of gradual unfreezing schedules, particularly [CI+3] [CI+32] [CI+321] [CI+3210], was crucial in boosting model performance in the late fusion setup, mirroring the success seen in early fusion. The slight differences in performance observed between CH *Base* and *Auto* augmentation approaches are also noteworthy. *Auto* frequently resulted in better outcomes. D data, by its nature, does not contain color or lighting information, making some of the applied augmentations less effective. However, the use of *Auto* on the CH training before applying *Base* for the FT phase yielded the best results.[14] Possibly, more complex augmentation techniques unfold their full benefits only within the context of a thorough training regimen. Nonetheless, it might be helpful to develop and test augmentation strategies specifically designed for RGB and D data.

The surgical finetuning approach suggested by Lee et al. [LCT+23] showed promising results, though it did not achieve top performance. It particularly excelled when freezing the middle layers (2 and 1) of the SwinV2 model, with most experiments achieving over 61% Acc@1. Interestingly, training layers 2 and 1 together in the CH *Auto* augmentation setup led to a further increase in Acc@1, reaching 62.97%, closely approaching the best results. This is noteworthy for two reasons. Firstly, it demonstrates that respectable results can be obtained without requiring more than two training runs, unlike the best performing model which underwent five runs. Secondly, it lends credence to Lee et al.'s hypothesis regarding the influence of the type of distribution shift in a dataset on the effectiveness of finetuning specific model subsets. In the case of the Topex dataset, which represents a feature-level shift with the same classes but different subpopulations in the test set (real vs. synthetic images), freezing the middle blocks of the model appears to be the most effective strategy [LCT+23].

Model	Pre-training	train scheme	CH transform	FT freezing schedule	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[C3210]	66.21	67.70	63.76
SwinV2	IN22K	CH-FT	Base	[C+3210]	66.16	67.60	63.57
SwinV2	**IN22K**	**CH-FT**	**Base**	**[C3][C32][C321][C3210]**	**69.70**	**70.50**	**67.03**
SwinV2	IN22K	CH-FT	Base	[C+3][C+32][C+321][C+3210]	66.79	67.71	64.26

Table E.17: **CH-FT Training-Late Fusion-Metric ZoeDepth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and ZoeDepth metric D data. The Metric data augmentation method was applied during FT training. Metrics are in %.

The integration of metric D data during CH-FT resulted in substantial performance increases across all metrics. With the application of *Base* augmentation during CH, paired with a comprehensive gradual unfreezing schedule ([C3] [C32] [C321] [C3210])[15], the model achieved an Acc@1 of 69.70% (see table E.17). Balanced Accuracy and F1 Score made similar strides. This marks a significant improvement over the results obtained using normalized D as well as a 10% increase in Acc@1 compared to the best performing Synthnet RGB model. The model

[14]The alternative strategy of employing *Auto* for CH as well as FT is not listed here, but did not improve Acc@1.

[15]It should be noted that the best performance was achieved after stage 3 and before the last stage, i.e. [C3] [C32] [C321] had the top performance.

was able to capitalize on the detailed spatial information by way of gradual and thorough training.

Model	Pre-training	train scheme	CH transform	FT freezing schedule	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[C3210]	63.59	64.56	61.35
SwinV2	IN22K	CH-FT	Base	[C+3210]	64.27	65.18	61.97
SwinV2	IN22K	CH-FT	Base	[C3][C32][C321][C3210]	67.04	67.82	64.95
SwinV2	IN22K	CH-FT	Base	[C+3][C+32][C+321][C+3210]	67.31	68.11	65.02
SwinV2	IN22K	CH-FT	Auto	[C3210]	67.54	68.22	66.00
SwinV2	IN22K	CH-FT	Auto	[C+3210]	67.12	67.86	65.48
SwinV2	IN22K	CH-FT	Auto	[C3][C32][C321][C3210]	67.60	68.60	66.22
SwinV2	**IN22K**	**CH-FT**	**Auto**	**[C+3][C+32][C+321][C+3210]**	**68.06**	**69.52**	**66.56**

Table E.18: **CH-FT Training-Late Fusion-Normalized Rendered Depth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth normalized D data. CH training was applied to the model with source-domain training data and rendered normalized D data. The AugMix data augmentation method was applied during FT training. Metrics are in %.

Model	Pre-training	train scheme	CH transform	FT freezing schedule	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH-FT	Base	[C3210]	58.77	59.98	56.47
SwinV2	IN22K	CH-FT	Base	[C+3210]	60.25	61.57	58.07
SwinV2	IN22K	CH-FT	Base	[C3][C32][C321][C3210]	59.00	60.57	56.53
SwinV2	**IN22K**	**CH-FT**	**Auto**	**[C+3][C+32][C+321][C+3210]**	**63.05**	**63.93**	**60.57**

Table E.19: **CH-FT Training-Late Fusion-Metric Rendered Depth.** Test Results for RGB-D Topex-Printer target-domain test set with ZoeDepth metric D data. CH training was applied to the model with source-domain training data and rendered metric D data. The Metric data augmentation method was applied during FT training. Metrics are in %.

The comparison between models trained on rendered versus estimated D data highlights subtle yet significant distinctions in performance and applicability. Specifically, models using normalized rendered D data (referenced in table E.18) showcased a notable edge, achieving an Acc@1 of 68.06%, which represents a 2.33% improvement over those trained with normalized ZoeDepth data. Contrary to expectations, the shift to metric rendered D data did not yield the anticipated increase in accuracy, managing only an Acc@1 of 63.05%, significantly trailing the performance achieved with ZoeDepth metric estimations. This suggests that the effectiveness of metric rendered D training data is limited by the nature of the estimated D test set. Without normalization the differences in the nature of these datasets are too stark.

This calls attention to the practical challenges of deploying rendered D data in real-world scenarios where only estimated D is available. It also shows the versatility of monocular D estimation techniques, which can be applied to both synthetic and real-world imagery. With ongoing advancements in these techniques, their utility is expected to grow. Even so, the gap in performance between models trained on rendered and estimated normalized D data, such as the 2.33% difference mentioned earlier, is relatively modest. This minimal gap attests to the efficacy of current D estimation methods and points towards a promising research trajectory aimed at further narrowing these differences.

Remarkably, the model trained on metric ZoeDepth data, when subjected to comprehensive and gradual training, outperformed all other configurations, showcasing the potential of D estimation technology to not just compete with, but in many cases, surpass the accuracy of models trained on rendered D.

E.4 Error Analysis

This chapter aims to systematically identify and analyze the predominant errors across the experiments' best-performing RGB-D adapted models, with a particular emphasis on the top CH-FT model utilizing normalized ZoeDepth D data.[16] The analysis begins by examining the error patterns in this model while ensuring that the patterns are consistent across other models. Subsequently, the discussion shifts towards exploring how differences in model configurations contribute to varying errors.

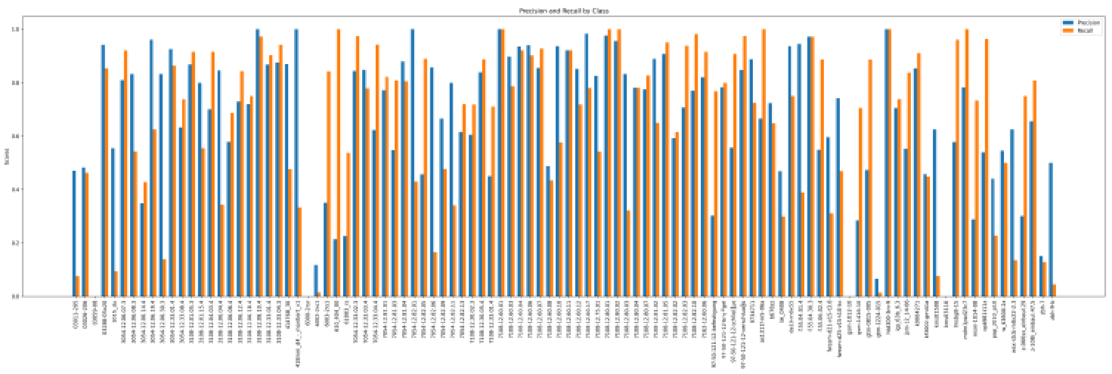


Figure E.2: Precision/Recall by Class for Best CH-FT Normalized D Data Model

To provide a clear, initial insight into class-specific performance, a bar chart showcasing precision and recall metrics for each class was produced (see figure E.2). This visualization is preferred over confusion matrices due to the large number of classes (102) in the Topex dataset, rendering confusion matrices illegible.

