UNIVERSITY OF CALIFORNIA SAN DIEGO

Learning Application-Oriented Classifiers for Multi-frame Visual Recognition

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Electrical and Computer Engineering

by

Anwesan Pal

Committee in charge:

Professor Henrik I Christensen, Chair
Professor Nuno Vasconcelos, Co-Chair
Professor Nikolay Atanasov

2019

The thesis of Anwesan Pal is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____
Co-Chair

_____
Chair

University of California San Diego

2019

DEDICATION

To my parents: Dipankar Pal and Lopamudra Pal

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I wish to extend my sincerest gratitude to my advisor, Dr. Henrik I Christensen for his help and guidance towards the work contained in this thesis. I have learnt a tremendous amount through the discussions that we have had together. This thesis would not have been what it is without him.

I am very grateful to the rest of the members of my committee namely, Dr. Nuno Vasconcelos and Dr. Nikolay Atanasov for their invaluable insights and constructive criticisms. I am also indebted to my fellow graduate students, Ahmed H. Qureshi, Carlos Nieto-Granda, Zhihang Ren, Yi Li, Shixin Li, James Smith and Priyam Parashar for their helpful comments and support.

I am eternally grateful to my parents, my elder brother and his wife for their constant motivation and unwavering faith in me. Their love and encouragement has been critical for the successful completion of my degree.

This thesis is currently being prepared for submission for publication of the material. Pal, Anwesan - The thesis author was the primary investigator and author of this material.

VITA

2017	Bachelor of Engineering, Indian Institute of Engineering Science and Technology, Shibpur
2017-2018	Teaching Assistant, University of California San Diego
2018-Present	Research Assistant, University of California San Diego
2019 (Expected)	Master of Science, University of California San Diego

PUBLICATIONS

Anwesan Pal, Carlos-Nieto Granda and Henrik I Christensen "DEDUCE: Diverse scEne Detection methods in Unseen Challenging Environments", submitted to *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019*.

Anwesan Pal, Zhihang Ren, Yi Li, Nuno Vasconcelos and Henrik I Christensen "A Deeper Look at Action Recognition Datasets and Algorithms via Representation Bias", submitted to *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Xinran Li*, **Anwesan Pal***, Ahmed Qureshi and Carlos-Nieto Granda "Optical Flow Based Video Classification with Visual Attention using Recurrent Convolutional Neural Network", *presented for a course project*, 2018.

*Equal Contribution

ABSTRACT OF THE THESIS

Learning Application-Oriented Classifiers for Multi-frame Visual Recognition

by

Anwesan Pal

Master of Science in Electrical and Computer Engineering

University of California San Diego, 2019

Professor Henrik I Christensen, Chair
Professor Nuno Vasconcelos, Co-Chair

Classification, a *supervised learning* problem, is a technique to categorize a given set of data points into two (binary classification) or more (multi-class classification) targets or labels, based on their characteristics. It has wide applications in the domain of Computer Vision and Robotics. Despite the tremendous success of classification in recent times, a fundamental question remains as to whether a classifier truly learns the correct representation (i.e. the *ground-truth representation*) needed to solve a problem. The uncertainty in learning often limits generalization of such methods across different domains. This thesis presents an in-depth study of how different classifiers leverage the intrinsic properties of a dataset in order to robustly model the data

distribution. The study further extends to the application of classifiers in two challenging domains - Visual Semantic Place Categorization and Human Activity Recognition.

The first part of the thesis focuses on the concept of bias in action recognition videos. This is illustrated by the fact that given just a single frame of a video, some actions can be easily identified, due to the presence of certain objects and the background. The second part generalizes classification into the domain of place categorization, where the focus is to learn semantic information about a robot's environment without any human guidance. This is done through a deep fusion of the scene and object attributes. Finally, an algorithm is proposed for human action recognition which incorporates the information learnt from optical flow and visual attention mechanism in order to perform classification.

Chapter 1

Introduction

Classification has always been an important topic in Machine Learning, and its applications are now widely spreading into the domains of Computer Vision and Robotics. Images and videos are an excellent source of semantic information since they can capture detailed contextual cues in a setting which is not possible through other types of sensors like LiDAR, Proximity, Gyroscope etc. Deep Learning models invoking Convolutional Neural Networks (CNNs) [LBBH98] have been very successful in tackling problems related to images including recognition, segmentation and detection [KSH12, CGGS12, FCNL12, GDDM14]. However, videos, or *multi-frame recognition* bring in more challenges in itself, such as huge computational cost, need for capturing long context, network architectural design constraints and lack of standard evaluation benchmarks. This thesis talks about these challenges in detail, with a special emphasis on what intrinsic properties are learnt by a deep learning model in order to solve the classification problem. The applications are discussed in the context of semantic place categorization and human action classification.

1.1 Bias in Action Recognition

The introduction of deep learning has enabled substantial advances in computer vision over the last few years. This is mostly because deep learning networks [KSH12, SZ14, SLJ+15, HZRS16] can leverage large amounts of training data in order to learn sophisticated *representations* of images and videos that enable the robust solution of problems such as object or action recognition. On the other hand, these networks raise challenges because it is not very clear as to what exactly they learn. This makes it difficult to know how far beyond their training they can generalize. One well known principle in machine learning is that a learning system can only be as good as the data on which it is trained. If the dataset is not representative of the entire example distribution, the network will overfit. This is known as dataset bias [TE11] and can be combated by increasing dataset size. With modern datasets [RDS+15, KRA+18, LMB+14, KH09], which tend to be very large in size, and therefore dataset bias is less of an issue.

There are, nevertheless, other sources of bias. One particularly difficult example is *representation bias*, i.e. the fact that the dataset favors some types of representations over others. Representation bias is unavoidable in recognition problems since the goal is to learn the optimal representation for the solution of the recognition problem, a.k.a. the *ground-truth representation*. While it is usually unknown, some properties of this representation can be inferred from the definition of the recognition problem. For example, it is unlikely that a system for the classification of horse breeds will need to learn a representation of humans [DT05], as opposed to that of motion. This is because running style of different horses can be informative of its breed. This, however, assumes an ideal dataset. In practice, the dataset may be such that certain horse breeds always appear with humans (e.g. domestic horses) while others do not (e.g. wild horses).

When this happens, a representation of humans, such as a person detector [TAS14], will achieve better than chance level performance for horse classification. In this case, the dataset is said to be *biased* for the human representation. In the extreme case, this might result in a very

high recognition accuracy utilizing only human representations. It is likely here that the deep learning system has solved the problem without building any representation of horses. Unlike dataset bias, a difficulty of representation bias is that it is not necessarily eliminated by simply making datasets larger. In general, the problem is that the data collection is itself biased. For example, pictures commonly found on the web may pit domestic horses with humans while wild horses in seclusion. Thus, if additional data is acquired online, the representation bias will only be reinforced.

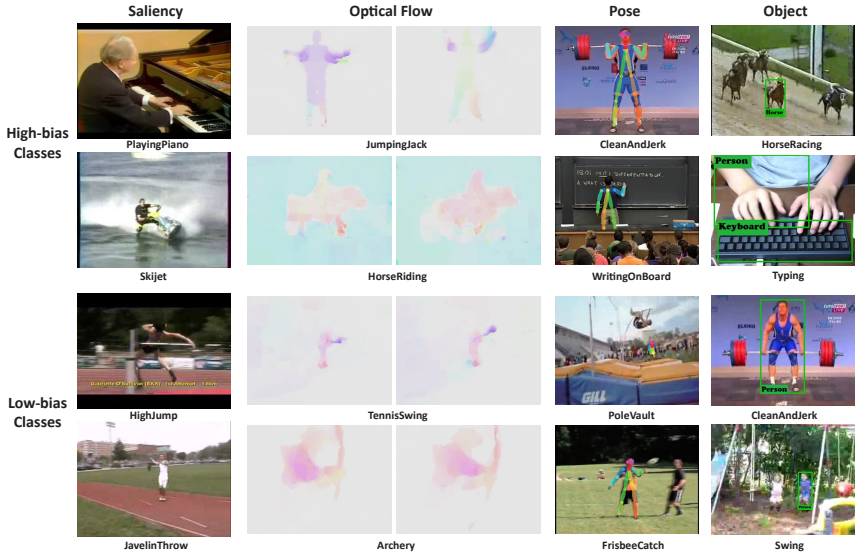


Figure 1.1: Representation bias illustrated by the frames of UCF101 classes. The features from these frames are given as inputs to the classifier. Top 2 rows correspond to the high-biased classes while the bottom 2 rows to the low-biased classes. Upon inspection, it is much simpler to classify the frames in the top rows into their respective classes as compared to the bottom rows.

