Leveraging Structural Cues for Better Training and Deployment in Computer Vision

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering
by Research

by

Shyamgopal Karthik
20161221
shyamgopal.karthik@research.iiit.ac.in

INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY
HYDERABAD

International Institute of Information Technology, Hyderabad
Hyderabad - 500 032, INDIA
April 2021

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "Leveraging Structural Cues for Better Training and Deployment in Computer Vision" by Shyamgopal Karthik, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. Vineet Gandhi

To my Family

Acknowledgments

I would first like to thank my advisor Prof. Vineet Gandhi for the freedom and flexibility that he gave me in the choice of topics as well as my approach to research. None of the work in this thesis could have been possible without the freedom that I got from him. I must also thank Dr. Puneet K. Dokania for our brief interactions which brought in some theoretical rigor and significantly enhanced the quality of our work. I also got a lot of insights into how research is carried out at the highest levels, something I hope to keep in mind for the future. I have also thoroughly enjoyed my classes and interactions with Prof. Aniket Alam and Prof. Kannan Srinathan which gave me a fresh perspective on many topics.

I must also thank the overall CVIT ecosystem for the unparalleled access to resources. In particular, the institute cluster “Ada”, named in honour of Ada Lovelace has been a vital aspect in all the work done as part of my thesis.

Most of my understanding of research has been shaped by my constant discussions with Ameya Prabhu. I find his deep intuition and understanding of research very unique and inspiring. I hope to continue working with him in the future doing simple things that work surprisingly well. I have also learnt a lot from Saujas whose erudition and scholarship I find remarkable. I must also thank him for his painstaking efforts in reviewing and editing every document that I wrote. I would also like to thank Aurobindo, Harish Krishna and Gokul for their constant support and advice especially during challenging times. I would also like to thank Abhinav for useful discussions.

I have thoroughly enjoyed working with Mohsin for most of my projects during my undergrad where I have learnt a great deal from his attention to detail and his sense of aesthetics. I have greatly benefited from our work with Ramkishore on multi-object tracking as well as our adventures in competitive programming with Anish. I have Unni to thank for helping me out whenever I have been stuck with any systems-related issue. I have been very fortunate to have many amazing friends with whom I have shared some wonderful memories over the years.

Finally, I must thank my family for their constant support, encouragement, prayers,and trust in my efforts which has kept me going all these years.

Abstract

With the growing use of computer vision tools in wide-ranging applications, it becomes imperative to understand and resolve issues in computer vision models when they are used in production settings for various applications. In particular, it is essential to understand that the model can be wrong quite frequently during deployment. Developing a better understanding of the mistakes made by a model can help mitigate and handle them without catastrophic consequences.

To investigate the severity of mistakes, we first explore this in a simple classification setting. Even in this setting, understanding the severity of mistakes of difficult to quantify, especially since manually defining pairwise costs does not scale well for large-scale classification datasets. Therefore most works have used class taxonomies/hierarchies, which allow pairwise costs to be defined using graph distances. There has been increasing interest in building deep hierarchy-aware classifiers, aiming to quantify and reduce the severity of mistakes and not just count the number of errors. However, most of these works require the hierarchy to be available during training and cannot adapt to new hierarchies or even small modifications to the existing hierarchy without having to re-train the model. We explore a different direction for hierarchy-aware classification – amending mistakes by making post-hoc corrections by resorting to the classical Conditional Risk Minimization(CRM). Surprisingly, we find that this method is a far more suitable alternative than the works on deep hierarchy-aware classification; CRM preserves the base model's top-1 accuracy and brings the most likely predictions of the model closer to the ground truth and is able to provide reliable probability estimates, unlike hierarchy-aware classifiers. We firmly believe that this serves as a very strong and useful baseline for future exploration in this direction.

We turn our attention to a crucial problem in many video processing pipelines: visual(single) object tracking. In particular, we explore the long-term tracking scenario where given a target in the first frame of the video; the goal is to track the object throughout a (long) video during which the object may undergo occlusion, vary in appearance, or go out-of-view. The temporal aspect of videos also makes it an ideal scenario to understand the accumulation of errors that would not be otherwise seen if every image is independent. We hypothesize that there are three crucial abilities that a tracker must possess to be effective in the long-term tracking scenario, namely Re-Detection, Recovery and, Reliability. The tracker must be able to re-detect the target when the target goes out of the scene, and returns must recover from failure and track an object contiguously to be of practical utility. We propose a set of novel and com-

prehensive experiments to understand each of these aspects which give a thorough understanding of the strengths and limitations of various state-of-the-art tracking algorithms.

We finally visit the problem of multi-object tracking. Unlike the problem of single-object tracking where the target is initialized in the first frame, the goal here is to track all objects of a *particular category*(such as pedestrians, vehicles, animals etc.). Since this problem does not require user-initialization, it has found use in wide-ranging real-time applications such as autonomous driving. The typical multi-object tracking pipeline follows the tracking-by-detection paradigm, i.e. an object detector is first used to detect all the objects in the scene. These detections are linked together to form the final trajectories using a combination of Spatio-temporal features and appearance/Re-Identification(ReID) features. The appearance features are extracted using a Convolutional Neural Network(CNN) trained on a corpus of labelled videos. Our central insight is that only the appearance model requires labelled videos in the entire pipeline, while the rest of the pipeline can be trained with just image-level supervision. Inspired by the recent successes in unsupervised contrastive learning which enforces the similarity in feature space between an image and its augmented version, we resort to a simple method that leverages the spatio-temporal consistency in videos to generate "natural" augmentations which are then used as pseudo-labels to train the appearance model. When integrated into the overall tracking pipeline, we find that this unsupervised appearance model can match the performance of its supervised counterparts in reducing the identity switches present in the trajectories, thereby saving costly video annotations that are impractical to scale up without sacrificing performance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the explosion of visual content in the form of images and videos on the internet and other platforms, computer vision solutions are being deployed in a wide variety of settings to automate and assist humans on tedious and time-consuming tasks. However, even with the best machine learning tools and large-scale training data, it is inevitable that these models make mistakes. Unlike humans, who can reduce the severity of the mistake or recover from their mistakes; many models can fail spectacularly, with catastrophic consequences. There has also been much work showing that small perturbations to the input can have catastrophic consequences to the predictions. However, these attacks typically require full access to the model, which is quite an impractical setting. There are 2 cases of catastrophic failures of deep neural networks in computer vision tasks in recent times which highlight the catastrophic failure of these models even in natural settings without any adversarial perturbations.

1.1 Two Motivating Examples

The first famous failure which is relevant to this thesis is when Google Photos categorized two people as 'gorillas" [1]. This incident, and Google's response of omitting the class as a label highlighted the lack of algorithmic solutions to tackle this challenge. It is imperative to understand that no model would achieve 100% accuracy on challenging real-world images. However, it is still vital that the model reduces the severity of mistakes and can understand when it is unsure about a prediction. The second aspect of models being able to provide calibrated uncertainty estimates is currently lacking in various state-of-the-art models.

The second failure, which also was extensively covered in the press is related to visual object tracking, which is much more specific to computer vision. Over the past few years, computer vision tools have become extremely popular for various sports broadcasting and analytics tasks. In particular, one ex-

[1] This was covered extensively by the press, e.g. `https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people`

ample is using visual object tracking to "track" important components of the game and then use these tracks for automatic editing and broadcasting. Recently, a football game that was using these tools for automatic ball tracking and broadcasting, encountered a catastrophic failure where the tracker started following the referee's head instead of the ball [2]. For the rest of the game, the entire broadcast just showed the referee instead of the actual football game that was happening.

While these incidents are specific to computer vision, the underlying issues are prevalent elsewhere in different settings where deep neural networks are increasingly being deployed. Therefore, it becomes crucial to understand the various failure modes in different algorithms and propose algorithmic approaches to tackle these issues.

In this thesis, inspired by these two examples, we look at methods that can better understand these issues and possibly alleviate some of these concerns.

1.2 Contributions

More formally, we make the following contributions:

1. We revisit an old post-hoc correction technique to incorporate hierarchies on a pre-trained image model. We also analyze theoretical guarantees for the top-1 accuracy with this post-hoc correction technique and investigate the calibration of hierarchy-aware models.
2. We propose novel experiments and metrics to understand the role of re-detection, recovery and reliability in long-term tracking scenarios.
3. We investigate the problem of multi-object tracking where we provide insights into how the overall training regime can get rid of trajectory level supervision. We further illustrate one concrete algorithm for the same and demonstrate that trajectory level supervision can be eliminated without sacrificing performance.

1.3 Background

1.3.1 Cost-Sensitive Learning

Not all mistakes are created equal. Some mistakes are much more costly than other mistakes. However, most machine learning work has focused on treating all classes equally, and therefore all mistakes equally costly. However, some work has been focused on cost-sensitive classification. In this setting,

[2] A full report of the story is available here: `https://www.theverge.com/tldr/2020/11/3/21547392/ai-camera-operator-football-bald-head-soccer-mistakes`

given N classes, a cost matrix C is constructed where C_{ij} is the cost for misclassifying class i as j. Typically C_{ii} is set to 0, and in many cases the costs are symmetric, therefore C_{ij} = Cji.

1.3.1.1 Binary Cost Sensitive Classification

The simplest case is where there are 2 classes, where $C_{00} = C_{11} = 0$ and $C_{01} \neq C_{10}$. Let p be the probability that a sample belongs to class 1 and $1 - p$ is the probability that the sample belongs to class 0. We can compute a threshold

$$\tau = \frac{C_{01}}{C_{01} + C_{10}}$$

. When $p \leq \tau$, we would classify the sample as class 0 and in other cases classify it as belonging to class 1. This would be optimal and the threshold and can be easily computed apriori. However, for the multi-class setting, the problem cannot be solved trivially. The methods that have been proposed for tackling this problem are broadly along the following lines:

1. The first set of methods aim to modify the classifier(support vector machine, neural networks etc.) itself to make it sensitive to the pairwise costs of misclassification.

2. The second line of approach is to modify the training procedure to make the resulting trained model cost-sensitive. This is typically done by reweighting the losses differently using the costs.

3. Finally, a generic trained model can also be made cost-sensitive by modifying the inference procedure akin to the binary classification case. This is perhaps the most general approach which can be applied as a post-hoc step to any pre-trained model.

1.3.1.2 Class Hierarchies for Cost-Sensitive Learning

The biggest issue holding back the widespread use of cost-sensitive learning in modern computer vision is that defining pairwise costs is a tedious process and does not scale well to the thousands of classes which are prevalent in the popular large-scale classification datasets. This requires millions of pairwise costs to be defined which is not practically feasible. An interesting alternative therefore was to leverage class hierarchies for this purpose. The semantic classes in computer vision datasets can be found in popular taxonomies such as WordNet or even the biological taxonomy. Therefore, we can construct a hierarchy H(which is a rooted tree), such that all the classes in the dataset form the leaf classes of the hierarchy. In such a setting, the graph distance between 2 classes can be used to define the pairwise cost for the classes. For instance, the ImageNet benchmark uses the height of the Lowest Common Ancestor(LCA) [3] to define the cost between 2 classes. Alternatively, the shortest distance, or the Jiang-Conrath distance can also be used for this purpose.