One notable observation from figure E.2 is that several classes consistently exhibit very low precision and recall. Classes such as "03059-08" (see figure E.4),

[16]Late Fusion, CH Auto transform, [C+3] [C+32] [C+321] [C+3210].

"6000-2rsr" (see figure E.3), and "kms03116" are particularly problematic. These problematic cases are not isolated instances, but recur across all the best performing ZoeDepth models. Moreover, there are several instances of high recall paired with low precision scores across models. This includes, but is not limited to, classes "6003-2rs1", "61803_rs", "gsm-1416-10", and "msm-1014-08" and hints at model biases towards specific classes, resulting in frequent misclassification of objects. The underlying reasons for these biases can primarily be attributed to Inter-Class Similarity and Train-Test Discrepancies.

E.4.1 Inter-Class Similarity

As discussed in Chapter C.2.2, the challenge of inter-class similarity can significantly impact model performance. Objects "gsm-1416-10" and "gsm-1012-10" are prime examples of this (see figure C.4). Their shapes appear identical. Their only distinguishing feature is size. The total absence of size reference points in the featureless background of the training data complicates the accurate perception of size. The D data seems incapable of correcting this issue. In the case of "gsm-1012-10", none of the 11 images in the test set were correctly identified, with 7 being misclassified as "gsm-1416-10". This issue persists even with the introduction of metric depth data, where the best-performing model erroneously classified 10 out of 11 test images as "gsm-1416-10".

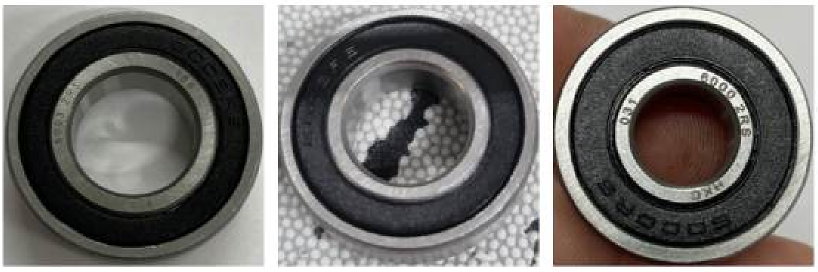


Figure E.3: Comparison of Test Objects 6003-2rs1 (left), 6002-2rs1 (middle) and 6000-2rsr (right)

This type of error extends to other objects, including "6000-2rsr", "6002-2rs1" and "6003-2rs1", as shown in Figure E.3. Out of 137 test images for "6002-2rs1", 111 were incorrectly identified as "6003-2rs1". Also "6000-2rsr" was predicted entirely incorrectly. Instead, the model predicted "6002-2rs1" or "6003-2rs1" for 6 out of its 12 test images. Models trained with metric D data did not show improvement, instead the "6003-2rs1" predictions for object "6002-2rs1" increased to 114, while the above mentioned error rate for "6000-2rsr" increased to 8 out of 12 images.

These findings suggest that the addition of D data alone, whether metric or normalized, is inadequate in addressing the issue of extreme inter-class similarity. It may require more innovative approaches, such as incorporating contextual size cues into the training dataset through synthetic rendering processes. This could potentially enhance the model's ability to distinguish objects based on size differences, thereby improving classification accuracy in cases of high inter-class similarity.

E.4.2 Train-Test Discrepancies

The other major reason for the consistent misclassification of certain objects across models can be attributed to train-test data discrepancies, which was briefly touched upon in chapter C.2.5. For instance, object "03059-08", is recurrently misclassified as "z-066sx_einbaul.29" (see figure E.4). When first glancing at the test images, this may seem unreasonable, given the apparent dissimilarity between the two objects. However, a closer look reveals a crucial detail. "03059-08" has a smooth surface in the training images. Despite this, the surface of the object in the test set is threaded. The model, trained on the smooth-surfaced images, struggles to recognize the threaded features in the test set. It finds a closer resemblance with the textured, grooved surface of "z-066sx_einbaul.29". A similar smooth/threaded discrepancy is observed with objects "03011-205" and "03026-206". They are incorrectly classified as "z-066sx_einbaul.29" 50 and 58 times out of respective 105 and 121 test images.

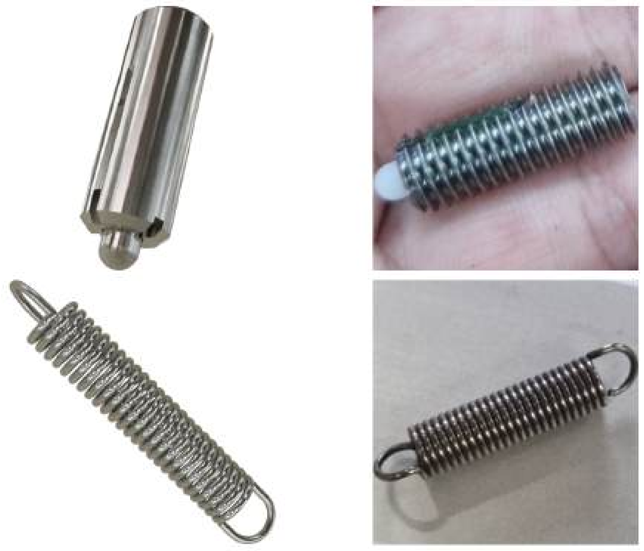


Figure E.4: 03059-08 (top row), z-066sx_einbaul.29 (bottom row), Train (left), Test (right)

Another notable discrepancy were missing elements in the training data, such as rubber casings. For instance, the training images of object "f.55.04.01.4" (refer to Figure E.5) do not contain the test images' defining rubber casing. This omission led to its frequent misclassification as object "610_404_00" which has a similar shape without the rubber casing. There are many more such examples.

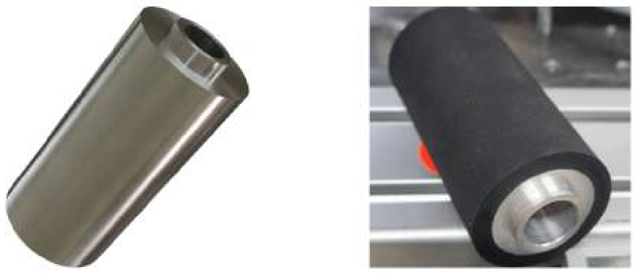


Figure E.5: Object f.55.04.01.4: Train example (left), Test example (right)

Color inconsistencies seem to further compound these challenges. For instance, object "97-50-121-12-befestigung" shown in figure C.11. Similarly, "jsm-12_14-06" transitions from black in the training dataset to yellow in the testing data. Other objects like "pap_2010_p10" or "kms01508" exhibit less pronounced, yet still significant, color discrepancies. Their synthetic versions are uniformly chrome while the test images display slight metallic color variations along with noticeable signs of wear and tear.

Surprisingly, these classes are not as prone to misclassification as might be expected. For example, "97-50-121-12-befestigung" maintains a recall score of over 70%, which increases to almost 90% with the use of metric depth data. Misclassification of "pap_2010_p10" as "jsm-12_14-06" appears to stem more from shape similarity, rather than color confusion. This can be assumed, because their colors match neither in the train nor in the test set. Likewise, "kms01508" is frequently misclassified as "gsm-1012-10" with both belonging to type *T4* (see appendinx G.1), suggesting that shape plays a more critical role in these errors than color.

To achieve better Acc@1, enhancements in synthetic rendering process are imperative. While ensuring the precise replication of object shapes is paramount, this analysis indicates that accurately rendering textures, and other characteristic features are also essential, even if the impact of color appears to be limited.