In domains such as action recognition, representation bias is a complex problem because there are many potential sources of bias. Some of these are illustrated in Figure 1.1, with examples from the UCF101 [SZS12] dataset. For example, action classes such as ”playing piano” or ”jet ski” can be recognized by inspection of a single image, while ”jumping jack” and ”riding a horse” are easily recognized from short term motion patterns, ”Clean and Jerk” and ”Writing on board” from human pose skeleton, and finally classes such as ”horse racing” and ”typing” from the presence of certain objects. In summary, UCF101 has bias towards static, motion, pose, and

object representations.

Given all this, the characterization of representation bias of existing datasets is important for at least two reasons. First, it provides pointers on how to create or modify datasets so as reduce or eliminate bias. For example, Li *et al*. [LLV18] have recently proposed a measure of representation bias and used it to show that a dataset (Diving48) explicitly collected to have low bias, does not have to be particularly large [KCS+17, GKM+17, MBG+18]. Second, by analyzing the performance of different action recogniton algorithms on datasets with quantified biases, it is possible to infer properties of the representation learned by the algorithm. This enables a better understanding of where these algorithms are likely to succeed and fail and can be exploited to produce new algorithms that address these limitations. Given the flexibility of modern deep learning systems, they are quite likely to exploit any existing biases to learn representations that solve the dataset instead of the real action recognition problem.

1.2 Visual Place Categorization

Scene recognition and understanding has been an important area of research in the Robotics & Computer Vision community for more than a decade now. Programming robots to identify their surroundings is integral to building autonomous systems for aiding humans in house-hold environments.

Kostavelis *et al*. [KG15] provided a survey of previous work in semantic mapping using robots in the last decade. According to their study, scene annotation augments topological maps based on human input or visual information of the environment. Bormann *et al*. [BJL+16] pointed out that the most popular approaches in room segmentation involve segmenting floor plans based on spatial regions.

An essential aspect of any spatial region is the presence of specific objects in it. Some examples include a bed in a bedroom, a stove in a kitchen, a sofa in a living room, etc. Niko *et*

al. [SBS+18] formulated the following three reasoning challenges that address the semantics and geometry of a scene and the objects therein, both separately and jointly: 1) Reasoning About Object and Scene Semantics, 2) Reasoning About Object and Scene Geometry, and 3) Joint Reasoning about Semantics and Geometry. The content in this thesis focuses on the first reasoning challenge and uses Convolutional Neural Networks (CNNs) as feature extractors for both scenes and objects. The goal is to design a system that allows a robot to identify the area where it is located using visual information in a manner similar to how a human being would.

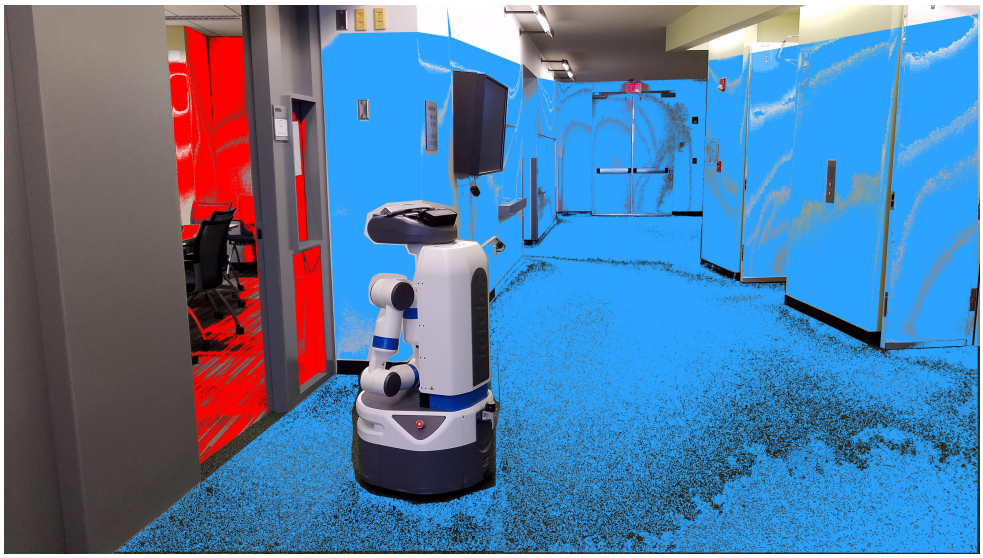


Figure 1.2: Visualization of the semantic mapping performed while the Fetch robot is navigating through the environment.

In this thesis, five different models of scene prediction have been considered, each developed through the integration of object and scene information from a surrounding in order to perform place categorization. To evaluate the robustness of these models, extensive experiments have been conducted on state-of-the-art still-image datasets, real-world videos captured via different hand-held cameras, and also those recorded using a mobile robot platform in multiple challenging environments. One such environment is shown in Figure 1.2, where the segmented regions correspond to the scenes detected. The results obtained from the experiments demonstrate that the proposed system can be generalized beyond the training data, in addition to being

impervious to object clutter, motion blur and varying light conditions.

1.3 Attention in Action Recognition

The study of attention has classically focused on the physical characteristics that determine whether stimuli are attended. William James [Jam50] defined attention as taking possession of one out of several simultaneously possible objects or train of thoughts by the mind. The modern study of attention continues to work within this broad definition. It is generally accepted that an individual is only aware of a small fraction of the information provided to the brain by the sensory systems.

The computer vision models of attention perform remarkably well in most of the applications where static images or images from a video stream are manually fed to the model in order to identify the most salient/task-relevant stimuli. In case of some real-time applications where the current visual input of the attention model is to be determined by the decision output of the model (i.e., the focus of attention) at the immediate past, the traditional methods of visual attention face severe limitations in a number of aspects [BMG06], [BMGK08] and[BKMG09]. Using a visual attention model as a component of robotic cognition is an example of such applications. In this case, the attention model should be able to locate the behaviorally relevant stimuli in an ongoing stream of visual input and respond to it, perform learning in an on-line fashion and with minimal human supervision, and apply the learned knowledge for guiding the attention behavior in arbitrary environmental settings.

1.4 Overview of the Thesis

Chapter 2 presents a study of the representation biases of existing datasets through their results as reported. Six types of representation biases are defined and a procedure to quantify

them is proposed. For implementation, nine of the most popular action recognition datasets are considered. The characterization of bias is performed both at the level of the entire dataset and that of individual classes. This provides a detailed picture of the origin of the biases that can be leveraged by dataset creators to produce better datasets, either by adding data to the existing datasets or by assembling new ones. In addition, a set of prototypical deep learning algorithms for action recognition are also considered, and their performance are characterized in terms of bias. This provides some insight on the representations learned by the algorithms. Finally, it is shown that by simple manipulation of a small number of classes, it is possible to undermine the "state of the art" standard for algorithm evaluation in computer vision.

Chapter 3 introduces a set of Diverse scEne Detection methods in Unseen Challenging Environments (DEDUCE) algorithms for performing visual semantic place categorization. These algorithms are then evaluated across different domains. The first experiments are conducted on two state of the art 2D image datasets. This is followed by more experiments on real world videos captured via professional and ordinary cell-phone cameras in house-hold settings. To further test the robustness of the algorithms, an experiment is also carried out on a recording of a movie, where the focus is not on the background, but instead on the people involved. Finally, data is recorded on a mobile robot navigating through an office setting and place categorization is performed without human guidance.

Chapter 4 provides a brief background of Optical Flow and the Neural Network architectures and algorithms implemented thus far for human action classification. This is followed by the explanation of the proposed Recurrent-Attentive Convolutional Neural Networks (R-ACNN) algorithm for video classification. Detailed description about the data collection and pre-processing, model architecture and hyper-parameter training are also provided. Finally, the results of the proposed algorithm is compared along with the other state of the art algorithms.

Chapter 2

Representation Bias in Datasets and Algorithms

2.1 Background

The learning process for action recognition can be affected by a wide variety of biases. Many of these biases can be divided broadly into two categories based on the type of features they affect—*static* and *dynamic*. To mitigate static biases, early action recognition datasets, such as KTH [SLC04], Weizmann [BGS+05] and Hollywood2 [MLS09], were captured under a controlled environment. However, most classes of these datasets can be recognized by simply analyzing motion in small spatiotemporal neighborhoods. Also, these datasets are small and prone to overfitting. Recently, many datasets have appeared which are larger in size and contain more diverse classes, such as UCF101 [SZS12], HMDB51 [KJSS13] and Kinetics [KCS+17]. Unfortunately, these datasets do not cover a wide range of the daily life actions. To overcome this problem, Charades [SVW+16] and Something-Something (v1 and v2) [GKM+17, MBG+18] change the traditional way of labeling action classes and bring in more daily life videos. In Charades, most videos are about day-to-day actions and every single video is assigned multiple

motion labels. For the Something-Something datasets, class names are not associated with any object or identity, such as *"Throwing something"*, *"Putting something on something"*, etc. Another newly introduced dataset, Diving48 [LLV18], consists of videos of competitive diving, and has the added advantage of containing low static cues.