[3] A formal definition of the Lowest Common Ancestor can be found here `https://en.wikipedia.org/wiki/Lowest_common_ancestor`

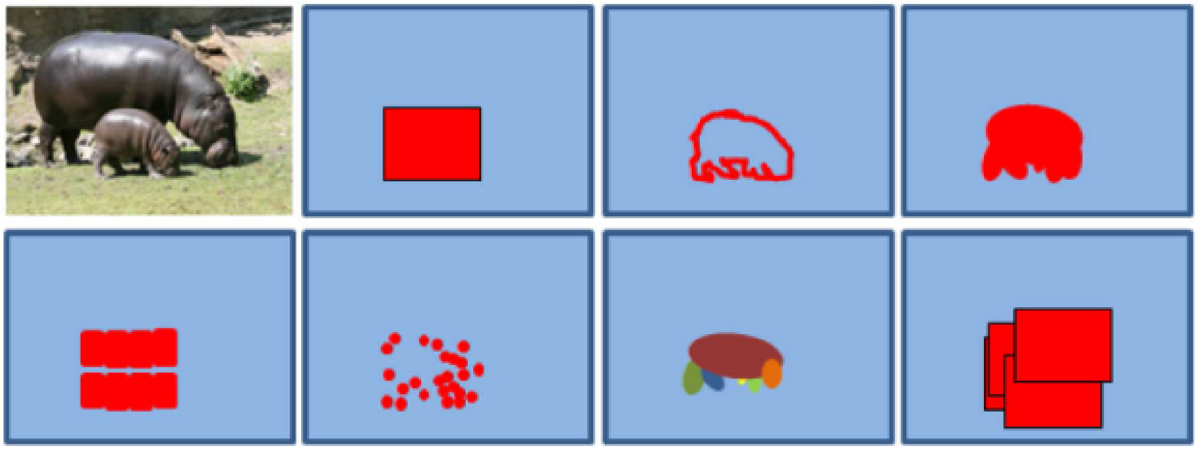


Figure 1.1: An example taken from [99] showing the different formulations of tracking: From left to right, top to bottom: target bounding box, target contour, target blob, patch-based,sparse set of salient features, parts, and multiple bounding boxes.

1.3.2 Visual Object Tracking

Understanding the temporal dynamics is essential for effective video understanding. Therefore one of the fundamental tasks in video understanding is object tracking. There have been a wide variety of formulations of this problem which is illustrated in Fig. 1.1. However, for this thesis, we restrict ourselves to the problem of tracking using bounding boxes i.e assigning a bounding box for the estimated target. Two popular problems with this formulation are single-object tracking and multiple object tracking.

1.3.2.1 Single-Object Tracking

The objective of the single-object tracking problem is to track a given bounding box in the first frame of a video for the rest of the sequence. This problem has received a lot of attention over the years. Some of the initial attempts tried to track the keypoints over the duration of the video. However, in recent times most methods have tried to learn object level representations from the first frame and then use this representation to track the object for the rest of the video. The use of features learnt from a Convolutional Neural Network has lead to a huge improvement in the performance as compared to the earlier handcrafted features. However, this problem still has a lot of scope for improvement especially when the duration of the video increases and the objects start going out of view and re-enter.

1.3.2.2 Multiple Object Tracking

In contrast to the single-object tracking problem, the problem of multi-object tracking aims to track all objects of a particular category for the duration of the video. The main challenge here is that there is no

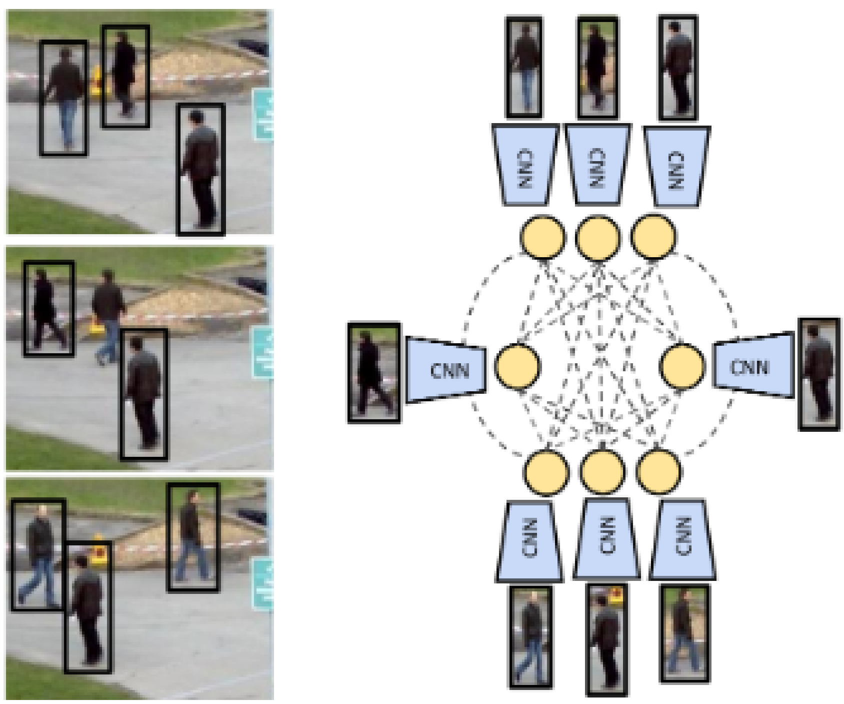


Figure 1.2: An example taken from [14] showing the overall pipeline of Multi-Object tracking algorithms. First an object detector is used to generate the detections in the image. Then using a combination of spatio-temporal and appearance features, the final tracks are generated.

track initialization, that is the tracker needs to identify all the relevant objects and track it. To make this task easier, a restriction is placed to ensure that only objects from a given vocabulary are required to be tracked. The most popular approach here has been tracking-by-detection. In essence, this problem is broken into two stages. In the first stage, a detector is used to detect all the relevant objects in the video. In the second stage, these detections are linked using a combination of spatiotemporal and appearance features to form the final trajectories. An illustration of this is given in Fig 1.2. In the past, Multi-Object Tracking has typically been limited to tracking pedestrians, vehicles and animals, but in recent times object detectors with wider vocabularies are being explored for this task.

1.4 Thesis Organization

The rest of the thesis is organized as follows: In Chapter 2, we investigate class hierarchies(derived from taxonomies) to define costs between classes using graph distances automatically. Here, we investigate methods to incorporate hierarchies into the image model in a post-hoc manner. In addition to this, we also investigate the calibration of various hierarchy-aware models. In Chapter 3, we investigate the problem of long-term visual object tracking where we place a special emphasis on understanding the failures of trackers and how they recover from a failure in tracking. Finally, in Chapter 4, we argue why Multi-Object Tracking is a more scalable setting that is more resistant to tracking failure. In addition to this, we propose a method to eliminate trajectory level supervision without sacrificing the pipeline's accuracy.

Overall, we hope that this thesis sheds light on various challenges prevalent when deep neural networks are deployed in practical settings. We also hope that this thesis provides some ideas and baselines which can be taken up in the future to address these issues in a variety of settings with more success.

Chapter 2

No-Cost Likelihoood Manipulation for Making Better Mistakes

2.1 Introduction

The conventional performance measure of accuracy for image classification treats all classes other than ground truth as equally wrong. However, some mistakes may have a much higher impact than others in real-world applications. An intuitive example being an autonomous vehicle mistaking a car for a bus is a *better mistake* than mistaking a car for a lamppost. Consequently, it is essential to integrate the notion of mistake severity into classifiers and one convenient way to do so it to use a taxonomic hierarchy tree of class labels, where severity is defined by a distance on the graph (e.g., height of the Lowest Common Ancestor) between the ground truth and the predicted label [29, 133]. This is similar to the problem of ranking classes and samples in a classification and retrieval setting, respectively. Consider the case of an autonomous vehicle ranking classes for a thin, white, narrow band (a pole, in reality). A top-3 prediction of "pole, lamppost, tree" would be a *better* prediction than "pole, person, building". Notice that the top-k class predictions would have at least $k - 1$ incorrect predictions here, and the aim is to reduce the severity of these mistakes, measured by the average hierarchical distance of the top k predictions each with the ground truth. [98] survey classical methods leveraging class hierarchy when designing classifiers across various application domains and illustrate clear advantages over the flat hierarchy classification, especially when the labels have a well-defined hierarchy.

There has been growing interest in the problem of deep hierarchy-aware image classification [4, 7]. These approaches seek to leverage the class hierarchy inherent in the large scale datasets (e.g., the ImageNet dataset is derived from the WordNet semantic ontology). Hierarchy is incorporated either using label embedding methods, hierarchical loss functions, or hierarchical architectures. These models are able to hierarchically align the top-k predictions of the model with the ground truth while trading off top-1 accuracy.

Upon close inspection of the top-1 predictions of state-of-the-art models for hierarchy-aware classification, we found that these approaches, in fact, are no better than the vanilla cross-entropy models in making better mistakes. Instead of reducing high-severity mistakes, they typically tend to introduce additional low-severity mistakes that eventually brings down the average mistake-severity metric. Because of the these additional mistakes, they also suffer from significant accuracy drop compared to the vanilla cross-entropy model. Additionally, we find these models to be highly uncalibrated (hence unreliable) which further limits their practical usability.

We explore a different direction for hierarchy-aware classification whereby we amend mistake severity at test time by making post-hoc corrections over the class likelihoods (e.g., softmax in the case of deep neural networks). Given a label hierarchy, we perform such amendments to the likelihood by applying the very well-known and classical approach called Conditional Risk Minimization (CRM). We found that CRM outperforms state-of-the-art deep hierarchy-aware classifiers by large margins at ranking classes with little loss in the classification accuracy. CRM is simple, requires addition of just one line of code to the standard cross-entropy model, does not require retraining of a network, and contains no hyperparameters whatsoever.

We would like to emphasize that we do not claim any algorithmic novelty as CRM has been well explored in the literature [31, Ch. 2]. Almost a decade ago, [29] had proposed a very similar solution using Support Vector Machines (SVMs) as classifiers over handcrafted features. However, this did not result in practically useful performance because of the lack of modern machine learning tools. Our primary goal is to bring this old, simple, and extremely effective approach back into the attention before we delve deeper into sophisticated ones with modern classifiers. Overall, our investigation into hierarchy-aware classification makes the following contributions:

- We highlight significant issues that have been overlooked by the past works that aim to leverage hierarchies to make better mistakes by first demonstrating how they do not really make better mistakes.

- We revisit an old post-hoc correction technique which significantly outperforms prior art when the ranking of the predictions made by the model are considered.

- We also investigate the reliability of prior art in terms of calibration and show that these methods are severely miscalibrated, limiting their practical usefulness.

