

Addressing the Labeled Data Scarcity Problem in Deep Learning

by

Mehran Khodabandeh

M.Sc., Simon Fraser University, 2015

B.Sc., Sharif University of Technology, 2013

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Computing Science
Faculty of Applied Sciences

© Mehran Khodabandeh 2023
SIMON FRASER UNIVERSITY
Summer 2023

Copyright in this work is held by the author. Please ensure that any reproduction or re-use is done in accordance with the relevant national copyright legislation.

Declaration of Committee

Name: Mehran Khodabandeh

Degree: Doctor of Philosophy (Computing Science)

Thesis title: Addressing the Labeled Data Scarcity Problem in Deep Learning

Committee:

Chair: Saeid Asgari
Adjunct Professor, Computing Science

Greg Mori
Supervisor
Professor, Computing Science

Ze-Nian Li
Committee Member
Professor, Computing Science

Binay Bhattacharya
Examiner
Professor, Computing Science

Michael S. Brown
External Examiner
Professor, Electrical Engineering and Computing Science
York University

Abstract

A significant challenge in machine learning and, more specifically, deep learning is the necessity for extensive labeled data. Given the high cost, time, and complexity of procuring such data, especially in computer vision applications, we focus on improving strategies for scenarios with limited data. We propose novel methodologies in three primary areas: Active Learning, Data Augmentation, and Domain Adaptation. Firstly, in the domain of Active Learning, we propose an algorithm that caters to cases with structured outputs, efficiently expanding the labeled dataset from a large pool of unlabeled data. By exploiting the inherent interdependencies in each sample, we can identify the most informative components for further labeling. Secondly, under Data Augmentation, we propose a unique algorithm for synthesizing realistic human action videos to enhance the training dataset for human action recognition tasks. Lastly, in the area of Domain Adaptation, we address the challenges of varying data distributions between training and deployment stages in object detection. A novel algorithm is proposed to enable object detectors adapt effectively using available unlabeled data from the target domain.

Keywords: Active Learning; Data augmentation; Domain adaptation; Group Activity Detection; Human Action Classification; Object Detection

Dedication

With heartfelt appreciation, I dedicate this dissertation to my parents, Ali and Roya, whose unwavering encouragement and sacrifices have paved the way for my academic success. To my loving wife, Momoka, whose belief in me, encouragement and patience has given me the strength to overcome challenges. To my brother, Mehrdad, whose sibling bond and friendship have provided inspiration throughout this journey. And to my dear friend, Masooud, whose intellectual discussions and constant support have shaped my thoughts.

Thank you for being my pillars of strength and for always being there for me. I am forever indebted to your love and support.

Acknowledgements

Firstly, I would like to express my profound gratitude to my supervisor, Prof. Greg Mori. His constant guidance and deep knowledge have been instrumental in shaping my research work. He not only guided me academically but also shared valuable life advice. His unwavering support extended into my personal life by giving me a research opportunity in Japan where I met my wife. He encouraged me in my entrepreneurial pursuits and constantly supported my PhD journey. My achievements, personally, professionally, and academically, would not have been possible without his help.

Secondly, I owe a heartfelt thanks to my wife. She has always stood by my side, cheering me on and encouraging me through the ups and downs of this thesis journey. I am truly grateful to have her as my partner, and I owe a significant part of my success to her unwavering dedication and love. Thank you for being my rock and for believing in me every step of the way. Thank you for being my solid foundation and for believing in me every step of the way.

I would like to express my deepest gratitude to my parents for their unwavering support and belief in my abilities. Their constant encouragement and sacrifices have been the driving force behind my pursuit of this PhD, and I am forever grateful for their love and guidance. Additionally, I would like to extend my thanks to my brother for his friendship and for stepping in to assist our parents during my absence.

I am deeply grateful to my mentors - Dr. Hossein Hajimirsadeghi, Dr. Arash Vahdat, Dr. Mani Ranjbar, Dr. Hamid Vaezi Joze, and Prof. Shinichi Satoh. Their guidance and mentorship throughout my PhD journey and the amazing internships that I did with them, have been crucial to the completion of this thesis.

I would also like to show my gratitude for my dear friends and colleagues whom I met during my university studies and internships. In particular, I am thankful to have spent quality time with Ali Madooei, Abbas Saadat, Hedayat Zarkoob, Abdollah Safari, Shahram Pourazadi, and Frederik Mallmann-Trenn.

I would like to express my heartfelt gratitude to my friend, Masoud Ramazani, for his profound impact on my personal and academic growth.

Lastly, I want to thank you Chad Gupta, for your endless support and vast knowledge. You have been a significant help.

To everyone I've mentioned and those I haven't, thank you. This journey wouldn't have been the same without you.

Table of Contents

Declaration of Committee	ii
Abstract	iii
Dedication	iv
Acknowledgements	v
Table of Contents	vi
List of Tables	ix
List of Figures	x
1 Introduction	1
1.1 Contributions	2
2 Related Work	4
2.1 Data Augmentation	4
2.1.1 Model free techniques	4
2.1.2 Model based techniques	8
2.1.3 Policy based techniques	10
2.2 Transfer Learning	13
2.2.1 Inductive Transfer Learning	13
2.2.2 Transductive Transfer Learning	19
2.3 Active Learning	23
2.4 Summary	26
3 Active Learning for Structured Prediction from Partially Labeled Data	28
3.1 Abstract	28
3.2 Introduction	28
3.2.1 Related Work	30
3.3 Active Learning for Structured Prediction	32
3.3.1 Stage 1: Structured Prediction Model	32

3.3.2	Stage 2: Instance Selection Strategy	34
3.4	Experiments	38
3.4.1	Datasets	38
3.4.2	Baselines	39
3.5	Results and Analysis	39
3.5.1	Quantitative Results	39
3.5.2	Comparison to Supervised Methods	40
3.5.3	Analysis	41
3.6	Discussion	44
3.6.1	Recent Advancements in Data Labelling Systems and Active Learning	44
3.6.2	Potential Future Directions	45
3.7	Summary	46
4	Dataset Augmentation for Human Action Classification	47
4.1	Abstract	47
4.2	Introduction	47
4.3	Related Works	49
4.3.1	Action Recognition	49
4.3.2	Video Generative Models	50
4.4	Method	51
4.4.1	Video Generation from Skeleton and Reference Appearance	51
4.4.2	Skeleton Trajectory Generation	53
4.5	Datasets and Action Recognition Methods	56
4.5.1	datasets	56
4.5.2	Action Recognition Methods	56
4.6	Experiments	57
4.6.1	Generated Trajectory	58
4.6.2	New Subjects	58
4.6.3	New Actions	58
4.6.4	dataset Expansion	60
4.6.5	Real World	60
4.7	Discussion and Future Work	61
4.8	Summary	62
5	A Robust Learning Approach to Domain Adaptive Object Detection	64
5.1	Abstract	64
5.2	Introduction	64
5.3	Previous Work	66
5.4	Method	67
5.4.1	Faster R-CNN	68

5.4.2	A Probabilistic View of Faster R-CNN	69
5.4.3	Robust Faster R-CNN	69
5.4.4	Image Classification:	72
5.5	Experiments	73
5.5.1	Adapting synthetic data to real world	75
5.5.2	Adapting normal to foggy weather	76
5.5.3	Adapting to a new dataset	76
5.6	Summary	76
6	Conclusion	79
6.0.1	Active Learning for Structured Prediction from Partially Labelled Data	79
6.0.2	Dataset Augmentation for Human Action Classification	80
6.0.3	Robust Learning for Domain-Adaptive Object Detection	81
6.0.4	Final note	81
	Bibliography	82

List of Tables

Table 2.1	Well-known algorithms in computer vision and their corresponding data augmentation techniques	5
Table 3.1	Results of our method with different number of epochs at each iteration. . .	41
Table 4.1	Action recognition on UT dataset using original data compared to generated from scratch data with proposed method in §4.4.1 and §4.4.2	58
Table 4.2	Performance comparison of multiple algorithms, trained on original data and additional subjects.	59
Table 4.3	Per class average accuracy for model trained by i3d using original training data from UT plus new action clip generated by our method using skeleton extracted from KTH training set.	60
Table 4.4	The comparison of dataset expansion by original data for UTK and SUB dataset.	61
Table 2	Quantitative results comparing our method to baselines for adapting from <i>Cityscapes</i> to <i>Foggy Cityscapes</i> . We record the average precision (AP) on the <i>Cityscapes</i> validation set. “Cls-Cor” represents “classification error correction”, Box-R stands for “Bounding Box Refinement” component, and FN-Cor stands for “False Negative Correction” component of our method. The last row shows the base detector’s performance if labeled data for target domain was available.	74
Table 1	Quantitative results comparing our method to baselines for adapting from <i>SIM 10K</i> dataset to <i>Cityscapes</i> . We record average precision (AP) on the <i>Cityscapes</i> validation set. The last row shows the base detector’s performance if labeled data for target domain was available.	75
Table 5.3	Quantitative comparison of our method with baselines for adapting from <i>KITTI</i> to <i>Cityscapes</i> . We record average precision (AP) on the <i>Cityscapes</i> test set. The last row gives the base detector’s performance if labeled data for the target domain was available.	77
Table 5.4	Quantitative comparison of our method with baselines for adapting <i>Cityscapes</i> to <i>KITTI</i> . We record average precision (AP) on the <i>KITTI</i> train set. The last row gives the base detector’s performance if labeled data for target domain was available.	77

List of Figures

Figure 2.1	Summary of the results obtained from ImageNet classification, ImageNet localization, and Pascal VOC 07 detection tasks using Mixup, Cutout, and our CutMix techniques. CutMix demonstrates a substantial improvement in the performance of these tasks. [250] © 2019 IEEE.	7
Figure 2.2	The model enhances the realism of synthetic images generated by a simulator by leveraging unlabeled data, while also maintaining the annotation information [191]. @ 2017 IEEE	9
Figure 2.3	Several t-SNE visualizations of ImageNet validation set using different features [34].	15
Figure 2.4	Mirsra et al. [138] experimented with several multi-task (two-task) architectures by dividing the ConvNet [113] at various layers for two sets of tasks. They compared the performance of these networks on each task to that of task-specific networks. They observed that the most effective multi-task architecture is dependent on the specific tasks involved and does not necessarily generalize well to different pairs of tasks. @ 2016 IEEE . . .	17
Figure 2.5	Kendall et al. [99] developed a method to combine multiple regression and classification loss functions for multi-task learning. Their architecture accepts a single monocular RGB image as input and generates pixel-wise classification, instance semantic segmentation, and an estimation of per-pixel depth. Multi-task learning can enhance accuracy compared to individually trained models, as cues from one task (e.g., depth) help regularize and boost the generalization performance in another domain (e.g., segmentation). @ 2018 IEEE	18
Figure 2.6	To enhance domain adaptation performance, Deep CORAL aligns the covariance matrices of the source and target domains in a deep neural network’s feature space. This is achieved by minimizing the CORAL loss term in the classification loss, which reduces the difference between the correlation matrices of the source and target feature domains[201].	22

Figure 3.1	Example of active learning for recognizing human actions. Given a partially labeled dataset (labeled examples in green, unlabeled red), we train a classifier to recognize the actions of each person and group in a scene. Our active learning criterion selects individual people or scenes to be annotated by the user (shown in yellow). These new data are added to the training dataset, and the process repeated.	29
Figure 3.2	Overview of the method. A structured prediction model is trained given the training dataset. This model produces a probability distribution for every node. In the second stage for each unlabeled node we estimate expected entropy reduction by fixing its value and running inference. We sort all nodes based on expected entropy reduction their labeling would produce. The top K nodes are selected and sent to user for labeling. These nodes can be either human action or scene labels. The new labeled data is then added to the training set and this process repeats.	33
Figure 3.3	In the above graph node m is unlabeled. In order to compute the <i>average entropy</i> of the graph if the label of node m was given and equal to 2 we do the following. First set the label distribution of the node m to $[0, 1, 0, 0, 0]$ ("2"). Then we perform inference using the current learned inference machine. This will produce the probability distributions of all nodes, \hat{P} . Then the <i>average entropy</i> is the mean entropy over all the nodes. In this figure the blue histograms are the action label distributions of the nodes obtained given the current model; and the green histograms are the probability distributions of the nodes after observing $m = 2$	37
Figure 3.4	Results of comparison of our method against baselines on <i>Volleyball Dataset</i> (a and b), <i>Collective Activity Dataset</i> (c and d). The y-axis is accuracy and x-axis is iteration. For all the methods we start from the same small initial labeled set so the accuracies of the first column are exactly the same. Our method outperforms the other methods in both action and scene accuracy ¹	40
Figure 3.5	Distribution of scene/action labels at each iteration	42
Figure 3.6	Examples of chosen variables in the Volleyball dataset. Each row shows one pair of before/after an active learning iteration. Left image shows scene with current labels (ground truth in green box, incorrectly predicted labels in yellow box). Yellow arrow shows variable chosen for labeling by oracle by our method. Note that selection of group activity (<i>scene</i> annotation) in top 2 rows or actions of specific people (bottom 2 rows) help in correcting other labels afterwards.	43

Figure 4.1	Our algorithm takes as input an action label, a set of reference images and an arbitrary background. The output is a generated video of the person in the reference image performing a given action. We approached this problem in two stages. Firstly (left side) a generative model trained on a small <i>labeled</i> dataset of skeleton trajectories of human actions, generates a sequence of human skeletons conditioned on the action label. Secondly (right side), another generative mode trained on an <i>unlabeled</i> set of human action videos, generates a sequence of photo-realistic frames conditioned on the given background, generated skeletons, and the person’s appearance given in the reference frames. This produces an arbitrary number of human action videos.	48
Figure 4.2	Network Architecture	52
Figure 4.3	Trajectory GAN network architecture.	54
Figure 4.4	Samples of generated skeleton sequences, conditioned on action label (e.g. throwing, hand waving, sitting).	55
Figure 4.5	Generated images on two different datasets.	57
Figure 4.6	The screen shot of a video generated by UTK expansion. The first row shows skeleton clips extracted from an arbitrary action. Second to fourth rows show the generated video for subjects from different clip carrying out that specific action.	59
Figure 4.7	Perspective transform example.	61
Figure 5.1	The robust learning approach consists of three phases. In phase 1, a detection module is trained using labeled data in the source domain. This detector is then used to generate noisy annotations for images in the target domain. In phase 2, the annotations assigned in phase 1 are refined using a classification module. Finally, in phase 3, the detector is retrained using the original labeled data and the refined machine-generated annotations in the target domain. Retraining is formulated to account for the possibility of mislabeling.	67

Figure 5.2 Qualitative comparison of our method with Faster R-CNN on the “*Cityscapes* → *KITTI*” experiment. Each column corresponds to a particular image in the *KITTI* test set. Top and bottom images in each column illustrate the bounding boxes of the cars detected by Faster R-CNN and our method respectively. In the first two columns our method corrects several false positives. In all cases our method successfully corrected the size/location of the bounding boxes (*e.g.* the rooflines in the third column). In the fourth and fifth examples, our method has detected cars that Faster R-CNN has missed. Nevertheless, false positives do occur (*e.g.* in column five), though the probability of those specific false positives is low (53% in this example). 74

Chapter 1

Introduction

Machine learning is a fast-growing field that is revolutionizing the way we address a wide range of real-world problems. It has been used in diverse tasks, including image recognition, natural language processing, anomaly detection, and even drug discovery. Deep learning, a branch of machine learning, has gained popularity thanks to its ability to automatically and effectively learn intricate patterns from vast amounts of data. Nonetheless, a significant obstacle in deep learning is the need for a massive quantity of labeled data to train models efficiently. In many real-life situations, obtaining labeled data can be difficult or expensive, leading to a constrained pool of accessible labeled data.

For example, in the field of computer vision, obtaining labeled data for autonomous vehicle systems may require extensive and costly manual annotation of images or videos, capturing various road conditions, weather, and potential hazards. In the domain of facial recognition, collecting labeled data could involve acquiring consent from individuals to use their images, which can be time-consuming and raise privacy concerns. Similarly, training models for object detection and segmentation often require large datasets with accurately labeled images of numerous objects and their corresponding boundaries. The process of annotating these images can be labor-intensive, requiring significant human effort to generate high-quality labeled datasets. These challenges with gathering data can hinder the development and deployment of machine learning models in real-world computer vision applications.

One might argue that, for industries such as autonomous driving, the cost of labeling data is negligible in comparison to the profits that can be garnered from training accurate models, justifying the cost. Additionally, recent advancements like Amazon Turk and other similar services have made the acquisition of labeled data more cost-effective. While these arguments hold true, the process is far from being free of charge. The cost can be prohibitive for small businesses and individual researchers who may lack the resources to afford large labeled datasets for custom tasks without access to already available public datasets. Furthermore, in many real-world applications such as medical imaging or disaster prediction, the events of interest are often rare or unique. These instances are underrepresented in large datasets, making it challenging to train a reliable model with these scarce labeled examples.

Moreover, not all labeled data can be directly applied to specific tasks due to differences in context, environment, or demographic. For instance, a model trained on traffic data from one city might not perform well in another city due to different driving patterns and infrastructure. Additionally, strict regulations and ethical considerations in fields like healthcare or finance often prevent the sharing and labeling of sensitive data.

We also cannot predict all the types of data and problems AI will encounter in the future. Preparing our models to work well with limited labeled data makes them more flexible and adaptable. Furthermore, methods that can learn from fewer labeled examples allow us to leverage the vast amount of unlabeled data more effectively, increasing the efficiency of our models.

In light of these challenges, this dissertation proposes novel approaches in Active Learning, Data Augmentation, and Domain Adaptation, specifically designed for scenarios where data scarcity is a concern. Despite the abundance of labeled data in some areas, it is crucial to continue research into learning from limited labeled data to ensure the broad applicability and versatility of AI. This research aims to contribute to this ongoing effort, addressing the scarcity of labeled data in deep learning.

In the realm of *Active Learning*, where a small amount of labelled data and a large pool of unlabelled data are available, the objective is to efficiently expand the labeled dataset in order to train a better model. We propose a new algorithm that is specific to the case where output is structured. We leverage the inherent interdependencies and relationships among the components of each sample and select the components that provide the most amount information, for further labelling.

For *Data Augmentation*, a standard strategy in deep learning that artificially expands the size of the labeled dataset, we introduce a method focused on synthesizing realistic human action videos. This specific algorithm is designed to strengthen the training data for human action recognition tasks.

Lastly, in the *Domain Adaptation* strategy, we target scenarios where a model is trained on a source domain and then deployed on a target domain that has a different distribution. Here, we suggest a novel algorithm to enable object detectors to adapt effectively using available unlabeled data from the target domain.

In the following chapters, we will delve into the details of these proposed methods.

1.1 Contributions

In this dissertation, we present three innovative methods to address the challenge of limited labeled data in different scenarios. The following summaries describe our contributions, with comprehensive discussions provided in the subsequent chapters:

- **Active Learning for Structured Prediction from Partially Labeled Data [101]:** We propose a general purpose active learning algorithm for structured prediction – gathering labeled data for training a model that outputs a set of related labels for an image/video. Active learning starts with a limited initial training set, then iterates querying a user for labels on unlabeled

data and retraining the model. We propose a novel algorithm for selecting data for labeling, choosing examples to maximize expected information gain based on belief propagation inference. This is a general purpose method and can be applied to a variety of tasks/models. As a specific example we demonstrate this framework for learning to recognize human actions and group activities in video sequences. Experiments show that our proposed algorithm outperforms previous active learning methods and can achieve accuracy comparable to fully supervised methods while utilizing significantly less labeled data.

- **Dataset Augmentation for Human Action Classification [102]:** Deep learning's increasing use in computer vision highlights the need for large, diverse datasets. This is especially challenging in human action recognition tasks, which require varied human appearances, actions, and environments. To address this, we've developed a new technique for generating data specifically for action recognition. We start with a small set of human action videos and generate new clips. Each video shows a "human subject" performing a variant of an "action" in a specific "background" or environment. By separately changing the "human subjects", "actions", and "backgrounds", we can generate many more realistic videos. Our method includes a unique approach to generating human skeleton trajectories. When combined with our video frame generation technique, this method allows us to create unlimited training data for action recognition, bypassing the need for expensive and time-consuming data collection. We tested our approach on two small human action recognition datasets and found significant improvements in the performance of existing action recognition methods.
- **A Robust Learning Approach to Domain Adaptive Object Detection [104]:** Domain shift is unavoidable in real-world applications of object detection. For example, in self-driving cars, the target domain consists of unconstrained road environments which cannot all possibly be observed in training data. Similarly, in surveillance applications sufficiently representative training data may be lacking due to privacy regulations. In this work, we address the domain adaptation problem from the perspective of robust learning and show that the problem may be formulated as training with noisy labels. We propose a robust object detection framework that is resilient to noise in bounding box class labels, locations and size annotations. To adapt to the domain shift, the model is trained on the target domain using a set of noisy object bounding boxes that are obtained by a detection model trained only in the source domain. We evaluate the accuracy of our approach in various source/target domain pairs and demonstrate that the model significantly improves the state-of-the-art on multiple domain adaptation scenarios on the SIM10K, Cityscapes and KITTI datasets.

Chapter 2

Related Work

In this chapter we aim to provide an overview of seminal papers addressing the challenge of handling limited labeled data in training deep learning models for computer vision tasks. While comprehensively covering the entire landscape of this critical research area is beyond the scope of this chapter, we aim to present key techniques and approaches that have significantly contributed to addressing this challenge.

2.1 Data Augmentation

Data augmentation can be formally defined mathematically as the process of applying a set of pre-defined transformation operations $t \in \mathcal{T}$ to a given dataset $\mathcal{D} := \{(x, l) \mid \}$ for $x \in \mathcal{X}$ and $l \in \mathcal{L}$, consisting of a set of training samples $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ and labels $\mathcal{L} = \{l_1, l_2, \dots, l_n\}$, where x_i is data and l_i is the corresponding label. The goal of data augmentation is to create additional training data $\mathcal{X}_t = \{t(x_1), t(x_2), \dots, t(x_n)\}, t \in \mathcal{T}$, without altering the original labels $\mathcal{L} = \{l_1, l_2, \dots, l_n\}$. Each transformation operation $t \in \mathcal{T}$ is carefully selected to ensure that the transformed sample $t(x_i)$ remains within the same sample space as the original data and retains the same semantic meaning as described by the original label l_i . Mathematically, $t \in \mathcal{T} : (\mathcal{X}, \mathcal{L}) \rightarrow (\mathcal{X}_t, \mathcal{L})$, where \mathcal{X} is the original dataset, \mathcal{X}' is the augmented dataset and \mathcal{T} is the set of transformation operations. This process is also known as label preserving transformation operations in image processing and computer vision literature.

Image augmentation has become a commonly used approach to enhance the performance of many computer vision algorithms. Well-known image classifier and object detection algorithms have been shown to benefit from basic image transformations like cropping, flipping, and others, resulting in improved robustness and accuracy. Well-known algorithms in computer vision and their corresponding data augmentation techniques are presented in Table 2.1.

2.1.1 Model free techniques

Data augmentation techniques have evolved over time, with researchers constantly exploring novel ways to enhance the performance of deep learning models. Simard et al. [193] focused on simple

Paper	Image augmentation method
AlexNet[113]	Translate, Flip, Intensity Changing, Crop
ResNet[70]	Crop, Flip
DenseNet[79]	Flip, Crop, Translate
MobileNet[78]	Crop, Elastic distortion
NasNet[264]	Cutout, Crop, Flip
ResNeSt[252]	AutoAugment, Mixup, Crop
DeiT[211]	AutoAugmentat, RandAugment, Random Erasing, Mixup, CutMix
Swin Transformer[129]	RandAugment, Mixup, CutMix, Random Erasing
Faster R-CNN[160]	Flip
YOLO[157]	Scale, Translate, Color space
SSD[128]	Crop, Resize, Flip, Color Space, Distortion
YOLOv4[7]	Mosaic, Distortion, Scale, Color space, Crop, Flip, Rotate, Random erase, Cutout, Hide-and-Seek, GridMask, Mixup, CutMix, StyleGAN

Table 2.1: Well-known algorithms in computer vision and their corresponding data augmentation techniques

techniques such as elastic distortion, which simulates the variability in handwriting by randomly distorting original images using local elastic transformations. This technique artificially increases the dataset size, leading to better generalization performance of the trained model on unseen data. Ciregan et al. [19] showed that augmenting dataset simply by randomly translating the image decrease error rates and improves overall performance on the MNIST dataset [29]; these techniques are only suitable for images.

Following these initial advancements, Krizhevsky et al. [113] proposed more sophisticated data augmentation techniques that contributed to the success of their deep learning model, AlexNet. Krizhevsky et al. [113] introduced two critical techniques in their groundbreaking paper "ImageNet Classification with Deep Convolutional Neural Networks", which helped improve the accuracy of their deep learning model, AlexNet. The first technique generates image translations and horizontal reflections by randomly extracting 224x224 pixel crops from the original 256x256 pixel images and flipping them horizontally, resulting in a 2048-fold augmentation. The second technique manipulates the intensities of the RGB channels by performing PCA on the RGB pixel values from the ImageNet [28] training set and adding multiples of the found principal components to the images, helping the model capture variations in lighting and color. These data augmentation techniques, along with dropout regularization, enabled them to train a large neural network that achieved state-of-the-art results on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [168] at the time.

Subsequently, Perez et al. [146] conducted a comprehensive study evaluating various augmentation techniques for image classification tasks. They evaluated several augmentation strategies, including geometric transformations, color jittering, and random erasing, ultimately finding that they all consistently improved the performance of deep learning models, and the best results were

achieved when multiple augmentation techniques were combined. This research opened the door for more innovative approaches to data augmentation, such as MixUp by Zhang et al. [253]. MixUp improves generalization performance by linearly interpolating pairs of training examples and their labels. Concretely, for two examples x_i and x_j with corresponding labels y_i and y_j , MixUp creates a new example x and label y by taking a linear combination of the two original examples and their labels: $x = \lambda x_i + (1 - \lambda)x_j$ and $y = \lambda y_i + (1 - \lambda)y_j$, where $\lambda \sim \text{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \infty)$; The mixup parameter α determines the extent of blending between feature-target pairs. As α approaches 0, it reverts back to the ERM (Empirical Risk Minimization) principle.. The paper shows that MixUp regularization improves the generalization performance of deep neural networks on various tasks, including image classification, object detection, and language modeling. The authors provide several theoretical analyses of MixUp, showing that it encourages the model to interpolate between training examples rather than simply memorizing them. Specifically, they show that MixUp promotes linear behavior in the intermediate layers of the network, and they provide a convexity argument that suggests MixUp can lead to a smoother loss landscape and better generalization. Additionally, they provide an interpretation of MixUp as a form of ensemble learning, where the model is trained on a convex combination of multiple training examples. The authors demonstrated that MixUp acts as a strong regularizer, improving the generalization of deep learning image classification models on several datasets, including CIFAR-10 [110], CIFAR-100 [111], and ImageNet [113] as well as object detection models on Pascal VOC [44] and MS COCO [124]. Furthermore, the authors demonstrate that MixUp can be combined with other regularization techniques, such as dropout and weight decay, to further improve performance. This method has been widely used in the computer vision domain

In parallel, DeVries and Taylor [33] developed another data augmentation technique for enhancing the generalization capabilities of Convolutional Neural Networks (CNNs). The proposed method, known as Cutout, involves randomly masking out contiguous regions within input images during the training process, thus forcing the network to learn more robust features that are invariant to partial occlusions by relying on the remaining unmasked regions. By applying this straightforward yet effective approach, the authors were able to achieve state-of-the-art performance on multiple benchmark datasets, including CIFAR-10, CIFAR-100, and SVHN. The paper provides comprehensive empirical evidence to demonstrate the efficacy of Cutout, showing that it consistently outperforms other regularization techniques like Dropout and data augmentation. Additionally, the authors highlight that Cutout is computationally efficient and can be seamlessly combined with existing regularization methods to further improve performance.

Yun et al. [250] introduced CutMix, a data augmentation technique for image classification tasks that involves cutting and pasting patches from different images and adjusting their labels. This technique further pushed the boundaries of data augmentation for image classification tasks. Specifically, given two input images I_a and I_b with ground-truth labels y_a and y_b , CutMix randomly selects a rectangular patch P_a from I_a and replaces it with a corresponding patch P_b from I_b , as shown in the following equation:

$$I_{\text{mixed}} = \mathcal{M}_{\text{cutmix}}(I_a, I_b) = \begin{cases} I_a(x, y) & \text{if } (x, y) \notin B \\ I_b(x, y) & \text{otherwise} \end{cases} \quad (2.1)$$

where B is the bounding box of the patch P_a , and (x, y) are the pixel coordinates in the image. To compute the ground-truth label y_{mixed} for the mixed image I_{mixed} , the authors use the area ratio λ between the two patches:

$$y_{\text{mixed}} = \lambda y_a + (1 - \lambda) y_b \quad (2.2)$$

where $\lambda = \text{area}(B)/\text{area}(I_a)$. In other words, the ground-truth label for the mixed image is a weighted average of the labels of the two original images, where the weight is determined by the area ratio of the two patches. This label adjustment technique helps ensure that the model learns to recognize objects that may appear partially in the mixed image, which can improve the model’s robustness and generalization performance. Additionally, CutMix encourages the model to learn more robust features by forcing it to focus on multiple objects in the same image and to learn more fine-grained boundary information. The authors show that CutMix provides better regularization and leads to improved performance compared to existing data augmentation methods, such as MixUp and CutOut, on various benchmark datasets, including CIFAR-10, CIFAR-100, and ImageNet. They also demonstrate that CutMix is less sensitive to the choice of hyperparameters than MixUp, making it more practical for use in real-world applications. Figure 2.1 shows performance of CutMix compared to the previous methods.





	ResNet-50	Mixup [48]	Cutout [3]	CutMix
Image				
Label	Dog 1.0	Dog 0.5 Cat 0.5	Dog 1.0	Dog 0.6 Cat 0.4
ImageNet Cls (%)	76.3 (+0.0)	77.4 (+1.1)	77.1 (+0.8)	78.6 (+2.3)
ImageNet Loc (%)	46.3 (+0.0)	45.8 (-0.5)	46.7 (+0.4)	47.3 (+1.0)
Pascal VOC Det (mAP)	75.6 (+0.0)	73.9 (-1.7)	75.1 (-0.5)	76.7 (+1.1)

Figure 2.1: Summary of the results obtained from ImageNet classification, ImageNet localization, and Pascal VOC 07 detection tasks using Mixup, Cutout, and our CutMix techniques. CutMix demonstrates a substantial improvement in the performance of these tasks. [250] © 2019 IEEE.