E.4.3 Impact of Blurriness on Model Accuracy

Special consideration was also given to the impact of image blurriness on performance, as discussed in chapter C.2.4. Contrary to expectations, an examination of the top five blur scores per class revealed that these classes exhibited a significantly higher recall than the average class.[17]

Object	Blur Score	Recall
gtm-0815-005	26%	86%
7054-12.33.04.4	23%	94%
7108-12.60.17	23%	81%
3108-12.06.06.4	22%	75%
jsm-12_14-06	21%	81%

Table E.20: Top Five Blur Rates per Class and Corresponding Recall.

The observed correlation coefficient of -0.065 between per class blur scores and recall, along with a high p-value of 0.514, points to a negligible linear relationship between these two variables. This is a first indication that the level of blurriness in the images does not meaningfully affect the model's classification accuracy. Further supporting this conclusion, a manual examination of various classes showed that images, even with considerable blur, were often correctly identified by the model. However, blur cannot be entirely dismissed as having no influence on classification accuracy. To conclusively determine the extent of blur's impact, additional statistical analyses would be necessary.

E.4.4 Class Type Errors

In pursuit of a better understanding of the model's ability to differentiate between different kinds of objects, efforts were made to categorized the 102 classes into 11 overarching types. This categorization relied on a simple visual inspection, utilizing a selection of test images to identify easily recognizable physical traits of each class. An object's function was not considered for categorization. Categorization was primarily based on shape. It should be noted that categories *T2*, *T3* and *T4* could be aggregated into a single group of ring shaped objects, which together would account for 27 classes. A description of each type along with images of classes that make up the types can be found in the appendix (see G.1). The distribution of classes within each type is displayed in table E.21.

[17]Blur scores were calculated with the blur detector developed by S. Alireza Golestaneh and Lina J. Karam [GK17]

Type	Number of Classes
T1	7
T2	8
T3	3
T4	16
T5	8
T6	4
T7	24
T8	2
T9	3
T10	17
T11	10

Table E.21: Distribution of Classes within each Type Category.

The first objective was to assess the model's proficiency in typewise object classification, which entails accurately predicting an object as either the correct class or as another class within the same type category. A confusion matrix offers a good overview of the model's behaviour (see figure E.6).

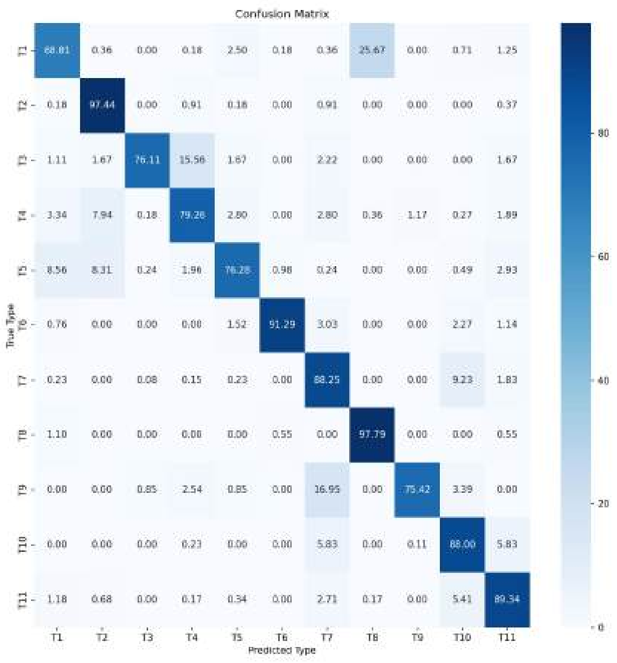


Figure E.6: Confusion Matrix Types CH-FT Normalized ZoeDepth

After gaining an initial understanding of how well the model assigns items to their respective types, precision and recall metrics per type were calculated. These were computed by first determining the precision and recall for each individual class within a type and then calculating the average of all classes in a type. In combination with the confusion matrix, this can help in identifying inter-class similarity issues. It is important to take into account the distribution of classes within each type. More classes increase the probability of intra-type errors.

Type Name	**Precision (%)**	**Recall (%)**
T1	59.5	50.6
T2	29.4	45.7
T3	48.4	35.3
T4	50.3	39.9
T5	78.5	71.6
T6	71.9	66.9
T7	82.7	77.8
T8	47.3	76.4
T9	94.6	69.3
T10	71.0	76.8
T11	77.1	81.4

Table E.22: Precision and Recall Averages for each Type in Percent for CH-FT Normalized ZoeDepth Model. Computed by Averaging the Metrics for each Class within the Type

The analysis reveals that *T2*, despite making few of out-of-type predictions, suffers from low precision and recall. This pattern suggests that while the model rarely confuses *T2* with objects from other categories, there is a significant challenge in distinguishing between different *T2* objects due to high inter-class similarity within this category. The confusion between "6000-2rsr" "6002-2rs1" and "6003-2rs1" (see figure E.3) is an example of this. On the other hand, despite also recording few out-of-type errors, *T7* demonstrates significantly better precision and recall rates. This suggests that the model is very effective at discriminating among different *T7* objects.

T3 and *T4* objects are both characterized by a high percentage of out-of-type errors, often being confused with *T4* and *T2* respectively, because of their common ring shape. They also score low on precision and recall. This underscores the difficulties in distinguishing between objects with a generic shape.

T10 items, despite the diversity within this category, exhibit low out-of-type errors and high precision and recall rates. This indicates that *T10* objects are both well-defined as a category and distinct from other types. It also demonstrates the model's strong capability to differentiate between various *T10* items. This might be attributed to their relatively diverse shapes or their unique hole patterns, a feature which is common across most *T10* objects. Interestingly, despite a more limited range of shapes, *T7* exhibits similar statistics. Both *T7* and *T10* have unique hole patterns in common.

E.4.5 Differences between Models

The purpose of examining the differences between models was to go beyond simply comparing Acc@1 scores. By examining comparative confusion matrices, insight into the types of mistakes each model was prone to making, and whether certain errors were consistent across models could be gained. A detailed comparison of recall and precision scores for each class additionally provides a clearer picture of each model's capabilities and potential areas for improvement.

This analysis acknowledges that the objective of this research project was not to explain the learning processes of the deployed models or to elucidate the relationship between specific adjustments and their predictions. It recognizes the significant challenge in identifying consistent, robust errors within such complex deep learning systems, as well as the dangers in drawing definitive conclusions from them. Consequently, any judgment on observed errors should be approached with caution.

Best Normalized ZoeDepth Models (CH vs. CH-FT):

The comparative evaluation of the CH and CH-FT late fusion models utilizing normalized ZoeDepth data reveals that the CH-FT model has improved precision and recall scores for the majority of classes, as illustrated in figure E.7. Specifically, the CH-FT model outperforms the CH model in precision in 58 classes, whereas the CH model has higher precision in 36 classes. The CH-FT model exhibits higher recall in 63 classes, while the CH model surpasses the CH-FT model in 22 classes. Reductions in recall are comparatively rare, indicating overall enhanced class recognition capabilities by the CH-FT model. Moreover, figure E.7 shows that it is uncommon for both recall and precision to be lower simultaneously, and on the few occasions that they are, the differences between models are minimal. Despite these disparities, it's important to note that there are trade-offs involved. For instance, items such as "03011-205" (an example of train-class discrepancy) and "kms01508" (an example of high inter-class similarity) exhibit significantly lower precision in the CH-FT model. This drop means that the model incorrectly identified other objects as "03011-205" and "kms01508". Perhaps it overemphasized features these classes share or it forgot features understood by the pre-trained model that allowed the CH model to differentiate between them.