2.2 Related Works

2.2.1 Cost-Sensitive Classification

Cost-sensitive classification assigns varying costs to different types of misclassification errors. The work by [1] groups cost-sensitive classifiers into three main categories. The first category specifically extends one particular classification model to be cost-sensitive, such as support vector machine [104] or decision tree [67]. The second category makes the training procedure cost-sensitive, which is typically achieved by assigning the training examples of different classes with different weights (rescaling) [139] or by changing the proportions of each class while training using sampling (rebalancing) [33]. The third category makes the prediction procedure cost-sensitive [30, 124]. Such direct cost-sensitive decision-making is the most generic: it considers the underlying classifier as a black box and extends to any number of classes and arbitrary cost matrices. Our work comes under the third category of post-hoc amendment. We study cost-sensitive classification in a large scale setting (e.g., ImageNet) and explore the use of a taxonomic hierarchy to obtain the misclassification costs.

2.2.2 Hierarchy Aware Classification

There is rich literature around exploiting hierarchies to improve the task of image classification. Embedding-based methods define each class as a soft embedding vector, instead of the typical one-hot vector. DeViSE [37] learn a transformation over image features to maximize the cosine similarity with their respective `word2vec` label embeddings. The transformation is learned using ranking loss and places the image embeddings in a semantically meaningful space. [2, 115] explore variations of text embeddings, and ranking loss frameworks. [4] project classes on a hypersphere, such that the correlation of class embeddings equals the semantic similarity of the classes. The semantic similarity is derived from the height of the lowest common ancestor (LCA) in a given hierarchy tree. Here, it may be relevant to note the work done on hyperbolic embeddings [82, 51, 66]. While we are unaware of any work that has been applied for "making better mistakes", it may be useful to explore this direction in the future since hyperbolic spaces are well-suited to embed hierarchies.

Another line of effort directly alters the loss functions or the algorithms/architectures. [133] propose a weighted (hierarchy-aware) multi-class logistic regression formulation. [107] optimize a context-sensitive loss to learn a separate distance metric for each node in the class taxonomy tree. [112] combine losses at different hierarchies of the tree by learning separate, fully connected layers for each level post a shared feature space. [11] add branches at different depths of AlexNet architecture to fuse losses at different levels of the hierarchy. [15] use conditional probability chains to derive a novel label encoding and a corresponding loss function.

Most deep learning-based methods overlook the severity of mistakes, and the evaluation revolves around counting the top-k errors. [7] has revived the interest in this direction by jointly analyzing the top-k accuracies with the severity of errors. They propose two modifications to cross-entropy to better capture the hierarchy: one based on label embeddings (Soft-labels) and the other, which factors the cross-entropy loss into the individual terms for each of the edges in the hierarchy tree and assigns different weights to them (Hierarchical cross-entropy or HXE).

We propose a generalized alternative. Our method uses models trained with vanilla cross-entropy loss and alters the decision rule to pick the class minimizing conditional risk. On similar lines, [29] study the effect of minimizing conditional risk on mean hierarchical cost. They leverage the ImageNet hierarchy for cost and compute posteriors by fitting a sigmoid function to the SVM's output or taking the percent of neighbours from a class for Nearest Neighbour classification. Our work investigates the relevance of CRM in the deep learning era and highlights the importance of looking beyond mean hierarchical costs and jointly analyzing the role of accuracy and calibration.

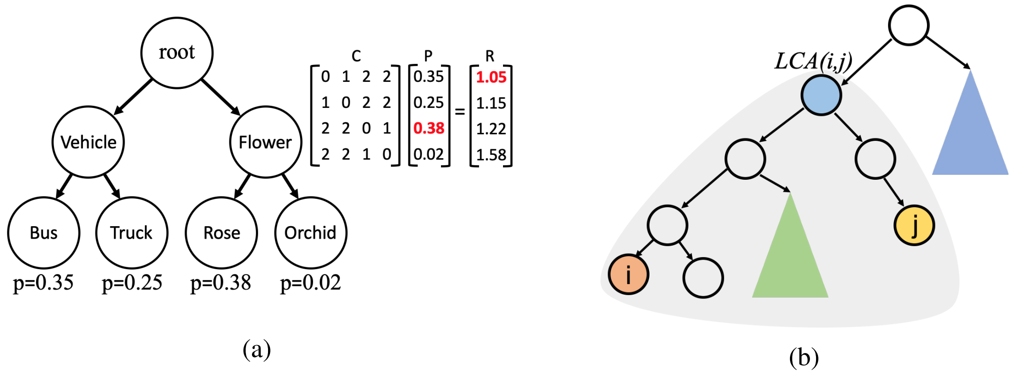


Figure 2.1: (a) Consider a four class hierarchy tree and corresponding leaf predictions obtained using a cross-entropy baseline. The risk computation is shown beside the tree. $\text{argmax}\, p(\mathbf{y}|\mathbf{x})$ predicts the class "rose", while the $\text{argmin}\ R(\mathbf{y} = k|\mathbf{x})$ predicts the class "bus". (b) Two nodes i and j and the subtree (shaded gray) originating at their lowest common ancestor $LCA(i, j)$.

2.2.3 Calibration of Deep Neural Networks

Networks are said to be well-calibrated if their predicted probability estimates are representative of the true correctness likelihood. Calibrated confidence estimates are important for model interpretability and its use in downstream applications. Platt scaling [85], Histogram binning [125] and Isotonic regression [126] are three common calibration methods. Although originally proposed for the SVM classifier, their variations are used in improving the calibration of neural networks [40]. Calibrated probability estimates are particularly important when cost-sensitive decisions are to be made [125] and are often measured using Expected Calibration Error (ECE) and Maximum Calibration Error (MCE) [79, 83].

We desire models with high accuracy that have low calibration error and make less severe mistakes. However, there is often a compromise. Studies in cost-sensitive classification [47] reveal a trade-off between costs and error rates. For instance, [7] trade the top-k error for a lower hierarchical cost. Similarly, reliability literature aims to obtain better calibrated deep networks while retaining top-k accuracy [96]. We further observe that methods like Soft-labels or Hierarchical cross-entropy successfully minimize the average top-k hierarchical cost, but result in poorly calibrated networks. In contrast, the proposed framework of post-hoc amendment retains top-k accuracies and good calibration, while significantly reducing the hierarchical cost.

2.3 Approach

The K-class classification problem comes with a training set $S = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{N}$, where label $\mathbf{y}_i \in \mathcal{Y} = \{1, 2, ..., K\}$. The classifier is a deep neural network parametrized by θ given by $f_\theta : \mathcal{X} \rightarrow p(\mathcal{Y})$ which maps the input samples to a probability distribution over the label space $\mathcal{Y}$. The $p(\mathbf{y}|\mathbf{x})$ is typically derived using a softmax function on the logits obtained for an input $\mathbf{x}$. Given $p(\mathbf{y}|\mathbf{x})$, the network minimizes cross-entropy with the ground truth class over samples from the training set, and uses SGD to optimize θ, forming the standard hierarchy-agnostic cross-entropy baseline. The decision rule is naturally given by $\underset{k}{\operatorname{argmax}}\, p(\mathbf{y}=k|\mathbf{x})$.

The classical CRM framework [31] can be adapted to image classification by taking the trained model with a given θ and incorporating the hierarchy information at deployment time. A symmetric class-relationship matrix $\mathbf{C}$ is created using the given hierarchy tree (which can either be drawn from the WordNet ontology or an application specific taxonomy), where $\mathbf{C}_{i,j}$ is the height of the lowest common ancestor ($LCA(\mathbf{y}_i, \mathbf{y}_j)$) between the classes i and j. The height is the number of edges between the given node and the furthest leaf. $\mathbf{C}_{i,j}$ is zero when $i = j$ and is bounded by the maximum height of the hierarchy tree.

Given an input $\mathbf{x}$, we obtain $p(\mathbf{y}|\mathbf{x})$ by passing it through the deep network $f_\theta(\mathbf{x})$. The only modification is in the decision rule, which now selects the class which minimizes the conditional risk $R(\mathbf{y}=k|\mathbf{x})$, given by:

$$\underset{k}{\operatorname{argmin}}\, R(\mathbf{y}=k|\mathbf{x}) = \underset{k}{\operatorname{argmin}} \sum_{j=1}^{K} \mathbf{C}_{k,j} \cdot p(\mathbf{y}=j|\mathbf{x})$$

For the ease of the reader, we illustrate a four-class example in Figure 2.1a, showing predictions from the cross-entropy baseline (leaf nodes); cost matrix for the given tree, and the risk computation. Given the probability of each class $p(\mathbf{y}|\mathbf{x})$, $\operatorname{argmin} R(\mathbf{y}|\mathbf{x})$ is the Bayes optimal prediction. It is guaranteed to achieve the lowest possible overall cost, i.e. lowest expected cost over all possible examples weighted by their probabilities [31, Ch. 2].

Depending on the cost-matrix and $p(\mathbf{y}|\mathbf{x})$, the top-1 prediction of the CRM applied on cross-entropy might differ from the top-1 prediction of the cross-entropy baseline. However, with deep neural networks, we observe that the top-1 probability of $p(\mathbf{y}|\mathbf{x})$ is greater than 0.5 in roughly 70% of the samples. Below we prove that in such situations where $\max p(\mathbf{y}|\mathbf{x})$ is higher than the sum of other probabilities, irrespective of the structure of the tree, the post-hoc correction (CRM) does not change the top-1 prediction. However, since the second highest probability is guaranteed to be less than 0.5 by definition, our correction can effectively re-rank the classes and drastically reduce hierarchical distance@k for $k > 1$ - as we later demonstrate through exhaustive experiments.