Expanding on the techniques described earlier, new approaches have been developed to further advance the field and enhance the effectiveness of data augmentation. Hendrycks et al. [72] proposed AugMix, which improves model robustness by applying multiple basic augmentations,

such as rotation and flipping, to an input image and probabilistically blending the results with the original image, generating a diverse set of training samples. The same year, Yin et al. [107] introduced PuzzleMix, which optimizes a mask for fusing two images, capturing salient information and underlying statistics to encourage better feature representations and generalization capabilities. Co-Mixup, proposed by Wang et al. [106], focuses on maximizing the salient signal of input images while maintaining diversity among the augmented images, using a co-regularized training objective to promote a more effective learning process. Subsequently, Luo et al. [24] introduced SuperMix, which utilizes the Newton iterative method to optimize a mask for fusing two images, exploiting salient regions to create richer training samples that improve model performance and generalization. Lastly, Ghiasi et al. [58] proposed Simple Copy-Paste, which randomly pastes object instances into images with large-scale jittering, forcing the model to learn more robust and invariant feature representations for enhanced performance on unseen data on the image segmentation task.

2.1.2 Model based techniques

Model-free data augmentation techniques have limitations that hinder their effectiveness in improving deep learning models. These techniques generate a limited range of plausible alternative data and rely on heuristic approaches, potentially missing important task-specific transformations. Model-based data augmentation techniques are being developed to address these limitations and significantly enhance the generalizability of deep learning models by learning task-specific transformations and discovering novel data augmentations.

Antoniou et al. [3] proposed the Data Augmentation Generative Adversarial Network (DAGAN), based on the assumption that samples from the same class can be transformed into each other using complex transformations. The DAGAN model aims to learn these complex transformations from a set of available data and apply them to a domain with limited labeled data, such as few-shot learning. In contrast to traditional data augmentation techniques, this method generates a wider variety of images while preserving the original data distribution and semantic information. The key contribution of this work is the DAGAN model, which combines the power of GANs with a conditional generation mechanism. The DAGAN consists of a generator network that creates new samples based on input images and a discriminator network that learns to differentiate between pairs of input images, determining if they are both real or if one is real and the other is fake. In essence, the discriminator learns to identify complex transformations between the images. The underlying assumption is that this transformation is valid for labeled data on hand, but not for a combination of a real and a generated image. The generator’s objective is to learn these complex transformations between real data points and apply them to input data, thereby generating new data points. The goal is to produce samples that deceive the discriminator into incorrectly identifying them as both real images. The authors illustrated the generator’s ability to generate diverse and visually plausible image samples from a single novel data point. They then demonstrated that data augmentation using DAGAN could enhance a standard classifier in low-data situations, significantly improving generalization performance on the Omniglot and EMNIST datasets.

Training machine learning models on synthetic images can be more cost-effective and efficient than using annotated real-world images. However, the performance of models trained on synthetic images often suffers due to differences between synthetic and real image characteristics. By enhancing the realism of synthetic images with the help of unlabeled real data, Shrivastava et al. [191] aim to improve the performance of machine learning models trained on large datasets without the need for expensive data collection or human annotation efforts. They developed the Simulated+Unsupervised (S+U) learning method to address the challenge of bridging the gap between synthetic and real image distributions. The developed framework, SimGAN, refines computer-generated images through a refiner network that employs both adversarial and self-regularization losses during training. The authors made key adjustments to the conventional Generative Adversarial Networks (GAN) approach to ensure training stability, maintain annotation details, and avoid artifact production by the refiner network. First, they introduced a self-regularization term to preserve the annotations of synthetic images. By adding this self-regularization loss to the adversarial loss, the method penalizes large changes between synthetic and refined images, ensuring the retention of important details from the original synthetic images. Second, they addressed the issue of training instability and artifact generation in traditional GANs, which use a discriminator that operates on the entire image. By limiting the discriminator’s receptive field to local regions instead of the whole image, the authors created multiple local adversarial losses per image, resulting in a more stable training process and fewer artifacts. Finally, they tackled the standard GAN training process’s instability due to the competing goals of the generator and discriminator networks. The authors proposed updating the discriminator using a history of refined images, rather than only those from the current refiner network, further enhancing training stability.

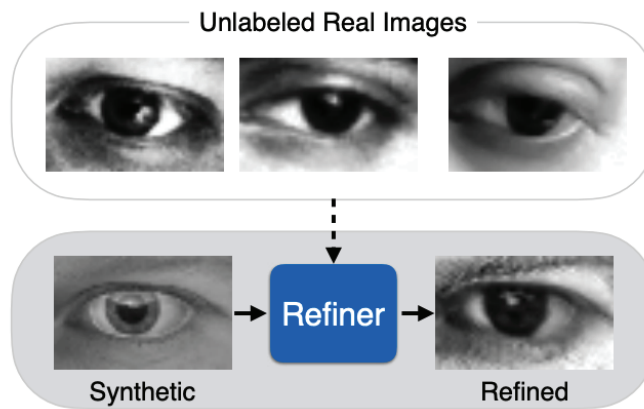


Figure 2.2: The model enhances the realism of synthetic images generated by a simulator by leveraging unlabeled data, while also maintaining the annotation information [191]. © 2017 IEEE

GANs have been used in the medical image domain, where labelled data is scarce. Bowles et al. [8] investigate the use of Generative Adversarial Networks (GANs) for augmenting training data in medical imaging applications. The authors focus on brain segmentation tasks and demonstrate that GAN-generated synthetic data can improve the Dice Similarity Coefficient (DSC) by 1 to 5

percentage points. The improvement is most significant when fewer than ten training image stacks are available. The study explores various aspects of GAN augmentation, such as the choice of segmentation network, comparison to rotation augmentation, the impact of synthetic data quantity, and the influence of available real data. The results suggest that GANs can benefit medical image segmentation tasks, and the method is transferable to multiple datasets. Frid-Adar et al. [51] proposed to use an augmentation scheme that combines standard image perturbation and synthetic liver lesion generation using Deep Convolutional GAN (DCGAN). They were able to generate realistic images of liver lesion and used for classification task. They demonstrated that using synthesized images significantly improved the classification performance.

In real-world datasets, it is not uncommon for some classes to have significantly fewer examples than others, resulting in a skewed distribution. This imbalance can adversely affect the performance of machine learning models, as they may develop a bias towards the majority class. To address this issue, Douzas et al. [37] introduce a novel method using Conditional Generative Adversarial Networks (CGANs), a type of generative model that generates synthetic samples based on class labels. By producing additional samples for under-represented classes, CGANs effectively balance the class distribution. The authors train a CGAN to learn the underlying data distribution and create realistic samples for minority classes, which are then used to augment the original dataset. This augmentation not only reduces the imbalance but also enhances the performance of various machine learning models. The experiments conducted on different imbalanced datasets and classification algorithms show that the proposed CGAN-based method outperforms existing techniques, such as the Synthetic Minority Over-sampling Technique (SMOTE). Overall, the CGAN approach excels in generating high-quality synthetic samples, leading to improved classification performance across diverse machine learning models.

2.1.3 Policy based techniques

The algorithms discussed in the previous two sections involve specific schemes that require domain knowledge to achieve better performance.. Typically, these methods use tailored operations with suitable magnitudes for image augmentation, catering to the specific characteristics of each dataset. However, the process of hyperparameter optimization can be both complex and time-intensive. An alternative approach is to create algorithms capable of determining the best augmentation strategies automatically. Such algorithms, referred to as policy-based optimization, can be classified into two groups: those based on reinforcement learning and those based on adversarial learning. Reinforcement learning-based methods identify the optimal strategy through reinforcement learning, while adversarial learning-based techniques select augmentation operations and magnitudes that lead to a high training loss but a low validation loss.

Cubuk et al. [22] introduced a seminal method called AutoAugment to automatically learn data augmentation policies by leveraging the training data itself In AutoAugment, a reinforcement learning-based search algorithm is employed to find the best augmentation policies. These policies consist of sequences of basic image operations, such as rotation, translation, shearing, and color

transformations, with specific probabilities and magnitudes. The search algorithm utilizes a recurrent neural network (RNN) controller to generate augmentation policies, which are then applied to the training dataset to create augmented images. These images are used to train "child" models for a limited number of epochs, and the resulting validation loss is used as feedback to update the controller using Proximal Policy Optimization (PPO). The process is iteratively performed, refining the controller's ability to propose better augmentation policies. The key idea behind AutoAugment is to use the data to learn the best augmentation policy rather than relying on human intuition or expertise. The authors demonstrate that AutoAugment leads to significant improvements in performance on various image classification tasks and datasets. The optimal augmentation policy discovered by AutoAugment for each dataset is different, which highlights the versatility of the approach. By employing a reinforcement learning-based search algorithm, AutoAugment can effectively explore various combinations and magnitudes of image operations, ultimately finding an optimal policy tailored to the specific dataset. This adaptability contributes to the improved performance of deep learning models across a range of image classification tasks, such as CIFAR-10, CIFAR-100, SVHN, and ImageNet.

Despite achieving favorable classification results on multiple datasets, AutoAugment's extensive training time is a significant drawback. Consequently, several research studies have investigated different approaches to address this problem. Fast AutoAugment [122] and Faster AutoAugment [68] present more efficient alternatives to the original AutoAugment method for discovering data augmentation policies. Fast AutoAugment uses Bayesian optimization on smaller proxy tasks to reduce search time, while Faster AutoAugment introduces Differentiable Augmentation Policy Search (DAPS), which uses backpropagation to learn policies in an end-to-end manner, eliminating the need for proxy tasks and additional optimization techniques. Both methods significantly reduce computational cost and search time compared to the original AutoAugment, while achieving competitive performance on benchmark datasets, demonstrating their effectiveness in enhancing the generalization of deep learning models.

In many cases, simpler and more efficient methods can lead to better results. One such example is the RandAugment technique proposed by Cubuk et al. [23]. This approach streamlines the data augmentation process by eliminating the need for a proxy task and reducing the search space to just two hyperparameters: the number of augmentation operations (N) and the magnitude of the applied augmentations (M). In RandAugment, the optimal operations and magnitudes are not explicitly searched for. Rather, during training, N augmentation operations are randomly selected from a fixed set of transformations (such as rotation, translation, and shearing) and applied to the input images. The magnitude of each transformation is determined by the M parameter, which is set by the user. RandAugment relies on the assumption that, by randomly selecting and composing various augmentation operations with the appropriate magnitude, the model will learn to generalize better. In practice, users might perform a coarse grid search or use validation performance to find suitable values for N and M . However, the search space for these two hyperparameters is much smaller and less computationally expensive compared to the search process in AutoAugment. Despite its simplicity,

RandAugment has been shown to perform on par with or outperform more complex augmentation methods, including AutoAugment, on various benchmark datasets such as CIFAR-10, CIFAR-100, and ImageNet. Its practicality and accessibility make RandAugment a valuable technique for a wide range of applications and users in the field of deep learning.

The main goal of image augmentation is to help train a model using a training dataset so it can perform well on a testing dataset. It is assumed that more challenging samples are beneficial, and those input images resulting in a higher training loss are regarded as such difficult samples. Adversarial learning-based image augmentation strives to develop a policy for generating these challenging samples by building upon the original training examples.

Xie et al. [240] propose a method called Adversarial Propagation (AdvProp) to improve the performance and robustness of image recognition models. AdvProp generates adversarial examples using a variant of the Projected Gradient Descent (PGD) method and incorporates them into the training process alongside clean examples. Adversarial examples are inputs to machine learning models that are intentionally designed to cause the model to make incorrect predictions. They are crafted by introducing small, imperceptible perturbations to input data, such as images, that lead the model to make incorrect classifications. Training on adversarial examples naturally degrades the performance due to the distribution discrepancy between clean and adversarial examples. To address this, the authors introduce two separate batch normalization layers, one for each type of example. This approach enables the model to better adapt to the different distributions and learn more robust features. The evaluation of AdvProp on benchmark datasets, including ImageNet, demonstrates that models trained with this method achieve better performance compared to models trained without adversarial examples or with traditional adversarial training methods.

In summary, the various data augmentation techniques discussed above, including model-free, model-based, and policy-based approaches, demonstrate the effectiveness of generating new training samples while preserving the original semantic information. These methods encompass a wide range of techniques. However, there are certain limitations associated with these methods. For instance, many of these techniques require domain knowledge and involve assumptions, such as selecting reasonable ranges for cropping, mixing, and applying transformations, which might not be universally applicable to all tasks or datasets. Furthermore, the optimization of hyperparameters can be complex and time-consuming, potentially limiting the adoption of these techniques in certain applications. Some data augmentation techniques have limitations due to their varying effectiveness across different domains and inability to remove biases in datasets. For instance, while vertical flipping works well in scene recognition, it may change labels in digit recognition (digit 9 vs 6). Additionally, in cases like waterbirds versus landbirds recognition, where background differences dominate, the model may learn to identify the background instead of the bird species, highlighting that data augmentation alone may not be sufficient to mitigate inherent biases. Despite these drawbacks, data augmentation has proven to be a powerful tool for enhancing the generalization capabilities and robustness of deep learning models across various domains and tasks.

2.2 Transfer Learning

Transfer learning is a powerful machine learning technique that leverages the knowledge gained from one domain or task, referred to as the source domain, to improve the performance of a model on a different but related domain or task, known as the target domain. The main idea behind transfer learning is to take advantage of the similarities between tasks or domains, enabling the model to learn from one context and apply that knowledge to another. This approach can significantly reduce training time, computational resources, and the need for large amounts of labeled data in the target domain.

Transfer learning can be broadly divided into two main groups based on the availability of labeled data in the target domain: inductive transfer learning and transductive transfer learning. In inductive transfer learning, labeled data are available in the target domain. This setting allows the model to be fine-tuned on the target task using the available labeled data, adapting the pre-trained model to better perform on the new task. On the other hand, transductive transfer learning deals with situations where labeled data are only available in the source domain. In this case, the challenge lies in adapting the model to the target domain while minimizing the distribution differences between the source and target domains, even in the absence of labeled data in the target domain. Both groups provide different strategies to address the challenges of transfer learning depending on the data available in the source and target domains.

2.2.1 Inductive Transfer Learning

Inductive transfer learning is a setting where labeled data are available in the target domain. This approach allows models to leverage the knowledge gained from the source domain while fine-tuning the model using labeled data from the target domain. Inductive transfer learning is particularly useful when the target task is similar to the source task, but the labeled data in the target domain is limited.

Inductive transfer learning can be further divided into two subcategories. Multi-task learning and Sequential transfer learning which we discuss in the following two sub-sections.

Sequential transfer learning

In sequential transfer learning, the model is first trained on a source task and then fine-tuned on the target task using the labeled data available in the target domain. This fine-tuning process can involve adjusting the entire network or just the final layers, depending on the degree of similarity between the source and target tasks. This approach has been successfully applied to various domains, including computer vision and natural language processing. One of the early and influential applications of sequential transfer learning in computer vision is the R-CNN method, which demonstrates the power of this approach in object detection tasks.

Li-Jia et al. [117] propose ObjectBank method, an early attempt at transfer learning in the field of computer vision. The ObjectBank method leverages pre-trained object detectors to generate semantically-rich image representations, which can be applied to various visual recognition tasks,

such as object recognition, scene classification, and event detection. By using a set of pre-trained detectors to extract high-level features from images, the Object Bank approach demonstrates the potential of transfer learning, where knowledge learned in one task is reused and adapted to improve performance on related tasks. While not as sophisticated as more recent deep learning-based transfer learning techniques, the Object Bank method was an important early work in the area of high-level visual representations and transfer learning.

Girshick et al. [63] introduced a groundbreaking work in the field of computer vision called R-CNN, which revolutionized object detection tasks by integrating region proposals with convolutional neural networks (CNNs) and yielding significantly improved performance. The R-CNN approach leverages transfer learning by pretraining a deep CNN on a large-scale dataset like ImageNet and then fine-tuning it on a smaller, target-specific dataset. This fine-tuning process updates the pretrained network's weights using backpropagation, aligning them with the target task and adapting the features learned from ImageNet to the target dataset's object classes, resulting in improved performance. The R-CNN method consists of three main components: region proposals, feature extraction, and classification. Selective Search generates candidate regions, reducing the search space compared to a sliding-window approach. Fixed-length feature vectors are extracted from each region proposal by warping it to a fixed size and passing it through the pretrained CNN, capturing high-level information from ImageNet. Finally, class-specific linear SVMs classify these feature vectors, while bounding box regression refines the objects' spatial locations, creating a cohesive object detection pipeline.

Building on the idea of repurposing features learned from large-scale datasets, Donahue et al. [34] explored the potential of such features for various novel tasks, in their seminal work DeCAF. They investigate the potential of repurposing features extracted from a deep convolutional network, trained on a large dataset with fixed set of object recognition tasks, for novel generic tasks that differ significantly from the original tasks and have limited labeled or unlabeled data. The authors analyze the semantic clustering of deep convolutional features for various tasks (Figure 2.3), such as scene recognition, domain adaptation, and fine-grained recognition challenges, and compare the efficacy of using different network levels to define a fixed feature. The clustering results demonstrate the generalizability and implicit semantic understanding of these features, indicating that they tend to group images into intriguing semantic categories, despite the fact that the network was not specifically trained for them. Additionally, they demonstrate that their multi-task deep learning framework, DeCAF, is able to learn powerful and generalizable features for semantic visual discrimination tasks. They demonstrate that the semantic understanding embedded in DeCAF features, enables simple linear classifiers to outperform state-of-the-art methods based on complex multi-kernel learning techniques with traditional hand-engineered features, across various visual recognition tasks.

The transferability of features in deep neural networks has also been investigated by Yosinski et al. [246], who sought to understand how these features generalize across different tasks and datasets. They discovered that first-layer features tend to be general and applicable to various datasets and tasks. However, as features progress through the layers, they become more task-specific and less

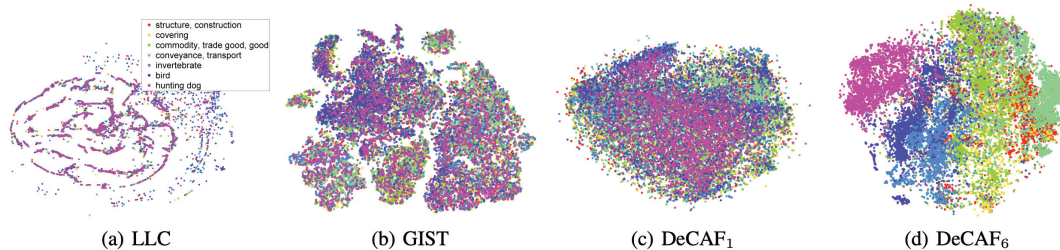


Figure 2.3: Several t-SNE visualizations of ImageNet validation set using different features [34].

transferable. The authors identify two factors that negatively impact transferability: 1) the specialization of higher layer neurons to their original task, and 2) optimization difficulties arising from splitting networks between co-adapted neurons. Higher layer neurons' specialization to their original task implies that they are less likely to be useful for other tasks, thus limiting the transferability of features learned in deep neural networks. The second factor, optimization difficulties, refers to the phenomenon where neurons in a deep neural network become co-adapted, working together to recognize specific features in the original dataset. When applied to a new dataset, these co-adapted neurons may not perform as effectively, leading to decreased performance. This unexpected finding underscores the importance of understanding feature learning and representation in deep neural networks. Additionally, the authors reveal that initializing a deep neural network with transferred features from a pre-trained network can enhance the network's performance on a new dataset, even if the pre-trained network was trained on an entirely different dataset. This result is significant, demonstrating the effectiveness of transfer learning and the potential for pre-training deep neural networks on large, general-purpose datasets to boost performance on specific tasks.

Further reinforcing the effectiveness of pretraining as a form of transfer learning, Razavian et al. [190] showcased its applicability to an even wider array of tasks. They demonstrated that a simple pipeline, which employs a pre-trained CNN as a feature extractor followed by a linear classifier, can achieve remarkable results across various recognition tasks, such as object and scene recognition, fine-grained recognition, attribute detection, and image retrieval. Their study indicates that this straightforward approach can outperform or match the performance of more complex state-of-the-art methods in numerous instances.

Multi-task learning

In multi-task learning, a model is trained to perform multiple tasks simultaneously, utilizing shared representations across tasks. This approach enables the model to achieve better performance and generalization on each individual task. This approach is particularly useful when the tasks are related, as the model can leverage the shared characteristics between them to enhance its performance. A common form of multi-task learning involves classification tasks, where a neural network extracts features from an input image and applies softmax to the final layer to predict the class to which the image belongs. This method has become a standard approach in deep learning, as it is more efficient

than training binary classifiers for each class individually. In this case, all tasks share a common characteristic, which is the presence of a label in the input image. However, this approach can also be extended to dissimilar tasks, such as segmentation and object detection.

One of the seminal works in multi-task learning was introduced by Caruana [12]. The author presented the concept of multi-task learning, emphasizing that concurrently learning multiple related tasks can lead to improved performance compared to learning them separately. The paper outlines a framework for multi-task learning and examines various applications. Caruana investigates methods for knowledge sharing between tasks, such as feature sharing, and using outputs as inputs and vice versa in neural networks. The author provides empirical evidence from diverse domains, illustrating the effectiveness of multi-task learning in enhancing generalization.

Kendall et al. [99] address the challenge of finding the optimal weighting between different task losses in multi-task learning. They demonstrate that performance heavily relies on the appropriate choice of these weightings, which is difficult to determine through manual tuning. The authors propose a novel method that leverages homoscedastic uncertainty to combine multiple loss functions and learn multiple objectives simultaneously. They model the loss functions for each task as a Gaussian distribution with a constant variance. These variances represent the homoscedastic uncertainty or task-dependent weighting for each task. The main idea is to learn the log of these variances $\log \sigma^2$ along with the model’s parameters during the training process. By optimizing these log variances, the model can automatically adjust the weights for each task’s loss function. The total multi-task loss function is a weighted sum of the individual task losses, where the weights are derived from the learned homoscedastic uncertainties. Specifically, the weight for each task is the inverse of its learned variance, which results in the following multi-task loss function:

$$Loss = \sum_i \left(\frac{1}{2\sigma_i^2} \right) L_i + \log(\sigma_i^2) \quad (2.3)$$

In this formula, L_i represents the loss for task i , and σ_i^2 is the learned variance (homoscedastic uncertainty) for that task. By minimizing this combined loss function, the model automatically learns the optimal weightings for each task, effectively balancing the different task losses during training. The authors showcase their approach in learning scene geometry and semantics using three tasks, illustrating its effectiveness in the context of complex multi-task learning scenarios.

Misra et al. [138] present strong evidence supporting multi-task learning’s effectiveness and propose an innovative approach to achieve it. Building on previous work that shared a subset of initial network layers to extract features for multiple tasks, they acknowledged the challenge of identifying the optimal subset. The authors conducted experiments using various network architectures in order to determine the optimal level of shared representation for task pairs. They achieved this by splitting a ConvNet at different layers and measuring the performance on each task for each resulting architecture. Their goal was to identify the most effective architecture that allowed for the greatest transfer of knowledge between tasks, while maintaining high performance on both tasks. Their discovery highlights that multi-task learning yields superior results compared to single-task learning,

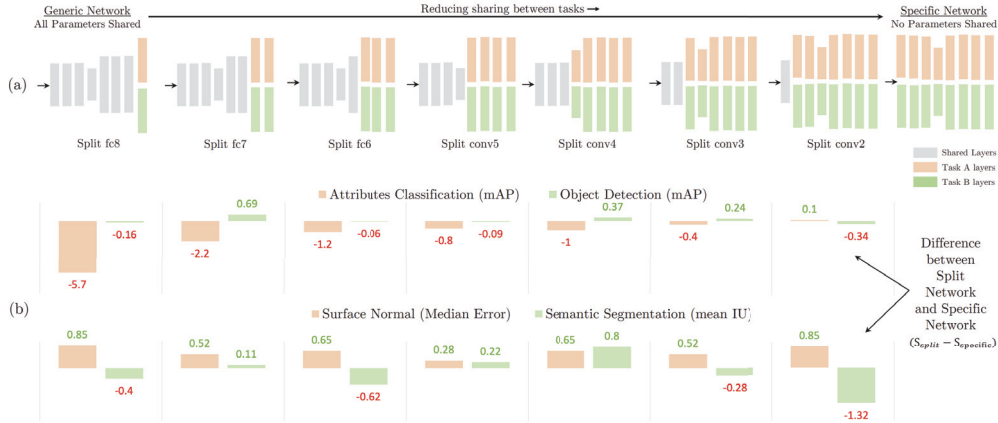


Figure 2.4: Mirsra et al. [138] experimented with several multi-task (two-task) architectures by dividing the ConvNet [113] at various layers for two sets of tasks. They compared the performance of these networks on each task to that of task-specific networks. They observed that the most effective multi-task architecture is dependent on the specific tasks involved and does not necessarily generalize well to different pairs of tasks. @ 2016 IEEE

and the most effective architectural split is contingent on the specific task pairings, which varies for each pair of tasks. However, identifying the optimal architecture for each task pair is cumbersome and impractical, as it requires enumerating all possible splits. To address this challenge, the authors introduced cross-stitch units, enabling a single network to capture various "Split" architectures and more. These units model shared representations by learning a linear combination of input activation maps from both tasks at each layer, with subsequent layers operating on this shared representation. Cross-stitch networks automatically learn the optimal combination of shared and task-specific representations, eliminating the need for brute-force enumeration and search, ultimately resulting in better performance than those obtained through exhaustive exploration.

Multi-task learning became increasingly popular in the following year. This led to Sebastian Ruder [166] publishing a survey that examines different MTL methods and offers valuable insights into the intuition behind MTL's success. Ruder highlights implicit data augmentation, attention focusing, and regularization as some of the reasons for MTL's effectiveness.

Implicit data augmentation is a crucial factor in effectiveness of MTL, as it effectively increases the sample size utilized for training the model. Since all tasks exhibit some degree of noise, training a model on a single task (task A, for example) aims to learn a good representation for that specific task, ideally disregarding data-dependent noise and generalizing well. However, different tasks have varying noise patterns, so a model trained on two tasks simultaneously can learn a more general representation. This prevents the risk of overfitting to task A, as learning tasks A and B jointly allows the model to obtain a superior representation (F) by averaging the noise patterns.

Attention focusing is another essential aspect of MTL's efficacy, particularly when a task is highly noisy, or the data is limited and high-dimensional. In such cases, it becomes challenging for a model to differentiate between relevant and irrelevant features. MTL aids the model in con-

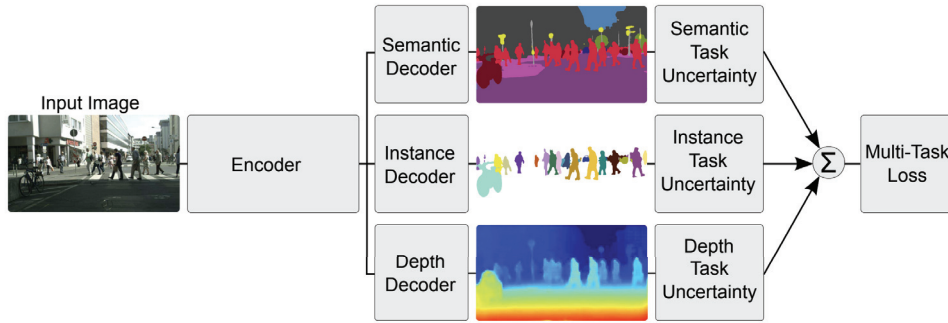


Figure 2.5: Kendall et al. [99] developed a method to combine multiple regression and classification loss functions for multi-task learning. Their architecture accepts a single monocular RGB image as input and generates pixel-wise classification, instance semantic segmentation, and an estimation of per-pixel depth. Multi-task learning can enhance accuracy compared to individually trained models, as cues from one task (e.g., depth) help regularize and boost the generalization performance in another domain (e.g., segmentation). @ 2018 IEEE

centrating on crucial features, as other tasks supply supplementary evidence for the significance or insignificance of those features.

Regularizer is the final key component of MTL's success. By introducing an inductive bias, MTL acts as a regularizer that helps to mitigate both overfitting and the model's capacity to fit random noise, which is known as the Rademacher complexity.

Building upon the ideas from previous methods to address network architecture and loss weight balancing challenges, Liu et al. [127] proposed a novel multi-task learning architecture that employs task-specific feature-level attention, called the Multi-Task Attention Network (MTAN). This architecture comprises a single shared network with a global feature pool and a soft-attention module for each task. These modules enable the learning of task-specific features from the global features while allowing features to be shared across different tasks. The MTAN can be built on any feed-forward neural network and is designed to address two main challenges in multi-task learning: Network Architecture (weight sharing mechanism) and Loss Function (loss weight balancing mechanism). The MTAN architecture encourages the learning of both task-shared and task-specific features, enabling the network to learn a generalizable representation while avoiding overfitting or underfitting. Additionally, the MTAN inherently balances the contributions of each task in the multi-task loss function, making it less sensitive to different weighting schemes. Authors demonstrate that the MTAN architecture achieves state-of-the-art performance in multi-task learning compared to existing methods and is less sensitive to various weighting schemes in the multi-task loss function. The architecture is evaluated on a variety of datasets and tasks, including image-to-image predictions and image classification tasks, showcasing its effectiveness and adaptability.

In recent years, numerous papers have proposed innovative methods to address the challenges in multi-task learning, Notable examples include, but are not limited to, HRNet [202] and Grad-

Norm [16], which present distinctive solutions aimed at enhancing the effectiveness of previous approaches. HRNet maintains high-resolution representations throughout the network for tasks like semantic segmentation and human pose estimation, preserving fine-grained details by connecting high-resolution feature maps in parallel with the low-resolution ones, thereby improving the performance of tasks requiring precise pixel-level predictions. On the other hand, GradNorm tackles the challenge of loss balancing by dynamically adapting loss weights during training. It calculates the gradient norms for each task and adjusts the task-specific loss weights to equalize the gradient norms across tasks or follow a target ratio. This adaptive loss balancing prevents tasks with larger gradients from dominating the learning process, enabling the network to learn multiple tasks more effectively.