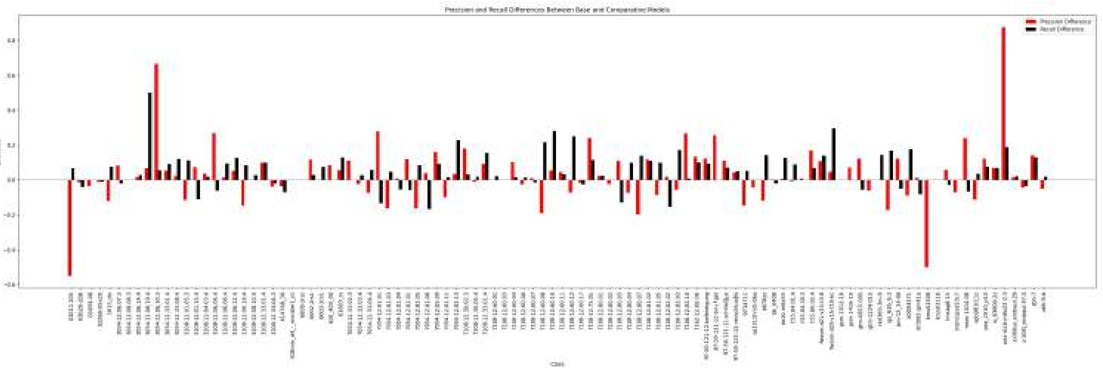


Figure E.7: Differences in Recall (black) and Precision (red) for Best CH and CH-FT Models with Normalized Depth. Negative values denote superior performance by CH model.

The CH-FT model showcased several instances of notable improvement in recall accompanied by slighter increases in precision. These instances often involve ring-shaped objects. Examples include "61803_rs", "7108-12.60.10", "7108-12.33.01.4", and "fwssm-d25-v15-t10-kc". Other ring-shaped objects like "fwssm-d21-v15-t3.8" and "zbh-7" exhibited significant enhancements in both metrics. This trend is corroborated by comparing the average recall and precision values per type for the CH model shown in table E.23 and the CH-FT model shown in table E.22.

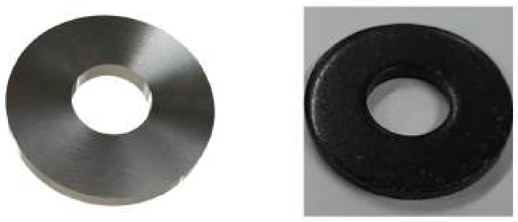


Figure E.8: Object wsx-stcb-m8x22-2.3: Train example (left), Test example (right)

In the case of the ring-shaped "wsx-stcb-m8x22-2.3" *T3* object (see figure E.8), the CH model originally could not correctly classify any instances, resulting in a recall and precision of 0%. Conversely, the CH-FT model marked a notable improvement, correctly classifying 7 out of 37 images, which elevated the recall to nearly 19% and the precision to 87.5%, with only a single error in classification. This development is particularly noteworthy considering the object underwent a color change from chrome in the training set to black in the test set. Furthermore, several test images included screws, obfuscating the washer object itself (see figure C.10). On closer inspection, the CH-FT model was able to recognize the class in images where the object was laid flat and viewed from a top-down perspective. However, it struggled to accurately classify it in images with added screws or altered camera angles. Similar to the CH model, the CH-FT model frequently mistook the object for other *T3* classes, specifically "gtm-0815-005" and "gtm-1224-015", which were consistently black in both their training and test images. Nevertheless, the ability of the CH-FT model to correctly identify the class in 7 instances signals an improvement, despite continuing to face difficulties with extreme cases of inter-class similarity and train-test discrepancies.

There are other examples of extreme inter-class similarity that the CH-FT model was able to successfully manoeuvre. For example, class "3054-12.06.19.4" saw a significant rise in recall, jumping from 22.5% to 72.5%. Simultaneously, its slightly slimmer counterpart, "f.55.06.02.4", experienced an increase in precision from 35.6% to 52.6%. These two *T6* objects are almost indistinguishable and were frequently confused for one another by the CH model. Determining how the CH-FT model was able to distinguish between these objects, and whether this improvement was a result of model enhancements or merely coincidental, is difficult to answer. The underlying mechanisms behind the CH-FT model's enhanced recognition capabilities are yet to be fully explored. Employing visualization techniques, such as Grad-CAM, could illuminate the model's decision-making process by identifying focus areas during object recognition. For example, improved detection of hole patterns may explain the observed advancements in identifying items like "7054-12.82.13" and other *T7* or *T10* items. This hypothesis invites further investigation into the model's learning dynamics.

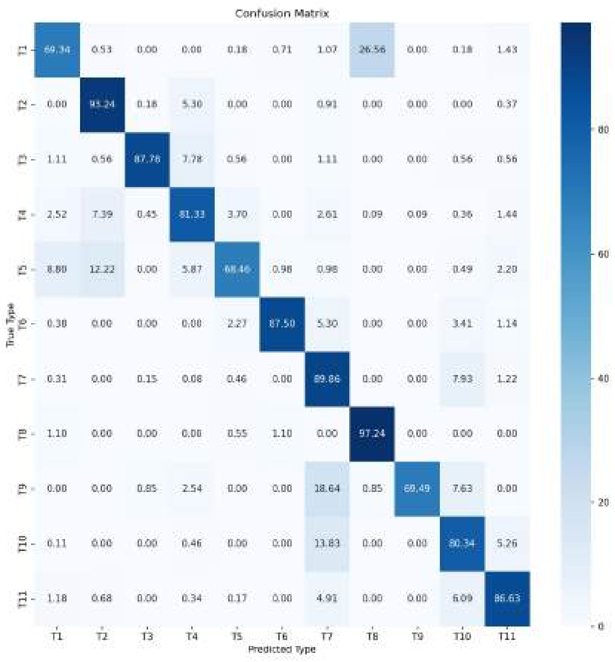


Figure E.9: Confusion Matrix Types CH Normalized ZoeDepth

The assessment of type performance metrics across both models highlights a trend of type-specific enhancements, particularly for *T2*, *T5*, *T6* and *T10* objects from CH to CH-FT. There was a notable decrease in out-of-type misclassifications for these categories, indicating improved recognition capabilities of the CH-FT model. These improvements were accompanied by increases in the average precision and recall scores per class (see table E.23), which could suggest that the CH-FT model not only distinguishes between types more effectively but also identifies objects within those types with greater accuracy. In general, the average recall for most types improved. However, the CH-FT model showed a notable increase in out-of-type misclassification errors for *T3* items (see figure E.9), often confusing them for *T4* objects. Though there are similarities in shape, the differences are stark and the errors surprising.

Type Name	**Precision (%)**	**Recall (%)**
T1	67.0	49.4
T2	26.8	40.2
T3	17.2	30.9
T4	47.4	33.9
T5	76.5	65.7
T6	69.0	54.8
T7	75.0	76.4
T8	48.8	77.0
T9	99.5	62.8
T10	70.9	69.4
T11	79.1	78.2

Table E.23: Precision and Recall Averages for each Type in Percent for CH Normalized ZoeDepth Model. Computed by Averaging the Metrics for each Class within the Type

All in all, by employing a CH-FT gradual unfreezing strategy, the model is able to leverage both its tuned feature extraction and tuned classification abilities. The approach mitigates catastrophic forgetting, balancing the retention of pre-learned features with the adaptation to new, task-specific nuances, resulting in enhanced precision and recall for most classes with only minimal reduction in others.

Best CH-FT ZoeDepth models (Normalized vs. Metric):

A comparison of the best late fusion CH-FT models using normalized and metric depth data shows that the metric model generally outperforms the normalized one in terms of precision and recall for most classes, as shown in figure E.7. The metric model boasts higher precision in 68 classes, the normalized model in 29 classes. For recall, the metric model outperforms the normalized model in 65 classes, compared to the latter's 23 classes. When delving into type performance metrics, the CH-FT Metric model's advantages are even more pronounced. It surpasses the normalized model in average precision and recall (see E.24) and reduces out-of-type misclassifications (see table E.11) across most object types. This indicates that the model is better at both distinguishing dissimilar as well as similar looking objects.