The notion that dataset bias can affect the performance of recognition systems has received some attention by the computer vision community. Torralba *et al*.[TE11] uncovered biases of several image datasets and related them to the poor generalization performance of state-of-the-art models. Gkioxari *et al*.[GGM15] began to utilize contextual biases to solve action recognition. Tommasi *et al*.[TPCT17] performed a thorough analysis of the common datasets and verified the present de-biasing methods. More recently, Feichtenhofer *et al*.[FPWZ18] utilized network visualizations to understand dataset bias and Stock *et al*.[SC18] proposed a way to avoid undesirable bias when learning a model.

Traditional action recognition algorithms utilize different types of representations to boost their performance. In many cases, it is difficult to determine what biases these representations can leverage. There are, however, examples where the model uses a very explicit combination of elementary representations. For example, two stream networks [SZ14] implement spatial and temporal streams, which capture the static and dynamic structure of video respectively, and combine these results at the final layer. These models can leverage either static or dynamic biases to achieve artificially good performance on a biased dataset. The P-CNN of [CLS15] utilizes a "guided" two stream network, which adds a pose representations to the mix. Because they are fairly explicit about their representations, these models are suitable tools for measuring the amount of bias of different datasets.

2.2 Understanding a Dataset

This section discusses the tools and the experiment setups used to measure bias. It also introduces the different biases considered and the algorithms tested.

2.2.1 Bias

Li *et al.*[LLV18] introduced the concept of representation bias, defining the amount of bias of a dataset $\mathcal{D}$ with respect to a representation ϕ as

$$\mathcal{B}(\mathcal{D},\phi) = log\frac{\mathcal{M}(\mathcal{D},\phi)}{\mathcal{M}_{rnd}}, \tag{2.1}$$

where $\mathcal{M}(\mathcal{D},\phi)$ is the performance obtained on $\mathcal{D}$ by a network pretrained to detect ϕ, and $\mathcal{M}_{rnd}$ is the random or chance-level performance. Li *et al.*[LLV18] proposed that $\mathcal{M}(\mathcal{D},\phi)$ be *test accuracy* when measuring the bias of datasets, and individual class *average precision* of the test set for class-level analysis. This definition makes sense for class-level analysis of a dataset. However, at the dataset level, using *mean average precision* is a more intuitive metric due to two reasons. Firstly, it matches the usage of average precision at the class-level. This expedites experiments on large datasets, since the dataset analysis follows easily from the class-level analysis. It also makes it possible to directly relate dataset-level to class-level biases. Secondly, since *precision* models the similarity among the constituent classes of a dataset, it is a preferable metric for determining sources of bias.

Chance level performance is defined as $1/C$ in [LLV18], where C is the number of classes. This does not account for unbalanced datasets. For this reason, the frequency of class i is defined as

$$f_i = \frac{n_i}{N}, \tag{2.2}$$

where n_i is the number of videos in the class and N the total number of videos in $\mathcal{D}$. At dataset-

level, chance performance $\mathcal{M}_{rnd}$ is defined as the average class frequency, while for class, it is the respective class frequency f_i.

2.2.2 Fairness

Bias is not a complete characterization of how a dataset or classes favour different representations. If $\mathcal{D}$ is an "easy" dataset, many representations will achieve high recognition rates on $\mathcal{D}$. A second property of interest is how the biases of the different representations compare. This is captured by the *fairness score* of $\mathcal{D}$. To define this, we denote by $\mathcal{B}_i^c$ the bias of class c for representation $i = 1, \ldots, r$. The fairness score ξ^c of the class is then determined by finding the bias distribution

$$\beta_i^c = \frac{\mathcal{B}_i^c}{\sum_j \mathcal{B}_j^c} \tag{2.3}$$

and computing its Kullback-Leibler divergence [KL51] with the uniform distribution $u_i = \frac{1}{r}, \forall i$, i.e.

$$\xi^c = D_{KL}(\beta_i || u_i) = \sum_i \beta_i^c log\left(\frac{\beta_i^c}{u_i}\right). \tag{2.4}$$

This is large when the bias distribution is "peaky," i.e. the class is much more biased for one representation than the others, and small when the distribution is uniform, i.e. the class is equally biased for all representations.

2.2.3 Representations

In this study, a representation ϕ is a classifier using a pre-trained convolutional neural network (CNN). A CNN can be thought as a combination of two stages: a feature extractor, which usually consists of many neural network layers, and a classification layer, usually implemented with a linear weight layer and a softmax function. The feature extractor defines the representations implemented by the network, the classifier leverages these representations to infer the class labels. If the CNN has high classification rate on $\mathcal{D}$, then $\mathcal{D}$ is said to be biased for the representations

implemented by the CNN. These are determined by the network architecture and the data used to train it. In general, a CNN can implement multiple representations. To guarantee that the network only implements one representation ϕ there is a need to keep both the CNN and its training as simple as possible. On the other hand, if the CNN is too simple it may not have the discriminant power required to detect biases. [LLV18] suggested the use of standard CNN models to define representations. We adopt a similar strategy, but expand the set of representations considered. The first set of representations is a set of three CNN feature extractors similar to those proposed in [LLV18]. In all cases, the CNN is the ResNet50 model of [HZRS16] and extracts a 2048-dimensional feature vector. This is obtained by removing the softmax layer after the network is trained.

Saliency bias - The first representation consists of a CNN pre-trained on the 1000 object classes of ImageNet [RDS+15]. While the network is trained on objects, we apply it to entire video frames, which can contain multiple objects or complex scenes. Hence, the network is more likely to be successful for close-ups of objects, and can be seen as a representation of object dominance in the scene, or *object saliency*. We denote this representation as ϕ_{sal}.

Scene bias - The second representation consists of a CNN pre-trained on the 365 scene classes of the Places365 scene classification dataset [ZLK+17]. This is a scene representation, denoted ϕ_{scene}.

Person bias - the third representation consists of a CNN pre-trained on the 204 classes of people attributes of the COCO-attributes dataset. This is a person representation, denoted ϕ_{person}.

Optical Flow bias - The fourth representation is based on optical flow and is implemented in two steps. The first consists of the Flownet 2.0. model for optical flow computation [IMS+17]. This model achieves a good trade-off between accuracy of the computed optical flow and speed. While other bench-mark methods such as TV-L1 [ZPB07], Brox [BBPW04] etc. may produce slightly more accurate flow maps, they are considerably slower to compute. The second step is an optical flow feature extractor, implemented with a Two-Stream CNN pre-trained on the UCF101

Table 2.1: Representation bias of the various datasets.

	KTH	Hollywood2	UCF-101	HMDB-51	Charades	Diving48	Kinetics	Something v1	Something v2
# Videos	599	2417	13320	6766	9848	18404	231984	108499	220847
# Classes	6	12	101	51	157	48	400	174	174
Saliency	1.39	**1.45**	**4.16**	**3.01**	**0.71**	**0.73**	**4.35**	**1.65**	**1.85**
Person	1.33	1.34	4.07	2.87	0.63	0.71	4.13	1.46	1.72
Scene	1.31	1.35	3.93	2.86	0.65	0.58	3.98	1.24	1.37
Optical Flow	**1.61**	1.18	2.74	1.89	0.22	0.45	1.68	0.83	1.78
Pose	1.06	0.61	1.18	1.02	0.11	0.14	0.39	0.14	0.10
Object	0.75	0.76	1.52	1.40	0.11	0.30	0.55	0.20	0.16
Avg	1.24	1.12	2.93	2.17	0.40	0.48	2.51	0.92	1.17

dataset [SZS12]. Given the optical flow maps, batches of 10 consecutive flow images, temporally separated by a stride of 10, are fed to the network. The implementation is based on [WXWQ15]. The output of the feature extraction stage of this network defines the optical flow representation, denoted ϕ_{flow}.

Pose bias - The fifth representation captures human pose. The feature extractor is the real-time multi-person 2D pose estimation algorithm of [CSWS17]. The frames are first sampled from the original video using a stride of 10 frames. The algorithm then extracts the pose of every person and represents it as an 18-joint skeleton. For each frame, a pose skeleton is chosen randomly and the x-y coordinates of each of the joints are concatenated. These concatenated coordinate vectors are used as pose features, defining the pose representation ϕ_{pose}.