2.2.2 Transductive Transfer Learning

Transductive transfer learning addresses the situation where labeled data are only available in the source domain, and the target domain has only unlabeled data, for example when obtaining labeled data for the target domain is expensive, time-consuming, or simply not feasible. The challenge lies in bridging the gap between the source and target domains without the benefit of labeled data in the target domain.

Domain adaptation

Domain adaptation algorithms generally fall into two categories: feature-based and instance-based. Feature-based approaches, which are more popular, focus on using training samples from both source and target domains to extract shared features or minimize the feature distribution distance. This typically involves calculating the distance between low-level features, selecting or re-weighting them, and then learning high-level feature combinations. Some feature-based methods also convert features into a new space to reduce distance between them. These methods perform well when source and target domains have strong connections, but can struggle when there is limited similarity between data samples. On the other hand, instance-based algorithms select similar instances from different domains to facilitate quality knowledge transfer. These methods often combine instance re-weighting and distribution distance metrics.

Learning from Instances. Essentially, instance-based learning algorithms involve transferring knowledge from the source domain to the target domain through reweighting or resampling source instances. These methods rely on two key assumptions: 1) a certain number of training instances in the source domain are relevant to the target domain, enabling their reuse, and 2) the conditional distributions of the source and target domains are identical. It is crucial to selectively choose samples that will be beneficial for the target domain task since not all source data can be reused for training the target model. Wang et al.'s [232] proposed four rules for designing instance-based domain adaptation: 1) Minimize the weighted empirical loss across source and target domains, 2) Allocate balanced weights to data points to prevent over-fitting caused by perturbations in the training data,

3) Assign more weight to the target sample since it will be used for testing, and 4) Distribute weights such that the performance gap between the domains is minimized.

Learning from Features

Nevertheless, the conditions required for instance-based algorithms do not consistently apply to many real-world problems [201, 244]. As a result, feature-based methods have been developed as an alternative.

Ghifary et al. [59] introduced the concept of an adaptation layer with their Domain Adaptive Neural Network (DaNN), a modified feedforward neural network. The loss function consists of two components: general loss and Maximum Mean Discrepancy (MMD) regularizer. The MMD measures the dissimilarity between two probability distributions based on their respective samples. The goal is to minimize the distribution mismatch between the source and target feature distributions in the latent space domain. The MMD loss can be calculated as follows:

$$\text{MMD}^2(P, Q) = |\mathbb{E}_x[\Phi(x)] - \mathbb{E}_y[\Phi(y)]|^2 \quad (2.4)$$

here P and Q are the source and target domain distributions, x and y are samples from these distributions, $\Phi(x)$ and $\Phi(y)$ are the mappings of x and y into the latent space, and \mathbb{E}_x and \mathbb{E}_y denote the expectations over the source and target domain samples, respectively. The MMD loss quantifies the distance between the means of the mapped samples in the latent space. While the DaNN outperformed similar models, its performance was limited due to its simplicity and shallow structure. To address this issue, Tzeng et al. [215] proposed a method, called Deep Domain Confusion (DDR) to improve domain adaptation in deep learning by focusing on learning domain-invariant features. To achieve this, they introduce a loss function that ensures the features extracted from both source and target domains have similar distributions, making them indistinguishable. This is accomplished by using the Maximum Mean Discrepancy (MMD) metric as a measure of divergence between the domains. The DDC method is integrated into the deep neural network architecture as a layer, allowing for simultaneous training with the main task on the source domain. This joint training process guides the model to learn domain-invariant features that are useful for both source and target domains. Additionally, they show that the evaluation metric could be used to determine the position and dimensionality of the adaptation layer.

Ganin et al. [53] proposed a similar approach for learning domain invariant features but with a different mechanism. They propose a method for unsupervised domain adaptation models using a gradient reversal layer (GRL) and a domain-adversarial training strategy. The model consists of a feature extractor, responsible for learning domain-invariant features, and a domain classifier, which attempts to distinguish between the source and target domains. The adversarial loss, incurred by the domain classifier, is minimized, encouraging the feature extractor to generate features that confuse the domain classifier. The GRL is inserted between the feature extractor and the domain classifier, reversing the gradient during backpropagation, effectively maximizing the domain classification error. In other words, while the domain classifier parameters are optimized to distinguish between

source and target domains, the reversed gradients are passed to the feature extractor to induce the opposite effect, producing features that confuse the classifier. By optimizing these competing objectives, the model learns domain-invariant features that are useful for the primary task while being less sensitive to differences between the source and target domains, resulting in improved generalization on the target domain data.

Drawing on the concept of Maximum Mean Discrepancy (MMD) as a metric for gauging differences between source and target features, Long et al. [130]. developed the Deep Adaptation Network (DAN) framework for domain adaptation. This method utilizes a multi-kernel MMD loss to minimize distribution discrepancies between source and target domains across multiple layers in a deep neural network. By freezing the initial layers that learn general features and fine-tuning the mid-layers, the last few layers in the network effectively benefit from this approach.

Another approach for training a model while simultaneously learning domain-invariant features is Deep CORAL (CORrelation ALignment), proposed by Sun et al. [201]. This method focuses on aligning the second-order statistics, specifically the covariance matrices, of the source and target domains within the feature space of a deep neural network. Deep CORAL works to minimize the distance between the source and target covariance matrices while maintaining the discriminative power of the deep features. This is accomplished by incorporating a CORAL loss term into the standard classification loss. This term measures the difference between the second-order statistics (covariance matrices) of the source and target domain feature distributions. The primary objective of the CORAL loss is to minimize this difference, which in turn helps align the source and target domain distributions. As a result, the performance in domain adaptation tasks is improved. The CORAL loss function can be calculated as follows:

$$CORAL_{loss} = \frac{1}{4d^2} \|C_s - C_t\|_F^2 \quad (2.5)$$

Where C_s and C_t are the covariance matrices of the source and target domain features, respectively. d is the dimensionality of the feature representations. $\|\cdot\|_F$ denotes the Frobenius norm, which is used to calculate the distance between the two covariance matrices.

Alternatively, multi-kernel MMD is employed to reduce the distribution mismatch between the source and target domains, with multiple adaptation layers utilized to enhance performance. A prime example of multi-kernel MMD-based architectures is the Domain Adaptive Hash (DAH) network in Venkateswara et al.'s [224] "Deep hashing network for unsupervised domain adaptation." This research, the first of its kind, leverages the feature learning capabilities of neural networks to learn representative hash codes for addressing domain adaptation problems. Notably, hashing techniques can convert high-dimensional data into binary codes, simplifying access and storage.

Tzeng et al. [214] collected all the ideas from previous methods and proposed a unified framework for unsupervised domain adaptation. They proposed that each method has three parts. The first part is the base model, whether it is discriminative [53, 59] or generative [126]. The second part is source and target mappings, whether they are tied [53, 59] or untied [126]. And the last part is

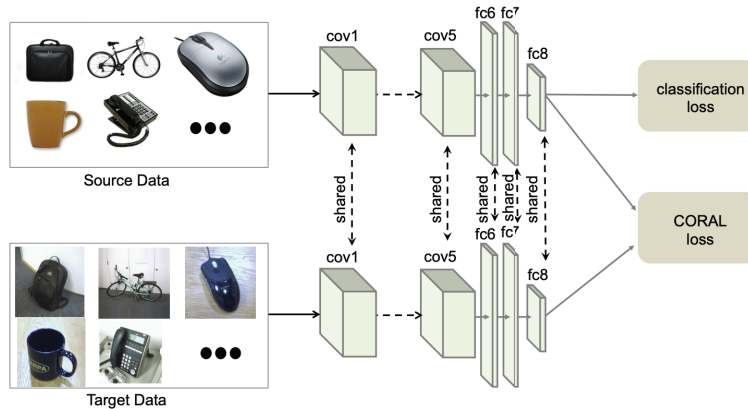


Figure 2.6: To enhance domain adaptation performance, Deep CORAL aligns the covariance matrices of the source and target domains in a deep neural network’s feature space. This is achieved by minimizing the CORAL loss term in the classification loss, which reduces the difference between the correlation matrices of the source and target feature domains[201].

the adversarial loss that makes sure the source and target mappings distributions match, examples are minmax [53], confusion [59], and GAN [126]. In this unified framework they proposed a novel method that uses a discriminative model, with unshared network weights and a GAN based adversarial loss. The framework comprises a source encoder (trained on source domain data), a target encoder (sharing the same architecture as the source encoder but initially untrained), and a domain discriminator (trained to distinguish between features from the source and target encoders). The training process involves iteratively training the source encoder with supervised learning, training the domain discriminator to recognize the features from the source and target encoders, and updating the target encoder to generate features that confuse the domain discriminator. This adversarial training process aims to align the features learned by the target encoder with those of the source encoder, resulting in features that are both discriminative for the classification task and domain-invariant across the source and target domains. at test time, the model uses the target encoder and the source classifier to predict the labels for the target domain.

Tzeng et al. [214] consolidated ideas from previous methods and proposed a unified framework for unsupervised domain adaptation, consisting of three key components. The first component is the base model, which can be either discriminative [53, 59] or generative [126]. The second component involves source and target mappings, which can be either tied [53, 59] or untied [126]. The third component is the adversarial loss that ensures the distributions of the source and target mappings match, such as min-max [53], confusion [59], or GAN-based [126] losses. Within this unified framework, the authors introduced a novel method that employs a discriminative model with unshared network weights and a GAN-based adversarial loss. The method includes a source encoder (trained on source domain data), a target encoder (sharing the same architecture as the source encoder but initially untrained), and a domain discriminator (trained to distinguish between features

from the source and target encoders). The training process iteratively updates the source encoder with supervised learning, trains the domain discriminator to recognize features from the source and target encoders, and refines the target encoder to generate features that confuse the domain discriminator. This adversarial training process aims to align features learned by the target encoder with those of the source encoder, producing features that are both discriminative for the classification task and domain-invariant across the source and target domains. During testing, the model utilizes the target encoder and the source classifier to predict labels for the target domain

2.3 Active Learning

In this section, we explore the seminal works of active learning, a machine learning technique that leverages an oracle to efficiently select and acquire labeled data that can enhance the model’s performance. By actively selecting the most informative data points, active learning reduces the need for massive amounts of labeled data and improves model accuracy. Consider a supervised learning problem with a labeled data pool \mathcal{L} and an unlabeled data pool \mathcal{U} . The goal of active learning is to iteratively select instances from \mathcal{U} to be labeled and added to \mathcal{L} , maximizing the learning algorithm’s performance.

Given a hypothesis space \mathcal{H} , a learning algorithm \mathcal{A} , and a selection strategy \mathcal{S} , the active learning process can be formulated as follows:

1. Train an initial model $h \in \mathcal{H}$ using the labeled data \mathcal{L} .
2. Estimate the informativeness of each instance $x \in \mathcal{U}$ using the selection strategy \mathcal{S} and the current model h .
3. Select the instance \hat{x}^* from \mathcal{U} with the highest informativeness score, i.e.

$$\hat{x}^* = \arg \max_{x \in \mathcal{U}} \mathcal{S}(x, h).$$
4. Obtain the label \hat{y} for the instance \hat{x}^* from an oracle, and update the labeled data as $\mathcal{L} = \mathcal{L} \cup \{(\hat{x}^*, \hat{y})\}$ and the unlabeled data as $\mathcal{U} = \mathcal{U} \setminus \{\hat{x}^*\}$.
5. Update the model h by retraining it with the updated labeled data \mathcal{L} .
6. Repeat steps 2-5 until a stopping criterion is met, such as a predefined budget for labeling instances or model performance convergence.

Active learning methods for classical machine learning typically involve sampling data instances one by one, which is suitable for traditional models such as support vector machines, decision trees, and logistic regression. Settles [184] has written a comprehensive survey on these pre-deep learning active learning techniques. However, this approach is not efficient for deep learning models, as they often require large-scale datasets and benefit from batch processing to optimize their training. Consequently, active learning strategies for deep learning (DeepAL) have been adapted to account

for these differences, focusing on batch sampling and other techniques that cater to the unique requirements of deep neural networks. In this section, we will primarily focus on active learning methods tailored for deep learning.

The batch-based query strategy forms the basis of DeepAL. A simplistic approach might involve continuously querying a batch of samples using a one-by-one strategy. For instance, Janz et al. [87] employs Bayesian Active Learning by Disagreement (BALD) to score the unlabeled samples and then select the top-k samples with the highest scores. BALD’s acquisition function is designed to measure the expected information gain from observing a new data point’s label, by calculating the difference in entropy between the expected posterior distribution and the expected entropy of the posterior distribution after label observation. This method is effective in identifying data points that are highly informative and can provide useful insights into the model parameters. However, this approach tends to select similar samples, which may lead to redundant information and limit the model’s learning potential. Additionally, the method treats each sample independently, without considering their correlation, which can result in suboptimal batch selections. To address these issues, Kirsch et al. [109] improves BALD by estimating the joint mutual information between multiple data points and model parameters, thereby considering the correlation between different query samples simultaneously. The main objective of BatchBALD is to maximize the mutual information between a batch of points \mathcal{B} and the model parameters θ . The acquisition function for a batch of $x_{1:b}$ with predicted labels $y_{1:b}$ is given by the mutual information between $y_{1:b}$ and θ , which can be written as:

$$\mathbb{I}(y_{1:b}, \theta | x_{1:b}, \mathcal{D}) = \mathbb{H}(y_{1:b} | x_{1:b}, \mathcal{D}) - \mathbb{E}_{p(\theta | \mathcal{D})}[\mathbb{H}(y_{1:b} | x_{1:b}, \mathcal{D}, \theta)] \quad (2.6)$$

The equation 2.6 shows that for high mutual information, the left term (model prediction entropy) should be high, and the right term (expectation of entropy over posterior) should be low. The first term, $\mathbb{H}(y_{1:b} | x_{1:b}, \mathcal{D})$, represents the entropy of the labels $y_{1:b}$ given the batch of points $x_{1:b}$ and the data \mathcal{D} . This term measures the uncertainty in the labels given the data points, and a higher value indicates the batch contains points, where model is marginally uncertain about their labels. The second term, $\mathbb{E}_{p(\theta | \mathcal{D})}[\mathbb{H}(y_{1:b} | x_{1:b}, \mathcal{D}, \theta)]$, represents the expected entropy of the labels $y_{1:b}$ given the batch of points $x_{1:b}$, specific model parameters θ , and the data \mathcal{D} , where the expectation is taken over the posterior distribution of the model parameters given the observed data \mathcal{D} . This term measures the average uncertainty in the labels for specific model parameters, and a lower value indicates that on average, the model is uncertain about the label of $y_{1:b}$ for given model parameters. In other words, different models (parameters) confidently disagree about the labels, so acquiring the labels $y_{1:b}$ for $x_{1:b}$ is expected to provide a significant information gain about the model parameters.

The uncertainty-based approach is a popular query strategy in Active Learning (AL) due to its simplicity and low computational complexity. Often employed in shallow models like SVM or KNN, it uses traditional uncertainty sampling methods to accurately measure a model’s uncertainty. In this strategy, the most uncertain samples are selected to form a batch query set. Margin sam-

pling is one example, where margin M is the difference between the highest and second-highest predicted probabilities for a sample. Smaller margins indicate greater uncertainty, and the top- K samples with the smallest margins are selected. Information entropy is another commonly used uncertainty measure, with larger entropy values indicating greater uncertainty. Thus, the top- K samples with the highest information entropy [184] are chosen. There are several DeepAL methods that use uncertainty-based sampling strategies [71, 156]. However, DeepFool Active Learning (DFAL) [40] highlights that conventional methods are vulnerable to adversarial examples. Instead, DFAL focuses on examining examples near the decision boundary and actively incorporates the information from these adversarial examples to estimate their distance to the boundary. This approach can effectively enhance the convergence speed of CNN training. However, it may lead to limited diversity in batch query samples, resulting in suboptimal or even ineffective deep learning model training. A potential solution is to adopt a hybrid query strategy that considers both the information volume and diversity of samples, either explicitly or implicitly.

Diverse Mini-Batch Active Learning [261], is one such method that incorporates informativeness into the K-means optimization objective by assigning weights. Additionally, it delves into a hybrid query approach that takes into account both the information volume and diversity of samples within a mini-batch sample query context.

Additionally, Ash et al. [5] proposed a method for achieving a balance between uncertainty and diversity in without the need for hand-tuned hyperparameters. The authors propose to measure the uncertainty of the model by calculating the gradient magnitude with respect to the parameters in the final (output) layer. This is done using the most likely label according to the model. The gradient magnitude indicates how sensitive the model's output is to changes in the input. Larger gradients signify higher uncertainty, meaning the model is less confident about the predicted label. To ensure that the selected batch of examples covers a diverse set of scenarios, the authors focus on gradients that span various directions. This helps the model improve its understanding of different aspects of the problem. Finally to build up the batch of query points based on the "hallucinated gradients" (imaginary gradients that haven't been observed yet), the authors employ the k-MEANS++ initialization technique. This method helps in selecting the initial cluster centers for the k-MEANS clustering algorithm. By using k-MEANS++ initialization, the authors can simultaneously capture the magnitude of a candidate gradient and its distance from previously included points in the batch.

Another group of active learning approaches explores the use of coresets. A coreset is a subset of a dataset that serves as a representative sample for the entire set. The concept, introduced by Phillips (2016) [148], involves utilizing coresets to approximate the full dataset in such a way that the output of an algorithm applied to the coreset closely resembles the output obtained from the entire dataset, with respect to a specific cost function. By executing the same algorithm on a smaller portion of the input data, coresets facilitate the development of efficient approximation algorithms. Many coreset methodologies draw inspiration from computational geometry

Geifman et al. [56] propose propose an active learning algorithm that is inspired by coreset dataset compression ideas. The method is based on the farthest-first traversal (also known as the

Gonzalez [65] algorithm) in the space of neural activations over a representation layer. The farthest-first traversal is a greedy algorithm used for the k-center clustering problem, where points are chosen iteratively, with each subsequent point being the one farthest away from the points already chosen. The main idea is that, for complex inputs like images or sound, it is more meaningful to consider the geometry of spaces induced by the representation layers of a deep neural network rather than the input space itself. In their approach, the authors aim to select informative samples based on their distance in the representation layer’s space.

Building upon prior work and drawing inspiration from a similar idea, Sener et al. [183] proposed a seminal deep active learning technique for batch sampling that was formulated as a core-set selection problem. The goal is to find a small subset from a large unlabeled dataset that allows a model to perform competitively when trained on it. authors derives a rigorous bound between the average loss over a chosen subset and the remaining data points based on their geometry. The proposed active learning algorithm aims to minimize this bound, which is equivalent to solving the k-Center problem, which involves finding a set of k centers in a metric space that minimizes the maximum distance between any point in the space and its closest center. The author employs an efficient approximate solution to this combinatorial optimization issue and evaluates the algorithm’s performance using three distinct image classification datasets, achieving state-of-the-art results by a significant margin.

In a recent study, Simeoni et al. [194] propose combining active learning with semi-supervised learning to fully utilize all available information, including unlabeled data, previously labeled data, and new data in each cycle of active learning. The authors first learn rich representations through unsupervised pretraining on the union of labeled and unlabeled data. In each active learning cycle, they employ semi-supervised methods to train the task learner. By comparing various acquisition strategies, the researchers discover that incorporating unlabeled data significantly enhances the performance of active learning methods.

2.4 Summary

To sum up, this chapter recognizes the difficulties stemming from limited data when training deep learning models, especially in the context of computer vision tasks. We have presented an overview of key approaches and techniques, each tailored for specific problem settings. Within each problem setting, these methods aim to tackle the issue by addressing it from different perspectives, thereby offering diverse solutions to the labelled data scarcity challenge.

We provided a summary of data augmentation techniques, transfer learning approaches, and active learning methods, which serve as key approaches to addressing the challenges of labelled data scarcity in deep learning models.

Data augmentation techniques, such as the widely used CutMix, MixUp, AutoAugment, and RandAugment, are applied to artificially increase the training dataset, subsequently enhancing the performance of deep learning models. As a standard part of training models, these techniques have

gained widespread adoption. However, they may face limitations due to insufficient diversity and the inability to address biases present in the dataset.

Pretraining, which is one of the most commonly used methods in training deep models, involves employing a model trained on an extensive dataset as the foundation for a new task. In contrast, multi-task learning focuses on training a single model on various related tasks to enhance performance across each individual task. The improvement in performance for individual tasks can be attributed to the implicit data augmentation, the regularization effect, and the enhancement of shared information between tasks that multi-task learning provides. Domain adaptation approaches aim to develop domain-agnostic data representations, facilitating the exchange of knowledge across disparate domains, further contributing to the effectiveness of deep learning models.

Active learning methods, on the other hand, prioritize expanding the labeled set while maximizing the information gained with minimal labeled data. These approaches involve querying an oracle, such as a human expert, to provide labels for the most informative subset of data, streamlining the learning process.

It is important to note that many of these techniques can be used simultaneously to optimize performance. For example, data augmentation, pretraining, and domain adaptation can be combined to achieve the best results.

In conclusion, although a variety of methods have been proposed to tackle the issue of data scarcity in training deep learning models, there remains significant potential for further advancements and improvements in this field.

Chapter 3

Active Learning for Structured Prediction from Partially Labeled Data

3.1 Abstract

In this work, our contribution lies in the development of a universal active learning algorithm designed for structured prediction. This method efficiently harnesses partially labeled data to train models producing interconnected labels for an image or video. We iterate between requesting user labels for unlabeled data and model retraining. Our novel algorithm, adept at selecting data for labeling, chooses instances that maximize expected information gain based on belief propagation inference. Applicable across a spectrum of tasks and models, we demonstrated its efficacy on the recognition of human actions and group activities in video sequences. Performance comparisons show our algorithm surpassing previous active learning methods and achieving accuracy on par with fully supervised methods, but using significantly less labeled data.

3.2 Introduction

Consider the activity recognition problem depicted in Fig. 3.1. Gathering large quantities of labeled data can lead to accurate classifiers that can predict what each person is doing and the overarching activity taking place in a scene.

However, naive approaches for labeling training data would be inefficient. Video data have significant redundancy; it seems unnecessary to label every single person in each frame of each video. Further, action categories vary significantly in frequency and intra-class variation. If images were selected randomly for labeling, copious quantities of similar walking poses would likely result. Finally, contextual inference that utilizes relationships between people in a scene should be accounted for when deciding whether a sample is necessary for learning.

For these reasons, we focus on active learning for building labeled datasets in structured prediction. In our framework the active learner benefits from partially labeled data. For example, in the task of activity recognition, we have video frames consisting of multiple people with both individual action and overarching activity labels. Our proposed algorithm selects which people and scenes

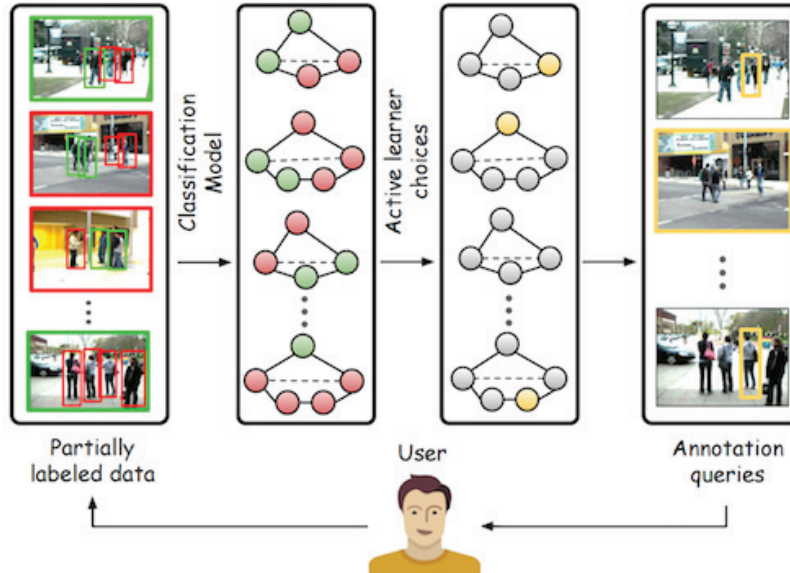


Figure 3.1: Example of active learning for recognizing human actions. Given a partially labeled dataset (labeled examples in green, unlabeled red), we train a classifier to recognize the actions of each person and group in a scene. Our active learning criterion selects individual people or scenes to be annotated by the user (shown in yellow). These new data are added to the training dataset, and the process repeated.

would be most informative to label. This allows labeling effort to hone in on the most important examples – such as unusual poses, categories with intra-class variation, key people in scenes, or ambiguous overarching scene labels.

While we demonstrate our active learning algorithm on human activity recognition, we note that structured prediction is a common task in computer vision. A variety of methods and algorithms have been developed in this vein. Recently, a number of approaches encode graphical model-style structured prediction inference within modern deep neural networks [262, 210, 85, 30]. Our algorithm follows in this line, showing that a novel criterion for active learning can result in highly effective inference for structured prediction with these state of the art techniques.

Active learning has deep roots within the vision/learning literature. The key ingredient of an active learning algorithm is the selection strategy used to choose which unlabeled examples to label. Although active learning has substantial literature, there has been limited work on active learning for structured prediction. Standard methods include uncertainty based sampling [133, 116], margin based sampling [180], expected model change [186, 225], and query-by-committee [187]. For example, a typical uncertainty based approach [133] would calculate the entropy of all the people in a scene separately and choose the ones with the highest entropy.

A key concept in graphical models and information theory is mutual information, which measures the amount of information gained about one random variable through the observation of another. In the context of a graphical model, an edge between two nodes signifies that they are not

independent and that they share some mutual information. In other words, knowing the state of one node can reduce uncertainty about the state of another. This dependency structure can lead to significant redundancies in the information carried by the nodes. For instance, in a densely connected graph, knowing the state of a single node might give us a considerable amount of information about the states of its neighbors. Similarly, in a hierarchical structure, higher-level nodes might summarize the information contained in a whole subset of lower-level nodes. Given this property, it's plausible to assume that we might not need to label every node in a graphical model to train an effective machine learning model.

To illustrate, consider a scenario where we are trying to understand the group activity of people in a scene. In this situation, each person could be a node in the graph, with edges representing the interaction between them. The group activity can be inferred by the individual actions of people, but in many cases, understanding the action of just one or two key individuals might be enough to infer the overall group activity. Hence, we might not need to label the actions of every individual (node) to predict the group activity effectively.

With this motivation, we propose the development of an active learning method designed to select for labeling those nodes expected to provide the most information. This strategy could be based on measures of expected information gain, identifying those nodes which, when labeled, would most significantly reduce the uncertainty about the unlabeled nodes. More specifically, we approximate the mutual information between each node and the rest of the graph by estimating how much the average entropy of the graph would be reduced if we "knew" the label of a particular node in the graph.

The main contributions of this work are (1) introducing a novel expected structured entropy reduction as a sample selection criterion for active learning, (2) developing an inference machine based on deep neural networks for efficiently computing this criterion, and (3) demonstrating the effectiveness of this algorithm for labeling data for human activity recognition

In the rest of the chapter, we first review related work (Sec. 3.2.1). Then we present our algorithm and the details of our proposed criterion (Sec. 3.3). We conduct empirical comparisons of our proposed method to the state of the art (Sec. 3.4) and conclude in Sec. 3.7.

3.2.1 Related Work

In this chapter we propose a novel approach for active learning in structured prediction, applied to human activity recognition. Each of these areas has seen substantial amounts of previous work. Below, we briefly review closely related previous work in each of these areas.

Active Learning: A large number of methods and applications of active learning have been explored. Active learning has been used for many tasks such as image classification [150], categorization of images and objects [94, 86], video segmentation [41, 45], and discovery of human interactions [105]. Another prominent use case is labeling large video or image datasets [245, 20, 228, 169]

There are numerous methods to measure the informativeness of samples. *Entropy* [189] is a standard measure, which has been used in many applications. For example, Holub *et al.* [77] devel-

oped a measurement based on minimum expected entropy. Varadarajan *et al.* [222] measure entropy reduction over the whole dataset. An alternate uncertainty criteria is least confidence, where the learner queries the instance that has the lowest probability for its most likely label [185]. Roth and Small [165] use the margin between the two most probable classes as an uncertainty measure. Sun *et al.* [203] estimate the joint distribution in a histogram-based method, however full supervision is needed, which could be expensive.

All these methods require the user/oracle to fully label an instance. However, we are interested in structured prediction tasks. Recent active learning methods that use partially labeled data include Luo *et al.* [133], in which the entropies of the marginal distributions are computed using belief propagation. Vezhnevets *et al.* [225] measured the (discrete) count of changes in labels of the nodes in a CRF, retraining the model for each instance and every possible label, on each iteration. We build on these approaches, by taking into account relations between examples, and select instances based on how much information they can provide to other examples as well. Further, rather than counts, we instead we measure expected entropy change which is a more informative, continuous value, within a more efficient belief propagation paradigm.

Structured Prediction: Many computer vision problems involve a set of output variables with relations among them. Structured prediction has a long history and recently has been addressed using deep learning. Zheng *et al.* [262] design a network called CRF-RNN which is basically a CRF that is formulated with Gaussian pairwise potentials, trainable in an end-to-end manner. Tompson *et al.* [210] build a model consisting of a MRF and a CNN that are jointly trained to exploit relationships between body parts for human pose estimation. Deng *et al.* [31] propose to represent factor graph in deep learning to model group activities. Jain *et al.* [85] use a generic method to train spatio-temporal graphs in a deep learning framework. Deng *et al.* [30] propose a sequential inference model in an RNN framework that is motivated by the traditional message passing inference algorithm. They utilize gates on edges between graph nodes to learn the structure of a graph. Our proposed method utilizes Deng *et al.*'s inference RNN-based inference technique.