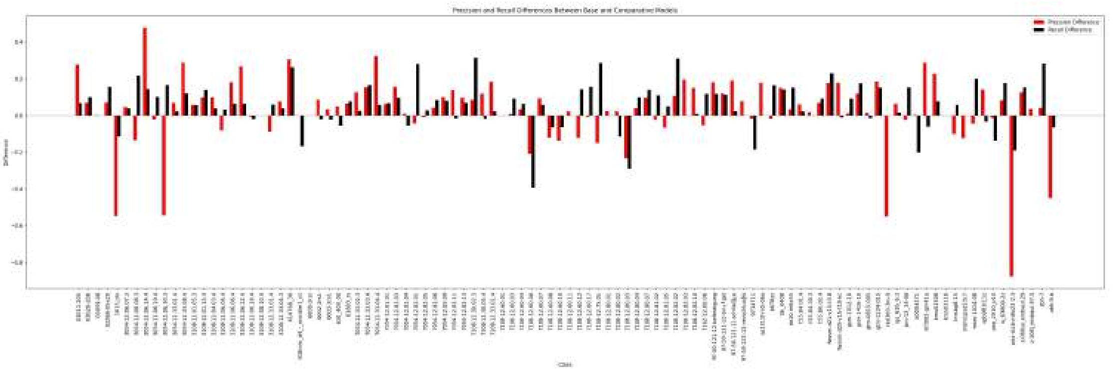


Figure E.10: Differences in Recall and Precision for Best CH-FT Models with Normalized or Metric Depth

While the CH-FT models demonstrate general improvement with metric depth data, it's crucial to acknowledge the normalized model's significantly better performance in specific classes. Notably, five classes experienced higher precision of over 45%. "wsx-stcb-m8x22-2.3" witnessed a complete fall to 0% in both recall and precision (refer to figure E.8) with metric model. The metric model was not able to maintain the gains in precision and recall that the normalized CH-FT model achieved over its CH counterpart, as described in **Best Normalized Depth Models (CH vs. CH-FT)**. This particular object's frequent confusion with "gtm-0815-005" and "gtm-1224-015" illustrates that some advancements achieved by the CH-FT models are not robust and heavily reliant on the exact type of depth data used. The attributes that led to successful identifications with the normalized data might not have been as effective or were altered in the context of metric depth.

Some exceptions aside, classes that scored low on recall and precision with the CH or normalized CH-FT models did not show marked improvement or deterioration with metric depth data, and vice versa. Examples include "03059-08", a case of train-test discrepancy, and ball bearings like "6000-2rsr", "6002-2rs1" (as shown in figure E.3), and "kms03116". These examples underscore the challenges all tested models face when dealing with high inter-class similarity or complex train-test discrepancy issues.

The metric depth model showcased its largest gains in both recall and precision over the normalized model across a diverse array of items such as "3054-12.06.14.4", "3054-12.33.08.4", "414768_30", "7054-12.33.03.4", "7054-12.33.04.4", "7108-12.30.02.3", "7108-12.82.02" and "fwssm-d21-v15-t3.8". These items encompass categories *T1*, *T3*, *T7* and *T10*. They lack shared defining features, suggesting the metric model's enhancements are broad-based rather than targeted corrections of specific errors prevalent in the normalized model.

A notable instance of improvement is "gtm-1224-015". The metric depth model corrected a critical error by the normalized model, which wrongly identified "gtm-1224-015" as "7108-12.33.01.4"—another ring-shaped item but with a significantly thicker ring. The metric model's ability to distinguish the objects hints at certain advantages it may have in delineating physical variations, such as the thickness.

Furthermore, the use of metric depth led to reduced out-of-type misclassifications for almost all types, but especially for *T1* and *T3*. For instance, there is a noticeable decrease in the misidentification of *T3* as *T4*, a distinction that should be very clear to a human observers. Similarly, *T7* objects are much less likely to be confused with *T10* objects, indicating that the model's enhanced depth perception aids in distinguishing flat objects from angular, curved or bent ones. However, it is noteworthy that despite these jumps, *T3* saw a downturn in average recall and precision per type. *T2* and *T4* may see improvements in average recall, but their precision scores also only remain steady. This may imply that although the metric model shows improvements in simpler cases, the models' inter-class struggles were not significantly ameliorated.

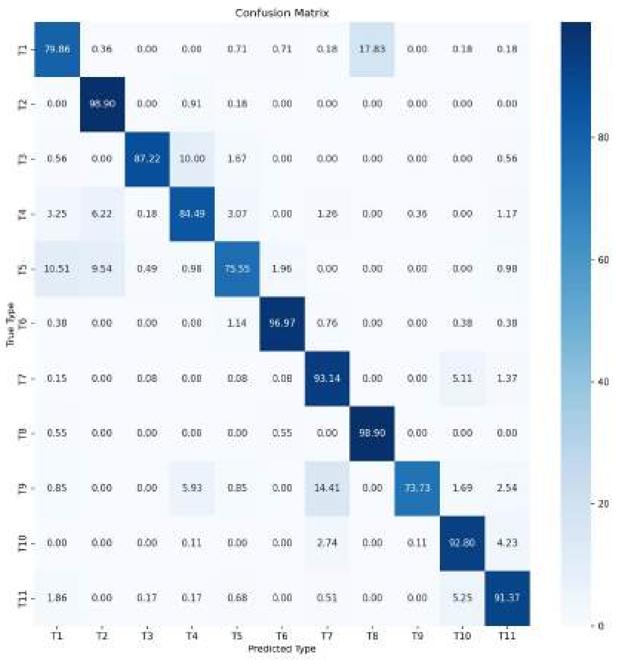


Figure E.11: Confusion Matrix Types CH-FT Metric ZoeDepth

Type Name	Precision (%)	Recall (%)
T1	69.6	61.0
T2	29.6	50.6
T3	24.5	32.6
T4	50.9	42.8
T5	78.1	70.3
T6	73.8	74.0
T7	88.8	85.6
T8	58.9	86.2
T9	78.9	57.6
T10	79.2	85.5
T11	84.1	85.9

Table E.24: Precision and Recall Averages for each Type in Percent for CH-FT Metric ZoeDepth Model. Computed by Averaging the Metrics for each Class within the Type

The fact that the metric model's enhancements are broadly observed does not exclude its effectiveness in tackling some high inter-class similarity challenges. For example, it increased the precision for "7108-12.82.03", without adversely affecting the recall for its lookalike "7108-12.82.02". Furthermore, the model corrected misidentifications between *T2* objects "b678zz" and "w_63800-2z", as well as "fwssm-d21-v15-t3.8" and "61803_rs". The model's ability to better recognize "f.55.04.01.4", despite significant differences between its training and testing images, as illustrated in Figure E.5 is astounding. It could be an indication of the model's enhanced capacity for discerning fine details. However, the specific mechanisms enabling the metric model to distinguish these subtle differences remains unknown and should be further explored.

E.5 Summary

This chapter critically examined the proposed methods of integrating depth data into the Synthnet image classification framework. It started with an overview of baseline models before outlining the different experimental configuration, including data augmentations, hyperparameter settings, and evaluation metrics. The experiments explored various modifications, including the use of different fusion techniques, types of D data, training schemes as well as freezing schedules. A thorough quantitative analysis assessed the models based on Acc@1, balanced accuracy, and F1-Macro, while also noting significant trends and changes across models. The chapter concluded with a detailed error analysis, pinpointing important mistakes made by the models. It also included an evaluation of performance against a novel categorization scheme designed to offer a clearer insight into model behaviour. Lastly, the chapter provided an in-depth comparison of error patterns and improvements across selected models, offering insights into how pipeline modifications influenced performance.

F Conclusion

The incorporation of D data into computer vision systems has marked a considerable advancement in enhancing model performances. Most prominently, the use of RGB-D images has led to improvements in the field of Salient Object Detection, as shown in recent surveys (for a comprehensive overview, refer to [ZFC$^+$23]). Despite these advancements, their application in image classification has not been as extensively researched, as remarked by Georgios Tziafas and Hamidreza Kasaei [TK23]. This paper aimed to address this gap by providing a detailed experimental evaluation of how D data can be successfully integrated into a transformer-based classification model, tailored to a particularly complex and challenging dataset.