Object bias - The final representation is based on object detection results produced by the Faster RCNN of [RHGS15]. This model is pre-trained on the 80 object classes of the MS-COCO dataset [LMB$^+$14]. For each frame, objects corresponding to the top 5 classification scores are selected. The bounding box information of these detections along with their classification scores are then concatenated and used as object features, defining the object representation ϕ_{object}.

Bias measurements Given a representation ϕ, the corresponding feature extractor is used to extract feature vectors from the training set of $\mathcal{D}$. These are fed to a linear classifier which is then trained with respect to the classes of $\mathcal{D}$. Next, the classifier is applied to the test set of $\mathcal{D}$ and the average precision of each class and its mean are recorded. This is the measure of performance $\mathcal{M}(\mathcal{D},\phi)$ used in (2.1) for bias calculation.

2.2.4 Algorithms

While the study of bias is important, it is also interesting to understand how bias affects the performance of different algorithms. In its largest generality, this is a daunting task, since there are many action recognition algorithms and these can be quite complex. Running a large number of algorithms on a large number of datasets is beyond the reach of our computational capabilities, at the moment. However, many algorithms are variants or combinations of simpler *prototypical* algorithms. Understanding how bias affects these algorithms provides insight on how it may affect the more complex variants. In this study, we considered the following prototypical algorithms. In all cases, we used a ResNet101 with the weights pretrained on ImageNet and finetuned on each dataset.

C2D-Spatial consists of the spatial stream of a Two-Stream CNN, implemented in isolation, by simple addition of a softmax layer to the C2D-Spatial feature extractor. To classify a video, n frames are randomly selected from it, fed through the network, and the output class scores averaged. We used $n = 3$ for training and $n = 19$ for inference. The training strategy is as per [SZ14], [WXWQ15] and [MCKA17].

C2D-Motion consists of the temporal stream of the Two-Stream CNN, implemented in isolation, as above. The network processes stacks of 20 optical flow maps (10 channels each of x and y motion components). For training, we randomly pass just one of these stacks, for inference, 19 stacks are uniformly segmented and softmax outputs averaged.

C2D-Two Stream averages the video-level predictions of the two models described above. Thus, this algorithm models the combination of static and dynamic features in a video.

Inflated 3D (I3D)-Spatial is the algorithm of [CZ17]. It consists of a 3D ConvNet which is the *inflated* version of the 2D Inception-v1 network [IS15]. The feature extractor takes in segments of 64 consecutive RGB video frames in order to produce a feature vector. For training, one segment is randomly chosen from each video while for inference, 10 uniformly distributed segments are passed into the network and the outputs are averaged. The model we used has been pretrained on

ImageNet and then fine-tuned on the Kinetics dataset, as well as on each of our datasets.

P-CNN [CLS15] is similar to the two stream network, but with the additional guidance of pose-joint locations. In both spatial and motion streams, image patches containing parts of the human body (e.g. right and left hand) are passed through the CNN, producing initial image features, which are then aggregated as *static* and *dynamic* features of corresponding body parts. Finally, they are concatenated into a single video feature vector. A linear SVM classifier is used for the final video classification.

2.3 Experiments

The representations and algorithms discussed above were used to perform a large number of experiments, whose results we describe in this section.

2.3.1 Representation bias

Datasets We started by characterizing the representation bias of 9 datasets. Table 2.1 gives the information regarding the number of videos and classes present in each of the datasets evaluated in the thesis. The train/test splits we used is the same as those reported in the respective sources of the datasets.

Bias Table 2.1 summarizes the biases of all datasets for all representations. A number of observations can be made from the table. First, all datasets other than KTH have largest bias for the saliency representation. This is likely because most action datasets tend to involve object interactions. Understanding these interactions frequently requires a close-up view of the objects. This explains the dominance of the saliency bias. On the other hand, many of these objects tend not to belong to the classes detected by the Faster RCNN. This suggests that they are not "popular" objects, which explains the low object bias. Second, comparing the datasets, it can be seen that UCF101 has the most average bias, while Charades and Diving48 have the least. This

can be explained by the fact that Charades contains lengthy videos with more than one label per video. Moreover, the labels are also quite different from one another - such as *"Looking outside the window"* combined with *"Someone is cooking something"*. As for the Diving dataset, the information within each frame is quite limited due to the similar environment and the objects related to the diving actions, which lowers down the biases from the static representations.

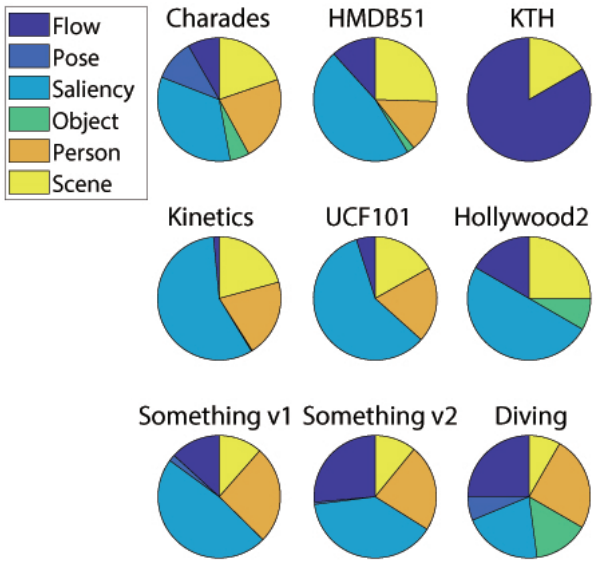


Figure 2.1: Dominant representation bias

Bias dominance: To obtain a more detailed picture of the bias within each dataset, we next performed an analysis of class bias. The pie-charts of Figure 2.1 illustrate the distribution of the class-level dominant representation bias for each of the datasets. They were obtained by identifying the strongest bias for each class, which is denoted the *dominant* bias, and counting the occurrences of these dominant biases across classes. A few conclusions can be taken from the figure. First, saliency bias is dominant for most datasets other than KTH. This agrees with the observations of Table 2.1. Second, the detailed picture is more nuanced than the broad picture painted by the table. While it is true that saliency usually dominates, this dominance has a large variation of intensity across datasets. On Kinetics and UCF101, saliency is dominant for more than half of the classes in the dataset. On the other hand, for Charades and Diving the saliency

slice is not much larger than some of the other slices.

Third, the second most dominant bias is, for most datasets, optical flow. The exceptions are again Kinetics and UCF101. This suggests that these datasets are heavily biased towards static representations. Among the remaining ones, Something-Something V2 and Diving have the best balance between static and dynamic biases. On the other hand, KTH is heavily biased to optical flow. Fourth, most datasets have very small bias towards object pose, which is explained by the limited vocabulary of the Faster RCNN, and zero pose bias. This suggests that body configurations do not play a major role in discriminating the action classes of these datasets. Exceptions are Charades and Diving, which are composed of actions of people doing things and athletes performing olympic dives. Finally, most datasets have a non-trivial amount of person bias. This is expected, since most actions involve humans. An important point to note here is that Figure 2.1 is also a reflection of the fairness score of a dataset $\mathcal{D}$ since it shows the distribution of the dominant class bias across the entire dataset. Thus, in this case, Charades and Diving are the most *fair* datasets since they have an almost uniform distribution among all the representations, while KTH is the most unfair.

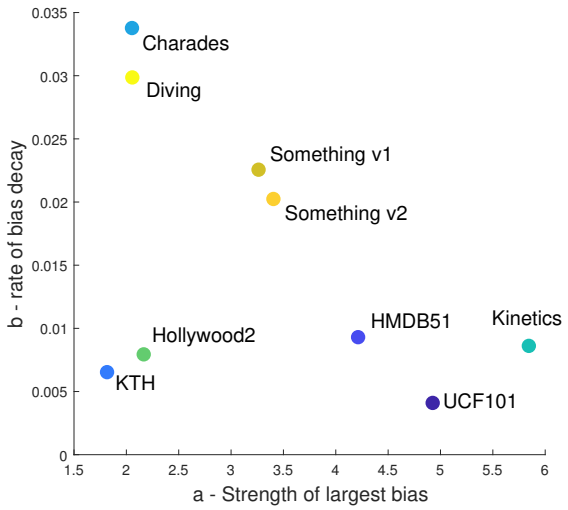


Figure 2.2: Characterization of bias skew of all datasets for saliency

Bias skew We next considered the question of whether, within each dataset, classes are uniformly

biased or contain a "bias skew," i.e. a small percentage of very biased classes and a long tail of unbiased ones. For each dataset and representation, we sorted the classes by descending bias and fit an exponential decay law, of the form $\mathcal{B}_i^c = a_i e^{-b_i c}$, where $\mathcal{B}_i^c$ is the bias of the class of rank c for representation i, to the resulting raked bias values, by least squares. The parameter a_i is a measure of the strength of the largest bias for representation i in the dataset, while b_i encodes the rate of bias decay, capturing how quickly the bias falls across the sorted classes. The scatter plot of Figure 2.2 depicts the values of largest bias (a) vs. bias decay (b) of the different datasets for the saliency representation (ϕ_{sal}). Charades and Diving have small maximum bias that decays very quickly. This is ideal behavior. On the other hand, UCF and Kinetics have large maximum bias, which decays slowly. This explains why these datases are so biased for saliency in Table 2.1. The remaining datasets "interpolate" between these two behaviors, with the exception of KTH and Hollywood2. These outliers have both small maximum bias and small decay. For KTH, this is explained by fact that it has very little bias for saliency, as shown by the pie chart of Figure 2.1. Hollywood2 is a more interesting case. While bias for the saliency representation is dominant in many classes (Figure 2.1), the bias distribution is quite uniform across classes.