Activity Recognition: In our work we focus on recognizing human actions and overarching group activities taking place in a scene. Previous work on this topic has shown that modeling this problem as a form of graphical model helps in providing contextual information for recognizing individual and group activities [2, 115, 171, 17, 172, 162]. Lan *et al.* [115] used latent variables in a max margin framework to find the most discriminative structure for the scene. Amer *et al.* [2] used grouping nodes to achieve the same goal. Shu *et al.* [192] utilized AND-OR graphs to recognize events and assign roles to the engaged people in noisy tracklets. Khamis *et al.* [100] combined track-level and frame-level information for the task of action recognition. Ramanathan *et al.* [154] use attention models to focus on the key player in sports videos. Deng *et al.* [30] used Recurrent Neural Networks to learn the structure of the people in a scene. In this work we propose a general active learning method for selecting the most informative nodes in structured data. Activity recognition methods can benefit from our approach since annotating videos/images with a full set of action and activity labels is time-consuming.

3.3 Active Learning for Structured Prediction

Successes in visual recognition hinge on the availability of quantities of labeled training data. Acquiring these labeled data efficiently is especially important in human activity recognition. Videos contain much redundant data, and significant structure exists amongst the people present in a scene. Leveraging information regarding the relationships between people and focusing labeling effort on more challenging categories present opportunities for efficiency.

Consider the example images in Fig. 3.2. Knowing that the action of one person in this scene is *walking* will assist in labeling other people in the scene. As another example, consider the volleyball scenes depicted in Fig. 3.6. Focusing labeling effort on rare, challenging classes such as *spiking* that help determine the group activity is advantageous.

We formalize our ideas within a typical active learning setting. We start with an initial labeled data set \mathcal{L} , and train a model. Next, we select for labeling additional data from an unlabeled pool \mathcal{U} . The learner will alternate between these two stages to build a progressively more accurate model. The key question we must answer is: *which* unlabeled instances from \mathcal{U} should we choose to label?

In our work the data are a set of video frames, in which we wish to predict the actions of each person and the overarching group activity taking place. In this structured prediction task, good samples to label are ones about which we are uncertain, and will help to disambiguate many other labels. Hence, our goal is to design an active learning algorithm that annotates the right human actions / group activity labels to improve the learned classifier.

To quantify this we use expected reduction in entropy as our selection criterion. Entropy refers to uncertainty. For example, if an instance has similar probabilities for different labels it has a high entropy. An instance with low entropy means the model can determine the label of that instance with high confidence. We select the most informative instance(s) based on how much entropy is reduced.

This is a simple idea and similar ideas have been used before. However, structured prediction presents new challenges: *how can we know how much labeling a particular unlabeled instance $y_u \in \mathcal{U}$, reduces the entropy?* This is the challenge that we address in this work. In a naive solution, for all possible labels of y_u a classifier is trained on $y_u \cup \mathcal{L}$ and the expected entropy is computed by taking a weighted average. Obviously, this is not feasible because it is time consuming (especially for deep learning) and secondly adding one datum is very noisy.

We address this challenge with an intuitive method based on belief propagation. In order to estimate the entropy if we knew the label of an unlabeled sample y_u , we consider it as an observation using every possible value for it, and perform belief propagation. Based on this intuition, for every unlabeled node y_u we calculate the average entropy reduction and select the ones that have the largest entropy reduction. We explain the details of this method below.

3.3.1 Stage 1: Structured Prediction Model

In this section we introduce the formulation of the structured prediction model and how we connect it to the active learning procedure.

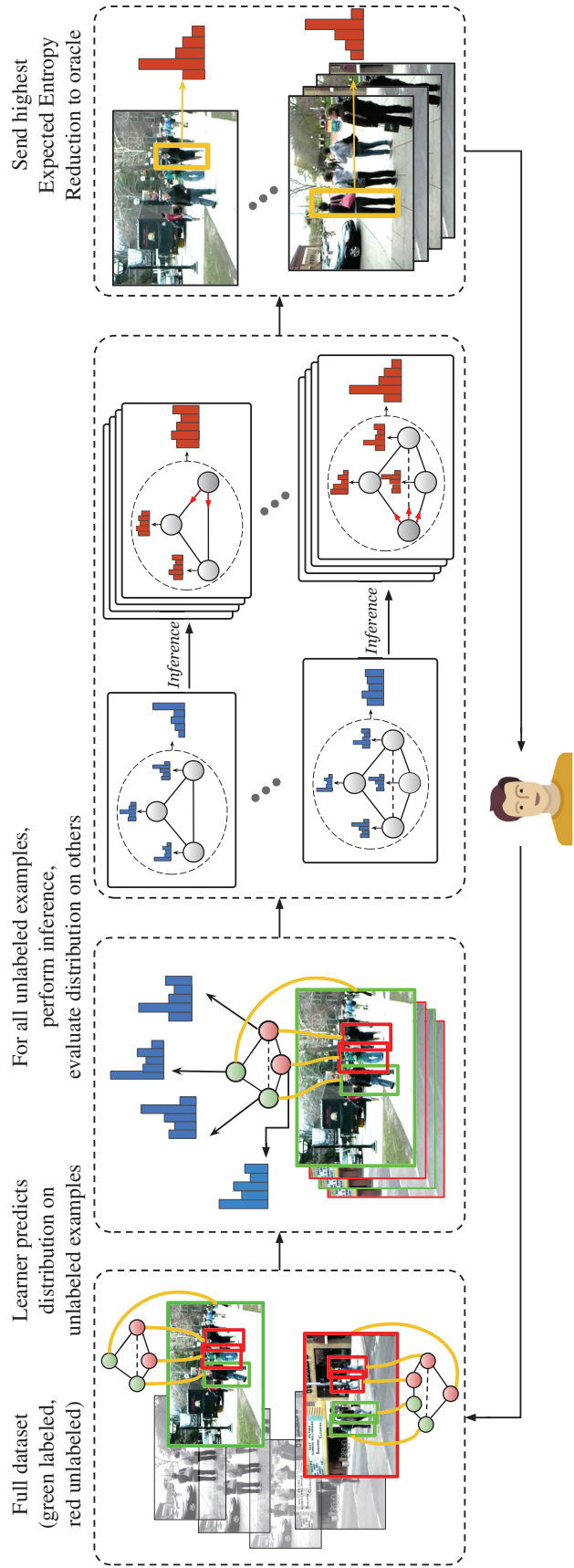


Figure 3.2: Overview of the method. A structured prediction model is trained given the training dataset. This model produces a probability distribution for every node. In the second stage for each unlabeled node we estimate expected entropy reduction by fixing its value and running inference. We sort all nodes based on expected entropy reduction their labeling would produce. The top K nodes are selected and sent to user for labeling. These nodes can be either human action or scene labels. The new labeled data is then added to the training set and this process repeats.

We formulate our model as follows. Denote the image observations as $X = \{x_i\}$ and the people and scene labels as $Y = \{y_i\}$. We define a graphical model describing the distribution over variables Y conditioned on the observation X :

$$P(Y|X; \theta) = \frac{1}{Z_X} \prod_{f \in F} \phi_f(y_f, x_f; \theta_f) \quad (3.1)$$

in which θ are the parameters of the model, F is the set of cliques, y_f , x_f , and θ_f are variables and parameters of a clique f . We utilize graphs that connect each action / group activity label to its corresponding person / scene image features, and a fully connected graph of relations between all people and the group activity in each scene.

In our work we model all distributions using convolutional neural networks. These include potentials that relate image observations to action/group activity labels, and potentials over sets of action/group activity labels in an image.

Inference in this graphical model is conducted using a forward pass through the neural network, which imitates a fixed set of rounds of belief propagation over the graph [30]. In this manner, we can estimate marginalized probability distributions $P(y_i|X; \theta)$ over a particular variable, or likewise $P(y_i|X, y_j = k; \theta)$ given a value for another label y_j .

Parameter Learning. In active learning, we normally only have a part of the data labeled. Denote the labeled set as $\mathcal{L} = \{y_l\}$. To learn the model parameters θ , we use the loss:

$$\ell(\theta) = - \sum_{y_l \in \mathcal{L}} \log P(y_l = y_l^* | X; \theta) \quad (3.2)$$

where y_l^* is the provided label for variable y_l . This function minimizes the negative log likelihood of the correct classes for the labeled data.

Based on the above formulation, in the next section, we provide the details of how our active learning algorithm chooses new data to be annotated by the oracle.

3.3.2 Stage 2: Instance Selection Strategy

Our objective is to select informative samples for labeling. These are samples that would significantly reduce ambiguity in labels across an entire scene. The previous section discussed how inference could be conducted provided we have certain node values.

The ideal scenario is to pick a node from a graph that holds the maximum information about the other nodes. This node is the one with the highest mutual information with the rest of the graph, meaning it carries the maximum information. To put it another way, knowing the label of this node would provide us with the most information compared to knowing the label of any other individual nodes. Let's assume a graphical model, g , with N nodes. We can frame the problem as follows:

$$y_i = \arg \max_i I(y_i, g) \quad (3.3)$$

Here, mutual information can be portrayed using the entropy of the graph, as illustrated below:

Algorithm 1 Compute Top K Most Informative Instances

```
1: procedure GETINSTANCES(COUNT: INT)
2:   for all frame  $f$  in dataset do
3:      $g \leftarrow$  BUILDGRAPH( $f$ )
4:      $\Phi^g \leftarrow$  GETEXPECTEDENTROPYREDUCTION( $g$ )
5:   end for
6:   return ARGMAXTOPK( $\Phi$ ,  $count$ )
7:   // Note:  $\Phi$  is a set of arrays. ArgmaxTopK searches this 2D structure and returns indexes of  $K$  largest
   instances (persons or scenes annotations)
8: end procedure
```

$$y_i = \arg \max_i H(g) - H(g|y_i) \quad (3.4)$$

In these equations, $H(g)$ represents the entropy of the graph, and $H(g|y_i)$ denotes the entropy of the graph given the label of node i . In essence, the most informative node is the one that, if its label were known, would most significantly reduce the entropy of the graph.

We now utilize this to compute expected entropy reduction to determine the informativeness of samples.

Assume a frame requiring N labels (person actions + group activity) is given; for ease of notation we assume each can take T possible values (actions / group activity). We can obtain the probability distributions $P(y_i|X; \theta)$ for each person and the group activity using the model from Sec. 3.3.1.

The entropy of node i with label y_i , $H(y_i)$ is defined as:

$$H(y_i) = - \sum_{j=1}^T P(y_i = j|X; \theta) \log P(y_i = j|X; \theta) \quad (3.5)$$

where $P(y_i = j|X; \theta)$ indicates the probability of assigning label j to the variable i . The *average entropy*, $\bar{H}(g)$, of a graph with N nodes is the mean of the individual node entropies:

$$\bar{H}(G) = \frac{1}{N} \sum_{i=1}^N H(y_i) \quad (3.6)$$

Expected average entropy. In a graph g with some unlabeled nodes, in order to find the most informative node, we need to find the node that reduces the *average entropy* of the graph if its label is known. Computing the actual *average entropy* of the graph if we label an example is not possible. This is because the labels that would be given to examples are not known, and if a certain label is given to an example it impacts the entropy over other nodes in the graph. Therefore, we approximate it with the *expected total entropy*.

The key idea behind the expected average entropy is the following. Suppose we choose to ask a user to label node i in graph g . We currently have a belief (probability distribution) over the possible

Algorithm 2 Compute Graph Expected Entropy Reduction

```
1: procedure GETEXPECTEDENTROPYREDUCTION(G: GRAPH)
2:    $P \leftarrow \text{DOINFERENCE}(g)$ 
3:    $\bar{H}(g) \leftarrow \text{GETAVERAGEENTROPY}(g, P)$ 
4:   for all node  $i$  in graph  $g$  do
5:     for label  $j$  in  $1..T$  do
6:       Fix the label of node  $i$  to  $j$  by changing probabilities
7:        $\hat{P} \leftarrow \text{DOINFERENCE}(g, i, j)$ 
8:        $H(g|y_i = j) \leftarrow \text{GETAVERAGEENTROPY}(g, \hat{P}, i, j)$ 
9:     end for
10:     $E[\bar{H}(g|y_i)] \leftarrow \sum_{j=1}^T P(y_i = j|X; \theta) \bar{H}(g|y_i = j)$ 
11:     $\Phi_i^g \leftarrow \bar{H}(g) - E[\bar{H}(g|y_i)]$ 
12:  end for
13:  return  $\Phi^g$ 
14:  // Note:  $\Phi^g$  is a tuple of nodes computations
15: end procedure
```

labels the user could give us for this node. For each of these possible resultant labels, we can estimate the average entropy that would remain. This is done by running our structured prediction model while fixing the value of node i . The expected average entropy computes a weighted average of these total entropies according to our current belief about node i .

In detail, denote by $\bar{H}(g|y_i = j)$ the average entropy of the graph g if the label of node i is known and is equal to j (symbol g is omitted for simplicity). In order to compute $\bar{H}(g|y_i = j)$, we fix the label of node i to j and run inference to obtain the probability distributions of all the other nodes. Then we compute the average entropy of the graph using Eqs. 3.5 and 3.6.

$$\bar{H}(g|y_i = j) = -\frac{1}{N} \sum_{\substack{n=1 \\ n \neq i}}^N \sum_{t=1}^T \hat{P}(y_n = t) \log \hat{P}(y_n = t) \quad (3.7)$$

where $\hat{P}(y_m = t) \equiv P(y_m = t|X, y_i = j; \theta)$ is the probability of node m having the label t , after fixing the label of node i to j . We can compute $\bar{H}(g|y_i = j)$ for all possible values of j (label of the node i).

The expected average entropy $\bar{H}(g|y_i)$ is defined as:

$$E[\bar{H}(g|y_i)] = \sum_{j=1}^T P(y_i = j|X; \theta) \bar{H}(g|y_i = j) \quad (3.8)$$

Fig. 3.3 illustrates this process. As a result, we can determine for each node i in the graph, what we expect to happen if this node were chosen for labeling by the user.

Expected Average Entropy Reduction. Finally, we choose to label the node(s) i that would result in the largest reduction in entropy. Denote by Φ the amount of information that nodes could provide to their graphs if labels are known. For a particular node i in a graph g we define the expected

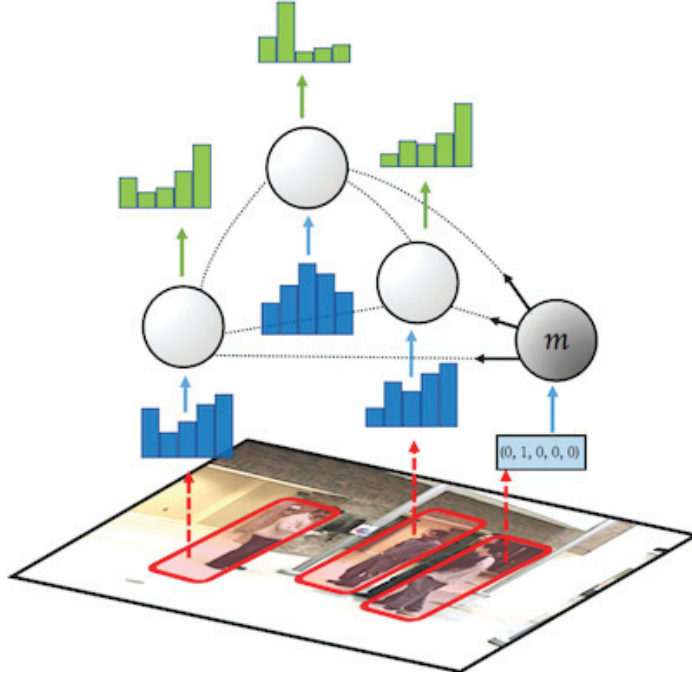


Figure 3.3: In the above graph node m is unlabeled. In order to compute the *average entropy* of the graph if the label of node m was given and equal to 2 we do the following. First set the label distribution of the node m to $[0, 1, 0, 0, 0]$ ("2"). Then we perform inference using the current learned inference machine. This will produce the probability distributions of all nodes, \hat{P} . Then the *average entropy* is the mean entropy over all the nodes. In this figure the **blue histograms** are the action label distributions of the nodes obtained given the current model; and the **green histograms** are the probability distributions of the nodes after observing $m = 2$.

average entropy reduction as:

$$\Phi_i^g = \bar{H}(g) - E[\bar{H}(g|y_i)] \quad (3.9)$$

Then, to add the next K training instances \mathcal{S} to our dataset, we get the top K most informative nodes in all available graphs from Φ (which is a set of arrays, for every graph, Φ_i^g is computed for its nodes). This can be formulated as the following:

$$\mathcal{S} = \text{ARGMAXTOPK}(\Phi, K) \quad (3.10)$$

where *ArgmaxTopK* returns the positions of the top K entries (corresponding to person or scene nodes) in Φ .

Summary. The complete algorithm is as follows. Given a structured prediction model with parameters θ_t , labeled set \mathcal{L}_t , and unlabeled set \mathcal{U}_t at iteration t , our active learning process evaluates each node in \mathcal{U}_t by determining how much entropy reduction we expect to obtain by labeling it. We select a set of nodes $\mathcal{C}_t \subseteq \mathcal{U}_t$ for labeling by the user according to this criterion. These nodes are annotated and labeled/unlabeled sets are updated: $\mathcal{L}_{t+1} = \mathcal{L}_t \cup \mathcal{C}_t$; $\mathcal{U}_{t+1} = \mathcal{U}_t \setminus \mathcal{C}_t$. Then the structured prediction model is re-trained and new parameters θ_{t+1} obtained, and the process repeated. Algo-

rithms 1 and 2 summarize this process. We provide a reference implementation in Caffe to enable reproduction of the results.

3.4 Experiments

We evaluated our method on the task of group activity recognition, which involves structured prediction of individual human actions and group activities. We utilize the Volleyball Dataset [81] and Collective Activity Dataset [18] to evaluate the performance of our method.

Below, we compare to other active learning methods, structured prediction baselines, and approaches based on entropy. We further analyze and interpret the performance of our method. The supplementary material contains details on training settings and results visualizations.

3.4.1 Datasets

Volleyball Dataset [81]: This dataset contains 4830 annotated frames from 55 volleyball videos with *nine* player action labels (waiting, setting, digging, falling, spiking, blocking, jumping, moving, standing) and *eight* team activity labels (right set, right spike, right pass, right winpoint, left winpoint, left pass, left spike, left set). The standing action represents around 70% of the action labels, making the dataset very imbalanced. The distribution of *action labels* and *scene labels* are illustrated in the right-most histogram of Figures 3.5b and 3.5c respectively. We follow the same datasets split as [81], in which 3493 frames are used for training and 1337 frames for testing. There is a maximum of 12 people in each frame, and the number of people varies from frame to frame. The total number of annotations in the training set is 44,294 (3493 scene labels and 40801 action labels). From the training set, we only use 696 frames for the initial fully labeled training data and the rest ($\approx 35,400$ annotations) is the unlabeled pool.

Collective Activity Dataset [18]: This dataset contains 44 videos of outdoor/indoor activities Crossing, Waiting, Talking, Queueing, and Walking. Each individual person could have one of the following 6 labels: Crossing, Waiting, Talking, Queueing, Walking, and Not Available. We use the same train/test split as [115]. The training set contains 1908 frames. The total number of annotations (scene/action labels) in the training set is 11,734. We only use 382 frames for the initial fully labeled set and the rest ($\approx 9,400$ annotations) is used as the “unlabeled” set.

A model is trained using the initial training set. Our method iteratively selects a number of scenes and queries the user to label certain people in some scenes (a total of K annotations per iteration). The number of people that are labeled at each iteration depends on the available resources that one has. We show results of $K = 1000$ (and $K = 500$ in the supplementary material).

3.4.2 Baselines

In order to verify the effectiveness of our method, we compared our model against baselines that: (1) utilize entropy in a non-structured prediction setting, (2) utilize alternative structured prediction criteria, and (3) standard active learning methods.

- *Separate Active (SA)* [133]: Select K nodes, including person nodes and scene nodes, with the highest entropy. We implement their “Separate active” method using approximate belief propagation for inference.
- *Least Confidence (LC)* [185]: Among all the person/scene nodes in all the graphs (frames), select K nodes that the trained model has the lowest confidence in their most probable labeling.
- *Margin (M)* [165]: select the K nodes amongst all the nodes in the unlabeled pool that have the lowest margin between their two most probable labellings.
- *Expected Change (EC)* [225]: select the K nodes that has the highest expected number of changed labels.
- *Random Sampling (RND)*: randomly select a batch of nodes (persons and/or scenes) in all graphs (frames).

The Least Confidence [185] and Margin [165] algorithms are originally proposed for simple prediction but we extended them for our structured prediction task. We examine performance of these methods over iterations of active learning. To enable fair comparison, the same amount of data/annotations are given to all the methods.

We trained a base model with the same initial training set (which is a small portion of the whole training set) for each method, and then grow the labeled training set by actively selecting K annotations per iteration.

3.5 Results and Analysis

Below, we compare our method quantitatively to baselines and perform analysis of the results.

3.5.1 Quantitative Results

Volleyball Dataset: Figures 3.4a and 3.4b show the results of our method and baselines on the Volleyball Dataset. Our proposed method exhibits similar performance, focusing on scene labeling performance first, outperforming all baseline methods at this task and eventually outperforming all at action labeling as well.

⁰The accuracy versus added annotation is not a monotonically increasing function. Due to existence of multiple local optima, the accuracy of the model could decrease. Also there is a possibility that the model overfits to the given training set.

Collective Activity Dataset: Figures 3.4c and 3.4d illustrate the comparison of our method against baselines on the Collective Activity Dataset. Note that performance on both individual action and group activity (scene) labeling are measured. Our method tends to focus its initial effort on labeling scenes, likely because of the gains in structured prediction performance that can be obtained in this manner. Overall, our method outperforms all baselines on this dataset on scene labeling and after a few iterations outperforms all baselines on action labeling as well.

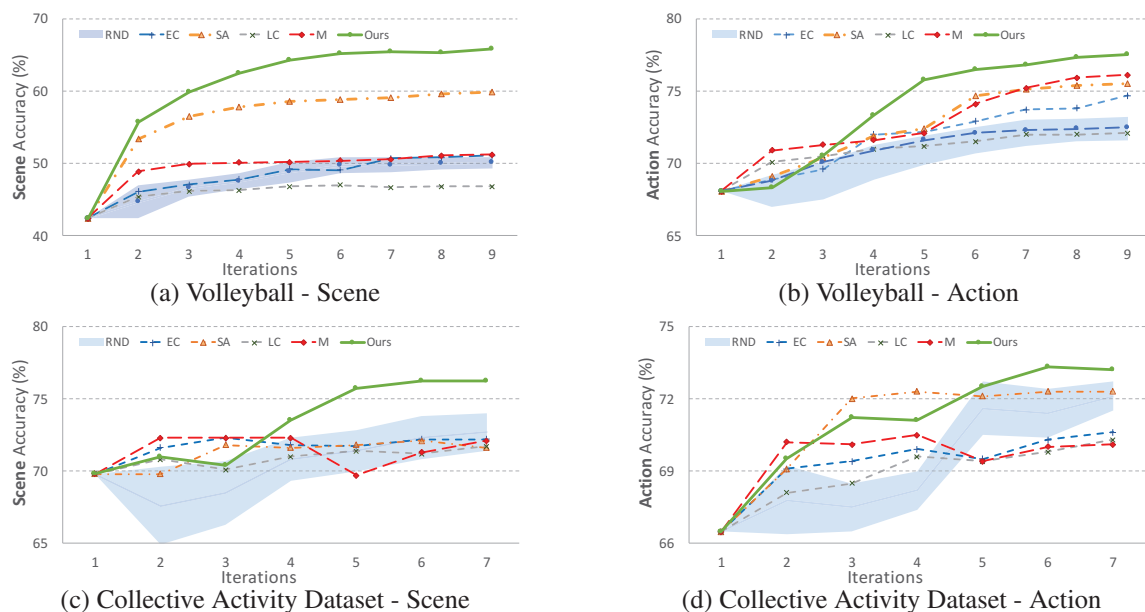


Figure 3.4: Results of comparison of our method against baselines on *Volleyball Dataset* (a and b), *Collective Activity Dataset* (c and d). The y-axis is accuracy and x-axis is iteration. For all the methods we start from the same small initial labeled set so the accuracies of the first column are exactly the same. Our method outperforms the other methods in both action and scene accuracy¹.

The full details of Figure 3.4 and a table of the accuracy that each method has achieved at each iteration is reported in the supplementary material.

3.5.2 Comparison to Supervised Methods

For these datasets alternate approaches use all available data supervised. Our experiments show that our method can save $\approx 70\%$ of annotation cost yet achieve similar accuracy as the state-of-the-art supervised methods.

In terms of fully supervised methods, Deng et al. [30] achieved 74.0% *scene accuracy* on the whole training set (11,734 annotations) using the “untied version” of their method. Our method achieves comparable accuracy (74.8%) after 3 iterations of adding 500 annotations (in total 1500 extra annotations) (Table 1 right-side in the supplementary material). In other words, using 33% of the data the same accuracy is achieved. Higher accuracy (76%) is obtained after labeling $\approx 44\%$ of the data. This shows the potential for active learning-based methods for this task.

		Number of Added Annotations									
		0	1000	2000	3000	4000	5000	6000	7000	8000	
Scene	Epochs										
	90	42.4	56.9	60.6	62.5	64.4	65.3	66.1	66.1	66.2	
	60	42.4	55.7	59.9	62.4	64.3	65.2	65.4	65.3	65.8	
	30	42.4	53.7	58.5	61.7	62.9	64.1	64.6	64.9	65.1	
	15	42.4	53.2	58.4	60.8	62.6	63.3	63.4	63.6	63.8	
Action	90	68.1	68.4	70.7	74.3	76.1	76.8	77.3	77.7	78.1	
	60	68.1	68.3	70.5	73.3	75.8	76.5	76.8	77.3	77.5	
	30	68.1	68.3	70.1	72.9	74.8	75.6	76.3	76.7	76.9	
	15	68.1	68.2	69.7	72.5	74.1	74.3	75.4	75.7	75.8	

Table 3.1: Results of our method with different number of epochs at each iteration.

Ibrahim *et al.* [81] obtain 68.1% accuracy (the non-temporal model) using 44,294 training annotations. Using much less training data, e.g. $\approx 35\%$ of the data we can achieve results comparable to this fully supervised method.

3.5.3 Analysis

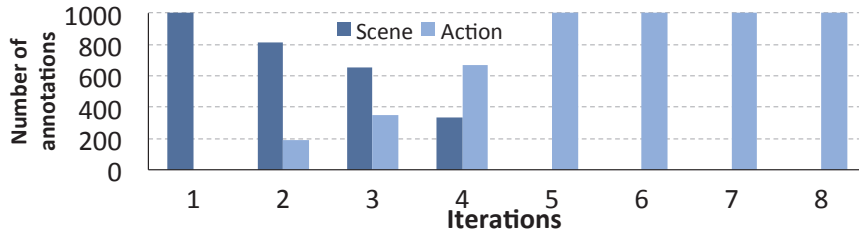
Analysis of Nodes Chosen: Figure 3.5a shows the number of *scene* and *action* nodes that have been selected by our active learner at each iteration. In the first few iterations, mostly scene nodes have been chosen for labeling and later on action nodes are selected. This makes intuitive sense given the impact that correct scene labels can have on the action labels in a video frame. The reason that our method doesn't outperform baselines in terms of action accuracy at first is that almost no new action annotations are provided to the learner. But after choosing action nodes our method outperforms the baselines.

The distributions of labeled action and scene classes at each iteration are illustrated in Figures 3.5b and 3.5c respectively. For the sake of comparison, the distribution of different classes in the whole training set is shown in the right side of Figures 3.5b and 3.5c.

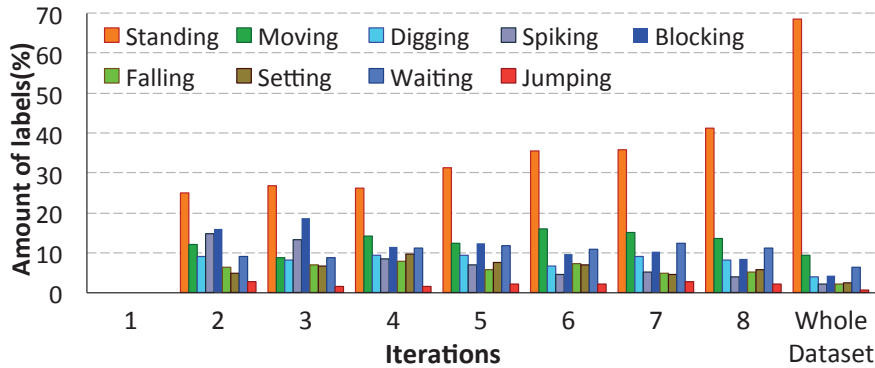
There is an interesting observation in Figure 3.5b that 70% of the action labels belong to the *standing* action, which is $\approx 28,000$ annotations. Looking at the distribution of the selected action labels in the first few iterations, the ratio of the *standing* action to other actions is significantly lower than its frequency in the dataset. This indicates that our algorithm considered this class as an easy class and a smaller number of instances were selected. On the other hand, only 1% of the training instances belong to the *jumping* class. Therefore the probability of choosing a sample from this class is below 1%. Nevertheless, our method has selected significantly more from the *jumping* class at each iteration, in order to model the relative complexity of this class.

Regarding the *standing* class, because the number of standing people is enormous compared to other classes, the probability of choosing this class is still higher; thus at each iteration more *standing* instances have been chosen compared to other classes such as *jumping*.

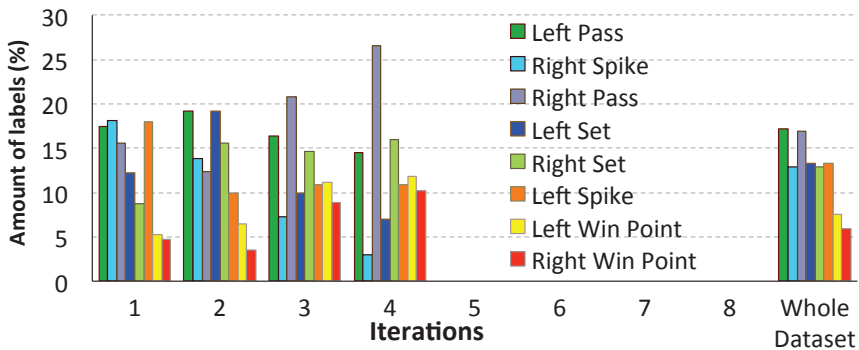
Effect of number of epochs in training: In this section we study how the number of epochs used in training affects the accuracy. For a given annotation set often the more epochs of stochastic gradient descent the better accuracy gets. There exist different work-flow patterns for acquiring labels from



(a) Number of scene/action annotations selected at each iteration for volleyball dataset experiment, in which 1000 annotations are added at each iteration.