At the onset of this research, several hypotheses were presented regarding methods as well as potential benefits of D data integration. This chapter revisits these hypotheses, evaluating their validity based on the experimental findings. Additionally, proposals for future research will be discussed.

F.1 Discussion

Synthnet Acc@1 improvement: The core hypothesis of this study posited that integrating D data into the RGB Synthnet model could enhance its Acc@1 by a noticeable margin. The outcomes largely validated this hypothesis, indicating a nuanced yet positive impact of D data on model performance. Specifically, the CH model exhibited a significant improvement of approximately 5% in accuracy when using a late fusion approach combined with Synthnet's data augmentations. Remarkably, this enhancement surged to 10% under specific conditions, such as reducing augmentations or leveraging metric depth. Early fusion models also demonstrated modest gains, affirming the beneficial role of D data across different fusion strategies. Notably, the most dramatic improvements were observed during full tuning, where D data contributions led to up to 25% accuracy gains in both early and late fusion models.

The extent of improvement, particularly the 25% increase under full tuning conditions, was unexpectedly high, albeit from a lower Synthnet baseline. It showed the substantial potential of D data in enhancing model accuracy, especially when models were trained from scratch or underwent extensive retraining.

The realization that not all improvements were uniform and depended critically on the deployment of sophisticated and gradually implemented freezing schedules was also insightful. This highlighted the complexity of effectively integrating D data into existing models.

Overall, the best results were observed during the hybrid CH-FT phase. This phase particularly showcased the strength of the late fusion approach and metric D data. The top result of over 69% Acc@1 came close to the RGB Synthnet model's best performance after adding domain adapation C-DAN and MCC techniques. Substantial gains over the Synthnet model can realistically be expected when employing these methods as well. Such findings illuminate the significant potential for D data to enhance classification accuracy, provided that the data is correctly harnessed and the model appropriately calibrated.

Effectiveness of Monocular Depth Estimation: The hypothesis concerning the effectiveness of monocular D estimation for improving classification tasks has been emphatically confirmed through this study. The integration of the ZoeDepth monocular D estimator significantly improved the models' performances as compared to the baseline Synthnet model, particularly when using metric D estimations. This enhancement underscores the potential of monocular depth estimation as a viable and efficient method for augmenting the capability of existing image classification frameworks, especially in scenarios where acquiring real D data is impractical or too costly.

One of the most striking revelations of this study was the ability of ZoeDepth's estimated D maps, produced from conventional RGB images, to compete with, and in some instances outperform, D data rendered from sophisticated Blender CAD models. It also highlighted the limitation of relying on CAD model data, namely, its inability to generate depth maps for real photographs, which often serve as the target domain. This outcome challenges the idea that D information must be acquired through high-precision methods to enhance model performance. It suggests that the quality of D maps generated by computational estimation methods like ZoeDepth can be exceptionally high, providing a viable alternative to more traditional D data sources.

Late Fusion vs. Early Fusion: This hypothesis posited that late fusion techniques would outperform early fusion strategies. The experimental outcomes substantiated the hypothesis, revealing a distinct advantage for late fusion approaches over early fusion methods. The effectiveness of the siamese-like architecture for separate feature extraction from RGB and D data demonstrated the importance of treating RGB and D information distinctly to capitalize on the unique properties of each data type.

The early fusion models, which combined RGB and D data at the input level before feature extraction, exhibited performance on par with the baseline RGB Synthnet Model.[18] However, in the context of more complex CH-FT configurations, early fusion methods barely matched the Synthnet RGB model. This outcome was sur-

[18]With the notable exception of the early fusion FT experiment's drastic improvement.

prising. The disappointing performance of the early fusion model suggests that a simplistic merging of RGB and D data might negate the potential advantages offered by D information. Nonetheless, there were also indications that with further experimentation and refinement, early fusion techniques could be improved, thereby enabling more effective cross-modal interaction.

More Elaborate Freezing Schedules Improve Model Performance: The study confirmed that applying more elaborate and dynamically adjusted freezing schedules could improve the efficiency of vision transformer models, especially when incorporating D data. This improvement was noted in both early and late fusion approaches. In contrast, basic freezing strategies failed to significantly enhance the performance of the RGB Synthnet model.

Gradual unfreezing strategies, where different layers of the transformer model were systematically unfrozen over the course of training, emerged as the most effective approach. This method allowed for the models to slowly adjust to the additional D data, thereby significantly outperforming more static freezing strategies.

The finding that surgical fine-tuning —though beneficial in certain contexts— did not surpass the outcomes of the best gradual unfreezing strategies was notable. It highlighted the complexity of optimizing transformer models for D data integration and the potential limitations of focusing too narrowly on specific layers without considering the broader dynamics of model training and adaptation. While surgical fine-tuning showed promise, it did not achieve the pinnacle of performance seen with the best gradual unfreezing strategies. However, it illuminated the potential benefits of layer-specific adjustments in response to the type of distribution shift encountered, suggesting areas for further exploration and refinement.

F.2 Future Work

The methodologies investigated in this thesis offer considerable potential for practical application. Tools like ZoeDepth present a promising avenue for enhancing system performance seamlessly without necessitating costly hardware upgrades or intricate data collection methods. Future research should aim to evaluate a variety of depth estimators across a broad spectrum of real-world datasets and environments to verify their reliability and adaptability.

In the closing phase of this project, a preliminary examination on the Topex dataset was conducted with a new depth estimation model released in January 2024 [YKH$^+$24]. The DepthAnything model was trained on a vast dataset comprising over 1.5 million labeled images and more than 62 million unlabeled images, showcasing robustness in handling diverse imaging conditions without the dependency on complex technical modules or extensive data collection processes.

A noteworthy observation during this project's cursory test runs with DepthAnything was the significantly reduced inference time of the model as compared to ZoeDepth. This efficiency indicates a potential for faster processing capabilities in real-world applications. The performance of DepthAnything in these preliminary experiments was found to be similar to that of ZoeDepth, suggesting that with further optimization, DepthAnything could achieve even better results (refer to table F.1).

Model	Pre-training	train scheme	CH transform	FT transform	FT freezing schedule	data	Acc@1	Balanced Acc	F1-Macro
SwinV2	IN22K	CH	Base	None	None	Norm	59.76	61.56	57.46
SwinV2	IN22K	CH	Auto	None	None	Norm	63.33	64.11	61.11
SwinV2	IN22K	CH	Base	None	None	Metric	55.48	57.21	53.18
SwinV2	IN22K	CH-FT	Base	Augmix	[C3210]	Norm	68.24	69.21	65.15
SwinV2	IN22K	CH-FT	Base	Augmix	[C+3210]	Norm	68.19	69.34	65.09
SwinV2	IN22K	CH-FT	Base	Augmix	[C3][C32][C321][C3210]	Norm	69.17	69.65	65.87
SwinV2	IN22K	CH-FT	Base	Augmix	[C+3][C+32][C+321][C+3210]	Norm	69.17	69.99	66.09
SwinV2	**IN22K**	**CH-FT**	**Auto**	**Augmix**	**[C3210]**	**Norm**	**70.83**	**71.16**	**68.19**
SwinV2	IN22K	CH-FT	Auto	Augmix	[C+3210]	Norm	68.24	68.81	65.87
SwinV2	IN22K	CH-FT	Auto	Augmix	[C3][C32][C321][C3210]	Norm	70.19	69.88	67.39
SwinV2	IN22K	CH-FT	Auto	Augmix	[C+3][C+32][C+321][C+3210]	Norm	69.77	70.47	67.40
SwinV2	IN22K	CH-FT	Base	Metric	[C3210]	Metric	63.52	63.91	60.22
SwinV2	IN22K	CH-FT	Base	Metric	[C+3210]	Metric	65.34	65.52	62.15
SwinV2	IN22K	CH-FT	Base	Metric	[C3][C32][C321][C3210]	Metric	62.98	63.44	59.96
SwinV2	IN22K	CH-FT	Base	Metric	[C+3][C+32][C+321][C+3210]	Metric	64.95	65.64	62.52

Table F.1: **Late Fusion-DepthAnything-Experiments.** Test Results for RGB-D Topex-Printer target-domain test set with DepthAnything D data. Training was applied to the model with source-domain training data with DepthAnything D data. CH learning rate is 1e-3. FT learning rate is 1e-5. Metrics are in %.