Theorem 1. *If* $\max(p(\mathbf{y}|\mathbf{x})) > 0.5$, *then* $\operatorname{argmin}_i \sum_{j=1}^{K} \mathbf{C}_{i,j} \cdot p(\mathbf{y}=j|\mathbf{x})$ *and* $\operatorname{argmax} p(\mathbf{y}|\mathbf{x})$ *are identical irrespective of the tree structure and lead to the same top-1 prediction.*

Proof. Consider the tree illustrated in Figure 2.1b; two leaf nodes (class labels) i,j and the subtree (T_{ij}) rooted at their Lowest Common Ancestor. Assuming the height of the $LCA(i,j)$ = h and $\operatorname{argmax} p(\mathbf{y}|\mathbf{x}) = i$, the risk $R(\mathbf{y}=j|\mathbf{x}) = R(j)$ is given as:

$$R(j) = h \cdot p(i) + \sum_{k \in T_{ij} \setminus \{i\}} C_{jk} \cdot p(k) + \sum_{\forall k \notin T_{ij}} C_{j,k} \cdot p(k)$$

Ignoring the cost of other nodes inside T_{ij}, we get $R(j) \geq h \cdot p(i) + \sum_{\forall k \notin T_{ij}} C_{j,k} \cdot p(k)$. Similarly, for the risk of class i:

$$R(i) \leq h \cdot (1 - p(i)) + \sum_{\forall k \notin T_{ij}} C_{i,k} \cdot p(k)$$

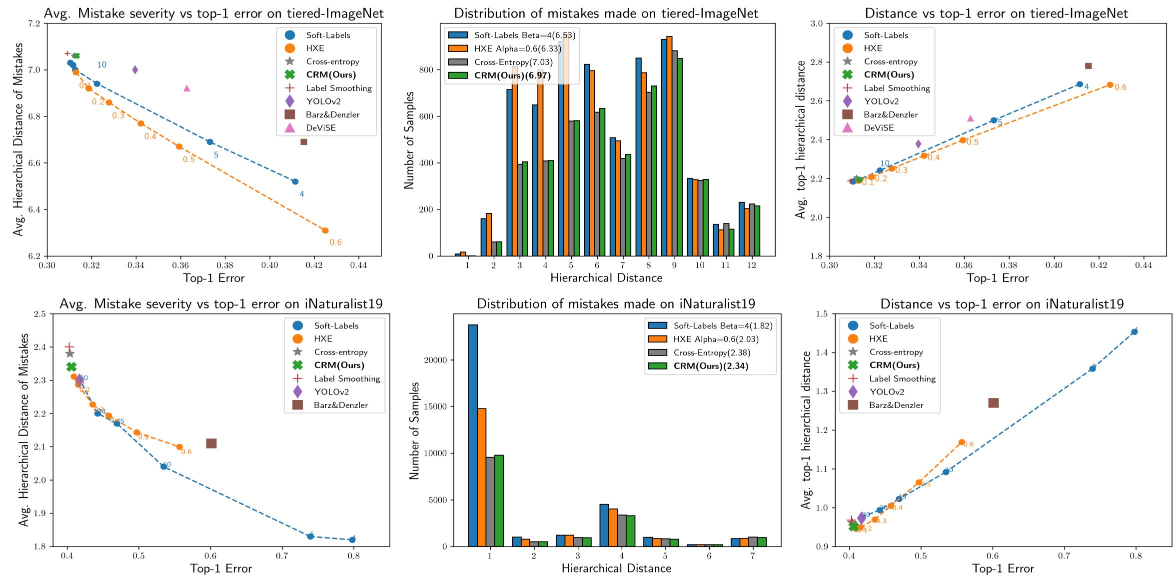
Outside the subtree rooted at $T_{i,j}$, $C_{i,k} = C_{j,k} \forall k$ and therefore without loss of generality we can say that $R(i) < R(j)$, if $p(i) > 0.5$. $\square$

2.4 Experiments

We evaluate our method on two large-scale hierarchy-aware benchmarks: (i) tieredImageNet-H for a broad range of classes and (ii) iNaturalist-H for fine-grained classification, both of which are complex enough to cover a large number of visual concepts. We closely follow the experimental pipeline from [7] including the train/validation/test splits, hyperparameters for training models, and evaluation metrics.

Experimental Details: All models are trained using a ResNet-18 architecture (pre-trained on ImageNet) using an Adam optimizer for 200K updates using a mini-batch of 256 samples, a learning rate of 10^{-5}, and standard data augmentation of flips and randomly resized crops. We train all the hierarchy-aware models – Hierarchical cross-entropy (HXE) [7], Soft-labels [7], YOLO-v2 [88], DeViSE [37], and [4] – along with a cross-entropy baseline. We pick the epoch corresponding to the lowest loss on the validation set along with two epochs preceding and succeeding it and report the average of the results obtained from these five checkpoints on the test set. Unlike [7] we do not preprocess the dataset to downsample the images to 224×224 as it noticeably reduces the accuracy. Instead we use `RandomResizedCrop()` augmentation to crop the images to a 224×224 resolution. This accounts for a small, but significant improvement in performance across models, thus leading to stronger baselines.

Metrics: We primarily focus on two major metrics: (i) top-1 error, (ii) average hierarchical distance@k, which is the mean LCA height between the ground truth and each of the k most likely classes. These metrics capture different views of the problem: top-1 error treats all classifier mistakes the same, whereas average hierarchical distance@1 captures a notion of mistake severity, i.e., *better* or *worse* mistakes. Average hierarchical distance@k captures the notion of ranking/ordering the predicted classes closer to the ground truth class. This metric is a natural extension of the hierarchical distance@1 metric proposed by [92] in the original ImageNet evaluation. We also investigate average mistake severity proposed by [7] which computes the hierarchical distance between the top-1 prediction and the ground truth for all the misclassified samples. Note that LCA is a log-scaled distance: an increment of 1.0 signifies an error of an entire level of the tree. In the simple case of a full binary tree, an increase by one level implies that the number of possible leaf nodes doubles.



(a) (b) (c)

Figure 2.2: (a) Trade-off between hierarchical distance@1 and the top-1 error. The mean here is taken only over the misclassified samples. (b) Histogram of the severity of top-1 mistakes (height of the LCA). The number in the bracket is the mean mistake severity. (c) Trade-off between hierarchical distance@1 and the top-1 error. The mean is taken over all the test samples (the hierarchical distance@1 is zero for correctly classified samples). The top and bottom row correspond to tiered-ImageNet and iNaturalist19 datasets respectively.

2.4.1 Hierarchical Distance of top-1 predictions

Hierarchy-aware classification methods typically seek to make better mistakes (less costly in terms of hierarchical distance). It is essential that the evaluation metric correctly measures this goal, i.e. a higher value of the evaluation metric should reflect that the model indeed makes better mistakes. We discover a serious flaw with the metric of computing average mistake severity over incorrectly classified samples employed in the prior art. We illustrate this in Figure 2.2.

In Figure 2.2a, following prior art, we evaluate different approaches only on the set of incorrectly classified samples (hence *different* test sets for different models as the mistakes will be different). It seems to indicate that recently proposed methods are able to achieve a good tradeoff between accuracy and mistake severity. We select the best performing models in terms of the mistake severity metric – Soft-labels with $\beta = 4$ and HXE with $\alpha = 0.6$ – and analyze the frequency of mistakes at different severity (illustrated in Figure 2.2b). Surprisingly, we observe that HXE and Soft-labels largely do not make better mistakes; they mostly make additional low-severity mistakes. This behaviour is not captured in Figure 2.2a as the metric here involves division by the number of mistakes made by a model. For example, say the high severity mistakes made by two models is exactly the same ($d_h > 0$) over the same number of mistakes ($m > 0$). Now, if the second model makes additional $n > 0$ mistakes with overall distance severity of $d_l > 0$, then it is straightforward to observe that $\frac{d_h}{m} \geq \frac{d_h + d_l}{m+n}$ if $\frac{d_h}{m} \geq \frac{d_l}{n}$. Implying,

the metric would prefer a model which is making additional low severity mistakes as long as the impact of the severity due to these additional mistakes is less than the impact by the high-severity ones.
We therefore revert to the hierarchical distance@1 proposed by [92] which computes the hierarchical distance over all the samples. As shown in Figure 2.2c, the best-performing ones in Figure 2.2a now show the highest hierarchical distance@1 as we account for the additional number of low-severity mistakes made. We refer the reader to Appendix (A.1)(Fig. A.1) for histogram plots across different values of α and β, clearly showing that the number of low-severity mistakes increases as the model moves farther from the cross-entropy baseline. In this modified and more reliable evaluation set-up, we observe CRM (ours) reduces mistake severity compared to cross-entropy, shifting the graver mistakes in cross-entropy towards the left of the histogram—resulting in a marginal shift of mistakes from buckets (9,10,12) to (3,6,7,8) in Imagenet and similarly from buckets (7,5,4) to (1) in iNaturalist19.
Overall, we observe that existing methods reduce the average mistake-severity metric by largely making additional low-severity or easily avoidable mistakes. This also explains why such models provide lower test accuracy. Resolving this issue, we see that no prior art significantly outperforms the cross-entropy baseline, and make better mistakes.

2.4.2 Hierarchical Distance of top-k predictions

We now compare the ordering of classes provided by each of these classifiers. Ranking predictions give us significant insight into how reliably the predictions align with the hierarchy, as discussed in Section 1. We measure the quality of ranking using the average hierarchical distance@k, for various values of k and present them in Figure 2.3a (left). We find that CRM significantly outperforms all the competing methods, giving the best hierarchically aligned models on the hierarchical distance@k for all k.
A better ranking of classes often comes with a significant trade-off with top-1 accuracy. We plot the hierarchical distance@k with top-1 accuracy for $k = 5$ and $k = 20$ in Figure 2.3a (middle and right) to better understand this trade-off. Interestingly, we observe that CRM improves ranking with almost no loss in top-1 performance and outperforms the other methods by a substantial margin.
An interesting extension is to analyze how dependent these approaches are to a given hierarchy, and how modifying the hierarchy might impact their behaviour. To test this, we randomly shuffle the classes at the leaf nodes of a given tree structure and compare ranking performance in Figure 2.3b. We observe that even though CRM does not explicitly use the hierarchy while training, it provides drastic reduction in the hierarchical distance@k compared to all the previous methods. High accuracy of CRM in this case is because of the fact that it is post-hoc and for highly confident models such as deep networks, its top-1 accuracy remains largely unchanged (refer Theorem 1). However, on the other hand, models depending on the tree-structure during training (directly or indirectly) will try to fit to the structure, which can be harmful in situations where the tree structure is not very reliable. For example, if the tree structure implies that 'cat' is closer to 'person' than a 'dog', then the models incorporating such information while learning the feature space might not be able to learn a robust classifier and might potentially end-up making more mistakes, as also validated in our experiments.

2.4.3 Impact of Label Hierarchy on the Reliability

Via various examples and insights, we have already established that models that make better mistakes are extremely important for real-world applications. However, in order for such models to be useful in safety-critical scenarios, they must be reliable. In other words, these models should be calibrated so that they are not wrong with high confidence—an extremely undesirable property of recent deep models. To this end, we analyse the reliability of the output probabilities of Softlabels, HXE, label smoothing, and CRM (which is same as cross-entropy) using well known and widely accepted metrics such as ECE (Expected Calibration Error) and MCE (Maximum Calibration Error) in Table 2.1. Softlabels and HXE, for example, show clear trends of increasing degradation in calibration on better class ranking (as measured by distance@k), i.e., *the more they attempt to adhere to the hierarchy, the less reliable their probability estimates become*. We additionally experiment with improving calibration in all the above models using temperature scaling. We observe that it reduces miscalibration as measured by the ECE and MCE scores, but most models remain far worse than the cross-entropy baseline. The cross-entropy baseline is not affected significantly by temperature scaling, indicating that it is calibrated to a large extent. Changes in ECE/MCE were unnoticeable when using the probability estimates corresponding to CRM predictions (taking $p(\mathbf{y}|\mathbf{x})$ corresponding to $\operatorname{argmin} R(\mathbf{y}|\mathbf{x})$) instead of maximum cross-entropy prediction.

These experiments clearly suggest that while the focus should turn into developing models that make better mistakes, we should also make sure that such models are reliable by understanding how incorporating the label hierarchy during training might impact the likelihood estimates.