(b) Amount of individual **action** labels chosen at each iteration for volleyball dataset. No action labels are added at iteration 1.



(c) Amount of **scene** labels chosen at each iteration for volleyball dataset. After iteration 4 no scene labels are added.

Figure 3.5: Distribution of scene/action labels at each iteration

an oracle (e.g. Amazon Mechanical Turk). These include recruiting workers at sparse intervals, inter-leaving human labeling with model training, and concurrent training and labeling.

In a deep learning framework, usually with the right choice of learning rate and weight decay, the solver achieves higher accuracies if *enough time* is spent on training. However, in active learning, there might be some cases where user wait time is constrained and the interaction time between a

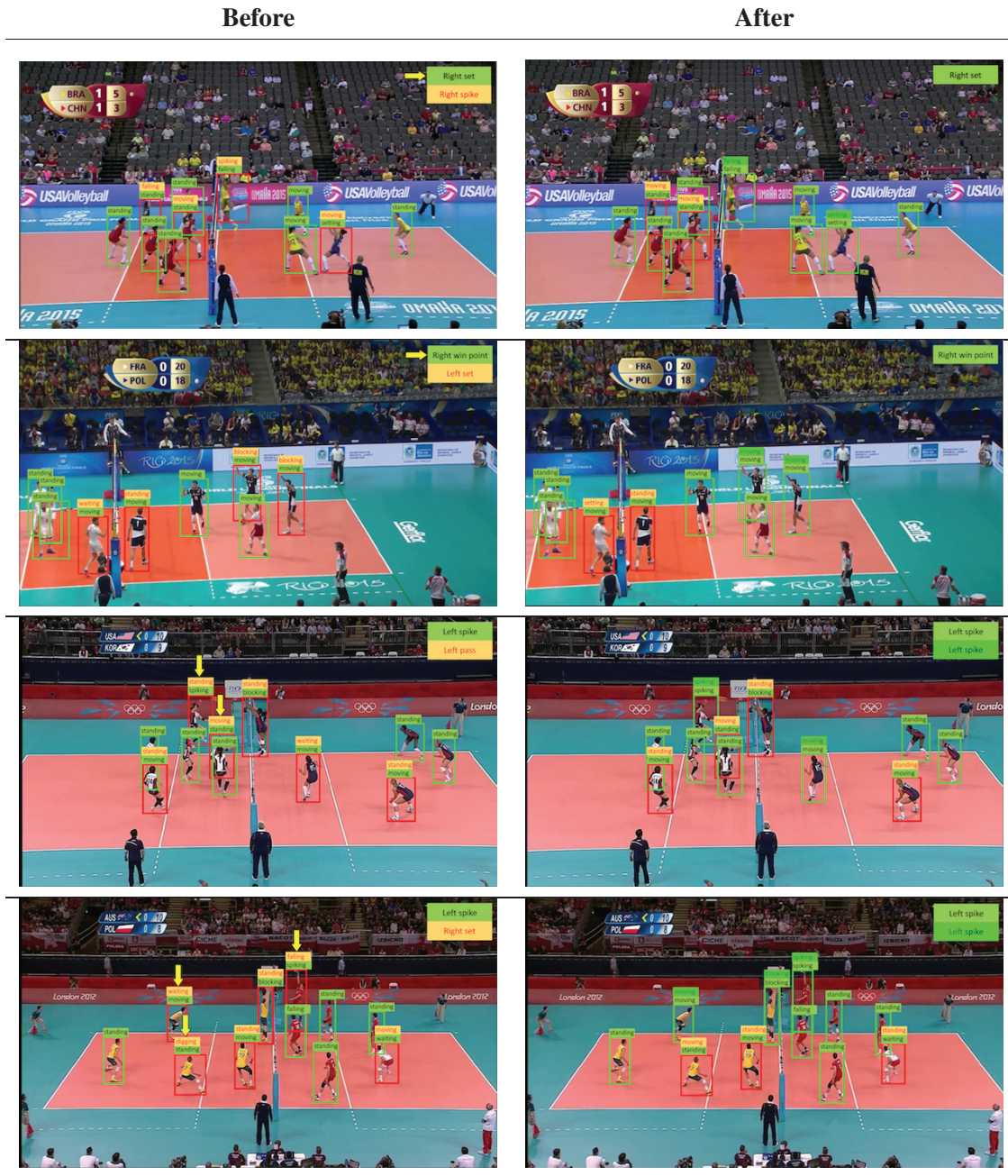


Figure 3.6: Examples of chosen variables in the Volleyball dataset. Each row shows one pair of before/after an active learning iteration. Left image shows scene with current labels (ground truth in **green box**, incorrectly predicted labels in **yellow box**). Yellow arrow shows variable chosen for labeling by oracle by our method. Note that selection of group activity (*scene* annotation) in top 2 rows or actions of specific people (bottom 2 rows) help in correcting other labels afterwards. user and the active learner is limited. In this case, less time should be spent on training². We have run

²This depends on the setting, one could also use crowdsourced services where the “wait time” might involve different labelers and not be relevant.

experiments that show the performance of our method for different numbers of epochs of learning in between each active data labeling cycle. We have reported our results in Table 3.1.

In terms of running time, the bottleneck of our pipeline is the training part not the instance selection part. In each iteration of our experiments on the Volleyball Dataset, training for 60 epochs takes 1 hour (only ≈ 5 minutes for sample selection and ≈ 55 minutes for training) on a GeForce GTX 1080 in a Caffe [89] implementation. There is an application-dependent trade-off based on worker recruitment, ramp-up, labeling time, etc. The results that we have provided for different numbers of epochs (running time) in Table 3.1, illustrate this accuracy-time trade-off.

3.6 Discussion

3.6.1 Recent Advancements in Data Labelling Systems and Active Learning

This research was completed in 2015. Since this time, there have been significant advances in the field of machine learning and data labelling. In this section we address the relevance of this research in the context of these advances.

The success of deep learning highlights the crucial role of data [112] in allowing machine learning models to tackle real-world problems. Deep models depend on the quality of the data they use. As a result, various specialized software tools for labeling data [170, 42, 196, 49, 175] have been developed to simplify the process and create large datasets suitable for training these models. These tools are designed to minimize the costs and human effort involved in labeling.

However, this leads to an important question: Is active learning still relevant with the advancements in data labeling software? A recent report [1] indicates that the global market for data collection and labeling is growing rapidly, with its value expected to increase from \$2.9 billion in 2023, to \$17.1 billion by 2030. Moreover, a recent study [173] conducted a comprehensive review of the available data labeling software, focusing specifically on image labeling software (ILS), over the past decade. The study revealed a substantial growth of 400% in the number of research papers related to ILS from 2013 to 2019.

This trend clearly highlights the ongoing importance of acquiring labeled data. Despite the cost and effort reductions brought about by data labeling software, the overall labeling costs on a global scale remain significant. Active learning strategies, which aim to reduce labeling costs, work alongside data labeling software instead of competing with it. Consequently, active learning remains an important area of research, even in domains such as computer vision and text analysis.

Furthermore, although the advancements in the ILS have considerably reduced the cost of labelling over the years, we must consider that these advancements are not uniformly distributed across all areas of computer vision. For instance, labeling in complex and highly variable domains like medical imaging, wildlife tracking, or astronomical imaging continues to present significant challenges due to the unique nature of these data and the specialized expertise needed for accurate labeling. As such, our active learning algorithm continues to hold great relevance for these challeng-

ing domains, as it helps prioritize the labeling of the most informative data points and thus optimize the utilization of limited expert time and resources.

Additionally, it is essential to consider the broader application of the principles of active learning beyond computer vision. While we utilized a group activity recognition problem in computer vision as a demonstration, the active learning algorithm developed is fundamentally a general-purpose approach. It can apply to any problem or domain where data can be represented by graphical models. This includes domains medical imaging, drug discovery [43], or other structured prediction problems. These areas continue to face issues of data sparsity and high labelling costs, underlining the continued relevance of our work.

Moreover, our algorithm’s utility is not limited to situations with scarce labeled data or high labelling costs. The selection of most informative data points using mutual information can also be beneficial in scenarios with abundant data. With the growth of data, the computational burden of training models increases. By focusing on more informative samples, we can potentially achieve comparable model performance with less computational resources and time.

In conclusion, while the landscape of computer vision and data labeling has evolved over the past eight years, the relevance and importance of our work in active learning remain considerable. The principles and methods we have developed not only continue to provide value in specific computer vision tasks and other structured prediction problems but also contribute to the broader advancement and understanding of machine learning as a field.

3.6.2 Potential Future Directions

This study has provided us with several possibilities for expanding and improving our existing knowledge and approach. One such possibility is related to our use of the concept of expected information gain. We have applied this concept by considering the mutual information between the most informative node and the rest of the graphical model. To do this, we relied on the entropy of the graph and conditional entropy. However, in our actual calculations, we resorted to an approximation of the entropy and conditional entropy within the graphical model. This decision was primarily driven by the considerable computational complexity involved in calculating all the conditional entropies. We can express the entropy of a joint distribution as:

$$H(X_1, X_2, \dots, X_N) = \sum_{i=1}^N H(X_i | X_1, X_2, \dots, X_{i-1}) \quad (3.11)$$

However, each node in our algorithm required an evaluation proportional to i^L for $H(X_i | X_1, X_2, \dots, X_{i-1})$, as each node could have L potential labels. One potential method for estimating the expected entropy without needing to make i^L evaluations is to only take into account the most probable label for each node, thereby approximating this conditional entropy with a single evaluation.

Another future direction for this work could be the integration of semi-supervised or self-supervised learning into our framework. These approaches, which make use of both labeled and

unlabeled data, could potentially offer additional efficiency and accuracy benefits, leading to more robust and reliable outcomes.

3.7 Summary

In this chapter, we presented a general purpose active learning approach for partially labeled structured prediction models that can be represented using graphical models. In our method, the initial models are built using a small training subset. Then, in an iterative manner we select a subset of people/scene labels to relearn better models. The selection strategy is based on a novel expected information gain criterion, which is computed from expected entropy reduction. Results demonstrate that our algorithm can improve upon baseline active learning approaches and achieve results competitive with fully supervised methods while using only a fraction of the labeled data.

Chapter 4

Dataset Augmentation for Human Action Classification

4.1 Abstract

Recognizing the demand for extensive and diverse datasets in deep learning for computer vision, we've designed a novel method to generate data tailored for action recognition tasks. Our approach starts with a minimal set of human action videos and generates new variations. By independently manipulating "human subjects", "actions", and "backgrounds", we generate a plethora of realistic videos. Our unique technique for creating human skeleton trajectories, combined with our video frame generation method, allows us to bypass resource-intensive data collection and create virtually unlimited training data for action recognition. Our tests on two small human action recognition datasets reveal significant improvements in the performance of existing action recognition methodologies.

4.2 Introduction

Following considerable advancements in face detection, face recognition, and object detection – technologies that have permeated our daily lives – computer vision researchers are now directing their efforts towards understanding videos, a challenge of higher dimensional complexity. While the utilization of advanced machine learning techniques has played a crucial role in achieving these remarkable accomplishments, it is important to acknowledge that their effectiveness heavily relies on the availability of sufficient training data. Large training datasets large datasets, like those encompassing millions of object and animal photographs [112], hundreds of thousands of faces [98], or millions of scenes [123], provided sufficient data for training complex neural networks. However, when confined to small datasets manually gathered by researchers, these outcomes cannot be replicated.

Compiling video datasets, especially those centered on human actions, poses further complications. Typical methods of creating a human action dataset include: (1) recording subjects performing a series of actions; (2) curating and labeling pre-existing online videos. E.g. UCF 101 [197], con-

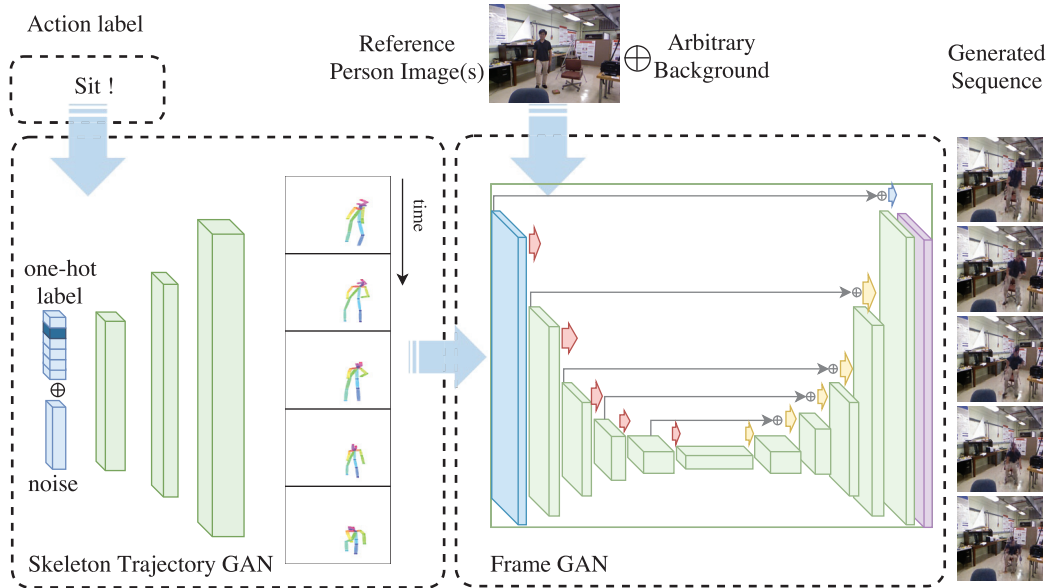


Figure 4.1: Our algorithm takes as input an action label, a set of reference images and an arbitrary background. The output is a generated video of the person in the reference image performing a given action. We approached this problem in two stages. Firstly (left side) a generative model trained on a small *labeled* dataset of skeleton trajectories of human actions, generates a sequence of human skeletons conditioned on the action label. Secondly (right side), another generative mode trained on an *unlabeled* set of human action videos, generates a sequence of photo-realistic frames conditioned on the given background, generated skeletons, and the person’s appearance given in the reference frames. This produces an arbitrary number of human action videos.

taining 101 actions of thousands of online clips, Hollywood2 [134] containing 12 actions in around 3 thousands clip extracted from movies, and the kinetics [97] including 400 actions from hundreds of thousands of YouTube videos; and (3) synthesizing 3D videos [61]. The first method suffers from scalability issues due to the number of subjects and capturing environment limitations. The second approach, while useful for algorithm benchmarking, may not align with real-world action recognition tasks like security surveillance or health monitoring, as each scenario carries unique settings and action sets. The third and more recent approach, despite showing promise, requires MoCap data and is costly.

In this work, we introduce a novel method of segmenting an action video clip into action, subject, and context. We demonstrate the feasibility of manipulating each segment independently and reassembling them into new clips using our proposed video generation model. Actions are represented by a series of skeletons, the context is a still image or video clip, and the subject is depicted by random images of the same individual.

Our technique enables the alteration of an action, subject, or context using arbitrary video clips, paving the way for the generation of unique action clips. This is particularly advantageous for action recognition models that require extensive datasets to enhance their accuracy. By utilizing a large set

of unlabeled data and a small pool of labeled data, we can synthesize a realistic training dataset for deep learning models.

To quantitatively evaluate our data generation technique, we applied it to UT Kinects [237], a human action dataset comprised of 200 video clips featuring 10 actions. We created new types of video clips by adding new subjects or actions or by expanding the current actions and subjects. Our results demonstrated that our generated data, used alongside existing data, could improve the performance of high-performing video representation networks (e.g. I3D [11] and C3D [212]) on action recognition tasks. To further demonstrate the efficacy of our algorithm, we extended its application to the SUB-Interact [249] datasets, which consist of videos showcasing interactions between two individuals.

The outline of this chapter is as follows. In section 4.3 we've described related works in action recognition, data augmentation and video generative model. Section 4.4 introduces our video generation methods as well as skeleton trajectory generation methods with samples and use cases. In section 4.5, we've discussed the datasets and action recognition methods used to evaluate our work. In section 4.6 we've presented the extensive experimental data backing our claims. Finally we conclude in section 4.8.

4.3 Related Works

4.3.1 Action Recognition

Human action recognition has drawn attention for some time. Before deep learning era of computer vision, many researchers tried to inflate successful 2D features or descriptors in order to solve this problem such as 3D SIFT [182], 3D bag of features [118] or dense trajectories [233]. A recent report [149] provides a comprehensive survey of these types of algorithms.

In this chapter, we will focus on deep learning networks due to their significant outperformance over traditional approaches. The area of video representation networks is particularly challenging and has not seen as many satisfactory advances as image representation network architecture. Various strategies have been employed to tackle this issue.

Some have utilized 2D (image-based) convolution layers [35, 248], while others have adopted 3D (video-based) kernels [88, 212, 11]. Regarding network inputs, some systems use solely RGB video [212], while others incorporate optical flow as an additional input [46, 11].

As for information propagation across frames, different methods are applied, including Long Short-Term Memory (LSTM)[35, 248] and feature aggregation[95]. These various approaches to video representation in deep learning networks highlight the diverse ways researchers are tackling this complex issue.

Data Augmentation Using synthetic data or data warping for training classifiers has been proven effective [112, 256, 193]. Sato *et al.* [179] proposes a method for training a neural network classifier using augmented data. Wong *et al.* [236] thoroughly investigated the benefits of data augmentation for classification tasks. In action recognition tasks, data is usually very limited, since

collecting and annotating videos is difficult. Although one can use our algorithm for data augmentation by generating videos varying in background, human appearance, and type of actions, this is not the purpose of our work. Unlike data augmentation that is limited to manipulating data, our method is capable of generating new data with new content and visual features.

4.3.2 Video Generative Models

Video generation has posed as a challenge for a number of years. The early work in the field focused on generating texture [36, 206, 235]. In recent years with the success of generative models in image generation such as GANs [66], VAEs [108, 161], Plug&Play Generative Networks [143], Moment Matching Networks [121], and PixelCNNs [221], a new window of opportunity has opened towards generating videos using generative models. In this chapter, we use GANs to generate human skeleton trajectories and realistic video sequences. GAN consists of a discriminator and a generator, trained in a 2-player zero-sum game. Although GANs have shown promising results on image generation [32, 151, 251, 126, 125], they have proven to be difficult to train. To address this issue, Arjovsky *et al.* [4] proposed Wasserstein GAN to combat mode collapse with more stability. Salimans *et al.* [178] introduced several tricks for training GANs. Karras *et al.* [96] proposed a novel method for training GANs through progressively adding new layers. Ronneberger *et al.* [164] proposed U-Net, a convolutional network for segmentation.

GANs have previously been used for video generation. There are two lines of work in video generation. First is video prediction where given the first few frames of a video, the goal is to predict the future frames. Several papers focus on producing pixel values conditioned on the past observed frames [243, 198, 144, 135, 93, 242, 226]. Another group of papers aimed at reordering the pixels from the previous frames to generate the new ones [220, 50].

In the second line of work, the goal is to generate a sequence of video frames conditioned on label, single frame, etc. Early attempts assumed video clips to be fixed length and embedded in a latent space [229, 176]. Tulyakov *et al.* [213] proposed to decompose motion from content and generate videos using a recurrent neural net. Our work is different from [213] where their model learns motion and content in the same network whereas we separated them completely. Furthermore, [213] is not capable of generating complex human motions. Also filling gaps in the background initially blocked by the person in the input video is a difficult task for this method. On the other hand, our method handles these challenges by completely separating appearance, background, and motion. Our work is somewhat similar to [231], which does video forecasting using pose estimation, by modeling the movement of human using a VAE and then using a GAN to predict the pixel value of the future frames.

Our work lies in the "video generation" category where we focus on employing video generation techniques to generate human action videos. In our proposed method we completely separate background, skeleton motion, and appearance, allowing us to model frame generation and skeleton trajectory independently. So, one would require labeled data and the other can benefit from unlimited unlabeled human action videos available on internet, respectively.

4.4 Method

We define problem as follows; given an action label l a small set of reference images $I = \{I_1, \dots, I_k\}$ each containing a human subject from which a sequence of video frames is generated featuring a human with the same appearance as the human in the reference image set I performing an action l . Modeling the (human/camera) motion and generating photo-realistic video frames may be challenging but knowing the location/motion of human skeletons in each frame would simplify it. Hence, we subdivided the problem into two simpler tasks (inspired by [213, 226]).

- The first task comprised of the reference images I , background image B , and a sequence of target skeletons $S = [S_1, S_2, \dots, S_n]$ employed to render photo-realistic video frames of the person in I moving according to S on background.
- The second task produced the target skeleton sequences for the first part. In another words, given action label l , a sequence of skeletons of a random person performing action l was generated.

By combining the two tasks, we created a novel algorithm that can generate arbitrary number of human action videos with varying backgrounds, human appearances, actions, and ways each action is performed.

4.4.1 Video Generation from Skeleton and Reference Appearance

In this section, we explain our algorithm used to generate a video sequence of a person based on given appearance (I) and a series of target skeletons (S) in an arbitrary background(B). In our proposed model, we use GAN conditioned on the appearance, the target skeleton, and the background. Our proposed generator network works in a frame-by-frame fashion, where each frame is generated independently from others. We have tried using LSTMs and RNNs to take into account smoothness of the videos. However, our experiments show frames that are generated separately are sharper as RNNs/LSTMS may introduce blurriness to the generated frames.

Generator Input. Our generator network needs a reference image of the person in order to generate images of the same person with arbitrary poses/backgrounds. However, one reference image may not have all the appearance information due to occlusions in some poses (e.g. face is not visible when the person is not facing the camera). To overcome this issue to some extent, we provided multiple reference images of the person to the network. In both training and testing, these images were selected completely at random, so that network would be responsible for choosing the right pieces of appearance features from the set of input images. These images could be selected with a better heuristic to produce better results though this is not in the scope of this work.

The reference images were pre-processed before incorporation into the network. First we extracted the human skeleton from each reference image I_i (using [10]), then used an offline transform to map the RGB pixel values of each skeleton part from the image to the target skeleton.

Also, a binary mask of where the transformed skeleton is located was created. All these images, $I^t = \{I_1^t, \dots, I_k^t\}$, along with the background, B , and the target skeleton, S_i were stacked.

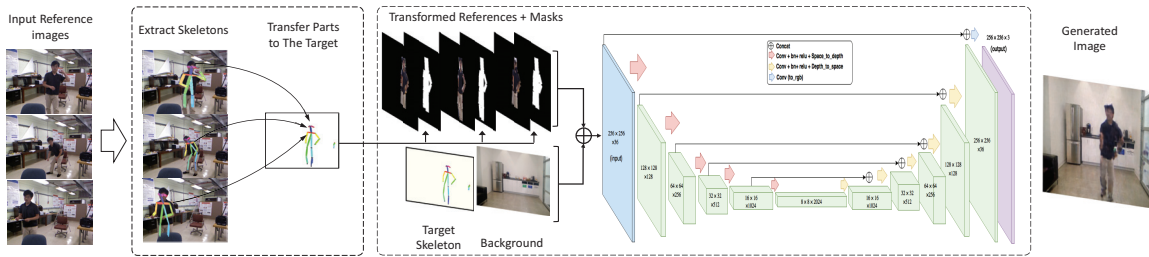
Conditional GAN. Inspired by pix2pix [84], we used a U-net style conditional GAN. The generator $G(C)$, is conditioned on the set of transformed images and corresponding masks, along with the background and target skeleton. The generator, G , maps $C = \{I_1^t, \dots, I_k^t, B, S_i\}$ to the target frame Y , such that it fools the discriminator, $D(C, Y)$. The discriminator, $D(C, Y)$, on the other hand is trained to discriminate between real images and the fake images generated by G . The architecture of the discriminator is illustrated in Figure 4.2b. The pipeline and architecture of the generator G is illustrated in Figure 4.2a. Figure 4.5a illustrates some of the results.

The objective function of GAN is expressed as:

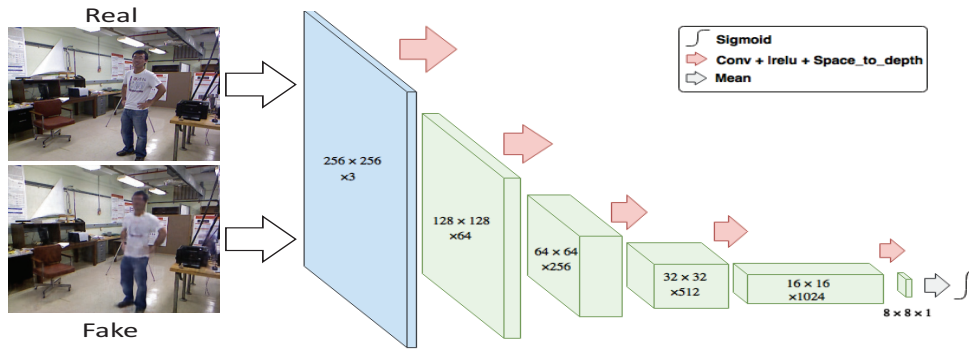
$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}_{c, y \sim P_{data}(c, y)}[\log D(c, y)] + \mathbb{E}_{c \sim P_{data}(c), z \sim P_z(z)}[1 - \log D(c, G(c, z))]$$

Following [84] we added an $L1$ loss to the objective function, which resulted in sharper generated frames.

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{c, y \sim P_{data}(c, y), z \sim P_z(z)}[||y - G(c, z)||]$$



(a) Architecture of the "generator", G , which takes as input background, target skeleton, and the transformed reference images to the target skeleton along with their masks.



(b) Architecture of the "discriminator", D , which takes as input generated image or ground truth and outputs "fake" or "real".

Figure 4.2: Network Architecture

In initial experiments, we noticed that using only $L1$ loss and GAN loss is not enough as the output background would be sharp but the region that the target person is supposed to be was blurry. Subsequently, we introduced a "Regional $L1$ loss" with a larger weight as following,

$$\mathcal{L}_R(G) = \mathbb{E}_{c,y \sim P_{data}(c,y), z \sim P_z(z)} [||\text{masked}(y) - \text{masked}(G(c, z))||]$$

where "masked" masks out the region where the person was located. This mask was generated based on the target skeleton, S_i , using morphological functions (erode, etc.).

Our final objective is as follows:

$$\mathcal{L}(G, D) = \mathcal{L}_{GAN}(G, D) + \lambda \mathcal{L}_{L1}(G) + \beta \mathcal{L}_R(G)$$

where λ and β are weights of $L1$ and R regional losses (in our experiments $\beta > \lambda$). and the goal is to solve the following optimization problem.

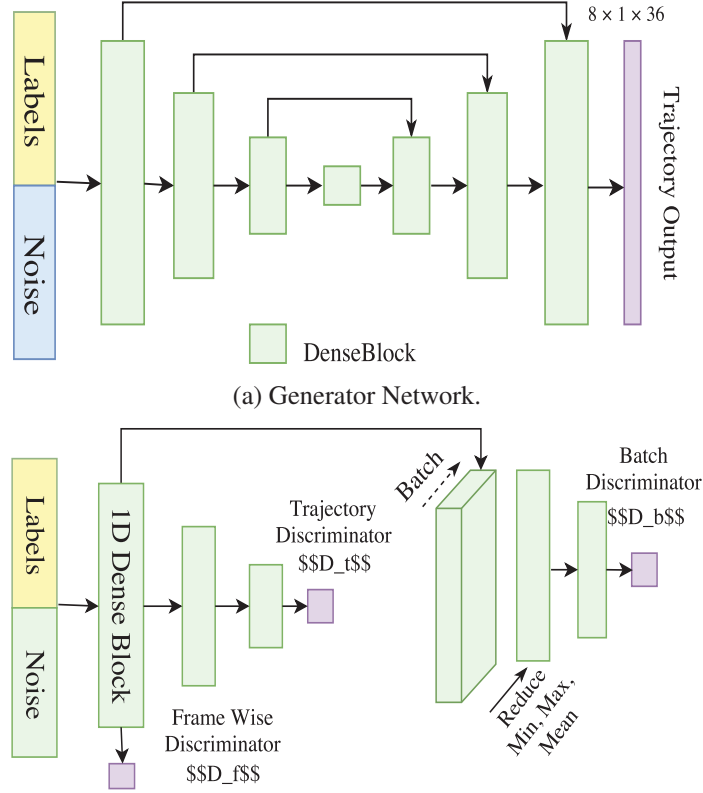
$$G^* = \arg \min_G \max_D \mathcal{L}(G, D) \tag{4.1}$$

Multi-person Video Generation In a nutshell, our algorithm merges transformed images of a person on an arbitrary pose with an arbitrary background in a natural photo-realistic way. We managed to go beyond simple one person human action videos and extended our method to *multi-person interaction videos* as well. For this purpose, we trained our model on a two person interaction dataset [249]. The only difference with single frame generation process is that in the pre-processing phase, for each person in the input reference image, we needed to know the corresponding skeleton in the target frame, we then transformed each person's body parts to his/her own body parts in the target skeleton. There are some challenges in this task such as occlusions in certain interactions (e.g. passing by, hugging, etc.). The dataset that we used contains these occlusions to some extent. Our method is able to handle relatively well some simple occlusions that occur in such interactions. We acknowledge that there is room for improvement in this area, but that would not fit in the scope of this work. Figure 4.5b illustrates some of the generated videos.

4.4.2 Skeleton Trajectory Generation

In the previous section, we explained how we designed a method that enables us to generate videos of an arbitrary person in any background based on any given sequence of skeletons. Although number of backgrounds and persons are unlimited, the number of labeled skeleton sequences are limited to the ones in the existing datasets. We propose a novel solution to this problem; using a generative model to learn the distribution of skeleton sequences conditioned on the action labels. This allows us to generate as many skeleton sequences as needed for the actions in the dataset. Figure 4.4 shows a few sample generated skeleton sequences.

We used small datasets for training our model. However, due to the nature of the problem and the limited amount of data, generating long sequences of natural looking skeletons proved challenging. Thus we aimed at generating relatively short fixed-length sequences. Having said that, training GAN in such way is still prone to problems such as mode collapse, divergence, etc. In designing the



(b) Trajectory Discriminator Network. The discriminator is the sum of three discriminators illustrated in this figure: $D = D_f + D_t + D_b$.