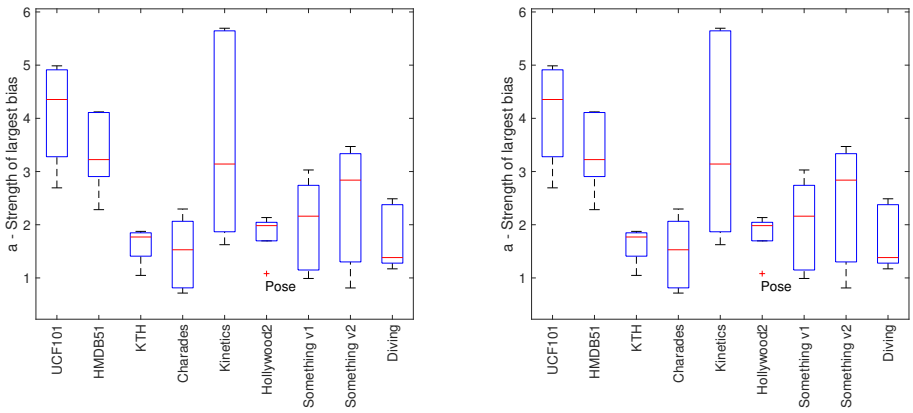


Figure 2.3: Box-plots of the strength of largest bias ("a") and the rate of bias decay ("b") for different datasets

Figure 2.3 summarizes the average and spread of the a-b values of each dataset across representations using a boxplot. With regards to the maximum bias a, UCF101 and HMDB51

have the highest median, while Kinetics has, by far, the largest range of values. This shows that the largest class bias is extremely strong for some representations. On the other hand, Charades and Diving have the smallest median max class bias. Regarding bias decay Charades has the largest median rate, indicating that it has the most skewed bias distribution. It is also interesting to note that the v2 for Something-Something has a greater median max bias a than v1 and an equivalent median decay rate b. This implies that the bias distribution of v2 upper bounds that of v1, and could be due to the different levels of granularity of the two datasets - more fine-grained action categories in v2, and even more fine-grained labeling via video captions as reported in [MBG+18]. Overall, Charades and Diving tend to have low and high values of "a" and "b" respectively, indicating that they have both a low overall bias and a low population of very highly biased classes.

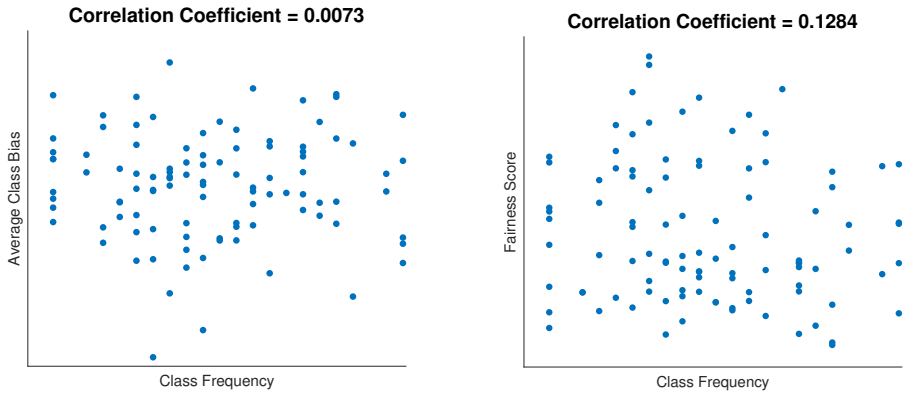


Figure 2.4: Correlation coefficients for average class bias and fairness score versus class frequency

Relations: We next considered whether bias and unfairness can be explained by dataset unbalance, i.e. unevenness of the class frequencies f_i. Figure 2.4 presents a scatter plot of the average bias of each class vs. its frequency and a similar plot for the fairness scores. The two plots show very weak correlation between either of the bias properties and class frequency. This is positive, because action recognition datasets tend to be unbalanced, just due to the fact that it is easier to find videos of some actions than of others. These plots suggest that dataset unbalance is not a

major reason of concern, when it comes to representation bias. On the other hand, it would be much easier to correct representation bias if it were mostly due to class inbalance. The fact that this is not the case, suggests that further studies like the one we conducted are likely to be needed, as well as algorithms that specifically target bias removal.

2.3.2 Effect of bias on algorithms

Beyond characterizing the representation bias of datasets, we investigated how this bias affects algorithms.

Dependencies: We started by investigating how the different algorithms leverage the various biases present in a dataset. For this reason, we computed the bias of a dataset $\mathcal{D}$ towards each algorithm $\mathcal{A}_i$ by defining $\mathcal{B}^c(\mathcal{D}, \mathcal{A}_i)$ to be the normalized (by f_i) average precision of $\mathcal{A}_i$ towards class c. In this way, we generalize the concept of bias calculation of a dataset towards both representations and algorithms. Thus, for a dataset such as HMDB51, we end up getting the matrix $\mathcal{B}(\mathcal{D}, \mathcal{A})$ of size 51x5 comprising of the bias of each of the 51 classes towards our 5 algorithms.

Representation/Algorithm similarity: The computation of the representation $\mathcal{B}^c(\mathcal{D}, \phi_i)$ and algorithmic $\mathcal{B}^c(\mathcal{D}, \mathcal{A}_j)$ biases, places algorithms and representations in a common space of dimension equal to the number of classes in the dataset. PCA was used to project this space on a 2D plane for visualizing the relationship between algorithms and representations. This helps to establish the kind of representations different algorithms favour. It can also be used to evaluate which algorithms might perform better on a new dataset, depending on the dominant representation bias it has.

From (2.1), we already have the bias of the classes towards the 6 representations. This results in a matrix $\mathcal{B}(\mathcal{D}, \phi)$ of size 51x6 for HMDB51. By concatenating $\mathcal{B}(\mathcal{D}, \phi)$ and $\mathcal{B}(\mathcal{D}, \mathcal{A})$, we get 11 vectors, each of 51 dimensions. These vectors are then projected onto the space of the two largest principal components. This is given by Figure 2.5. There are a few notable observa-

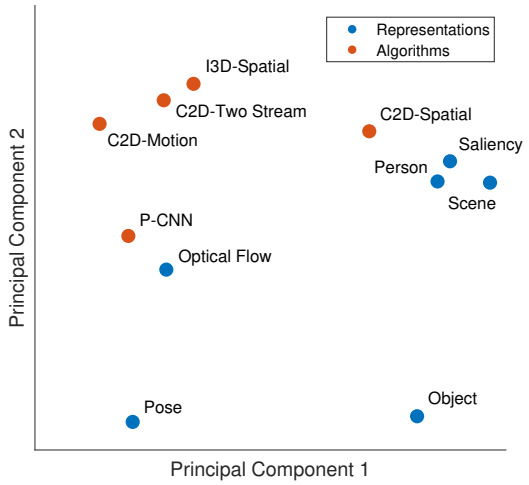


Figure 2.5: PCA visualization of the algorithms and representations projected onto the space of the two largest Principal Components

tions here. Firstly, the C2D-Spatial as well as the I3D-Spatial algorithms lie close to the static representations while the C2D-Motion and PCNN are closer to the dynamic representation,ϕ_{flow} and pose joint representation, ϕ_{pose} respectively. Secondly, it is intuitive to see that C2D-Two Stream, which is the combination of the Spatial and Motion C2D algorithms, indeed lies around the middle. Thirdly, the Object representation, ϕ_{object}, is located far away from all the algorithms. It shows that there are not a lot of action recognition algorithms which consider just object locations for classification purpose. This is in contrast to the Saliency representation ϕ_{sal}, which is close to multiple algorithms. Thus, this result highlights the importance of object saliency over just its location.