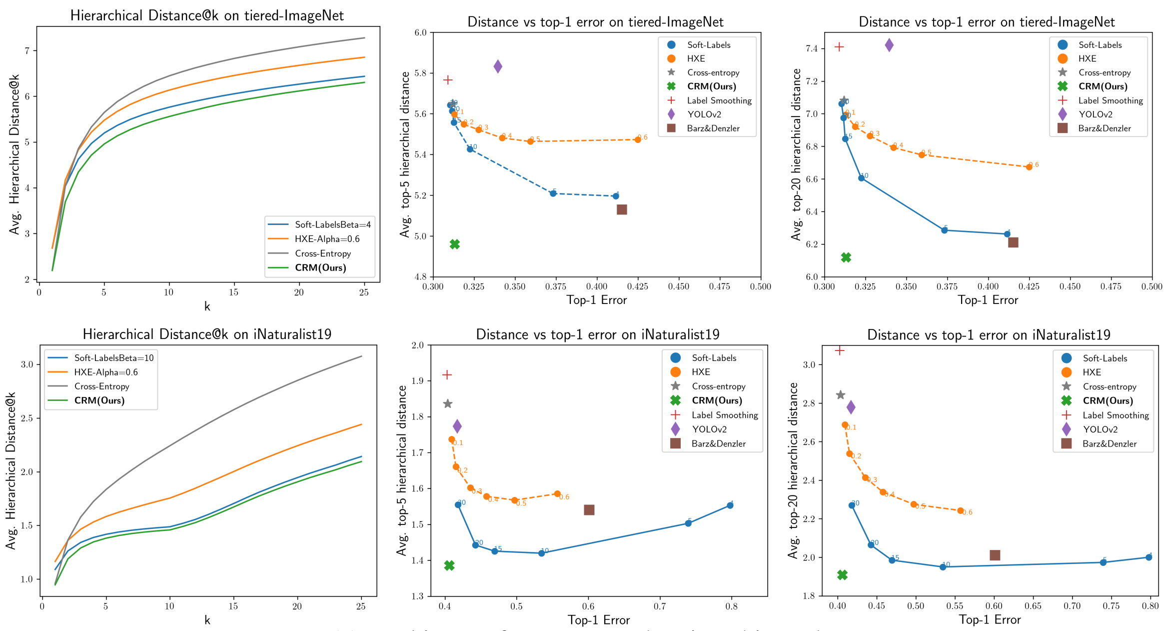
2.5 Conclusion

Our paper identifies two major problems in existing approaches for tackling hierarchy-aware classification: (i) making better mistakes and (ii) better ranking predictions. We study post-hoc amendments for hierarchy aware image classification, different from the three dominant paradigms: hierarchy-aware losses, hierarchy-aware architectures, and label embedding methods. We make a case for exhausting classical alternatives before moving to more sophisticated options.

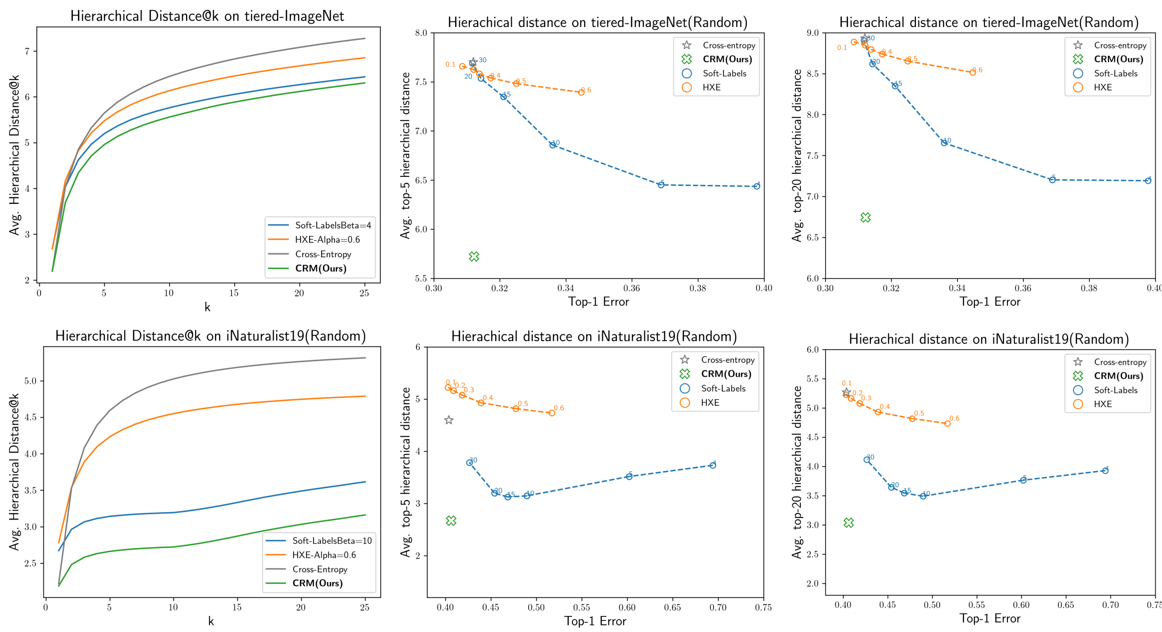
Regarding making better mistakes, we demonstrate that no recent hierarchy-aware classifier helps make better mistakes, contrary to the motivation. We illustrate and correct the issue with the mistake severity metric and show that CRM achieves modest improvements. We provide insights on why it is hard to improve further in this direction under similar constraints.

In terms of better ranking predictions, our proposed post-hoc correction consistently outperforms state-of-the-art methods in deep hierarchy-aware image classification by large margins in terms of decrease in hierarchical distance@k, with little to no loss in top-1 accuracy. We find the direction of post-hoc corrections promising because it can simultaneously deliver calibration, accuracy, and better class ranking efficiently with surprisingly little tradeoffs in either. We hope that CRM will act as a strong baseline in future efforts.

Overall, the literature on hierarchy-aware image classification has shown the effectiveness of the WordNet hierarchy in improving performance. However, previous works assume that all edges in the tree are equally important. A future avenue for exploration is learning weights of the edges in the tree to compute a more effective measure of misclassification cost.



(a) Ranking performance on the given hierarchy



(b) Ranking performance on a randomly class-shuffled hierarchy.

Figure 2.3: Left: Average hierarchical distance@k with varying k. Middle: Average hierarchical distance@5 vs top-1 error. Right: Average hierarchical distance@20 vs top-1 error.

Loss Function	tieredImageNet-H				iNaturalist-H			
	ECE		MCE		ECE		MCE	
	pre T	post T	pre T	post T	pre T	post T	pre T	post T
Soft-labels (β=15)	7.05%	4.79%	18.55%	13.03%	34.64%	16.63%	54.86%	26.87&
Soft-labels (β=10)	29.55%	6.36%	39.95%	22.07%	29.55%	19.87%	39.95%	33.29%
Soft-labels(β=5)	58.99%	10.92%	83.53%	26.86%	24.53%	17.20	88.32%	55.68%
Soft-labels (β=4)	57.16%	11.12%	86.92%	27.44%	19.29%	11.46%	19.29%	56.06%
HXE (α=0.2)	1.53%	1.53%	5.84%	5.84%	4.37%	1.50%	7.73%	3.61%
HXE (α=0.4)	2.44%	2.44%	5.48%	5.48%	1.13%	1.13%	2.62%	2.62%
HXE(α=0.5)	3.95%	2.61%	7.84%	5.40%	2.46%	2.46%	6.77%	6.77%
HXE (α=0.6)	6.25%	3.28%	10.75%	6.58%	5.24%	5.24%	11.20%	11.20%
Label-Smoothing	9.61%	2.33%	15.43%	6.13%	4.93%	1.11%	7.35%	3.35%
Cross-Entropy	1.61%	1.61%	4.27%	4.27%	4.32%	1.42%	8.18%	3.26%

Table 2.1: ECE and MCE for the various models on the tiered-ImageNet and iNaturalist19 datasets before and after temperature scaling. The optimal temperature was found on the validation set and results reported on the test set.

Chapter 3

Exploring 3 R's of Long-term Tracking: Re-detection, Recovery and Reliability

3.1 Introduction

Visual tracking is a fundamental problem in computer vision and has rapidly progressed in the recent past with the onset of deep learning. However, progress is still far from matching practitioner needs, which demands consistent and reliable long-duration tracking. Interestingly, most existing works evaluate their performance on datasets consisting of multiple short clips. For instance, the most commonly used OTB dataset has an average length of about 20 seconds [114] per clip. Work by Moudgil and Gandhi [77] observed a sharp performance drop when the trackers were evaluated on long sequences. Following works [34, 106, 69] also make similar observations and suggest that we need alternate ways to evaluate and analyze long term tracking performance.

Based on these works [77, 34, 106, 69] we hypothesize that three properties are crucial for an improved long term tracking performance. First is the ability to re-detect the target if it is lost. *Re-detection* is crucial to handle situations where the target object goes out of the frame and reappears. It is also essential to re-initiate tracking when the target object is lost due to occlusions or momentary tracking failures. The second key aspect is the ability of the tracker to distinguish between the actual target and distractor or background clutter. This aspect is vital for consistency in tracking as well as for *recovery* from failures. Figure 3.1 illustrates an example where chance plays a crucial role in recovery. We believe that scrutinizing the nature of failures and recoveries will aid improved tracking performance. The third key aspect is *Reliability*, which connects to the ability for consistent and contiguous tracking. Contiguity suggests the ability of the tracker to track for a long duration without any failure. Consistency indicates the accuracy of tracking over time. Tracking in the long-duration video allows us to study factors like a slow accumulation of error which is difficult to observe in short sequences. Several applications like video surveillance or virtual camera simulation from static camera [39] require precise tracking for long time. Surprisingly, none of the current evaluation strategies focus on these three crucial aspects of *Re-detection*, *Recovery*, and *Reliability*.