Figure 4.3: Trajectory GAN network architecture.

generator and discriminator networks, we have taken into account these problems (e.g. introduced batch diversity in the discriminator, created multiple discriminators, etc.).

Skeleton Trajectory Representation. Each skeleton consists of 18 joints. We represented each skeleton with a 1×36 vector (a flattened version of 18×2 matrix of joints coordinates). We normalized the coordinates by dividing them by "height" and "width" of the original image.

Generator Network. We used a conditional GAN model to generate sequences of skeletal positions corresponding to different actions. Our generator has a "U" shape architecture where input consists of action label and noise, and output is a $8 \times 1 \times 36$ tensor representing a human skeleton trajectory with 8 time-steps.

Based on our results, providing a vector of random noise for each time step helps the generator to learn and generalize better. So the input noise, z , is a tensor with size $8 \times 1 \times 128$; drawn from a uniform distribution. The one-hot encoding of action label, l , is replicated and concatenated to the 3rd dimension of the z . The rest is a "U" shaped network with skip connections that maps the input (z, l) to a skeleton sequence S . Figure 4.3a illustrates the network architecture. We also used Dense-net [79] blocks in our network.

Discriminator Network. Architecture of discriminator is three-fold. The base for discriminator is 1D convolutional neural net along the time dimension. In order to allow discriminator to distinguish "human"-looking skeletons, we used sigmoid layer on top of fully-convolutional net. To discriminate "trajectory", we used set of convolutions along the time with stride 2, shrinking output to one $1 \times 1 \times C$ containing features of the whole sequence. To prevent mode collapse, first we grouped fully convolutional net outputs across batch dimension. We then used min, max and mean operations across batch, and provided these statistical information to the discriminator. This method seems to provide enough information about distribution of values across batch and allows to change batch size during training. For detailed discriminator architecture see Figure 4.3b.

Our objective function is:

$$\mathcal{L}_T(G, D) = \mathbb{E}_{l, s \sim P_{data}(l, s)}[\log D(l, s)] \\ + \mathbb{E}_{l \sim P_{data}(l), z \sim P_z(z)}[1 - \log D(l, G(l, z))]$$

where l and s are action label and skeleton trajectories, respectively. We aim to solve the following:

$$G^* = \arg \min_G \max_D \mathcal{L}_T(G, D)$$

In this work, we have shown that generative models can be adopted to learn human skeleton trajectories. We trained a Conditional GAN on a very small dataset (200 sequences) and managed to generate natural looking skeleton trajectories conditioned on action labels. This can be used to generate a variety of human action sequences that don't exist in the dataset. However, our work is limited to a fixed number of frames. Thus for future work, we'll work to improve our method so that it'll accommodate longer sequences varying in length. We also explained that in addition to

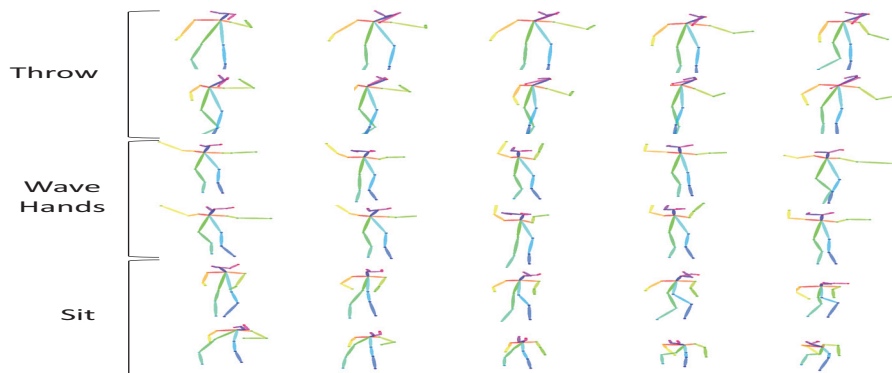


Figure 4.4: Samples of generated skeleton sequences, conditioned on action label (e.g. throwing, hand waving, sitting).

the generated skeletons, we can also use real skeleton sequences from other sources (other datasets, current dataset but different subjects) to largely expand existing datasets.

4.5 Datasets and Action Recognition Methods

4.5.1 datasets

In this chapter, we've claimed to expand small amount of action videos by addition of new generated videos. We targeted smaller action recognition datasets and expanded them to meet the large data load requirements of recent action recognition algorithms such as UCF 101 [197], the kinetics [97] or NTU RGB+D [188]. This eliminates the need for time and cost inefficient data acquisition processes.

UT Kinects [237]: One of the datasets widely used in our experiments is UT Kinects which includes 10 action labels: Walk, Sit-down, Stand-up, Trow, Push, Pull, Wave-hand, Carry and Clap-hand. There are 10 subjects that perform each of these action twice in front of a rig of RGB camera and Kinect. Therefore in total they are 200 action clips of RGB and depth though depth is ignored. All videos are taken in office environment with similar lighting condition and the position of the camera is fixed.

For the training setup, 2 random subjects were left out (20%, used for testing) and the experiments were carried out using 80% of the subjects. The reported results are the average of six individual runs. The 6 train/test runs are constant throughout our experiment.

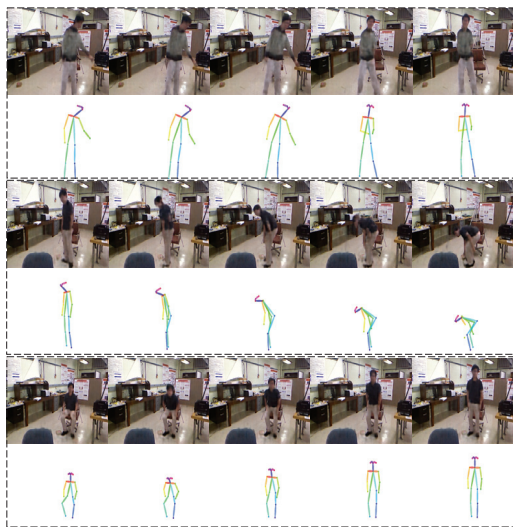
SUB Interact [249]: Since our methods work with multiple human subjects in a scene, we picked SUB Interact. It is a kinect captured human activity recognition dataset depicting two person interaction. It contains 294 sequences of 8 classes (Kicking, Punching, Hugging, Shaking-hand, Approaching, departing and Exchanging objects) with subject independent 5-fold cross validation. The original data includes RGB, depth and skeleton but we only use RGB for our purpose. We used a 5-fold cross validation throughout our experiments and reported the average accuracy.

KTH [181]: KTH action recognition dataset was commonly used at the early stage of action recognition. It includes 600 low resolution clips of 6 actions: Walk, Wave-hand, Clap-hand, Jogging, running and boxing which are divided in train, test and validation. The first three action labels are shared with UT dataset while the last three are new. We used this dataset to add new action to UT dataset and for cross dataset evaluation.

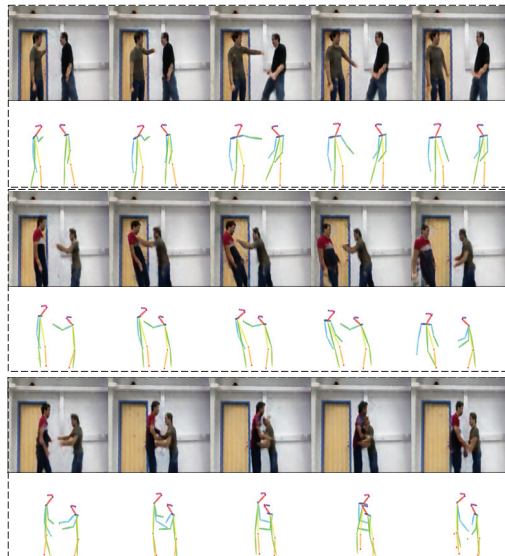
4.5.2 Action Recognition Methods

We used the following deep learning networks which have previously shown decent performance on recent action recognition datasets.

Convolutional 3D (C3D) [212]: is a simple and efficient 3-dimensional ConvNet for spatiotemporal feature which shows decent performance on video processing benchmarks such as action recognition in conjunction with large amount of training data. We used their proposed network with 8 convolutional layers, 5 pooling layers and 2 fully connected layers with 16-frames of 112×112



(a) UT dataset. Subjects from the same dataset.



(b) SBU dataset. None of the subjects exist in this dataset.

Figure 4.5: Generated images on two different datasets.

RGB input. They released a network pre-trained on UCF Sport [197] which we used for our experiments aimed at training from scratch, denoted as C3D(p) vs. C3D(s). Unfortunately we can not couldn't converge the C3D when we trained from scratch on UT dataset but it converged successfully on SUB.

Inflated 3D ConvNets (I3D) [11] : is a more complex model which has recently been proposed as the state-of-the-art for action recognition task. It builds upon Inception-v1 [83], but inflates their filters and pooling kernels into 3D. It is a two-stream network which uses both RGB and optical flow input with 224×224 inputs. We only used RGB for simplicity. They released a network pre-trained on ImgeNet [28] followed by the Kinetics [97]. We used this for our experiments aimed at training from scratch, denoted as I3D(p) vs. I3D(s).

We use data augmentation by translation and clipping as mentioned in [11] for all experiments. For training, we only used the original clips as test, making sure there was no generated clips with skeletons or subjects (subject pair) from test data in each run.

4.6 Experiments

So far, we have introduced our video generation method which enable us to generate new action clips for the action recognition training process. In this section, we show different scenarios for generating new data and running experiments for each to see if adding the generated data to a training process can improve the accuracy of the action recognizer. We applied our proposed video generation models to all the experiments using skeletons. The skeletons were trained using data from UT and SBU datasets as well as 41 un-annotated clips (between 10 to 30 seconds) that we captured

from our colleagues. For future works, we will train our model again using a large amount of data from web. But the time being, we are satisfied with the current model as higher resolution for action recognition is currently unnecessary. Our technique for generating new action video clips has the capacity of running experiments with numerous varying settings. Here, we show five experiments which may be quantitatively evaluated.

4.6.1 Generated Trajectory

The first experiments is a combination of our proposed video generation technique and skeleton trajectory generation. We generated around 200 random skeleton trajectories from action labels in UT dataset using the method mentioned in §4.4.2. Each of these skeleton trajectories generated a video by proposed video generation applied to a person in UT dataset, meaning our new dataset is doubled with half of it being the generated data. We then trained our model by I3D and C3D using training setting mentioned in §4.5.1. Table 4.1 shows about 3% improvement for I3D with and without training data as well as significant improvement (by 15%) for C3D network which is less complex.

Method	Org.	Org. + Gen.
I3D(s)	64.58%	67.50%
I3D(p)	86.25%	89.17%
C3D(p)	55.83%	70.83%

Table 4.1: Action recognition on UT dataset using original data compared to generated from scratch data with proposed method in §4.4.1 and §4.4.2

4.6.2 New Subjects

A common method to augment a video dataset involves recruiting new individuals to perform a series of actions on camera. Diversification in body shape, clothing, and behavior [6] can significantly improve the generalizability of machine learning (ML) methods. In this experiment, our goal was to virtually add new subjects to the dataset. We collected small, unannotated clips from ten distinct individuals and input them as new subjects into our proposed video generation method. For the UT dataset, each subject was replaced by a new one across all of their actions, similar to adding ten new subjects to the UT dataset. The same process was conducted with the SUB dataset to double its size, the only difference being that each pair was replaced with a new subject pair. Figure 4.5b displays some of the new subjects along with their generated action videos from the SBU dataset. The results are presented in Table 4.2.

4.6.3 New Actions

In real computer vision problems, one might decide to add a new label class after the data collection process has been done. Adding a new label action to a valid dataset could cost the same as gathering

	UT		SBU	
	Org.	Exp.	Org	Exp.
I3D(s)	64.58%	67.08%	86.48%	91.23%
I3D(p)	86.25%	89.17%	97.30%	98.65%
C3D(s)	-	-	83.52%	87.00%
C3D(p)	55.83%	70.43%	92.02%	96.25%

Table 4.2: Performance comparison of multiple algorithms, trained on original data and additional subjects.

a dataset from scratch as all the subjects are needed for re-acting that single action. In this experiment, we tried to introduce a new action labeled to UT dataset. As mentioned in §4.5.1, UT consists 10 action labels. We used training data from a third dataset called KTH [181] in order to generate 3 new actions, running, jogging and boxing, in addition to that of the UT. For each subject in UT dataset and each of these 3 new action, we randomly picked 5 action clips from KTH training data clips and extracted the skeleton by OpenPose [10] where in addition to input background image, we generated 150 new action clips from our dataset. We then trained a new model using I3D by pre-



Figure 4.6: The screen shot of a video generated by UTK expansion. The first row shows skeleton clips extracted from an arbitrary action. Second to fourth rows show the generated video for subjects from different clip carrying out that specific action.

trained network where in each run we used training data from original set and all the data generated for the new set of actions. Since the KTH data is grey scaled images, we randomly grey scaled both the original and the generated training clips in the training phase. For each run, we found per class accuracy for UT test set (refer to §4.5.1 for explaining UT train/test) as well as KTH test sets. Table 4.3 shows average of the per class accuracy for both test sets. We may consider KTH test results as a measure of cross dataset accuracy for walk, wave-hand and clap-hand. Our trained network on new action labels *boxing*, *running* and *jogging* achieved 72.14%, 44.44% and 63.20%, respectively. This indicates that the new actions in the dataset performed as good as the data captured by camera.

Action	UTK Test	Label	KTH Test
Walk	91.67%	Walk	67.18%
Wave-hand	100.0%	Wave-hand	58.59%
Clap-hand	91.67%	Clap-hand	28.90%
Push	33.33%	Boxing	72.14%
Pull	58.33%	Running	44.44%
Pick-up	100.0%	Jogging	63.20%
Sit-down	87.50%		
Stand-up	95.83%		
Threw	54.17%		
Carry	79.17%		

Table 4.3: Per class average accuracy for model trained by i3d using original training data from UT plus new action clip generated by our method using skeleton extracted from KTH training set.

4.6.4 dataset Expansion

So far, we’ve shown that using our proposed method we can generate video clips with any number of arbitrary action videos and subjects. In an action dataset with N subjects carrying out M distinct actions, there will be $M \times N$ video actions. when applied to our proposed method of action video generation, the N subjects and the $M \times N$ video actions will result in generation of $M \times N^2$ video actions comprising of $M \times N$ original videos while the rest is generated videos. This approach enabled us to expand UT Kinect dataset from 200 clips to 4000 clips and SUB Interact from 283 clips to 5943 using only the original dataset. We trained I3D and C3D using our expanded dataset as described in §4.5.1. Table 4.4 shows the result of this experiment.

Figures 4.6 shows a screen shot of the clips from UTK and SUB datasets. The first row shows skeleton clips extracted from an arbitrary action while rows 2-4 show the generated video for subjects from different clip performing that specific action.

4.6.5 Real World

In this section, we carried out 4 different experiments on 2 datasets for bench-marking. Although in all experiments, the generated data improved the network performance, we believe none of the

	UTK		SBU	
	Org.	Exp.	Org	Exp.
i3d(s)	64.58%	69.58%	86.48%	93.54%
i3d(p)	86.25%	90.42%	97.30%	99.13%
c3d(s)	-	-	83.52%	86.03%
c3d(p)	55.83%	71.25%	92.02%	97.41%

Table 4.4: The comparison of dataset expansion by original data for UTK and SUB dataset.

experiments show the actual strength and convenience of our proposed methods in real world scenarios. In both datasets, as well as other commonly used small datasets, the environmental setup for data acquisition such as distance from camera view [92] and light condition were kept as uniformly as possible for both test and train video clips. This would be unattainable in real life data acquisitions. A way of overcoming this obstacle would be to collect diverse sets of data for strong neural network models. We’ve previously shown that by partitioning the video to action, subject and context allows us to easily manipulate the background or change the camera view. In this experiment, We applied perspective transform on skeleton while using diverse backgrounds. Although the model trained with these data did not outperform our previous experiments, a live demo showed it to be better for unseen cases, qualitatively. Figure 4.7 illustrates an input skeleton and its perspective transform as well as the generated clip.

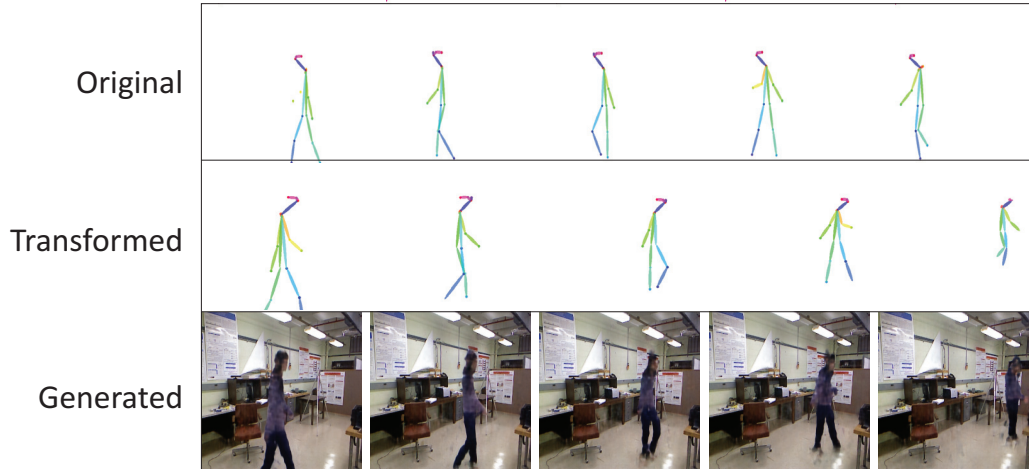


Figure 4.7: Perspective transform example.

4.7 Discussion and Future Work

One of the key contributions of this chapter was the introduction of a novel method for generating realistic human action videos. However, since the completion of this project in 2017, there have been significant advancements in image generation techniques. Specifically, diffusion models such

as DALL-E [155], Imagen [174], Stable Diffusion [163], etc. have emerged as effective tools for generating images based on natural language prompts. Given these advancements in the field of generative learning since the completion of our project, it is important to discuss the relevance and limitations of our work.

In terms of limitations, our approach was hinged on the utilization of Generative Adversarial Networks (GANs) for generating skeleton trajectories, a task for which GANs were not intrinsically designed. Training GANs for this specific application proved to be a formidable challenge, frequently leading to mode collapse [13, 209] despite rigorous hyper-parameter tuning, incorporation of batch diversity loss, and a reduction of time steps.

A potential improvement for future iterations of this work might involve the application of Variational Autoencoders (VAEs) or Diffusion models instead of GANs for skeleton generation. VAEs, although renowned for lower reconstruction accuracy compared to GANs, tend to have better mode coverage[239]. The low reconstruction accuracy for 2D coordinates of human skeleton is not expected to negatively impact the training of human action videos on the synthesized videos. Moreover, Diffusion models have proven to be proficient in generating high-resolution images, which can potentially replace the conditional GAN in our model for refining the quality of synthesized action videos.

With these technological advances since the original research, questions about the relevance of our work in 2023 naturally arise. Our techniques, although no longer novel, continue to contribute valuable insights to the field. It is critical to understand that our core contribution lies not within a novel method for generating human action videos, but within the demonstration of how data augmentation using generative learning can significantly enhance the performance of human action recognition models. As such, with continued advancement in generative learning, we anticipate an escalating improvement in the performance of action recognition models trained using synthesized videos generated by these increasingly sophisticated generative models.

At present, state-of-the-art video generation techniques are not yet adept at producing realistic, controllable human action videos, especially for longer durations with accurate labels [73, 74]. These current methods still necessitate the application of certain engineering techniques, such as the decomposition of the video into subject, action, and background. This was an approach proposed and used in our original work. Consequently, the video generation technique presented in this work remains potentially beneficial, especially if paired with state-of-the-art generative models like VAEs or Diffusion models. Therefore, we maintain that our work retains relevance in the continually evolving landscape of generative learning, particularly in the context of human action recognition models.

4.8 Summary

In this chapter, we've introduced a novel way to partition an action video clip into action, subject and context. We showed that we can manipulate each part separately, reassemble them with our

proposed video generation model into new clips and use as an input for action recognition models which require large data. We can change an action by extracting it from an arbitrary video clip, generate it through our proposed skeleton trajectory model or by applying perspective transform on existing skeleton. Additionally, we can change the subject and the context using arbitrary video clips. We demonstrated that by using synthesized videos in the training set, the accuracy of action recognition models can be significantly improved.

Chapter 5

A Robust Learning Approach to Domain Adaptive Object Detection

5.1 Abstract

Considering the inevitability of domain shift in real-world applications of object detection, we've formulated a robust learning approach to the domain adaptation problem, treating it as training with noisy labels. We introduce a robust object detection framework, which exhibits resilience to noise in bounding box class labels, locations, and size annotations. To accommodate domain shift, the model is trained on the target domain using a set of noisy object bounding boxes provided by a detection model trained solely in the source domain. Evaluation of our approach on various source/target domain pairs reveals considerable improvements in the state-of-the-art across multiple domain adaptation scenarios, as demonstrated on the SIM10K, Cityscapes, and KITTI datasets.

5.2 Introduction

Object detection lies at the core of computer vision and finds application in surveillance, medical imaging, self-driving cars, face analysis, and industrial manufacturing. Recent advances in object detection using convolutional neural networks (CNNs) have made current models fast, reliable and accurate.

However, domain adaptation remains a significant challenge in object detection. In many discriminative problems (including object detection) it is usually assumed that the distribution of instances in both train (source domain) and test (target domain) set are identical. Unfortunately, this assumption is easily violated, and domain changes in object detection arise with variations in viewpoint, background, object appearance, scene type and illumination. Further, object detection models are often deployed in environments which differ from the training environment.

Common domain adaptation approaches are based on either supervised model fine-tuning in the target domain or unsupervised cross-domain representation learning. While the former requires additional labeled instances in the target domain, the latter eliminates this requirement at the cost of two new challenges. Firstly, the source/target representations should be matched in some space (e.g.,

either in input space [263, 75] or hidden representations space [54, 214]). Secondly, a mechanism for feature matching must be defined (*e.g.* maximum mean discrepancy (MMD) [141, 132], \mathcal{H} divergence [14], or adversarial learning).

In this work, we approach domain adaptation differently, and address the problem through robust training methods. Our approach relies on the observation that, although a (primary) model trained in the source domain may have suboptimal performance in the target domain, it may nevertheless be used to detect objects in the target domain with some accuracy. The detected objects can then be used to retrain a detection model on both domains. However, because the instances detected in the target domain may be inaccurate, a robust detection framework (which accommodates these inaccuracies) must be used during retraining.

The principal benefit of this formulation is that the detection model is trained in an unsupervised manner in the target domain. Although we do not explicitly aim at matching representations between source and target domain, the detection model may implicitly achieve this because it is fed by instances from both source and target domains.

To accommodate labeling inaccuracies we adopt a probabilistic perspective and develop a robust training framework for object detection on top of Faster R-CNN [160]. We provide robustness against two types of noise: i) mistakes in object labels (*i.e.*, a bounding box is labeled as person but actually is a pole), and ii) inaccurate bounding box location and size (*i.e.*, a bounding box does not enclose the object). We formulate the robust retraining objective so that the model can alter both bounding box class labels and bounding box location/size based on its current belief of labels in the target domain. This enables the robust detection model to refine the noisy labels in the target domain.

To further improve label quality in the target domain, we introduce an auxiliary image classification model. We expect that an auxiliary classifier can improve target domain labels because it may use cues that have not been utilized by the original detection model. As examples, additional cues can be based on additional input data (*e.g.* motion or optical flow), different network architectures, or ensembles of models. We note however, that the auxiliary image classification model is only used during the retraining phase and the computational complexity of the final detector is preserved at test time.

The contributions of this chapter are summarized as follows: i) We provide the first (to the best of our knowledge) formulation of domain adaptation in object detection as robust learning. ii) We propose a novel robust object detection framework that considers noise in training data on both object labels and locations. We use Faster R-CNN[160] as our base object detector, but our general framework, theoretically, could be adapted to other detectors (*e.g.* SSD [128] and YOLO [158]) that minimize a classification loss and regress bounding boxes. iii) We use an independent classification refinement module to allow other sources of information from the target domain (*e.g.* motion, geometry, background information) to be integrated seamlessly. iv) We demonstrate that this robust framework achieves state-of-the-art on several cross-domain detection tasks.

5.3 Previous Work

Object Detection: The first approaches to object detection used a sliding window followed by a classifier based on hand-crafted features [26, 47, 227]. After advances in deep convolutional neural networks, methods such as R-CNN [63], SPPNet [69], and Fast R-CNN [62] arose which used CNNs for feature extraction and classification. Slow sliding window algorithms were replaced with faster region proposal methods such as selective search [216]. Recent object detection methods further speed bounding box detection. For example, in Faster R-CNN [160] a region proposal network (RPN) was introduced to predict refinements in the locations and sizes of predefined anchor boxes. In SSD [128], classification and bounding box prediction is performed on feature maps at different scales using anchor boxes with different aspect ratios. In YOLO [157], a regression problem on a grid is solved, where for each cell in the grid, the bounding box and the class label of the object centering at that cell is predicted. Newer extensions are found in [254, 158, 25]. A comprehensive comparison of methods is reported in [80]. The goal of this chapter is to increase the accuracy of an object detector in a new domain regardless of the speed. Consequently, we base our improvements on Faster R-CNN, a slower, but accurate detector.¹

Domain Adaptation: was initially studied for image classification and the majority of the domain adaptation literature focuses on this problem [39, 38, 114, 67, 64, 48, 200, 130, 131, 54, 52, 60, 9, 140, 119]. Some of the methods developed in this context include cross-domain kernel learning methods such as adaptive multiple kernel learning (A-MKL) [39], domain transfer multiple kernel learning (DTMKL) [38], and geodesic flow kernel (GFK) [64]. There are a wide variety of approaches directed towards obtaining domain invariant predictors: supervised learning of non-linear transformations between domains using asymmetric metric learning [114], unsupervised learning of intermediate representations [67], alignment of target and domain subspaces using eigenvector covariances [48], alignment the second-order statistics to minimize the shift between domains [200], and covariance matrix alignment approach [234]. The rise of deep learning brought with it steps towards domain-invariant feature learning. In [130, 131] a reproducing kernel Hilbert embedding of the hidden features in the network is learned and mean-embedding matching is performed for both domain distributions. In [54, 52] an adversarial loss along with a domain classifier is trained to learn features that are discriminative and domain invariant.

There is less work in domain adaptation for object detection. Domain adaptation methods for non-image classification tasks include [55] for fine-grained recognition, [15, 76, 258, 230] for semantic segmentation, [103] for dataset generation, and [136] for finding out of distribution data in active learning. For object detection itself, [241] used an adaptive SVM to reduce the domain shift, [153] performed subspace alignment on the features extracted from R-CNN, and [14] used Faster RCNN as baseline and took an adversarial approach (similar to [52]) to learn domain invari-

¹Our adoption of faster R-CNN also allows for direct comparison with the state-of-the-art [14].

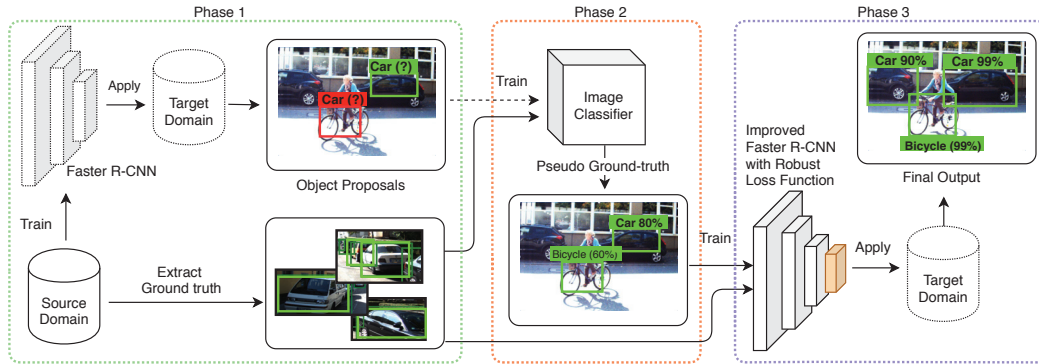


Figure 5.1: The robust learning approach consists of three phases. In phase 1, a detection module is trained using labeled data in the source domain. This detector is then used to generate noisy annotations for images in the target domain. In phase 2, the annotations assigned in phase 1 are refined using a classification module. Finally, in phase 3, the detector is retrained using the original labeled data and the refined machine-generated annotations in the target domain. Retraining is formulated to account for the possibility of mislabeling.

ant features jointly on target and source domains. We take a fundamentally different approach by reformulating the problem as noisy labeling. We design a robust-to-noise training scheme for object detection which is trained on noisy bounding boxes and labels acquired from the target domain as pseudo-ground-truth.

Noisy Labeling: Previous work on robust learning has focused on image classification where there are few and disjoint classes. Early work used instance-independent noise models, where each class is confused with other classes independent of the instance content [142, 139, 145, 199, 259, 247]. Recently, the literature has shifted towards instance-specific label noise prediction [238, 137, 217, 218, 219, 223, 207, 90, 27, 159]. To the best of our knowledge, ours is the first proposal for an object detection model that is robust to label noise.

5.4 Method

Following the common formulation for domain adaptation, we represent the training data space as the source domain (\mathcal{S}) and the test data space as the target domain (\mathcal{T}). We assume that an annotated training image dataset in \mathcal{S} is supplied, but that only images in \mathcal{T} are given (*i.e.* there are no labels in \mathcal{T}). Our framework, visualized in Fig. 5.1, consists of three main phases:

1. **Object proposal mining:** A standard Faster R-CNN, trained on the source domain, is used to detect objects in the target domain. The detected objects form a proposal set in \mathcal{T} .
2. **Image classification training:** Given the images extracted from bounding boxes in \mathcal{S} , we train an image classification model that predicts the class of objects in each image. The resulting classifier is used to score the proposed bounding boxes in \mathcal{T} . This model aids in

training the robust object detection model in the next phase. The reason for introducing image classification is that i) this model may rely on representations different than those used by the phase one detection model (*e.g.*, motion features) or it may use a more sophisticated network architectures, and ii) this model can be trained in a semi-supervised fashion using labeled images in \mathcal{S} and unlabeled images in \mathcal{T} .