Bias-based Clustering analysis: One of the primary objectives of the work done here is to look deeper at a dataset in order to find similarities in its constituent classes in terms of bias. This motivates the idea of grouping or *clustering* the classes of a dataset according to the representation bias values and visualizing the performance of each cluster. This is shown in Figure 2.6. On the left side, we have the t-SNE plot [MH08] depicting 3 clusters of the UCF101 classes found according to the k-means clustering algorithm. The performance of the 5 algorithms

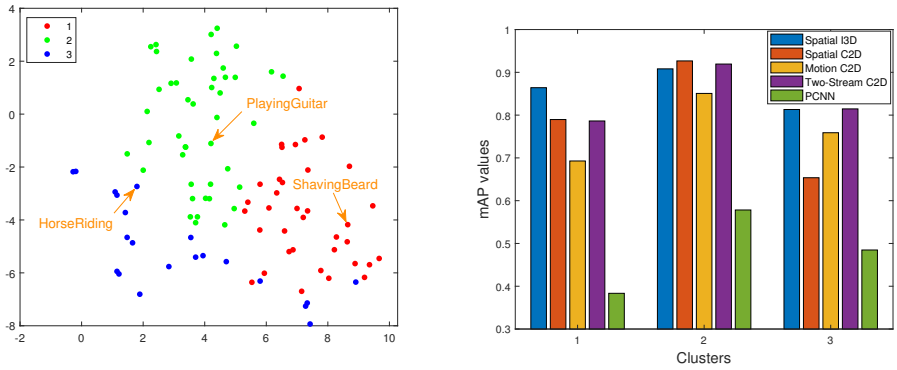


Figure 2.6: t-SNE visualization of clusters of UCF101 classes and the corresponding algorithm performance on each cluster

are then plotted on the right side for each of these clusters. It is observed that while the overall performance of C2D-Spatial is better than C2D-Motion for UCF101, this is not the case for cluster 3. This is possibly due to the presence of classes with short-term motion (such as *HorseRiding*). On the contrary, for cluster 2, the C2D-Spatial actually performs better than both the Two-Stream as well as I3D algorithms. This is a striking observation since I3D usually outperforms most C2D algorithms by a long margin [CZ17] .

2.3.3 Effect of Dataset Manipulation on Algorithms

The above sections mention the performance of the *complete* dataset towards different representation biases and algorithms. Here, we introduce a novel technique of manipulating a dataset according to bias, in order to see a change in the performance of algorithms. During our implementation, the Spatial stream of C2D outperformed the Motion stream for UCF101. From Figure 2.5, it can be seen that the spatial algorithms rely more on ϕ_{sal} while the motion algorithm on ϕ_{flow}. This is understandable since the former focus more on the static features while the latter on short-term dynamics. Thus, theoretically, removing the classes according to ϕ_{sal} should have some negative impact in the performance of the spatial algorithms. In our experiment, we therefore picked only the high-biased classes such as *PlayingPiano*, *Skijet* etc. (as shown in

Figure 1.1) which also have relatively low bias for ϕ_{flow} and recomputed the performance of each of these algorithms.

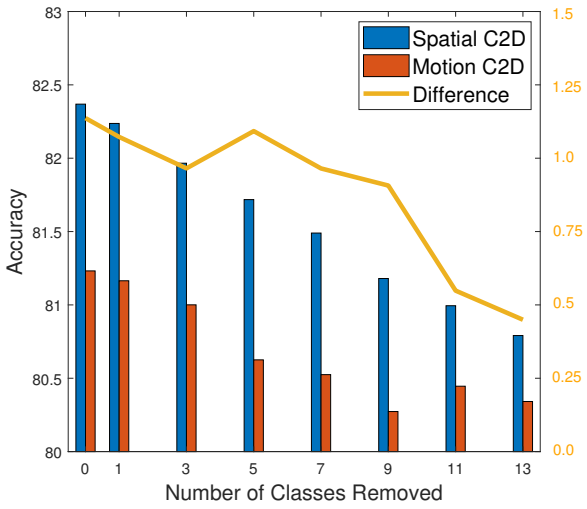


Figure 2.7: Performance of two algorithms on UCF101 and their differences, with biased classes removed.

This is reflected in Figure 2.7 for the different number of classes removed. The curve overlaid on the bins denotes the difference in accuracy of the spatial and motion algorithms. Clearly, the difference decreases with the number of classes removed. This suggests that a simple manipulation of classes may indeed change/alter the performance of algorithms. This is an interesting result since it shows that relative algorithm performances are affected by the biases of the dataset. This is important, due to the prevalence of the "state of the art" standard in the evaluation of vision algorithms. This practice encourages the promotion of algorithms that could be best at leveraging representation biases of the dataset, instead of solving the actual task. To make matters worse, datasets collected using similar data collection set-ups are likely to exhibit similar biases. For example, many data collection efforts use the same source (YouTube) and similar protocols (Turk annotations, etc.). While most recent efforts have focused on collecting *more* data, this will not help if the new large datasets are as biased as the old and small ones.

2.4 Conclusion

In this chapter, we studied different types of representation biases on most state-of-the-art action recognition datasets and found that they vary a lot with respect to different representations. A new metric, the fairness score, was also introduced to assess a dataset based on bias, and it was further proven that both this metric and the average bias of a dataset is uncorrelated with class frequency. We further established that the concentration of high-biased classes vary among datasets and this information can be utilized to reduce the overall bias of the dataset. The second part of the chapter rounds up the state-of-the-art algorithms and their relationship with different representations. We proved that algorithms tend to perform better in the presence of their preferred representations. Thus, when a new action recognition dataset is introduced, the optimal algorithm can be found based on this dataset's bias. Finally, we showed that an algorithm's performance is subject to the classes it is exposed to. Therefore, by removing biased classes of a dataset, it is totally possible to change or even alter the relative performance of different algorithms.

Chapter 3

Diverse Scene Detection

3.1 Background

Semantic place categorization using only visual features has been an important area of research for robotic applications [TMFR03, WCR09]. In the past, many robotics researchers focused on place recognition tasks [PMC08, SI07] or on the problem of scene recognition in computer vision [OT01, QT09].

Quattoni and Torralba [QT09] introduced a purely vision-based place recognition system, which improves the performance of the global gist descriptor by detecting prototypical scene regions. However, the size, shape and location of up to ten object prototypes has to be labeled and learned in advance for this system to work. Also, the labeling task is very work-intensive, and the approach of having fixed regions is only useful in finding objects in typical views of the scene. This makes the system ill-suited for robotics applications. Since we want to deal with flexible positions of objects, we apply a visual attention mechanism that can locate important regions in a scene automatically.

A number of different approaches have been proposed to address the problem of classifying environments. One popular approach adopted is to use Feature Matching with Simultaneous

Localization and Mapping (SLAM). Ekvall *et al*. [EKJ07] and Tong *et al*. [TSY17] demonstrated a strategy for integrating spatial and semantic knowledge in a service environment using SLAM and object detection & recognition based on Receptive Co-occurrence Histograms. Espinace *et al*. [EKSR10] presented an indoor scene recognition system based on a generative probabilistic hierarchical model using the contextual relations to associate objects to scenes. Kollar *et al*. [KR09] utilize the notion of object-object and object-scene context to reason about the geometric structure of the environment in order to predict the location of the objects. The performance of the object classifiers is improved by including geometrical information obtained from a 3D range sensor that facilitates a focus of attention mechanism, in addition. However, these approaches only identify the place based on the specific objects detected and the hierarchical model that is used to link the objects with the place. In contrast to their method, our algorithm is not limited to a small number of recognizable objects.

Recently, there have been several approaches to Scene Recognition using Neural Networks. Liao *et al*. [LKW$^+$16, LKW$^+$17] used Convolutional Neural Networks (CNNs) to recognize the environment based on object occurrence for semantic reasoning, but their system information and mapping results are not provided. Sun *et al*. [SMTA18] proposed a Unified Convolutional Neural Network which performs Scene Recognition and Object Detection. Luo *et al*. [LC18] developed a semantic mapping framework utilizing spatial room segmentation, CNNs trained for object recognition, and a hybrid map provided by a customized service robot. Niko *et al*. [SDM$^+$16] proposed a transferable and expandable place categorization and semantic mapping system that requires no environment-specific training. Mancini *et al*. [MBCR18] addressed Domain Generalization (DG) in the context of semantic place categorization. They also provide results of state-of-the-art algorithms on the VPC Dataset [WCR09] that we compare to. However, most of these results do not test their algorithms on a wide variety of platforms. This is the main focus of the work shown in this chapter. The experiments are conducted, both for static images, and dynamic real world videos captured using hand-held cameras and robots.