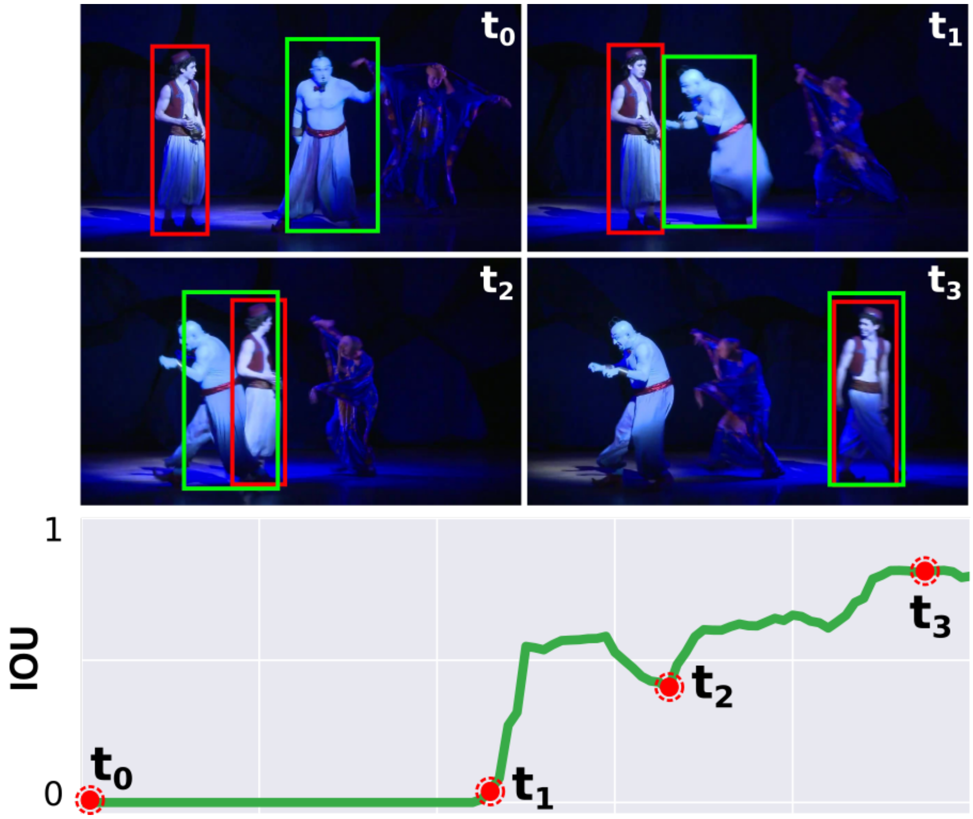


Figure 3.1: A typical example of a chance based recovery in Alladin sequence from TLP [77] dataset. SiamRPN(green) is tracking the incorrect object and has zero overlap with the target(red) in the start. It switches to tracking the target when they pass through each other. We study such chance based recoveries in long-term setting both qualitatively and quantitatively. Best viewed in colour.

For instance, the most prevalent metrics are Success and Precision plots, which measure the number of frames with Intersection Over Union (IoU) greater than a threshold and the mean distance from the center of the ground truth respectively. Both these metrics do not reflect anything specific about the 3R's. Recent work by Lukezic *et al.* [69] studied the efficacy of the search region expansion strategy of different trackers. However, the evaluation is performed in a synthetic experimental setup (designed by padding with gray values) and may not be an indicator of performance in real-world scenarios. Valmadre *et al.* [106] improves the evaluation strategy by explicitly handling the cases where the target is not visible/absent from the frame. Other recent efforts [77, 34] identify the aforementioned key issues, but they do not provide any way to evaluate these properties comprehensively.

In this work, we propose two novel evaluation metrics focused on the re-detection ability and the aspect of continuous and consistent long term tracking. Furthermore, we present more in-depth insights into the failure and recovery of different trackers, explicitly addressing the role of distractors. Since shorter sequences are inappropriate to address these concerns, we use the Track Long and Prosper (TLP) [77]

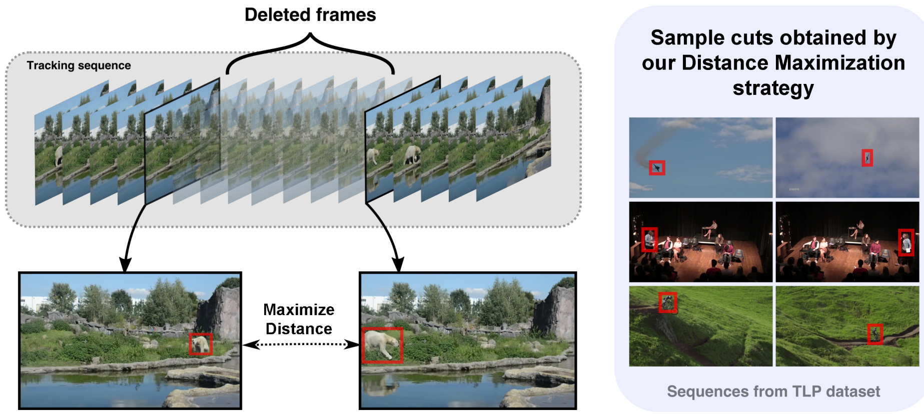


Figure 3.2: A cut is introduced by removing a set of contiguous frames from a tracking sequence. This introduces a sudden change of position of the ground truth object as shown in the left diagram. The red bounding box shows the position of the target object, before and after the cut. We maximize the amount of target shift by minimizing the distance between these bounding boxes. We evaluate the trackers ability to re-detect the object after the cut. Few more examples from TLP dataset with simulated cuts are shown on the right.

dataset for the experiments. The main advantage of TLP is that the average sequence length is longest among other densely annotated datasets [45, 34, 69]. Long duration videos present cases of multiple failures and recoveries for each video, which allows for a deeper analysis. Our contributions include:

1. We propose a novel way to quantitatively evaluate the re-detection abilities of a tracker by simulating cuts (an abrupt transition from a frame to another) in original videos. We propose a method to search challenging locations for placing the cut by maximizing the distance between the ground truth bounding boxes in the frame before and after the cut. Different trackers are then evaluated on their ability to recover/re-detect, and the time they take to recover.

2. We formally study the chance factor in recoveries post-failure. We analyze the role of distractors in failures and recoveries and the co-incidences which aid tracking. For example, it often happens in long sequences that tracker loses the target at some location and freezes there. If by chance the target passes the same location (after a while), the tracker starts tracking it again. Our study aims to quantify such behavior.

3. We propose 3D Longest Subsequence Measure (3D-LSM), as a novel metric for quantifying tracking performance. It measures the longest contiguous sequence successfully tracked at a given precision and allowed failure tolerance. The 3D-LSM allows for a direct visual interpretation of tracking results in the form of a 2D image.

3.2 Related Work

Tracking Datasets: There are a large variety of datasets for object tracking. Most commonly used datasets are OTB50[114] and OTB100[113]. They consist of short videos from generic real-world scenarios. ALOV300[99] increased diversity by including 300 short sequences. The average video length in ALOV dataset is only 9 seconds. NFS[38] dataset included sequences recorded at high frame rate (240fps). UAV[78] introduced a dataset from sequences shot from an aerial vehicle. Moudgil and Gandhi[77] TLP dataset with 50 sequences, focusing on long-duration tracking (significantly increasing length of individual sequences). LTB35[69] and OxUvA[106] then followed emphasizing the need to focus on long term tracking. LaSOT[34] significantly increased the size of the dataset with over 1000 sequences. GOT10k[45] then followed by proposing a dataset with 10000 sequences, including objects from 563 different classes.

Tracking Methods: We list some notable attempts which led to significant progress in long term tracking. Collins *et al.* [25] proposed the idea of using the neighborhood around the ground truth for discriminative feature learning. This idea was later formalized into tracking by detection frameworks [3]. Kalal *et al.* [49] proposed TLD framework of learning detector from initial tracking, maintaining the confidence of local tracking based on feature point tracks, and switching to detection in low confidence scenarios. TLD tracker was one of the first attempts to elegantly handle the re-detection problem, which is crucial for long term tracking. The consistency aspect was then improved by employing an ensemble of classifiers [128]. These methods maintain several weak classifiers, often initiated at different checkpoints to account for appearance variations of the target.

Another popular direction is Discriminative Correlation Filter (DCF) based tracking [13, 28]. These methods exploit the properties of circular correlation (efficiently performed in Fourier domain) for training a regressor in a sliding-window fashion. Recent progress in DCF's is driven by integrating multi-resolution shallow and deep features maps to learn the correlation filters [27, 10, 105]. Another fundamental contribution is the use of Siamese networks for visual object tracking [8, 42]. The GOTURN tracker [42] uses the Siamese architecture to directly regress the bounding box locations given two cropped images from previous and current frames. The SiamFC tracker [8] transforms the exemplar image and the large search image using the same function and outputs a map by computing similarity in the transformed domain. These efforts [8, 42] do not include any online updates and are extremely efficient in terms of computation. The Siamese framework was further augmented by employing Region Proposal Networks (RPN)[60, 61] which significantly improves the accurate prediction of scale and aspect ratio of the bounding boxes.

Another pioneering effort came from Nam *et al.* [80], who introduced the idea of treating the tracking problem as classifying candidate windows sampled around the previous target region. Several recent efforts have explored the variations of the Tracking Learning Detection (TLD) framework. Nebehay *et al.* [81] proposed a mechanism to drift by filtering outlier correspondences. A combination of short term CF tracker with additional components (e.g., an explicit re-detection module) have been explored [68, 71]. Zhang *et al.* [132] employed an offline trained regression network as the short-term component and an online-trained verification network to detect tracking failure and start image wide detection. Yan *et al.* [119] show significant computational improvements by replacing the online verification network with an offline trained Siamese verification network.

Tracking Metrics: Early works relied on the precision metric[3, 114] for quantifying the tracking performance, which computes the pixel distance between the center of the ground truth and the prediction. This was convenient since it required only annotating the center of the target and not the whole bounding box. However, since this does not account for the scale and aspect ratio, the success metric[114] was introduced. It measures the percentage of frames where the Intersection Over Union (IoU) of the predicted and ground truth bounding boxes is more than a threshold. Failure rate [55] was then introduced to address the continuity and consistency aspect of tracking. In failure rate measure, a manual operator reinitializes the tracker upon every failure. The number of required manual interventions per frame is recorded as the quantitative measure. However, due to the need of manual interventions, it is unscalable for long sequences (in large datasets). For a more detailed review and analysis of metrics for short-term tracking, we would refer the reader to work by Cehovin *et al.* [16].

A few evaluation metrics have been proposed targeting long-duration tracking. Valmadre *et al.*[106] introduced True Positive Rate(TPR), True Negative Rate(TNR) and took their geometric mean. To have a single representative metric accounting for the trackers which do not predict absent labels, they proposed a modified metric called maximum geometric mean metric. However, the metric is biased towards the ability of a tracker to predict absent labels.

Lukezic *et al.*[69] introduced tracking recall and precision and used this to give a tracking F1 score. However, their definition of a long term tracker is limited to the ability of a tracker to predict absence, and the proposed metric does not focus on the continuity and consistency aspect of tracking. We believe the ability to track for long-duration consistently even when the target object is always present has been overlooked in these previous efforts [69, 106]. Lukezic *et al.*also proposed an experiment to quantify the re-detection ability of a tracker. However, their experiment mainly focuses on the search strategy with no appearance changes. Here, we seek to quantify the re-detection ability in the wild. Moudgil and Gandhi [77] proposed the Longest Subsequence Measure (LSM), which quantifies the longest contiguous segment successfully tracked in the sequence. Here, we propose an extension of it called 3D-LSM, which allows comparing trackers visually.

3.3 Re-detection in the Wild

This experiment is designed to quantify a tracker's ability to re-detect the object after it is lost (either because the target goes of the view or due to momentary failures).