3. **Robust object detection training:** In this phase a robust object detection model is trained using object bounding boxes in \mathcal{S} and object proposals in \mathcal{T} (from phase one) that has been rescored using the image classification (from phase two).

We organize the detailed method description as follows. Firstly, we introduce background notation and provide a description of Faster R-CNN in Sec. 5.4.1 to define the model used in phase one. Secondly, a probabilistic view of Faster R-CNN in Sec. 5.4.2 provides a foundation for the robust object detection framework presented in Sec. 5.4.3. This defines the model used in phase three. Lastly, the image classification model used in phase two is discussed in Sec. 5.4.4.

Notation: We are given training images in \mathcal{S} along with their object bounding box labels. This training set is denoted by $\mathbf{D}_{\mathcal{S}} = \{(\mathbf{x}^{(s)}, \mathbf{y}^{(s)})\}$ where $\mathbf{x}^{(s)} \in \mathcal{S}$ represents an image, $\mathbf{y}^{(s)}$ is the corresponding bounding box label for $\mathbf{x}^{(s)}$ and s is an index. Each bounding box $\mathbf{y} = (y_c, \mathbf{y}_l)$ represents a class label by an integer, $y_c \in \mathcal{Y} = \{1, 2, \dots, C\}$, where C is the number of foreground classes, and a 4-tuple, $\mathbf{y}_l \in \mathcal{R}^4$, giving the coordinates of the top left corner, height, and width of the box. To simplify notation, we associate each image with a single bounding box.²

In the target domain, images are given without bounding box annotations. At the end of phase one, we augment this dataset with proposed bounding boxes generated by Faster R-CNN. We denote the resulting set by $\mathbf{D}_{\mathcal{T}} = \{\mathbf{x}^{(t)}, \tilde{\mathbf{y}}^{(t)}\}$ where $\mathbf{x}^{(t)} \in \mathcal{T}$ is an image, $\tilde{\mathbf{y}}^{(t)} \in \mathcal{Y}$ is the corresponding proposed bounding box and t is an index. Finally, we obtain the image classification score obtained at the end of phase two for each instance in $\mathbf{D}_{\mathcal{T}}$ from $p_{img}(y_c | \mathbf{x}, \tilde{\mathbf{y}}_l)$ which represents the probability of assigning the image cropped in the bounding box $\tilde{\mathbf{y}}_l$ in \mathbf{x} to the class $y_c \in \mathcal{Y} \cup \{0\}$ which is one of the foreground categories or background.

5.4.1 Faster R-CNN

Faster R-CNN [160] is a two-stage detector consisting of two main components: a region proposal network (RPN) that proposes regions of interests (ROI) for object detection and an ROI classifier that predicts object labels for the proposed bounding boxes. These two components share the first convolutional layers. Given an input image, the shared layers extract a feature map for the image. In the first stage, RPN predicts the probability of a set of predefined anchor boxes for being an object or background along with refinements in their sizes and locations. The anchor boxes are

²This restriction is for notational convenience only. Our implementation makes no assumptions about the number of objects in each image.

a fixed predefined set of boxes with varying positions, sizes and aspect ratios across the image. Similar to RPN, the region classifier predicts object labels for ROIs proposed by the RPN as well as refinements for the location and size of the boxes. Features passed to the classifier are obtained with a *ROI-pooling* layer. Both networks are trained jointly by minimizing a loss function:

$$\mathcal{L} = \mathcal{L}_{RPN} + \mathcal{L}_{ROI}. \quad (5.1)$$

\mathcal{L}_{RPN} and \mathcal{L}_{ROI} represent losses used for the RPN and ROI classifier. The losses consist of a cross-entropy cost measuring the mis-classification error and a regression loss quantifying the localization error. The RPN is trained to detect and localize objects without regard to their classes, and the ROI classification network is trained to classify the object labels.

5.4.2 A Probabilistic View of Faster R-CNN

In this section, we provide a probabilistic view of Faster R-CNN that will be used to define a robust loss function for noisy detection labels. The ROI classifier in Faster R-CNN generates an object classification score and object location for each proposed bounding box generated by the RPN. A classification prediction $p_{cls}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$ represents the probability of a categorical random variable taking one of the disjoint $C + 1$ classes (*i.e.*, foreground classes plus background). This classification distribution is modeled using a softmax activation. Similarly, we model the location prediction $p_{loc}(\mathbf{y}_l|\mathbf{x}, \tilde{\mathbf{y}}_l) = \mathcal{N}(\mathbf{y}_l; \bar{\mathbf{y}}_l, \sigma\mathbf{I})$ with a multivariate Normal distribution³ with mean $\bar{\mathbf{y}}_l$ and constant diagonal covariance matrix $\sigma\mathbf{I}$. In practice, only $\bar{\mathbf{y}}_l$ is generated by the ROI classifier which is used to localize the object.

5.4.3 Robust Faster R-CNN

To gain robustness against detection noise on both the label (y_c) and the box location/size (\mathbf{y}_l), we develop a refinement mechanism that corrects mistakes in both class and box location/size annotations. The phase three detection model is trained using these refined annotations.

If the training annotations are assumed to be noise-free then both p_{cls} and p_{loc} are used to define the maximum-likelihood loss functions in Eq. 5.1. In the presence of noisy labels, $\arg \max p_{cls}$ and $\arg \max p_{loc}$ may disagree with the noisy labels but nevertheless correctly identify the true class or location of an object. Additionally, we also have access to the image classification model p_{img} from phase 2 that may be more accurate in predicting class labels for proposed bounding boxes in \mathcal{T} since it is trained using information sources different from the primary detection model. The question then is how to combine p_{cls} , p_{loc} from Faster R-CNN and p_{img} from the image model to get the best prediction for the class and location of an object?

³This assumption follows naturally if the L_2 -norm is used for the localization error in Eq. 5.1. In practice however, a combination of L_2 and L_1 norms are used which do not correspond to a simple probabilistic output.

Vahdat [217] has proposed a regularized EM algorithm for robust training of image classification models. Inspired by this approach, we develop two mechanisms for correcting classification and localization errors, based on the assumption that when training a classification model on noisy labeled instances, the distribution over true labels should be close to both the distributions generated by the underlying classification model and an auxiliary distribution obtained from other sources. Since the accuracy of the learned classification model improves during training, the weighting of these information sources should shift during training.

Classification Error Correction: We seek a distribution, $q(y_c)$, which is close to both the classification model of Faster R-CNN and the image classification model p_{img} , that is trained in phase two. We propose the following optimization objective for inferring $q(y_c)$

$$\min_q \text{KL}(q(y_c)||p_{cls}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)) + \alpha \text{KL}(q(y_c)||p_{img}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)). \quad (5.2)$$

KL denotes the Kullback-Leibler divergence and $\alpha > 0$ balances the trade-off between two terms. With large values of α , q favors the image classification model (p_{img}) over Faster R-CNN predictions (p_{cls}), and with smaller α , q favors p_{cls} . Over the course of training, α can be changed to set a reasonable balance between the two distributions.

The following result provides a closed-form solution to the optimization problem in Eq. 5.2:

Theorem 1. *Given two probability distributions $p_1(z)$ and $p_2(z)$ defined for the random variable z and positive scalar α , the closed-form minimizer of*

$$\min_q \text{KL}(q(z)||p_1(z)) + \alpha \text{KL}(q(z)||p_2(z))$$

is given by:

$$q(z) \propto (p_1(z)p_2^\alpha(z))^{\frac{1}{\alpha+1}} \quad (5.3)$$

Proof. Here, we prove the theorem for a continuous random variable defined in domain Ω .

$$\begin{aligned} \min_q \quad & \text{KL}(q(z)||p_1(z)) + \alpha \text{KL}(q(z)||p_2(z)) \\ &= \int_{\Omega} q(z) \log \frac{q(z)}{p_1(z)} dz + \alpha \int_{\Omega} q(z) \log \frac{q(z)}{p_2(z)} dz \\ &= (\alpha + 1) \int_{\Omega} q(z) \log \frac{q(z)}{[p_1(z)p_2^\alpha(z)]^{\frac{1}{\alpha+1}}} dz \\ &= (\alpha + 1) \text{KL}(q(z) || \frac{1}{Z} [p_1(z)p_2^\alpha(z)]^{\frac{1}{\alpha+1}}) + C \end{aligned}$$

where Z is the normalization for $(p_1(z)p_2^\alpha(z))^{\frac{1}{\alpha+1}}$ and C is a constant independent of q . The final KL is minimized when Eq. 5.3 holds. \square

Using Theorem. 1, the solution to Eq. 5.2 is obtained as the weighted geometric mean of the two distributions:

$$q(y_c) \propto (p_{cls}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l) p_{img}^\alpha(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l))^{\frac{1}{\alpha+1}}. \quad (5.4)$$

Since both $p_{cls}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$ and $p_{img}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$ are categorical distributions (with softmax activation), $q(y_c)$ is also a (softmax) categorical distribution whose parameters are obtained as the weighted mean of the logits generated by p_{cls} and p_{img} , i.e., $\sigma((\mathbf{l}_{cls} + \alpha \mathbf{l}_{img}) / (1 + \alpha))$ where σ is the softmax and \mathbf{l}_{cls} and \mathbf{l}_{img} are the corresponding logits. Setting $\alpha = \infty$ in Eq. 5.4 sets $q(y_c)$ to $p_{img}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$ while $\alpha = 0$ sets $q(y_c)$ to $p_{cls}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$. During training we reduce α from large to smaller values. Intuitively, at the beginning of the training, $p_{cls}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$ is inaccurate and provides a poor estimation of the true class labels, therefore by setting α to a large value we guide $q(y_c)$ to rely on $p_{img}(y_c|\mathbf{x}, \tilde{\mathbf{y}}_l)$ more than p_{cls} . By decreasing α throughout training, q will rely on both p_{cls} and p_{img} to form a distribution over true class labels.

Bounding Box Refinement: Eq. 5.4 refines the classification labels for the proposal bounding boxes in the target domain. Here, we provide a similar method for correcting the errors in location and size. Recall that Faster R-CNN’s location predictions for the proposal bounding boxes can be thought as a Normally distributed $\mathcal{N}(\mathbf{y}_l; \tilde{\mathbf{y}}_l, \sigma \mathbf{I})$ with mean $\tilde{\mathbf{y}}_l$ and constant diagonal covariance matrix $\sigma \mathbf{I}$. We let $p_{init}(\mathbf{y}_l|\mathbf{x}, \tilde{\mathbf{y}}_l) := \mathcal{N}(\mathbf{y}_l; \tilde{\mathbf{y}}_l, \sigma \mathbf{I})$ denote the initial detection for image \mathbf{x} . At each iteration Faster R-CNN predicts a location for object using $p_{loc}(\mathbf{y}_l|\mathbf{x}, \tilde{\mathbf{y}}_l) = \mathcal{N}(\mathbf{y}_l; \tilde{\mathbf{y}}_l, \sigma \mathbf{I})$ for image \mathbf{x} and the proposal $\tilde{\mathbf{y}}_l$. We use the following objective function for inferring a distribution q over true object locations:

$$\min_q \text{KL}(q(\mathbf{y}_l) || p_{loc}(\mathbf{y}_l|\mathbf{x}, \tilde{\mathbf{y}}_l)) + \alpha \text{KL}(q(\mathbf{y}_l) || p_{init}(\mathbf{y}_l|\mathbf{x}, \tilde{\mathbf{y}}_l)) \quad (5.5)$$

As with Eq. 5.2, the solution to Eq. 5.5 is the weighted geometric mean of the two distributions.

Theorem 2. *Given two multivariate Normal distributions $p_1(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_1, \boldsymbol{\Sigma})$ and $p_2(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_2, \boldsymbol{\Sigma})$ with common covariance matrix $\boldsymbol{\Sigma}$ defined for the random variable \mathbf{z} and a positive scalar α , the weighted geometric mean $q(\mathbf{z}) \propto (p_1(\mathbf{z}) p_2^\alpha(\mathbf{z}))^{\frac{1}{\alpha+1}}$ is also Normal with mean $(\boldsymbol{\mu}_1 + \alpha \boldsymbol{\mu}_2) / (\alpha + 1)$ and covariance matrix $\boldsymbol{\Sigma}$.*

Proof. By the definition of the Normal distribution, we have:

$$\begin{aligned} q(\mathbf{z}) &\propto (p_1(\mathbf{z}) p_2^\alpha(\mathbf{z}))^{\frac{1}{\alpha+1}} \\ &\propto e^{-\frac{1}{2} \left[\frac{1}{\alpha+1} (\mathbf{z} - \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}^{-1} (\mathbf{z} - \boldsymbol{\mu}_1) + \frac{\alpha}{\alpha+1} (\mathbf{z} - \boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}^{-1} (\mathbf{z} - \boldsymbol{\mu}_2) \right]} \\ &\propto e^{-\frac{1}{2} \left[\mathbf{z}^T \boldsymbol{\Sigma}^{-1} \mathbf{z} - 2 \mathbf{z}^T \boldsymbol{\Sigma}^{-1} \left(\frac{\boldsymbol{\mu}_1 + \alpha \boldsymbol{\mu}_2}{\alpha+1} \right) \right]} \\ &\propto e^{-\frac{1}{2} \left(\mathbf{z} - \left(\frac{\boldsymbol{\mu}_1 + \alpha \boldsymbol{\mu}_2}{\alpha+1} \right) \right)^T \boldsymbol{\Sigma}^{-1} \left(\mathbf{z} - \left(\frac{\boldsymbol{\mu}_1 + \alpha \boldsymbol{\mu}_2}{\alpha+1} \right) \right)} \end{aligned}$$

Hence, $q(\mathbf{z}) = \mathcal{N}(\mathbf{z}; (\boldsymbol{\mu}_1 + \alpha \boldsymbol{\mu}_2) / (\alpha + 1), \boldsymbol{\Sigma})$ □

Using Theorem. 2, the minimizer of Eq. 5.5 is:

$$q(\mathbf{y}_l) = \mathcal{N}(\mathbf{y}_l; (\bar{\mathbf{y}}_l + \alpha \tilde{\mathbf{y}}_l)/(\alpha + 1), \sigma \mathbf{I}). \quad (5.6)$$

This result gives the refined bounding box location and size as the weighted average of box location/size extracted from phase one and the current output of Faster R-CNN. Setting $\alpha = \infty$ ignores the current output of Faster R-CNN while $\alpha = 0$ uses its output as the location. At training time, we initially set α to a large value and then gradually decrease it to smaller values. In this way, at early stages of training q relies on p_{init} because it’s more accurate than the current estimation of the model, but as training progresses and p_{loc} becomes more accurate, q relies more heavily on p_{loc} .

Training Objective Function: We train a robust Faster R-CNN using $\mathbf{D}_S \cup \mathbf{D}_T$. At each mini-batch update, if an instance belongs to \mathbf{D}_S then the original loss function of Faster R-CNN is used for parameter update. If an instance belongs to \mathbf{D}_T then $q(y_c)$ in Eq. 5.4 and $q(\mathbf{y}_l)$ in Eq. 5.6 are used to refine the proposed bounding box annotations. $q(y_c)$ is used as the soft target labels in the cross entropy loss function for the mis-classification term and $(\bar{\mathbf{y}}_l + \alpha \tilde{\mathbf{y}}_l)/(\alpha + 1)$ is used as the target location for the regression term. The modifications are made only in the ROI classifier loss function because the RPN is class agnostic.

False Negative Correction: Thus far, the robust detection method only refines the object proposals generated in phase one. This allows the model to correct false positive detections, *i.e.*, instances that do not contain any foreground object or that contain an object from a class different than the predicted class. However, we would also like to correct false negative predictions, *i.e.*, positive instances of foreground classes that are not detected in phase one.

To correct false negative instances, we rely on the hard negative mining phase of Faster R-CNN. In this phase a set of hard negative instances are added as background instances to the training set. Hard negatives that come from \mathbf{D}_S are actually background images. However, the “background” instances that are extracted from \mathbf{D}_T may be false negatives of phase one and may contain foreground objects. Therefore, during training for negative samples that belong to \mathbf{D}_T , we define $p_{img}(y_c)$ to be a softened one-hot vector by setting the probability of a background to $1 - \epsilon$ and the probability of the other class labels uniformly to ϵ/C . This is used as a soft target label in the cross-entropy loss.

5.4.4 Image Classification:

Phase two of our framework uses an image classification model to re-score bounding box proposals obtained in phase one. The image classification network is trained in a semi-supervised setting on top of images cropped from both \mathbf{D}_S (clean training set) and \mathbf{D}_T (noisy labeled set). For images in \mathbf{D}_S , we use the cross-entropy loss against ground truth labels, but, for images in \mathbf{D}_T the cross-entropy loss is computed against soft labels obtained by Eq 5.2, where the weighted geometric mean

between predicted classification score and a softened one-hot annotation vector is computed. This corresponds to multiclass extension of [217] which allows the classification model to refine noisy class labels for images in $\mathcal{D}_{\mathcal{T}}$.

Note that both $\mathcal{D}_{\mathcal{S}}$ and $\mathcal{D}_{\mathcal{T}}$ have bounding boxes annotations from foreground classes (although instances in $\mathcal{D}_{\mathcal{T}}$ have noisy labels). For training the image classification models, we augment these two datasets with bounding boxes mined from areas in the image that do not have overlap with bounding boxes in $\mathcal{D}_{\mathcal{S}}$ or $\mathcal{D}_{\mathcal{T}}$.

5.5 Experiments

To compare with state-of-the-art methods we follow the experimental design of [14]. We perform three experiments on three source/target domains and use similar hyper-parameters as [14]. We use the Faster R-CNN implementation available in the object detection API [80] source code. In all the experiments, including the baselines and our method, we set the initial learning rate to 0.001 for 50,000 iterations and reduce it to 0.0001 for the next 20,000 iterations (a similar training scheme as [14]). We linearly anneal α from 100 to 0.5 for the first 50,000 iterations and keep it constant thereafter. We use InceptionV2 [205], pre-trained on ImageNet [28], as the backbone for Faster R-CNN. In one slight departure, we set aside a small portion of the training set as validation for setting hyper-parameters. InceptionV4 [204] is used for the image classification phase with initial learning rate of 3×10^{-4} that drops every 2 epochs by a factor 0.94. We set the batch size to 32 and train for 300 000 steps.

Baselines: We compare our method against the following progressively more sophisticated baselines.

- **Faster R-CNN** [160]: This is the most primitive baseline. A Faster R-CNN object detector is trained on the source domain and tested on the target domain so that the object detector is blind to the target domain.
- **Pseudo-labeling** [82]: A simplified version of our method in which Faster R-CNN is trained on the source domain to extract object proposals in the target domain, and then based on a pre-determined threshold, a subset of the object proposals are selected and used for fine-tuning Faster R-CNN. This process can be repeated. This method corresponds to the special case where $\alpha = 0$ is fixed throughout training. The original method in [82] performs a progressive adaptation, which is computationally extensive. Since our method and the previous state-of-the-art method perform only one extra fine-tuning step, we perform only one repetition for a fair comparison.
- **Feature Learning** [14]: This state-of-the-art domain adaptation method reduces the domain discrepancy by learning robust features in an adversarial manner. We follow the experimental setup used in [14].

Method	Cls-Cor	Box-R	FN-Cor	person	rider	car	truck	bus	train	motorcycle	bicycle	mAP
Faster R-CNN[160]				31.69	39.41	45.81	23.86	39.34	20.64	22.26	32.36	31.92
Pseudo-labeling[82]				31.94	39.94	47.97	25.13	39.85	27.22	25.01	34.12	33.90
Feature Learning [14]				35.81	41.63	47.36	28.49	32.41	31.18	26.53	34.26	34.70
Noisy Labeling (Ours) :	✓	✗	✗	34.82	41.89	48.93	27.68	42.53	26.72	26.65	35.76	35.62
	✓	✓	✗	35.26	42.86	50.29	27.87	42.98	25.43	25.30	35.94	36.06
	✓	✓	✓	35.10	42.15	49.17	30.07	45.25	26.97	26.85	36.03	36.45
Faster R-CNN[160] trained on target				40.63	47.05	62.50	33.12	50.43	39.44	32.57	42.43	43.52

Table 2: Quantitative results comparing our method to baselines for adapting from *Cityscapes* to *Foggy Cityscapes*. We record the average precision (AP) on the *Cityscapes* validation set. ‘‘Cls-Cor’’ represents ‘‘classification error correction’’, Box-R stands for ‘‘Bounding Box Refinement’’ component, and FN-Cor stands for ‘‘False Negative Correction’’ component of our method. The last row shows the base detector’s performance if labeled data for target domain was available.

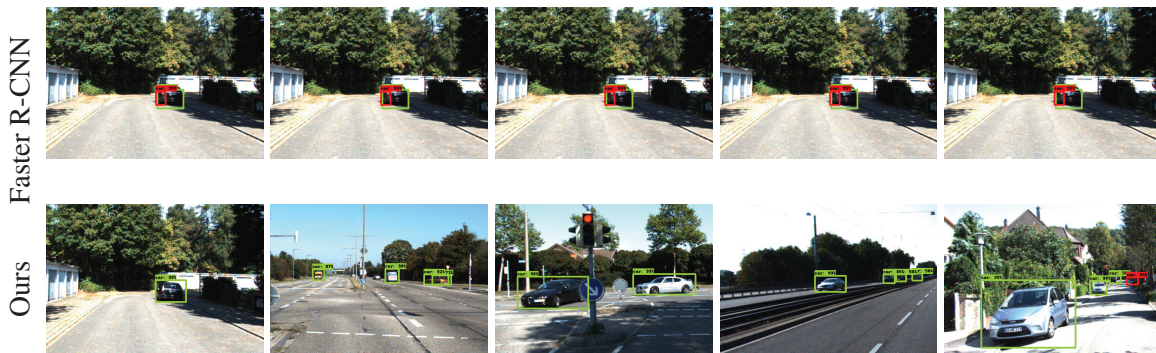


Figure 5.2: Qualitative comparison of our method with Faster R-CNN on the ‘‘*Cityscapes* \rightarrow *KITTI*’’ experiment. Each column corresponds to a particular image in the *KITTI* test set. Top and bottom images in each column illustrate the bounding boxes of the cars detected by Faster R-CNN and our method respectively. In the first two columns our method corrects several false positives. In all cases our method successfully corrected the size/location of the bounding boxes (e.g. the rooflines in the third column). In the fourth and fifth examples, our method has detected cars that Faster R-CNN has missed. Nevertheless, false positives do occur (e.g. in column five), though the probability of those specific false positives is low (53% in this example).

Datasets: Following [14] we evaluate performance on multi- and single-label object detection tasks using three different datasets. Depending on the experiment, some datasets are used as both target and source domains and some are only used as either the source or target domain.

- **SIM 10K** [91] is a simulated dataset containing 10,000 images synthesized by the Grand Theft Auto game engine. In this dataset, which simulates car driving scenes captured by a dash-cam, there are 58,701 annotated car instances with bounding boxes. We use 10% of these for validation and the remainder for training.

- *Cityscapes* [21] is a dataset⁴ of real urban scenes containing 3,475 images captured by a dash-cam, 2,975 images are used for training and the remaining 500 for validation. Following [14] we report results on the validation set because the test set doesn't have annotations. In our experiments we used the tightest bounding box of an instance segmentation mask as ground truth. There are 8 different object categories in this dataset including *person, rider, car, truck, bus, train, motorcycle and bicycle*.
- *Foggy Cityscapes* [177] is the foggy version of *Cityscapes*. The depth maps provided in *Cityscapes* are used to simulate three intensity levels of fog in [177]. In our experiments we used the fog level with highest intensity (least visibility). The same dataset split used for *Cityscapes* is used for *Foggy Cityscapes*.
- *KITTI* [57] is another real-world dataset consisting of 7,481 images of real-world traffic situations, including freeways, urban and rural areas. Following [14] we used the whole dataset for both training, when it is used as source, and test, when it is used as target.

5.5.1 Adapting synthetic data to real world

In this experiment, the detector is trained on synthetic data generated using computer simulations and the model is adapted to real world examples. This is an important use case as it circumvents the lack of annotated training data common to many applications (*e.g.* autonomous driving). The source domain is *SIM 10K* and the target domain is *Cityscapes* dataset (denoted by “*SIM 10K* → *Cityscapes*”). We use the validation set of *Cityscapes* for evaluating the results. We only train the detector on annotated *cars* because *cars* is the only object common to both *SIM 10K* and *Cityscapes*.

<i>SIM 10K</i> → <i>Cityscapes</i>				
Method	Cls-Cor	Box-R	FN-Cor	AP
Faster R-CNN[160]				31.08
Pseudo-labeling[82]				39.05
Feature Learning [14]				40.10
Noisy Labeling (Ours) :	✓	✗	✗	41.28
	✓	✓	✗	41.83
	✓	✓	✓	42.56
Faster R-CNN[160] trained on target				68.10

Table 1: Quantitative results comparing our method to baselines for adapting from *SIM 10K* dataset to *Cityscapes*. We record average precision (AP) on the *Cityscapes* validation set. The last row shows the base detector’s performance if labeled data for target domain was available.

⁴This dataset is usually used for instance segmentation and not object detection.

Table 1 compares our method to the baselines. We tested our method with “Classification Error Correction (Cls-Cor)”⁵, with or without the “Bounding Box Refinement (Box-R)” and “False Negative Correction (FN-Cor)” components. The state-of-the-art Feature Learning [14] method has +1.05% improvement over the basic Pseudo-labeling[82] baseline. Our best performing method has a +3.51% improvement over the same baseline yielding more than triple the improvement over the incumbent state-of-the-art.

5.5.2 Adapting normal to foggy weather

Changes in weather conditions can significantly affect visual data. In applications such as autonomous driving, the object detector must perform accurately in all conditions [177]. However, it is often not possible to capture all possible variations of objects in all weather conditions. Therefore, models must be adaptable to differing weather conditions. Here we evaluate our method and demonstrate its superiority over the current state-of-the-art for this task. We use *Cityscapes* dataset as the source domain and *Foggy Cityscapes* as the target domain (denoted by “*Cityscapes* → *Foggy Cityscapes*”).

Table. 2 compares our method to the baselines on multi-label domain adaptation. The categories in this experiment are *person, rider, car, truck, bus, train, motorcycle, bicycle*. Average precision for each category along with the mean average precision (mAP) of all the objects are reported. Our method improves Faster R-CNN mAP by +4.53%, while the state-of-the-art’s improvement is +2.78%.

5.5.3 Adapting to a new dataset

The previous examples of domain adaptation (synthetic data and weather change) are somewhat specialized. However, any change in camera (*e.g.* angle, resolution, quality, type, etc.) or environmental setup can cause domain shift. We investigate the ability of our method to adapt from one real dataset to another real dataset. We use *Cityscapes* and *KITTI* as the source and target domain in two separate evaluations. We denote the experiment in which *Cityscapes* is the source domain and *KITTI* is the target domain by “*Cityscapes* → *KITTI*”, and vice versa by “*KITTI* → *Cityscapes*”.

Tables 5.3 and 5.4 compare average precision on the *car* class, the only common object. Our method significantly outperforms the state-of-the-art in both situations (*Cityscapes* ⇌ *KITTI*). Qualitative results of our method on the *KITTI* test set are shown in Figure 5.2.

5.6 Summary

Domain shift can severely limit the real-world deployment of object-detection-based applications when labeled data collection is either expensive or infeasible. We have proposed an unsupervised

⁵Turning off Cls-Cor reduces our approach to a method similar to Pseudo-labeling[82] with similar performance. To maintain robustness to label noise, we run all experiments with Cls-Cor component.

<i>KITTI</i> \rightarrow <i>Cityscapes</i>				
Method	Cls-Cor	Box-R	FN-Cor	AP
Faster R-CNN[160]				31.10
Pseudo-labeling[82]				40.23
Feature Learning [14]				40.57
Noisy Labeling (Ours) :	✓	✗	✗	42.03
	✓	✓	✗	42.39
	✓	✓	✓	42.98
Faster R-CNN[160] trained on target				68.10

Table 5.3: Quantitative comparison of our method with baselines for adapting from *KITTI* to *Cityscapes*. We record average precision (AP) on the *Cityscapes* test set. The last row gives the base detector’s performance if labeled data for the target domain was available.

<i>Cityscapes</i> \rightarrow <i>KITTI</i>				
Method	Cls-Cor	Box-R	FN-Cor	AP
Faster R-CNN[160]				56.21
Pseudo-labeling[82]				73.84
Feature Learning [14]				73.76
Noisy Labeling (Ours) :	✓	✗	✗	76.36
	✓	✓	✗	76.93
	✓	✓	✓	77.61
Faster R-CNN[160] trained on target				90.13

Table 5.4: Quantitative comparison of our method with baselines for adapting *Cityscapes* to *KITTI*. We record average precision (AP) on the *KITTI* train set. The last row gives the base detector’s performance if labeled data for target domain was available.

approach to mitigate this problem by formulating the problem as robust learning. Our robust object detection framework copes with labeling noise on both object classes and bounding boxes. State-of-the-art performance is achieved by robust training in the target domain using a model trained only in the source domain. This approach eliminates the need for collecting data in the target domain and integrates other sources of information using detection re-scoring.

Chapter 6

Conclusion

In this dissertation, we address the scarcity of labeled data in deep learning. The rise of deep learning has underscored the necessity for large data sets. Yet, obtaining large labeled datasets remains a challenge for certain applications, such as autonomous driving, anomaly detection, and medical imaging. This is due, in part, to the expertise required for annotations in specialized domains, such as medical imaging, making it costly and time-intensive. In more standard fields like image classification or object detection, there exists a range of categories with few examples that are difficult to acquire. Moreover, privacy and ethical concerns further exacerbate data scarcity in image labeling.

This research identifies data scarcity as a pressing concern in both academia and industry and subsequently reviews prevalent strategies to combat this issue. These strategies are classified into three main categories: Transfer Learning, which involves leveraging knowledge from well-labeled domains to those with scant data, and encompasses sub-categories like multitask learning and domain adaptation; Active Learning, which seeks to intelligently and efficiently label data from a smaller labeled set; and Data Augmentation, which artificially enlarges dataset size to bolster training of deep models.

We present three novel methodologies targeting the aforementioned categories—Active Learning, Data Augmentation, and Transfer Learning (specifically Domain Adaptation)—to alleviate the constraints posed by limited labeled data.