3.2 Proposed Methodology

We consider a set of five different models, abbreviated as Diverse scEne Detection methods in Unseen Challenging Environments (DEDUCE), for place categorization. Each model is derived from two base modules, one based on the PlacesCNN [ZLK+17] and the other being an Object Detector-YOLOv3 [RF18]. The classification model can be formulated as a supervised learning problem. Given a set of labeled training data $\mathcal{X}^{tr} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2)...(\mathbf{x}_N, \mathbf{y}_N)\}$, where $\mathbf{x}_i$ corresponds to the data samples and $\mathbf{y}_i$ to the scene labels, the classifier should learn the discriminative probability model

$$p(\hat{\mathbf{y}}_j | \Phi(\mathcal{X}^{tr})) \tag{3.1}$$

where $\hat{\mathbf{y}}_j$ corresponds to the j-th predicted scene label and $\Phi = \{\phi_1, \phi_2...\phi_t\}$ are the set of different feature representations obtained from the $\mathbf{x}_i$. This trained model should be able to correctly classify a set of unlabelled test samples $\mathcal{X}^{te} = \{\mathbf{x}_1, \mathbf{x}_2...\mathbf{x}_M\}$. It is to be noted that while the goal of each of our five models is to perform place categorization, it is the Φ which varies across them. We now describe the two base modules, and how our five models are derived and trained from them. The complete network architecture is given in Figure 3.1.

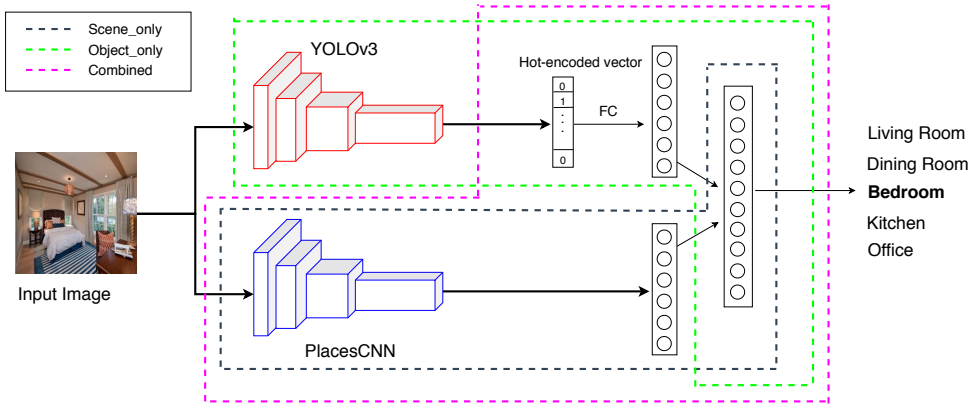


Figure 3.1: Model Architecture. The highlighted regions represent the portion of the network which was trained for the respective models.

3.2.1 Scene Recognition

For scene recognition, we use the PlacesCNN model. The base architecture is that of Resnet-18 [HZRS16] which has been pre-trained on the ImageNet dataset [DDS+09] and then fine-tuned on the Places365 dataset [ZLK+17]. We choose seven classes out of the total 365 classes, which are integral to the recognition of indoor home/office environments - Bathroom, Bedroom, Corridor, Dining room, Living room, Kitchen and Office. We use the official training and validation split provided for our work. The training set consists of 5000 labelled images for each scene-class, while the test set contains 100 images for each scene.

3.2.2 Object Detection

Object detection is a domain that has benefited immensely from the developments in deep learning. Recent years have seen people develop many algorithms for object detection, some of which include YOLO [RDGF16, RF17, RF18], SSD [LAE+16], Mask RCNN [HGDG17], Cascade RCNN [CV18] and RetinaNet [LGG+17]. We work with the YOLOv3 [RF18] detector here, mainly because of its speed, which makes real-time processing possible. It is a Fully Convolutional Network (FCN), and employs the Darknet-53 architecture which has 53 convolution layers, consisting of successive 3x3 and 1x1 convolutional layers with some shortcut connections. The network used here has been pre-trained to detect the 80 classes of the MS-COCO dataset [LMB+14].

3.2.3 Place Categorization models

Scene Only

The first model which we use consists of only the pre-trained and fine-tuned PlacesCNN with a simple Linear Classifier on top of it. This model accounts for a holistic representation of a scene, without specifically being trained to detect objects. Thus, the feature vector for this model

is given by $\Phi_{scene} = \phi_s$.

Object Only

The second model acts a Scene classifier using only the information of detected objects. There is no separate training performed here to identify the individual scene attributes. For this purpose, we create a codebook of the most common COCO-objects seen in all the seven scenes. This is shown in Table 3.1. It is to be noted that every object has been associated to only one scene, thereby making it a *landmark*. For this model, the feature representation is given by $\Phi_{obj} = \phi_{\{obj\}}$ where $\{obj\}$ is the set of objects detected in the image.

Table 3.1: Top landmark objects (non-human) for the seven different scene classes

Bathroom	Toilet	Sink	-	-
Bedroom	Bed	-	-	-
Corridor	-	-	-	-
Dining Room	Dining Table	Chair	Wine Glass	Bowl
Kitchen	Oven	Microwave	Refrigerator	-
Living Room	Sofa	Vase	-	-
Office	TV-Monitor	Laptop	Keyboard	Mouse

Scene+Attention

In this model, we compute the activation maps for the given image of a scene, and using those, we try to visualize where the network has its focus during scene classification. From the output of the final block convolutional layer (layer 4) of the WideResnet architecture [ZK16], we get the 14x14 feature blobs which retain the spatial information corresponding to the whole image. Our model is similar to the *soft* attention mechanism of [XBK$^+$15] in the sense that here too, we assign the weights to be the output of a softmax layer, thereby associating a probability distribution to it. However, since we are not classifying based on a sequence of images, we do not employ a recurrent network to compute the sequential features. Instead, we simply utilize the weights of the final FC layer and take its dot product with the feature blobs to obtain the

heatmap. The final step is to upsample this 14x14 heatmap to the input image size, and then overlay it on top to obtain the activation mask $m(x_n)$ of the input image x_n. Therefore, the feature representation for this model is $\Phi_{attn.} = \phi_{m(x_n)}$. The basic architecture is given in Figure 3.2.

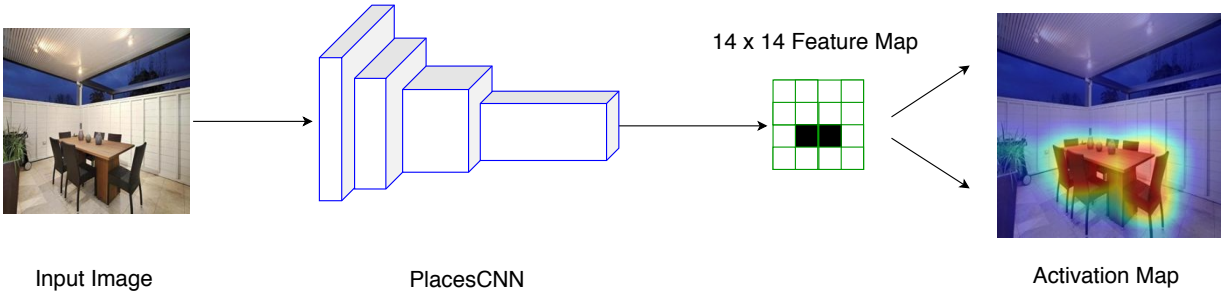


Figure 3.2: Architecture for Generation of the Activation Map. The 14x14 feature maps obtained from the block layer 4 of WideResNet are combined with the weights from the final FC layer, and then their dot product id upsampled to the image size and overlaid on top to get the activation maps

Combined

In this model, we use the PlacesCNN mentioned above as a feature extractor to give the semantics of a scene. In addition, the YOLO detector gives us the information regarding the objects present in the image. Given an image of a scene, our model creates a hot-encoded vector of 80 dimensions, corresponding to the object classes of MS-COCO, with only the indices of the detected objects set to 1. We then concatenate this vector along with that of the output of the scene feature extractor, and train a Linear Classifier on top of it. Since we combine the two different features of scene and objects, the feature representation here is given by $\Phi_{comb.} = \{\phi_s, \phi_{\{obj\}}\}$.