Had this project been extended, several other straightforward measures could have potentially led to further substantial improvements in model performance. To begin with, repeating the experiments with varied random seeds would have strengthened the research by providing a clearer understanding of the model's robustness. Implementing mixed training sets that predominantly consist of synthetic data with a sprinkling of real images has been show to lead to enhancements [Nik19] and could have easily been implemented. Developing and evaluating bespoke data augmentations tailored to RGB-D data could have expanded and strengthened the dataset. It can also be assumed that the hyperparameters optimized for the RGB Synthnet pipeline might not perform as well on the added D dataset. Adjusting the hyperparameters could potentially overcome the limitations associated with merely copying the settings from the Synthnet RGB model. Exploring alternative model fusion techniques, such as employing addition strategies or max pooling instead of straightforward concatenation, may have refined the process of integrating D data. Moreover, delving into advanced fusion methods beyond late and early fusion, such as deep fusion strategies, could be a promising avenue to pursue. Although domain adaptation strategies such as C-DAN and MCC had shown impressive outcomes in the SynthNet study, this research did not implement them. There are compelling arguments that such strategies could work just as well on the updated models with additional D data. How much actual performance improvement this would result in would have to be tested.

This project also tried to keep in mind the importance of developing models that are not just precise, but also efficient and and capable of working effectively in various conditions and settings. Further research should consider the applicability and scalability of the applied techniques across different datasets (e.g., ModelNet-10) and across a range of vision transformer model sizes (from mini to large Swin models for example).

Lastly, the application of visualization tools, such as Grad-CAM, for detailed error analysis would offer profound insights into the decision-making processes of models and the influence of D data on classification decisions. This approach could unveil critical aspects of how D information modifies feature maps, providing a clearer understanding of model behavior and identifying areas for improvement.

Bibliography

[BBW$^+$23] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth, 2023.

[DBK$^+$21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[DDS$^+$09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, K. Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[FBAW21] Shariq Farooq Bhat, Ibraheem Alhashim, and Peter Wonka. Adabins: Depth estimation using adaptive bins. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2021.

[FHSR$^+$21] Di Feng, Christian Haase-Schutz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, March 2021.

[Fir16] Michael Firman. Rgbd datasets: Past, present and future, 2016.

[GBV20] Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification: an overview, 2020.

[GK17] S. Alireza Golestaneh and Lina J. Karam. Spatially-varying blur detection based on multiscale fused and sorted transform coefficients of gradient magnitudes, 2017.

[HM22] Alfred Hagberg and Mustaf Musse. Instance segmentation on depth images using swin transformer for improved accuracy on indoor images, 2022.

[HMC$^+$20] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty, 2020.

[JCP22] Xingzhao Jia, Dongye Changlei, and Yanjun Peng. Siatrans: Siamese transformer network for rgb-d salient object detection with depth image classification, 2022.

[JSWL22] Junguang Jiang, Yang Shu, Jianmin Wang, and Mingsheng Long. Transferability in deep learning: A survey, 2022.

[JWLW20] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation, 2020.

[KMBB23] Teerath Kumar, Alessandra Mileo, Rob Brennan, and Malika Bendechache. Image data augmentation approaches: A comprehensive survey and future directions, 2023.

[KRJ$^+$22] Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. Fine-tuning can distort pretrained features and underperform out-of-distribution, 2022.

[KWSS22] Donghyun Kim, Kaihong Wang, Stan Sclaroff, and Kate Saenko. A broad study of pre-training for domain generalization and adaptation, 2022.

[LCT$^+$23] Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts, 2023.

[LCWJ18] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation, 2018.

[LHL$^+$22] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin transformer v2: Scaling up capacity and resolution, 2022.

[LLC$^+$21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[LPVG23] Chen Cecilia Liu, Jonas Pfeiffer, Ivan Vulić, and Iryna Gurevych. Improving generalization of adapter-based cross-lingual transfer with scheduled unfreezing, 2023.

[LTHX22] Zhengyi Liu, Yacheng Tan, Qian He, and Yun Xiao. Swinnet: Swin transformer drives edge-aware rgb-d and rgb-t salient object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(7):4486–4497, July 2022.

[Nik19] Sergey I. Nikolenko. Synthetic data for deep learning, 2019.

[OB21] Juri Opitz and Sebastian Burst. Macro f1 and macro f1, 2021.

[Pel21] Peltarion. Self-attention video. `https://peltarion.com/blog/data-science/self-attention-video`, 2021. Accessed: 2023-12-22.

[PG22] Jo Plested and Tom Gedeon. Deep transfer learning for image classification: a survey, 2022.

[PUK^{+}17] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge, 2017.

[RHH^{+}23] Dennis Ritter, Mike Hemberger, Marc Hönig, Volker Stopp, Erik Rodner, and Kristian Hildebrand. Cad models to real-world images: A practical approach to unsupervised domain adaptation in industrial object classification, 2023.

[RLH^{+}20] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer, 2020.

[RPM^{+}22] Alina Roitberg, Kunyu Peng, Zdravko Marinov, Constantin Seibold, David Schneider, and Rainer Stiefelhagen. A comparative analysis of decision-level fusion for multimodal driver behaviour understanding, 2022.

[Rud19] Sebastian Ruder. *Neural transfer learning for natural language processing*. PhD thesis, National University of Ireland, Galway, June 2019.

[STF22] Shiv Shankar, Laure Thompson, and Madalina Fiterau. Progressive fusion for multimodal integration, 2022.

[TK23] Georgios Tziafas and Hamidreza Kasaei. Early or late fusion matters: Efficient rgb-d fusion in vision transformers for 3d object recognition, 2023.

[VCS17] Igor Vasiljevic, Ayan Chakrabarti, and Gregory Shakhnarovich. Examining the impact of blur on recognition by convolutional networks, 2017.

[VSP^{+}23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[WLB19] Isaac Ronald Ward, Hamid Laga, and Mohammed Bennamoun. Rgb-d image-based object detection: from traditional methods to deep learning techniques, 2019.

[YKH+24] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data, 2024.

[ZFC+23] Tao Zhou, Deng-Ping Fan, Ming-Ming Cheng, Jianbing Shen, and Ling Shao. RGB-D Salient Object Detection: A Survey. GitHub repository, 2023. Accessed: 2023-11-30.

G Appendix

G.1 Type Categorization

Type 1 (T1): Metal rods with external threading.

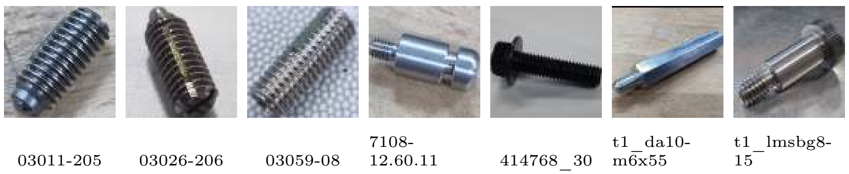


Figure G.1: Collection of T1 Classes

Type 2 (T2): Ball bearings, which consist of spherical rollers enclosed between two rings.

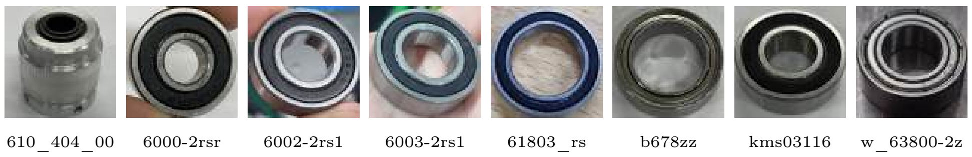


Figure G.2: Collection of T2 Classes

Type 3 (T3): Washers, which are flat disks with holes.