Setup: We select a segment from a sequence, and delete it, thereby introducing a cut (illustrated in Figure 3.2). We evaluate the tracker's performance on the segment after the cut to evaluate the re-detection ability of the tracker. For each sequence from the TLP dataset, we cut a segment that maximizes the L2 norm of the center locations between the target bounding boxes before and after the cut. The duration of the cut is fixed to 300 frames. We empirically find that 300 frames allow the target to move far away from the tracker's search region without significantly varying the other aspects in the scene. Keeping a similar context around the target helps to keep the focus on the re-detection ability (the context can change dramatically in long sequences if the length of the omitted sequence is large). The proposed re-detection scheme is quite general and can be applied even on datasets that do not have target disappearances at all.

Evaluation: In all the experiments the tracker is initialized 100 frames before the cut. We choose 100 frames so that the tracker starts stable tracking before the cut. It also allows trackers with online updates to build a reasonable representation of the target object. We also make sure that there are no critical challenges in this duration of 100 frames such as heavy occlusion, clutter, etc. to avoid tracker failure in these 100 frames. After the cut, the tracker is continued to run on the sequence for another 200 frames and its performance on this segment is evaluated. We define "recovery" when the IoU of the tracker with the target reaches 0.5. To make a relative comparison of the trackers on the re-detection task, we report the following metrics.

1. Total number of sequences (out of the total 50 TLP sequences) in which a tracker is able to recover within the remaining 200 frames.

2. Total number of sequences where the recovery is "quick," i.e., the recovery happens within 30 frames (1 second).

3. Average number of frames a tracker takes to recover successfully.

We perform this experiment on TLP dataset with the following trackers: SPLT [119], MBMD [132], FuCoLoT [68], ATOM [26], MDNet [80], SiamRPN [61], ECO [27], CMT [81], LCT [71], and TLD [49]. SPLT, MBMD, FuCoLoT, CMT, LCT and TLD are long-term trackers with explicit re-detection ability; ATOM is the current top performing tracker on the long-term benchmark LaSOT, while MDNet, SiamRPN and ECO are the top performing trackers on other benchmarks [77, 113, 56]. This selection presents all the prevalent tracking approaches: correlation filter based trackers [27, 71, 68], end to end classification with online updates [80], offline trained Siamese trackers with region proposals [61], low level feature tracking with online learned detector [49, 81] and combination of multiple offline/online trained components [26, 132, 119]. The same set of trackers are used in all the following experiments as well.

Results and Discussion: Our results are summarized in Table 3.1. SPLT gives the best results, followed by FuCoLoT and MBMD. Since the base framework of SPLT and MBMD is the same as SiamRPN, the significant improvements (from SiamRPN to SPLT) can be attributed to the additional verification and re-detection module. An explicit re-detection module also improves CF-based trackers (as seen in FuCoLoT). CMT and TLD dominate in re-detection experiments studied in previous works [69]; however, they give poor results in our experiment. We empirically observe that CMT and TLD fail to adapt to appearance changes that occur before and after the cut, possibly because of the weak appearance

Tracker	Quick recoveries ↑	Total recoveries ↑	Avg. recovery length (# frames) ↓
SPLT [119]	**20**	**36**	19
FuCoLoT [68]	10	33	55
MBMD [132]	15	27	28
ATOM [26]	12	25	34
CMT [81]	9	14	22
TLD [49]	6	10	**8**
MDNet [80]	5	13	48
ECO [27]	4	7	28
SiamRPN [61]	2	7	39
LCT [71]	2	7	143

Table 3.1: Results for the re-detection experiment (out of 50 sequences).

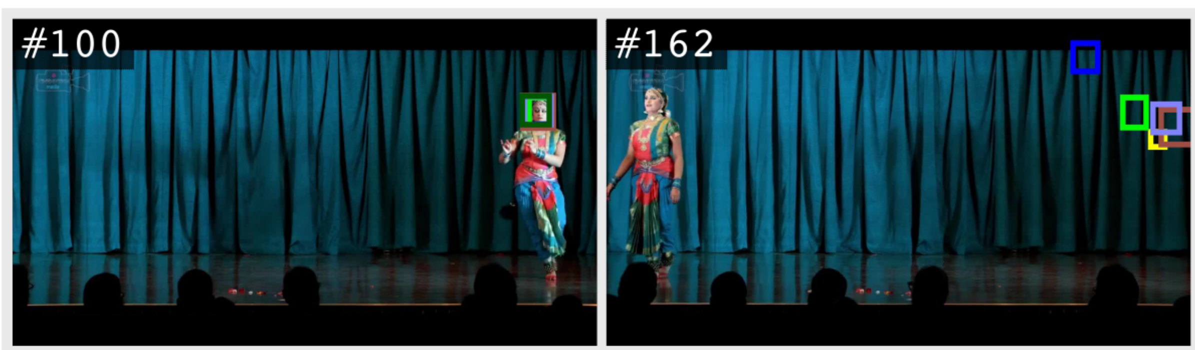


Figure 3.3: The figure illustrates a simulated cut in the Bharatanatyam sequence from TLP dataset. The cut can be seen as a representation of a situation where the performer exits the stage and enters from another end. None of the 6 trackers was able to recover in this sequence, even with the exact same background and a single target object.

model used in the detector. Adapting to appearance changes during re-detection is essential in real-world settings and previous synthetically designed experiments [69] do not account for this aspect. Other trackers like ECO, MDNet, and SiamRPN are limited by their search range and only recover if the target object comes within their search range after the cut. ATOM, on the other hand, uses a larger search area (25 times the area of target object bounding box) and hence recovers more often. Qualitatively, we observe sequences with background clutter or with distractors prove to be the most challenging for all the trackers. Re-detection results are also poor on targets that are small in size.

Related Publications

1. **Shyamgopal Karthik**, Ameya Prabhu, Puneet K. Dokania, Vineet Gandhi. *No Cost Likelihoood Manipulation for Making Better Mistakes.* In the **Ninth International Conference on Learning Representations (ICLR)** 2021. Accepted.

2. **Shyamgopal Karthik**, Abhinav Moudgil, Vineet Gandhi. *Exploring 3 R's of Long-term Tracking: Re-detection, Recovery and Reliability*. In the **IEEE Winter Conference on Applications of Computer Vision (WACV)** 2020.

3. **Shyamgopal Karthik**,Ameya Prabhu, Vineet Gandhi. *Simple Unsupervised Multi-Object Tracking*. In **Arxiv** 2020.

Bibliography

[1] N. Abe, B. Zadrozny, and J. Langford. An iterative method for multi-class cost-sensitive learning. In *KDD*, 2004.

[2] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015.

[3] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 983–990. IEEE, 2009.

[4] B. Barz and J. Denzler. Hierarchy-based image embeddings for semantic image retrieval. In *WACV*, 2019.

[5] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking without bells and whistles. In *ICCV*, pages 941–951, 2019.

[6] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008.

[7] L. Bertinetto, R. Mueller, K. Tertikas, S. Samangooei, and N. A. Lord. Making better mistakes: Leveraging class hierarchies with deep networks. In *CVPR*, 2020.

[8] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016.

[9] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *ICIP*, pages 3464–3468, 2016.

[10] G. Bhat, J. Johnander, M. Danelljan, F. Shahbaz Khan, and M. Felsberg. Unveiling the power of deep tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 483–498, 2018.

[11] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren. Do convolutional neural networks learn class hierarchy? *IEEE transactions on visualization and computer graphics*, 2017.

[12] E. Bochinski, V. Eiselein, and T. Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, pages 1–6, 2017.

[13] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2544–2550. IEEE, 2010.

[14] G. Brasó and L. Leal-Taixé. Learning a neural solver for multiple object tracking. In *CVPR*, 2020.

[15] C.-A. Brust and J. Denzler. Integrating domain knowledge: using hierarchies to improve deep classifiers. In *ACPR*, 2019.

[16] L. Čehovin, M. Kristan, and A. Leonardis. Is my new tracker really better than yours? In *IEEE Winter Conference on Applications of Computer Vision*, pages 540–547. IEEE, 2014.

[17] V. Chari, S. Lacoste-Julien, I. Laptev, and J. Sivic. On pairwise costs for network flow multi-object tracking. In *CVPR*, pages 5537–5545, 2015.

[18] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, et al. Hybrid task cascade for instance segmentation. In *CVPR*, pages 4974–4983, 2019.

[19] L. Chen, H. Ai, Z. Zhuang, and C. Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*, pages 1–6, 2018.

[20] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, pages 3029–3037, 2015.

[21] W. Choi and S. Savarese. A unified framework for multi-target tracking and collective activity recognition. In *ECCV*, pages 215–230, 2012.

[22] P. Chu, H. Fan, C. C. Tan, and H. Ling. Online multi-object tracking with instance-aware tracker and dynamic model refreshment. In *WACV*, pages 161–170, 2019.

[23] P. Chu and H. Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *ICCV*, pages 6172–6181, 2019.

[24] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera. Deep learning in video multi-object tracking: A survey. *Neurocomputing*, 2020.

[25] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *IEEE transactions on pattern analysis and machine intelligence*, 27(10):1631–1643, 2005.

[26] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. Atom: Accurate tracking by overlap maximization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4660–4669, 2019.

[27] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017.

[28] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 4310–4318, 2015.

[29] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei. What does classifying more than 10,000 image categories tell us? In *ECCV*, 2010.

[30] P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *KDD*, 1999.

[31] R. O. Duda and P. E. Hart. Pattern recognition and scene analysis, 1973.

[32] I. Elezi, S. Vascon, A. Torcinovich, M. Pelillo, and L. Leal-Taixe. The group loss for deep metric learning. *arXiv preprint arXiv:1912.00385*, 2019.

[33] C. Elkan. The foundations of cost-sensitive learning. In *IJCAI*, 2001.

[34] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5374–5383, 2019.

[35] K. Fang, Y. Xiang, X. Li, and S. Savarese. Recurrent autoregressive networks for online multi-object tracking. In *WACV*, pages 466–475, 2018.

[36] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, pages 1–8, 2008.

[37] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NeurIPS*, 2013.

[38] H. K. Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey. Need for speed: A benchmark for higher frame rate object tracking. *arXiv preprint arXiv:1703.05884*, 2017.

[39] V. Gandhi, R. Ronfard, and M. Gleicher. Multi-clip video editing from a single viewpoint. In *Proceedings of the 11th European Conference on Visual Media Production*, page 9. ACM, 2014.

[40] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *ICML*, 2017.

[41] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[42] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, pages 749–765. Springer, 2016.

[43] R. Henschel, L. Leal-Taixé, D. Cremers, and B. Rosenhahn. Improvements to frank-wolfe optimization for multi-detector multi-object tracking. *arXiv preprint arXiv:1705.08314*, 2017.

[44] R. Henschel, L. Leal-Taixé, D. Cremers, and B. Rosenhahn. Fusion of head and full-body detectors for multi-object tracking. In *CVPR-W*, pages 1428–1437, 2018.

[45] L. Huang, X. Zhao, and K. Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *arXiv preprint arXiv:1810.11981*, 2018.