6.0.1 Active Learning for Structured Prediction from Partially Labelled Data

In the realm of Active Learning, we introduced an algorithm tailored for structured prediction, efficiently leveraging partially labeled data. Our findings demonstrate that for classification problems producing structured outputs, acquiring labels for every component of the structures in training data isn't essential. This stems from the inherent information sharing and redundancy across nodes in a graphical model. From an information theory standpoint, we proposed a candidate selection strategy based on our introduced metric, "expected entropy reduction". This metric measures the informativeness of the nodes in a graph, emphasizing those nodes offering the most information about the rest of the graph. Experimental results confirm that our algorithm outperforms preceding active learning approaches.

Limitations and Future Directions:

One primary constraint of our technique is its current impracticality within computer vision. Given the iterative nature of our model training and node informativeness measurement, it may be more resource-intensive than simply procuring labels for all images, especially considering recent advances in image labeling tools [173]. Yet, in certain domains such as drug discovery, recommendation systems, and autonomous driving, the concept of active learning for structured prediction remains relevant. A future research direction would be adapting our method’s variants to domains outside of computer vision. Notably, in the context of named entity recognition, active learning for structured prediction remains an active research area [260, 152].

Another key limitation of our methodology is that it did not consider the diversity of samples, a pivotal aspect extensively explored within the active learning framework [261, 5, 183]. As an extension, these considerations can be integrated into the selection strategy along with the informativeness metric.

6.0.2 Dataset Augmentation for Human Action Classification

Human action classification stands as a pivotal task in computer vision. Yet, compared to image classification, annotating videos is both more resource-intensive and time-consuming. In this research, we introduce a data augmentation method to enhance the performance of human action classifiers. Central to our approach is a conditional GAN, designed to generate video frames, utilizing a human skeleton, reference human images, and a background image as inputs.

This methodology synthesizes realistic human action videos from a limited dataset, augmenting the training data without necessitating extensive data collection and labelling. Demonstrated successes on two distinct human action recognition datasets highlight the efficacy of our approach in augmenting conventional action recognition algorithms.

A pivotal factor driving the effectiveness of frame-based video generation techniques for human action classification is the current superior advancement of generative models for images over those for videos. Coupled with the richer availability of labeled data for images than for videos, there’s a unique opportunity to transfer knowledge from the image to the video domain. This is especially possible by harnessing generative models pretrained on vast amount of labeled image datasets, while accounting for frame consistency and overall video coherence.

Limitations and Future Directions:

Contemporary video generation techniques, during the time of this research, didn’t offer the desired proficiency for the task. Nevertheless, substantial advancements have been made in recent years such as "Imagen Video" [74], "Make a Video" [195], hinting at a promising direction involving these emerging methods. However, frame-based video generation is still a viable option.

A significant drawback of our approach resides in the separation between the background and action, leading to potential discrepancies; an illustrative case being the 'sitting' action not accompanied by appropriate background elements like chairs. State-of-the-art diffusion models, such as ControlNet [255] and DreamBooth [167], have displayed capabilities to yield consistent images

given only an input human skeleton or the subject’s image. Their outputs, in terms of quality, often surpass those from GANs.

Furthermore, our efforts to train a GAN for generating skeleton trajectories encountered challenges, notably the persistent issue of mode collapse. Despite implementing various techniques, like truncating trajectory lengths and integrating a batch diversity loss, we didn’t get the desired results. Recent literature indicates that sophisticated models, such as VAE [147] and diffusion model [208], may pave the way for superior quality in skeleton trajectory generation.

6.0.3 Robust Learning for Domain-Adaptive Object Detection

In addressing domain shift within Domain Adaptation, we presented a method addressing a significant challenge in object detection’s real-world application. Our technique stands as an initial contribution in domain adaptation for object detection, emphasizing refining pseudo-labels.

The framework introduced is characterized by its robustness and probabilistic approach, distinct for its simplicity and resilience against label noise in bounding boxes. Our strategy involves a two-fold process:

Extracting pseudo-labels in the target domain with an object detector pretrained on the source domain. Iteratively refining these pseudo-labels grounded on model predictions. Empirical results on datasets like *SIM 10K*, *Cityscapes*, and *KITTI* confirm our method’s superiority over existing state-of-the-art techniques in various domain adaptation scenarios.

Limitations and Future Directions: Our approach’s iterative nature presents potential risks, including the model reinforcing its own errors. We’ve counteracted this by strategic hyperparameter tuning. As an example, we refrain from annealing alpha to zero, ensuring the model doesn’t wholly lean on its predictions during training. Another tactic involves relying on high-confidence samples, achieved by setting stringent thresholds on objectness scores. Looking ahead, refining pseudo-labels remains an active area for exploration. Potential avenues include leveraging temporal consensus during training epochs [257] or integrating teacher-student paradigms combined with source-target domain feature alignment [120]

6.0.4 Final note

In conclusion, our approaches collectively contribute towards enhancing the performance and efficiency of machine learning models in scenarios where labeled data is limited or hard to acquire. We hope that the methods presented here will catalyze further research in overcoming the constraints of labeled data scarcity and will enhance the real-world applicability of machine learning models across a variety of fields.

Bibliography

- [1] Data collection and labeling market size & share report 2030. <https://www.grandviewresearch.com/industry-analysis/data-collection-labeling-market>. Accessed: 2023-7-14.
- [2] Mohamed Rabie Amer, Peng Lei, and Sinisa Todorovic. Hirf: Hierarchical random field for collective activity recognition in videos. In *European Conference on Computer Vision*, pages 572–585. Springer, 2014.
- [3] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.
- [4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.
- [5] Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019.
- [6] Mohammad Bagheri, Qigang Gao, Sergio Escalera, Albert Clapes, Kamal Nasrollahi, Michael B Holte, and Thomas B Moeslund. Keep it accurate and diverse: Enhancing action recognition performance by ensemble learning. In *CVPRW*, pages 22–29, 2015.
- [7] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.
- [8] Christopher Bowles, Liang Chen, Ricardo Guerrero, Paul Bentley, Roger Gunn, Alexander Hammers, David Alexander Dickie, Maria Valdés Hernández, Joanna Wardlaw, and Daniel Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks. *arXiv preprint arXiv:1810.10863*, 2018.
- [9] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *ICCV*, pages 754–763, 2017.
- [10] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.
- [11] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. *arXiv preprint arXiv:1705.07750*, 2017.
- [12] Rich Caruana. *Multitask learning*. Springer, 1998.

- [13] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.
- [14] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3339–3348, 2018.
- [15] Yuhua Chen, Wen Li, and Luc Van Gool. Road: Reality oriented adaptation for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7892–7901, 2018.
- [16] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *International conference on machine learning*, pages 794–803. PMLR, 2018.
- [17] Wongun Choi and Silvio Savarese. A unified framework for multi-target tracking and collective activity recognition. In *European Conference on Computer Vision*, pages 215–230. Springer, 2012.
- [18] Wongun Choi, Khuram Shahid, and Silvio Savarese. What are they doing?: Collective activity classification using spatio-temporal relationship among people. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 1282–1289. IEEE, 2009.
- [19] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3642–3649. IEEE, 2012.
- [20] Brendan Collins, Jia Deng, Kai Li, and Li Fei-Fei. Towards scalable dataset construction: An active learning approach. In *European Conference on Computer Vision*, pages 86–98. Springer, 2008.
- [21] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.
- [22] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.
- [23] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.
- [24] Ali Dabouei, Sobhan Soleymani, Fariborz Taherkhani, and Nasser M Nasrabadi. Supermix: Supervising the mixing data augmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13794–13803, 2021.
- [25] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in neural information processing systems*, pages 379–387, 2016.

- [26] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.
- [27] Mostafa Dehghani, Arash Mehrjou, Stephan Gouws, Jaap Kamps, and Bernhard Schölkopf. Fidelity-weighted learning. In *International Conference on Learning Representations (ICLR)*, 2018.
- [28] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.
- [29] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.
- [30] Zhiwei Deng, Arash Vahdat, Hexiang Hu, and Greg Mori. Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016.
- [31] Zhiwei Deng, Mengyao Zhai, Lei Chen, Yuhao Liu, Srikanth Muralidharan, Mehrosan Roshtkhari, and Greg Mori. Deep structured models for group activity recognition. In *2015 The British Machine Vision Conference (BMVC)*, 2015.
- [32] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015.
- [33] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.
- [34] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655. PMLR, 2014.
- [35] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, pages 2625–2634, 2015.
- [36] Gianfranco Doretto, Alessandro Chiuso, Ying Nian Wu, and Stefano Soatto. Dynamic textures. *International Journal of Computer Vision*, 51(2):91–109, 2003.
- [37] Georgios Douzas and Fernando Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, 91:464–471, 2018.
- [38] Lixin Duan, Ivor W Tsang, and Dong Xu. Domain transfer multiple kernel learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):465–479, 2012.
- [39] Lixin Duan, Dong Xu, Ivor Wai-Hung Tsang, and Jiebo Luo. Visual event recognition in videos by learning from web data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1667–1680, 2012.

- [40] Melanie Ducoffe and Frederic Precioso. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841*, 2018.
- [41] Suyog Dutt Jain and Kristen Grauman. Active image segmentation propagation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [42] Abhishek Dutta and Andrew Zisserman. The via annotation software for images, audio and video. In *Proceedings of the 27th ACM international conference on multimedia*, pages 2276–2279, 2019.
- [43] Moe Elbadawi, Simon Gaisford, and Abdul W Basit. Advanced machine-learning techniques in drug discovery. *Drug Discovery Today*, 26(3):769–777, 2021.
- [44] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- [45] Alireza Fathi, Maria Florina Balcan, Xiaofeng Ren, and James M Rehg. Combining self training and active learning for video segmentation. In *Eds. Jesse Hoey, Stephen McKenna and Emanuele Trucco, In Proceedings of the British Machine Vision Conference (BMVC 2011)*, volume 29, pages 78–1, 2011.
- [46] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1933–1941, 2016.
- [47] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.
- [48] Basura Fernando, Amaury Habrard, Marc Sebban, and Tinne Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *Proceedings of the IEEE international conference on computer vision*, pages 2960–2967, 2013.
- [49] Niklas Fiedler, Marc Bestmann, and Norman Hendrich. Imagetagger: An open source online platform for collaborative image labeling. In *RoboCup 2018: Robot World Cup XXII 22*, pages 162–169. Springer, 2019.
- [50] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in Neural Information Processing Systems*, pages 64–72, 2016.
- [51] Maayan Frid-Adar, Eyal Klang, Michal Amitai, Jacob Goldberger, and Hayit Greenspan. Synthetic data augmentation using gan for improved liver lesion classification. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 289–293. IEEE, 2018.
- [52] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *arXiv preprint arXiv:1409.7495*, 2014.
- [53] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

- [54] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.
- [55] Timnit Gebru, Judy Hoffman, and Li Fei-Fei. Fine-grained recognition in the wild: A multi-task domain adaptation approach. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 1358–1367. IEEE, 2017.
- [56] Yonatan Geifman and Ran El-Yaniv. Deep active learning over the long tail. *arXiv preprint arXiv:1711.00941*, 2017.
- [57] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.
- [58] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2918–2928, 2021.
- [59] Muhammad Ghifary, W Bastiaan Kleijn, and Mengjie Zhang. Domain adaptive neural networks for object recognition. In *PRICAI 2014: Trends in Artificial Intelligence: 13th Pacific Rim International Conference on Artificial Intelligence, Gold Coast, QLD, Australia, December 1-5, 2014. Proceedings 13*, pages 898–904. Springer, 2014.
- [60] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 597–613. Springer, 2016.
- [61] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F Troje. Movi: A large multipurpose motion and video dataset. *arXiv preprint arXiv:2003.01888*, 2020.
- [62] Ross Girshick. Fast r-cnn. pages 1440–1448, 2015.
- [63] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.
- [64] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2066–2073. IEEE, 2012.
- [65] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical computer science*, 38:293–306, 1985.
- [66] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [67] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 999–1006. IEEE, 2011.

- [68] Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXV 16*, pages 1–16. Springer, 2020.
- [69] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *European conference on computer vision*, pages 346–361. Springer, 2014.
- [70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [71] Tao He, Xiaoming Jin, Guiguang Ding, Lan Yi, and Chenggang Yan. Towards better uncertainty sampling: Active learning with multiple views for deep convolutional neural network. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1360–1365. IEEE, 2019.
- [72] 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. *arXiv preprint arXiv:1912.02781*, 2019.
- [73] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.
- [74] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv:2204.03458*, 2022.
- [75] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017.
- [76] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016.
- [77] Alex Holub, Pietro Perona, and Michael C Burl. Entropy-based active learning for object recognition. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW’08. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2008.
- [78] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [79] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.
- [80] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, volume 4, 2017.

- [81] Mostafa S. Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. Hierarchical deep temporal models for group activity recognition. *arXiv preprint arXiv:1607.02643*, 2016.
- [82] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5001–5009, 2018.
- [83] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.
- [84] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.
- [85] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *CVPR*, 2016.
- [86] Prateek Jain and Ashish Kapoor. Active learning for large multi-class problems. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 762–769. IEEE, 2009.
- [87] David Janz, Jos van der Westhuizen, and José Miguel Hernández-Lobato. Actively learning what makes a discrete sequence valid. *arXiv preprint arXiv:1708.04465*, 2017.
- [88] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *PAMI*, 35(1):221–231, 2013.
- [89] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.
- [90] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Regularizing very deep neural networks on corrupted labels. In *International Conference on Machine Learning (ICML)*, 2018.
- [91] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *arXiv preprint arXiv:1610.01983*, 2016.
- [92] Imran N Junejo, Emilie Dexter, Ivan Laptev, and Patrick Perez. View-independent action recognition from temporal self-similarities. *PAMI*, 33(1):172–185, 2011.
- [93] Nal Kalchbrenner, Aaron van den Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. *arXiv preprint arXiv:1610.00527*, 2016.
- [94] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with gaussian processes for object categorization. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

- [95] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.
- [96] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- [97] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.
- [98] Ira Kemelmacher-Shlizerman, Steven M Seitz, Daniel Miller, and Evan Brossard. The megaface benchmark: 1 million faces for recognition at scale. In *CVPR*, pages 4873–4882, 2016.
- [99] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018.
- [100] Sameh Khamis, Vlad I Morariu, and Larry S Davis. Combining per-frame and per-track cues for multi-person action recognition. In *European Conference on Computer Vision*, pages 116–129. Springer, 2012.
- [101] Mehran Khodabandeh, Zhiwei Deng, Mostafa S Ibrahim, Shinichi Satoh, and Greg Mori. Active learning for structured prediction from partially labeled data. *arXiv preprint arXiv:1706.02342*, 2017.
- [102] Mehran Khodabandeh, Hamid Reza Vaezi Joze, Ilya Zharkov, and Vivek Pradeep. Diy human action dataset generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.
- [103] Mehran Khodabandeh, Hamid Reza Vaezi Joze, Ilya Zharkov, and Vivek Pradeep. Diy human action dataset generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1448–1458, 2018.
- [104] Mehran Khodabandeh, Arash Vahdat, Mani Ranjbar, and William G. Macready. A robust learning approach to domain adaptive object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- [105] Mehran Khodabandeh, Arash Vahdat, Guang-Tong Zhou, Hossein Hajimirsadeghi, Mehrrsan Javan Roshtkhari, Greg Mori, and Stephen Se. Discovering human interactions in videos with limited data labeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–18, 2015.
- [106] Jang-Hyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. *arXiv preprint arXiv:2102.03065*, 2021.
- [107] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learning*, pages 5275–5285. PMLR, 2020.

- [108] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [109] Andreas Kirsch, Joost Van Amersfoort, and Yarin Gal. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning. *Advances in neural information processing systems*, 32, 2019.
- [110] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).
- [111] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-100 (canadian institute for advanced research).
- [112] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [113] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.
- [114] Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1785–1792. IEEE, 2011.
- [115] Tian Lan, Yang Wang, Weilong Yang, Stephen N Robinovitch, and Greg Mori. Discriminative latent models for recognizing contextual group activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1549–1562, 2012.
- [116] David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.
- [117] Li-Jia Li, Hao Su, Yongwhan Lim, and Li Fei-Fei. Object bank: An object-level image representation for high-level visual recognition. *International journal of computer vision*, 107:20–39, 2014.
- [118] Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3d points. In *CVPRW*, pages 9–14. IEEE, 2010.
- [119] Wen Li, Zheng Xu, Dong Xu, Dengxin Dai, and Luc Van Gool. Domain generalization and adaptation using low rank exemplar svms. *IEEE transactions on pattern analysis and machine intelligence*, 40(5):1114–1127, 2018.
- [120] Yu-Jhe Li, Xiaoliang Dai, Chih-Yao Ma, Yen-Cheng Liu, Kan Chen, Bichen Wu, Zijian He, Kris Kitani, and Peter Vajda. Cross-domain adaptive teacher for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7581–7590, 2022.
- [121] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1718–1727, 2015.

- [122] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugmentation. *Advances in Neural Information Processing Systems*, 32, 2019.
- [123] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.
- [124] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.
- [125] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017.
- [126] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pages 469–477, 2016.
- [127] Shikun Liu, Edward Johns, and Andrew J Davison. End-to-end multi-task learning with attention. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1871–1880, 2019.
- [128] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.
- [129] 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. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.
- [130] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.
- [131] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. In *Advances in Neural Information Processing Systems*, pages 136–144, 2016.
- [132] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2208–2217. JMLR. org, 2017.
- [133] Wenjie Luo, Alex Schwing, and Raquel Urtasun. Latent structured active learning. In *Advances in Neural Information Processing Systems*, pages 728–736, 2013.
- [134] Marcin Marszałek, Ivan Laptev, and Cordelia Schmid. Actions in context. In *CVPR*, 2009.
- [135] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.
- [136] Arash Mehrjou, Mehran Khodabandeh, and Greg Mori. Distribution aware active learning. *arXiv preprint arXiv:1805.08916*, 2018.

- [137] Ishan Misra, C. Lawrence Zitnick, Margaret Mitchell, and Ross Girshick. Seeing through the human reporting bias: Visual classifiers from noisy human-centric labels. In *CVPR*, 2016.
- [138] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003, 2016.
- [139] Volodymyr Mnih and Geoffrey E. Hinton. Learning to label aerial images from noisy data. In *International Conference on Machine Learning (ICML)*, pages 567–574, 2012.
- [140] Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, page 3, 2017.
- [141] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4500–4509, 2018.
- [142] Nagarajan Natarajan, Inderjit S. Dhillon, Pradeep K. Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.
- [143] Anh Nguyen, Jason Yosinski, Yoshua Bengio, Alexey Dosovitskiy, and Jeff Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. *arXiv preprint arXiv:1612.00005*, 2016.
- [144] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in Neural Information Processing Systems*, pages 2863–2871, 2015.
- [145] Giorgio Patrini, Alessandro Rozza, Aditya Menon, Richard Nock, and Lizhen Qu. Making neural networks robust to label noise: A loss correction approach. In *Computer Vision and Pattern Recognition*, 2017.
- [146] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.
- [147] Mathis Petrovich, Michael J Black, and Gül Varol. Action-conditioned 3d human motion synthesis with transformer vae. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10985–10995, 2021.
- [148] Jeff M Phillips. Coresets and sketches. *arXiv preprint arXiv:1601.00617*, 2016.
- [149] Ronald Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010.
- [150] Guo-Jun Qi, Xian-Sheng Hua, Yong Rui, Jinhui Tang, and Hong-Jiang Zhang. Two-dimensional active learning for image classification. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.
- [151] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

- [152] Puria Radmard, Yassir Fathullah, and Aldo Lipani. Subsequence based deep active learning for named entity recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4310–4321, 2021.
- [153] Anant Raj, Vinay P Namboodiri, and Tinne Tuytelaars. Subspace alignment based domain adaptation for rcnn detector. *arXiv preprint arXiv:1507.05578*, 2015.
- [154] Vignesh Ramanathan, Jonathan Huang, Sami Abu-El-Haija, Alexander Gorban, Kevin Murphy, and Li Fei-Fei. Detecting events and key actors in multi-person videos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, June 2016.
- [155] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.
- [156] Hiranmayi Ranganathan, Hemanth Venkateswara, Shayok Chakraborty, and Sethuraman Panchanathan. Deep active learning for image classification. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3934–3938. IEEE, 2017.
- [157] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- [158] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. *arXiv preprint*, 2017.
- [159] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning (ICML)*, 2018.
- [160] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [161] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and variational inference in deep latent gaussian models. In *International Conference on Machine Learning*, 2014.
- [162] Alexandre Robicquet, Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *European Conference on Computer Vision*, pages 549–565. Springer, 2016.
- [163] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.
- [164] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015.
- [165] Dan Roth and Kevin Small. Margin-based active learning for structured output spaces. In *European Conference on Machine Learning*, pages 413–424. Springer, 2006.
- [166] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

- [167] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22500–22510, 2023.
- [168] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
- [169] Olga Russakovsky, Li-Jia Li, and Li Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [170] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77:157–173, 2008.
- [171] MS Ryoo and JK Aggarwal. Stochastic representation and recognition of high-level group activities. *International journal of computer Vision*, 93(2):183–200, 2011.
- [172] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *arXiv preprint arXiv:1701.01909*, 2017.
- [173] Christoph Sager, Christian Janiesch, and Patrick Zschech. A survey of image labelling for computer vision applications. *Journal of Business Analytics*, 4(2):91–110, 2021.
- [174] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.
- [175] Asaad F Said, Vinay Kashyap, Namrata Choudhury, and Farshad Akhbari. A cost-effective, fast, and robust annotation tool. In *2017 IEEE applied imagery pattern recognition workshop (AIPR)*, pages 1–6. IEEE, 2017.
- [176] Masaki Saito and Eiichi Matsumoto. Temporal generative adversarial nets. *arXiv preprint arXiv:1611.06624*, 2016.
- [177] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, pages 1–20, 2018.
- [178] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.
- [179] Ikuro Sato, Hiroki Nishimura, and Kensuke Yokoi. Apac: Augmented pattern classification with neural networks. *arXiv preprint arXiv:1505.03229*, 2015.
- [180] Tobias Scheffer, Christian Decomain, and Stefan Wrobel. Active hidden markov models for information extraction. In *International Symposium on Intelligent Data Analysis*, pages 309–318. Springer, 2001.

- [181] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *ICPR*, volume 3, pages 32–36. IEEE, 2004.
- [182] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360. ACM, 2007.
- [183] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.
- [184] Burr Settles. Active learning literature survey. 2009.
- [185] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics, 2008.
- [186] Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Advances in neural information processing systems*, pages 1289–1296, 2008.
- [187] H Sebastian Seung, Manfred Opper, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM, 1992.
- [188] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *CVPR*, pages 1010–1019, 2016.
- [189] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.
- [190] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.
- [191] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.
- [192] Tianmin Shu, Dan Xie, Brandon Rothrock, Sinisa Todorovic, and Song-Chun Zhu. Joint inference of groups, events and human roles in aerial videos. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4576–4584. IEEE, 2015.
- [193] Patrice Y Simard, David Steinkraus, John C Platt, et al. Best practices for convolutional neural networks applied to visual document analysis. In *Icdar*, volume 3. Edinburgh, 2003.
- [194] Oriane Siméoni, Mateusz Budnik, Yannis Avrithis, and Guillaume Gravier. Rethinking deep active learning: Using unlabeled data at model training. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 1220–1227. IEEE, 2021.
- [195] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

- [196] Jaemin Son, Sangkeun Kim, Sang Jun Park, and Kyu-Hwan Jung. An efficient and comprehensive labeling tool for large-scale annotation of fundus images. In *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis: 7th Joint International Workshop, CVII-STENT 2018 and Third International Workshop, LABELS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Proceedings 3*, pages 95–104. Springer, 2018.
- [197] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.
- [198] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, pages 843–852, 2015.
- [199] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.
- [200] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, volume 6, page 8, 2016.
- [201] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision—ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8–10 and 15–16, 2016, Proceedings, Part III 14*, pages 443–450. Springer, 2016.
- [202] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019.
- [203] Qing Sun, Ankit Laddha, and Dhruv Batra. Active learning for structured probabilistic models with histogram approximation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3612–3621, 2015.
- [204] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [205] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.
- [206] Martin Szummer and Rosalind W Picard. Temporal texture modeling. In *Image Processing, 1996. Proceedings., International Conference on*, volume 3, pages 823–826. IEEE, 1996.
- [207] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [208] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. *arXiv preprint arXiv:2209.14916*, 2022.
- [209] Hoang Thanh-Tung and Truyen Tran. Catastrophic forgetting and mode collapse in gans. In *2020 international joint conference on neural networks (ijcnn)*, pages 1–10. IEEE, 2020.

- [210] Jonathan J Tompson, Arjun Jain, Yann LeCun, and Christoph Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in neural information processing systems*, pages 1799–1807, 2014.
- [211] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.
- [212] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, pages 4489–4497, 2015.
- [213] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. *arXiv preprint arXiv:1707.04993*, 2017.
- [214] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.
- [215] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.
- [216] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.
- [217] Arash Vahdat. Toward robustness against label noise in training deep discriminative neural networks. In *Neural Information Processing Systems (NIPS)*, 2017.
- [218] Arash Vahdat and Greg Mori. Handling uncertain tags in visual recognition. In *International Conference on Computer Vision (ICCV)*, 2013.
- [219] Arash Vahdat, Guang-Tong Zhou, and Greg Mori. Discovering video clusters from visual features and noisy tags. In *European Conference on Computer Vision (ECCV)*, 2014.
- [220] Joost van Amersfoort, Anitha Kannan, Marc’Aurelio Ranzato, Arthur Szlam, Du Tran, and Soumith Chintala. Transformation-based models of video sequences. *arXiv preprint arXiv:1701.08435*, 2017.
- [221] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. In *Advances in Neural Information Processing Systems*, pages 4790–4798, 2016.
- [222] Balakrishnan Varadarajan, Dong Yu, Li Deng, and Alex Acero. Maximizing global entropy reduction for active learning in speech recognition. In *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, pages 4721–4724. IEEE, 2009.
- [223] Andreas Veit, Neil Aldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6575–6583. IEEE, 2017.

- [224] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.
- [225] Alexander Vezhnevets, Joachim M Buhmann, and Vittorio Ferrari. Active learning for semantic segmentation with expected change. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3162–3169. IEEE, 2012.
- [226] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. *ICLR*, 1(2):7, 2017.
- [227] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.
- [228] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204, 2013.
- [229] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016.
- [230] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2517–2526, 2019.
- [231] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. *arXiv preprint arXiv:1705.00053*, 2017.
- [232] Boyu Wang, Jorge Mendez, Mingbo Cai, and Eric Eaton. Transfer learning via minimizing the performance gap between domains. *Advances in neural information processing systems*, 32, 2019.
- [233] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *CVPR*, pages 3169–3176. IEEE, 2011.
- [234] Yifei Wang, Wen Li, Dengxin Dai, and Luc Van Gool. Deep domain adaptation by geodesic distance minimization. *arXiv preprint arXiv:1707.09842*, 2017.
- [235] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 479–488. ACM Press/Addison-Wesley Publishing Co., 2000.
- [236] Sebastien C Wong, Adam Gatt, Victor Stamatescu, and Mark D McDonnell. Understanding data augmentation for classification: when to warp? In *Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on*, pages 1–6. IEEE, 2016.
- [237] L. Xia, C.C. Chen, and JK Aggarwal. View invariant human action recognition using histograms of 3d joints. In *CVPRW*, pages 20–27. IEEE, 2012.
- [238] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

- [239] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. *arXiv preprint arXiv:2112.07804*, 2021.
- [240] Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L Yuille, and Quoc V Le. Adversarial examples improve image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 819–828, 2020.
- [241] Jiaolong Xu, Sebastian Ramos, David Vázquez, and Antonio M López. Domain adaptation of deformable part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 36(12):2367–2380, 2014.
- [242] T Xue, J Wu, K Bouman, and B Freeman. Probabilistic modeling of future frames from a single image. In *NIPS*, 2016.
- [243] Tianfan Xue, Jiajun Wu, Katherine Bouman, and Bill Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2016.
- [244] Hongliang Yan, Yukang Ding, Peihua Li, Qilong Wang, Yong Xu, and Wangmeng Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2272–2281, 2017.
- [245] Rong Yan, Jie Yang, and Alexander Hauptmann. Automatically labeling video data using multi-class active learning. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 516–523. IEEE, 2003.
- [246] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.
- [247] Xiyu Yu, Tongliang Liu, Mingming Gong, and Dacheng Tao. Learning with biased complementary labels. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 68–83, 2018.
- [248] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702, 2015.
- [249] Kiwon Yun, Jean Honorio, Debaleena Chattopadhyay, Tamara L Berg, and Dimitris Samaras. Two-person interaction detection using body-pose features and multiple instance learning. In *CVPRW*, pages 28–35. IEEE, 2012.
- [250] Sangdoon Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019.
- [251] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016.

- [252] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2736–2746, 2022.
- [253] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
- [254] Liliang Zhang, Liang Lin, Xiaodan Liang, and Kaiming He. Is faster r-cnn doing well for pedestrian detection? In *European Conference on Computer Vision*, pages 443–457. Springer, 2016.
- [255] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543*, 2023.
- [256] Xi Zhang, Yanwei Fu, Andi Zang, Leonid Sigal, and Gady Agam. Learning classifiers from synthetic data using a multichannel autoencoder. *arXiv preprint arXiv:1503.03163*, 2015.
- [257] Xiao Zhang, Yixiao Ge, Yu Qiao, and Hongsheng Li. Refining pseudo labels with clustering consensus over generations for unsupervised object re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3436–3445, 2021.
- [258] Yang Zhang, Philip David, and Boqing Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *The IEEE International Conference on Computer Vision (ICCV)*, volume 2, page 6, 2017.
- [259] Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Neural Information Processing Systems (NIPS)*, 2018.
- [260] Zhisong Zhang, Emma Strubell, and Eduard Hovy. Data-efficient active learning for structured prediction with partial annotation and self-training. *arXiv preprint arXiv:2305.12634*, 2023.
- [261] Fedor Zhdanov. Diverse mini-batch active learning. *arXiv preprint arXiv:1901.05954*, 2019.
- [262] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.
- [263] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.
- [264] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.