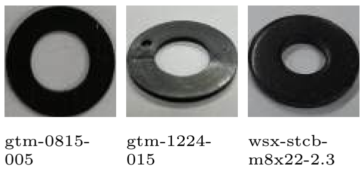


Figure G.3: Collection of T3 Classes

Type 4 (T4): Cylindrical sleeves or rings, often with a smooth exterior and a central hole, designed to fit over a rod or shaft.

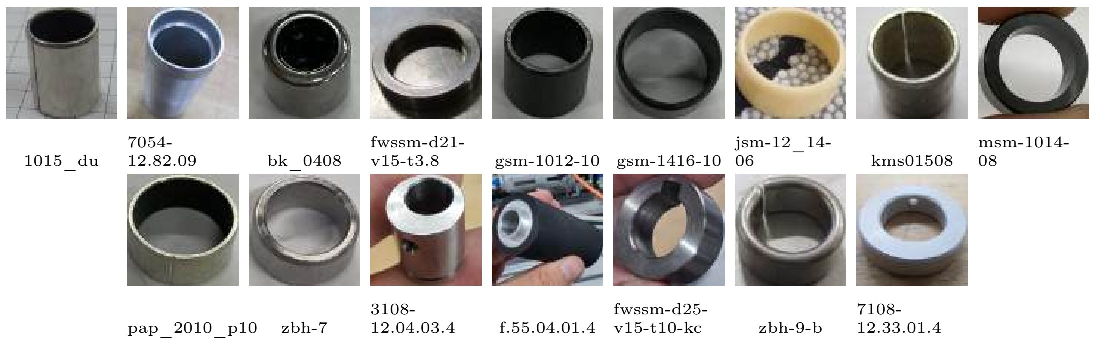


Figure G.4: Collection of T4 Classes

Type 5 (T5): Metal rods with internal threading.

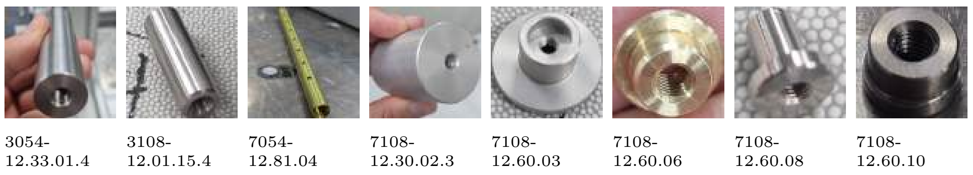


Figure G.5: Collection of T5 Classes

Type 6 (T6): Metal rods without internal or external threading

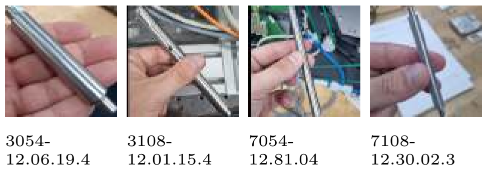


Figure G.6: Collection of T6 Classes

Type 7 (T7): Metal plates with holes or slots for mounting or assembly. They have a predominantly flat shape but can vary in thickness and size.

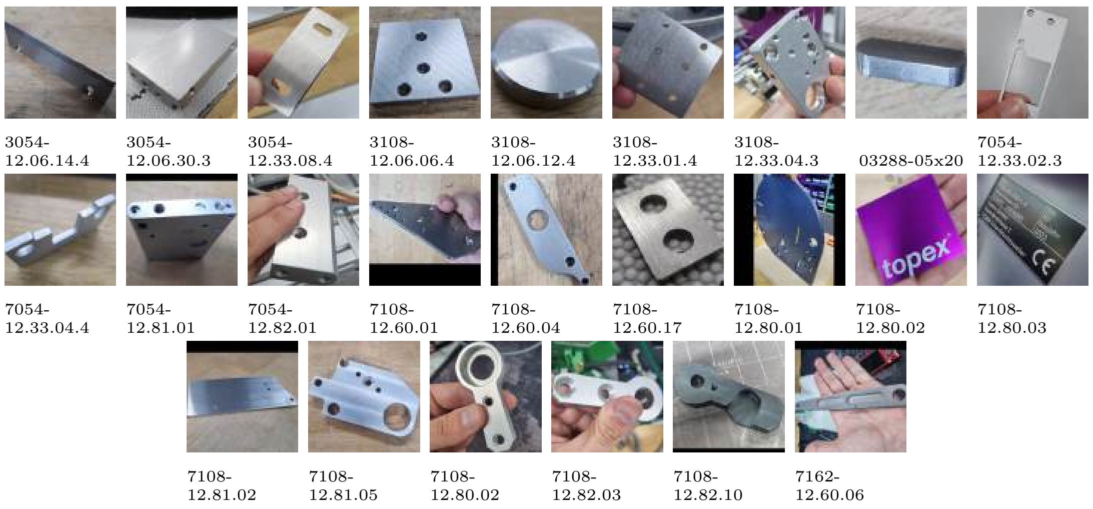


Figure G.7: Collection of T7 Classes

Type 8 (T8): Metal springs coiled into a helical shape with loops or hooks at each end.

Figure G.8: Collection of T8 Classes

Type 9 (T9): Flexible looped components made from rubber or similar material.

Figure G.9: Collection of T9 Classes

Type 10 (T10): Mechanical brackets featuring a range of geometric shapes with holes for fastening or assembly purposes. They have a rigid structure with angular, curved or bent profiles.

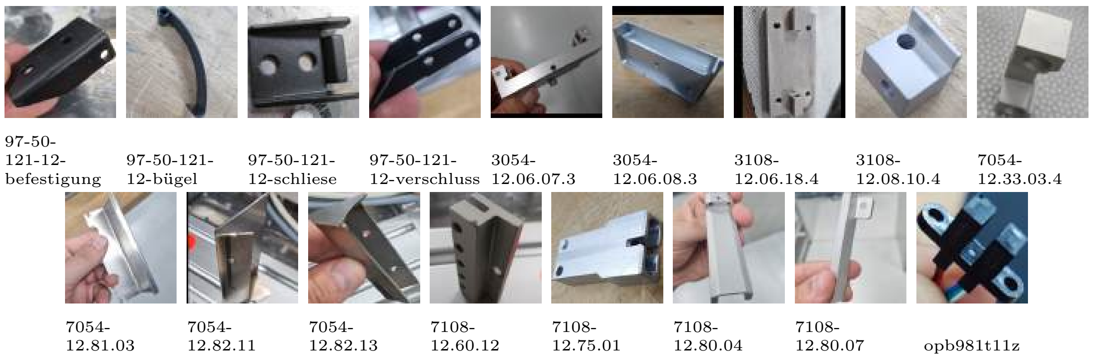


Figure G.10: Collection of T10 Classes

Type 11 (T11): Miscellaneous items that cannot be easily categorized and bear little resemblance to other objects in the dataset.

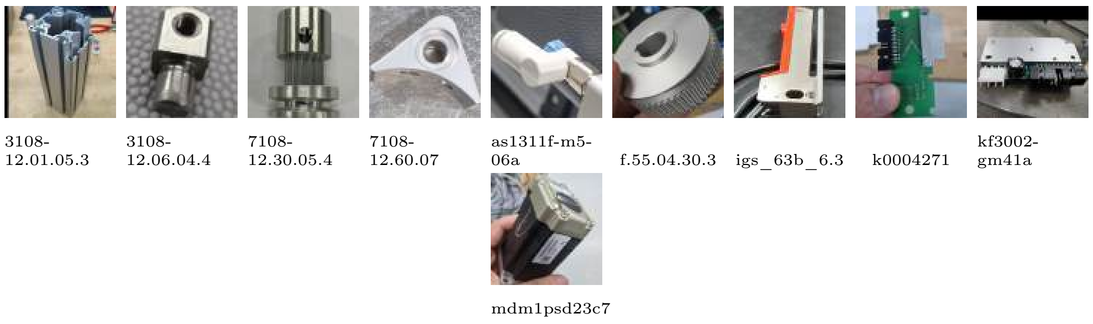


Figure G.11: Collection of T11 Classes