[46] P. Huang, S. Han, J. Zhao, D. Liu, H. Wang, E. Yu, and A. C. Kot. Refinements in motion and appearance for online multi-object tracking. *arXiv preprint arXiv:2003.07177*, 2020.

[47] T.-K. Jan, D.-W. Wang, C.-H. Lin, and H.-T. Lin. A simple methodology for soft cost-sensitive classification. In *KDD*, 2012.

[48] L. Jing and Y. Tian. Self-supervised visual feature learning with deep neural networks: A survey. *arXiv preprint arXiv:1902.06162*, 2019.

[49] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409–1422, 2011.

[50] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *TPAMI*, 42(1):140–153, 2018.

[51] V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky. Hyperbolic image embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6418–6428, 2020.

[52] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *ICCV*, pages 4696–4704, 2015.

[53] C. Kim, F. Li, and J. M. Rehg. Multi-object tracking with neural gating using bilinear lstm. In *ECCV*, pages 200–215, 2018.

[54] M. Kim, S. Alletto, and L. Rigazio. Similarity mapping with enhanced siamese network for multi-object tracking. *arXiv preprint arXiv:1609.09156*, 2016.

[55] M. Kristan, S. Kovacic, A. Leonardis, and J. Pers. A two-stage dynamic model for visual tracking. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(6):1505–1520, 2010.

[56] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey, et al. The sixth visual object tracking vot2018 challenge results. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 0–0, 2018.

[57] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler. Learning by tracking: Siamese cnn for robust target association. In *CVPR-W*, pages 33–40, 2016.

[58] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*, 2015.

[59] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *ICCV-W*, pages 120–127, 2011.

[60] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. *arXiv preprint arXiv:1812.11703*, 2018.

[61] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, 2018.

[62] M. Li, X. Zhu, and S. Gong. Unsupervised person re-identification by deep learning tracklet association. In *ECCV*, pages 737–753, 2018.

[63] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *CVPR*, 2014.

[64] X. Li, A. Wu, and W.-S. Zheng. Adversarial open-world person re-identification. In *ECCV*, pages 280–296, 2018.

[65] Y. Lin, X. Dong, L. Zheng, Y. Yan, and Y. Yang. A bottom-up clustering approach to unsupervised person re-identification. In *AAAI*, pages 8738–8745, 2019.

[66] S. Liu, J. Chen, L. Pan, C.-W. Ngo, T.-S. Chua, and Y.-G. Jiang. Hyperbolic visual embedding learning for zero-shot recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9273–9281, 2020.

[67] S. Lomax and S. Vadera. A survey of cost-sensitive decision tree induction algorithms. *ACM Computing Surveys (CSUR)*, 2013.

[68] A. Lukežič, L. Č. Zajc, T. Vojíř, J. Matas, and M. Kristan. Fucolot–a fully-correlational long-term tracker. In *Asian Conference on Computer Vision*, pages 595–611. Springer, 2018.

[69] A. Lukežič, L. Č. Zajc, T. Vojíř, J. Matas, and M. Kristan. Now you see me: evaluating performance in long-term visual tracking. *arXiv preprint arXiv:1804.07056*, 2018.

[70] C. Ma, C. Yang, F. Yang, Y. Zhuang, Z. Zhang, H. Jia, and X. Xie. Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-gru for multiple object tracking. In *ICME*, pages 1–6, 2018.

[71] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5388–5396, 2015.

[72] L. Ma, S. Tang, M. J. Black, and L. Van Gool. Customized multi-person tracker. In *ACCV*, pages 612–628, 2018.

[73] A. Maksai and P. Fua. Eliminating exposure bias and metric mismatch in multiple object tracking. In *CVPR*, pages 4639–4648, 2019.

[74] S. Manen, M. Gygli, D. Dai, and L. Van Gool. Pathtrack: Fast trajectory annotation with path supervision. In *ICCV*, pages 290–299, 2017.

[75] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[76] A. Milan, L. Leal-Taixé, K. Schindler, and I. Reid. Joint tracking and segmentation of multiple targets. In *CVPR*, pages 5397–5406, 2015.

[77] A. Moudgil and V. Gandhi. Long-term visual object tracking benchmark. *arXiv preprint arXiv:1712.01358*, 2017.

[78] M. Mueller, N. Smith, and B. Ghanem. A benchmark and simulator for uav tracking. In *European conference on computer vision*, pages 445–461. Springer, 2016.

[79] M. P. Naeini, G. F. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *AAAI*, 2015.

[80] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.

[81] G. Nebehay and R. Pflugfelder. Clustering of static-adaptive correspondences for deformable object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2784–2791, 2015.

[82] M. Nickel and D. Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347, 2017.

[83] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. In *ICML*, 2005.

[84] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. Aggarwal, H. Lee, L. Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR*, pages 3153–3160, 2011.

[85] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 1999.

[86] J. Redmon, S. Divvala, R. Girshick, and A. 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.

[87] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016.

[88] J. Redmon and A. Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017.

[89] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[90] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, pages 91–99, 2015.

[91] E. Ristani and C. Tomasi. Features for multi-target multi-camera tracking and re-identification. In *CVPR*, pages 6036–6046, 2018.

[92] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[93] O. Russakovsky, L.-J. Li, and L. Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2121–2131, 2015.

[94] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *ICCV*, pages 300–311, 2017.

[95] S. Schulter, P. Vernaza, W. Choi, and M. Chandraker. Deep network flow for multi-object tracking. In *CVPR*, pages 6951–6960, 2017.

[96] S. Seo, P. H. Seo, and B. Han. Learning for single-shot confidence calibration in deep neural networks through stochastic inferences. In *CVPR*, 2019.

[97] J. Si, H. Zhang, C.-G. Li, J. Kuen, X. Kong, A. C. Kot, and G. Wang. Dual attention matching network for context-aware feature sequence based person re-identification. In *CVPR*, pages 5363–5372, 2018.

[98] C. N. Silla and A. A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 2011.

[99] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1442–1468, 2013.

[100] C. Song, Y. Huang, W. Ouyang, and L. Wang. Mask-guided contrastive attention model for person re-identification. In *CVPR*, pages 1179–1188, 2018.

[101] C. Su, J. Li, S. Zhang, J. Xing, W. Gao, and Q. Tian. Pose-driven deep convolutional model for person re-identification. In *ICCV*, pages 3960–3969, 2017.

[102] Y. Suh, J. Wang, S. Tang, T. Mei, and K. Mu Lee. Part-aligned bilinear representations for person re-identification. In *ECCV*, pages 402–419, 2018.

[103] S. Tang, M. Andriluka, B. Andres, and B. Schiele. Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, pages 3539–3548, 2017.

[104] H.-H. Tu and H.-T. Lin. One-sided support vector regression for multiclass cost-sensitive classification. In *ICML*, 2010.

[105] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2805–2813, 2017.

[106] J. Valmadre, L. Bertinetto, J. F. Henriques, R. Tao, A. Vedaldi, A. W. Smeulders, P. H. Torr, and E. Gavves. Long-term tracking in the wild: A benchmark. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 670–685, 2018.

[107] N. Verma, D. Mahajan, S. Sellamanickam, and V. Nair. Learning hierarchical similarity metrics. In *CVPR*, 2012.

[108] C. Vondrick and D. Ramanan. Video annotation and tracking with active learning. In *NeurIPS*, pages 28–36, 2011.

[109] S. Wang and C. C. Fowlkes. Learning optimal parameters for multi-target tracking with contextual interactions. *IJCV*, 122(3):484–501, 2017.

[110] Z. Wang, L. Zheng, Y. Liu, and S. Wang. Towards real-time multi-object tracking. *arXiv preprint arXiv:1909.12605*, 2019.

[111] N. Wojke, A. Bewley, and D. Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, pages 3645–3649, 2017.

[112] H. Wu, M. Merler, R. Uceda-Sosa, and J. R. Smith. Learning to make better mistakes: Semantics-aware visual food recognition. In *ACM MM*, 2016.

[113] Y. Wu, J. Lim, and M. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, Sept. 2015.

[114] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[115] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele. Latent embeddings for zero-shot classification. In *CVPR*, 2016.

[116] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *ICCV*, pages 4705–4713, 2015.

[117] J. Xu, Y. Cao, Z. Zhang, and H. Hu. Spatial-temporal relation networks for multi-object tracking. In *ICCV*, pages 3988–3998, 2019.

[118] Y. Xu, A. Osep, Y. Ban, R. Horaud, L. Leal-Taixé, and X. Alameda-Pineda. How to train your deep multi-object tracker. In *CVPR*, 2020.

[119] B. Yan, H. Zhao, D. Wang, H. Lu, and X. Yang. 'skimming-perusal' tracking: A framework for real-time and robust long-term tracking. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[120] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*, pages 2129–2137, 2016.

[121] M. Ye, J. Shen, G. Lin, T. Xiang, L. Shao, and S. C. Hoi. Deep learning for person re-identification: A survey and outlook. *arXiv preprint arXiv:2001.04193*, 2020.

[122] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *ECCV*, pages 36–42, 2016.

[123] Q. Yu, G. Medioni, and I. Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *CVPR*, pages 1–8, 2007.

[124] B. Zadrozny and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *KDD*, 2001.

[125] B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, 2001.

[126] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *KDD*, 2002.

[127] A. R. Zamir, A. Dehghan, and M. Shah. Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*, pages 343–356, 2012.

[128] J. Zhang, S. Ma, and S. Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In *European conference on computer vision*, pages 188–203. Springer, 2014.

[129] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, pages 1–8, 2008.

[130] S. Zhang, Y. Gong, J.-B. Huang, J. Lim, J. Wang, N. Ahuja, and M.-H. Yang. Tracking persons-of-interest via adaptive discriminative features. In *ECCV*, pages 415–433, 2016.

[131] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu. A simple baseline for multi-object tracking. *arXiv preprint arXiv:2004.01888*, 2020.

[132] Y. Zhang, D. Wang, L. Wang, J. Qi, and H. Lu. Learning regression and verification networks for long-term visual tracking. *arXiv preprint arXiv:1809.04320*, 2018.

[133] B. Zhao, F. Li, and E. P. Xing. Large-scale category structure aware image categorization. In *NeurIPS*, 2011.

[134] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *ICCV*, 2015.

[135] Z. Zheng, X. Yang, Z. Yu, L. Zheng, Y. Yang, and J. Kautz. Joint discriminative and generative learning for person re-identification. In *CVPR*, pages 2138–2147, 2019.

[136] K. Zhou and T. Xiang. Torchreid: A library for deep learning person re-identification in pytorch. *arXiv preprint arXiv:1910.10093*, 2019.

[137] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang. Omni-scale feature learning for person re-identification. In *ICCV*, pages 3702–3712, 2019.

[138] X. Zhou, V. Koltun, and P. Krähenbühl. Tracking objects as points. *arXiv:2004.01177*, 2020.

[139] Z.-H. Zhou and X.-Y. Liu. On multi-class cost-sensitive learning. *Computational Intelligence*, 2010.

[140] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang. Online multi-object tracking with dual matching attention networks. In *ECCV*, pages 366–382, 2018.