Scene+N-best objects

Our final model is similar to the above in the sense that here also, we use both the PlacesCNN and the YOLO detector. However, this model does not need to be retrained again and so, it is significantly faster. For this model, we place a certain confidence threshold on the scene detector, and only when the probability of classification is below this threshold, we search

for the information about specific objects in the scene (as obtained from Table 3.1). The reason for introducing this as a new model is two-fold. Firstly, we eliminate the scenario of looking at every object present since it is often redundant, given the semantics of the scene. Secondly, this is similar to how we as human beings operate when we come across an unknown scene. The feature representation for this model is given by $\Phi_{N-best} = \{\phi_s, \phi_{\{N-obj\}}\}$.

3.3 Experiments and Results

We evaluated our five models described above on a number of platforms. In this section, we first describe our training procedure, and then talk about the different experiment settings used for evaluation.

3.3.1 Training Procedure

As mentioned in Section 3.2, the base architecture for our scene classifier is the ResNet-18 architecture. The data pre-processing and training process is similar to [ZLK+17]. We used the Stochastic Gradient Descent (SGD) optimizer with an initial learning rate of 0.1, momentum of 0.9, and a weight decay of 10^{-4}. For the Φ_{scene} and the $\Phi_{attn.}$ models, the training was performed for 90 epochs with the learning rate being decreased by a factor of 10 every 30 epochs. The $\Phi_{comb.}$ model converged much faster and so, it was only trained for 9 epochs, with the learning rate reduced by 10 times after every 3 epochs. For all the 3 training procedures, the cross-entropy loss function was optimized, which minimizes the cost function given by

$$J(\hat{\mathbf{y}}_j, \mathbf{y}_j) = -\frac{1}{N}(\sum_{j=1}^{N} \mathbf{y}_j \odot log(\hat{\mathbf{y}}_j)) \tag{3.2}$$

The training process was carried out on a NVIDIA Titan Xp GPU using the PyTorch framework. The performance of the five DEDUCE algorithms on the test set of Places365 is

shown in Table 3.2 for the seven classes chosen.

Table 3.2: Accuracy in percentage of DEDUCE on Places365 dataset

Scenes	Φ_{scene}	Φ_{obj}	$\Phi_{attn.}$	$\Phi_{comb.}$	Φ_{N-best}
Dining room	79	**94**	75	79	80
Bedroom	90	74	90	90	**91**
Bathroom	**92**	65	**92**	91	**92**
Corridor	94	90	**99**	96	94
Living Room	**84**	25	68	80	**84**
Office	85	29	76	**94**	83
Kitchen	**87**	62	70	**87**	**87**
Avg	87.3	62.6	81.4	**88.1**	87.3

3.3.2 Experiment Settings

In order to check the robustness of our models, we further evaluated their performance on two state-of-the-art still-image datasets.

SUN Dataset

The SUN-RGBD dataset [SLX15] is one of the most challenging scene understanding datasets in existence. It consists of 3,784 images using Kinect v2 and 1,159 images using Intel RealSense cameras. In addition, there are 1,449 images from the NYUDepth V2 [SHKF12], and 554 manually selected realistic scene images from the Berkeley B3DO Dataset [JKJ+13], both captured by Kinect v1. Finally, it has 3,389 manually selected distinguished frames without significant motion blur from the SUN3D videos [XOT13] captured by Asus Xtion. Out of this, we sample the seven classes of importance and use the official test split to evaluate our models. We only consider the RGB images for this work since our training data doesn't have depth information. The performance is summarized in Table 3.3.

Upon comparison with Table 3.2, which contains the results on the Places365 dataset where our models were fine-tuned, a number of observations can be made which are consistent for both the datasets. Firstly, the $\Phi_{comb.}$ model performs the best. This is intuitive since here,

Table 3.3: Accuracy in percentage of DEDUCE on SUN dataset

Scenes	Φ_{scene}	Φ_{obj}	$\Phi_{attn.}$	$\Phi_{comb.}$	Φ_{N-best}
Dining room	65.2	**83.7**	53.3	67.4	72.8
Bedroom	43.7	36.5	**48.9**	**48.9**	47.3
Bathroom	94.5	87.0	**97.3**	96.6	95.2
Corridor	44.4	**67.6**	**67.6**	44.4	41.7
Living Room	58.8	24.0	43.6	**59.2**	58.8
Office	84.0	12.6	75.8	**90.6**	80.6
Kitchen	77.1	63.5	63.9	**83.8**	77.4
Avg	66.8	53.6	64.3	**70.1**	67.7

the scene classification is done using the combined training of both the information about the scene attributes and the object identity. Secondly, the Φ_{obj} model works the best for the *Dining Room* class, even though its overall performance is the worst. This trend can be attributed to the fact that dining rooms can be easily identified by the presence of specific objects, whereas the scene attributes might throw in some confusion (for instance when the kitchen/living room is partially visible in the image of a dining room). Thirdly, for *Corridor*, the performance of the $\Phi_{attn.}$ model is best for both the datasets. This supports the fact that in order to classify a scene like a corridor, viewing only a small portion of the image close to the vanishing point is sufficient. Finally, the Φ_{N-best} model performs just as good or better than the Φ_{scene} model. This proves that presence of objects does indeed improve the scene classification. For the best performance using the Φ_{N-best} model, the threshold was set to 0.5 for the *Places* dataset while it was 0.6 for the *SUN* dataset. The reason for the higher confidence on scene attributes for *Places* dataset is most likely due to the fact that the scene classifier itself was fine tuned on it.

VPC Dataset

The Visual Place Categorization dataset [WCR09] consists of videos captured autonomously using a HD camcorder (JVC GR-HD1) mounted on a rolling tripod. The data has been collected from 6 different home environments, and three different floor types. The advantage of this dataset is that the collected data closely mimics that of the motion of a robot - instead of focusing on

captured frames or objects/furniture in the rooms, the operator recording the data just traversed across all the areas in a room while avoiding collision with obstacles. For comparison with the state-of-the-art algorithms, we test our methods only on the five classes which are present in all the homes - Bathroom, Bedroom, Dining room, Living room and Kitchen. Table 3.4 contains the results for the individual home environments for these five classes. For the AlexNet [KSH12] and ResNet [HZRS16] models, we adopt the same training procedure as in [MBCR18]. It can be seen from the table that our models perform better than the rest in all but one of the home environments and much better in the overall performance.

Table 3.4: VPC Dataset: Average Accuracy across the 6 home environments

Networks	H1	H2	H3	H4	H5	H6	avg.
AlexNet	49.8	53.4	49.2	64.4	41.0	43.4	50.2
AlexNet+BN	54.5	54.6	55.6	69.7	41.8	45.9	53.7
AlexNet+WBN	54.7	51.9	61.8	70.6	43.9	46.5	54.9
AlexNet+WBN*	53.5	54.6	55.7	68.1	44.3	49.9	54.3
ResNet	55.8	47.4	64.0	69.9	42.8	50.4	55.0
ResNet+WBN	55.7	49.5	**64.7**	70.2	42.1	52.0	55.7
ResNet+WBN*	56.8	50.9	64.1	69.3	45.1	51.6	56.5
Ours (Φ_{scene})	**63.7**	57.3	63.7	**71.4**	60.2	65.9	63.7
Ours ($\Phi_{comb.}$)	**63.7**	**60.7**	64.5	70.7	**65.7**	**68.8**	**65.7**

Table 3.5 further compares our models with all other baseline algorithms tested on the VPC dataset. The reported accuracies are the average over all the six home environments. We first consider the methods described in [WCR09], which use SIFT and CENTRIST features with a Nearest Neighbor Classifier, and also exploit temporal information between images by coupling them with Bayesian Filtering (BF). Next, we look at the approach of [FET12] where Histogram of Oriented Uniform Patterns (HOUP) is used as input to the same classifier. [YW12] proposed the method of using object templates for visual place categorization, and reported results

Table 3.5: VPC Dataset: Comparison with State Of The Art

Method	[WCR09]				[FET12]	[YW12]		AlexNet			ResNet		**Ours**	
Config.	SIFT	SIFT+BF	CE	CE+BF	HOUP	G+BF	G+O(SIFT)+BF	Base	BN	WBN*	BN	WBN*	**Scene-only**	**Combined**
Acc.	35.0	38.6	41.9	45.6	45.9	47.9	50	50.2	53.7	54.3	55.0	56.5	**63.7**	**65.7**

for Global configurations approach with Bayesian Filtering (G+BF), and that combined with the object templates (G+O(SIFT)+BF). Ushering the deep learning era, AlexNet [KSH12] and ResNet [HZRS16] architectures give better results, both with their base models, as well as the Batch Normalized (BN) and the Weighted Batch Normalized versions [MBCR18]. However, comparisons with our Φ_{scene} and $\Phi_{comb.}$ models show that our methods beat all the other results by significant margins.

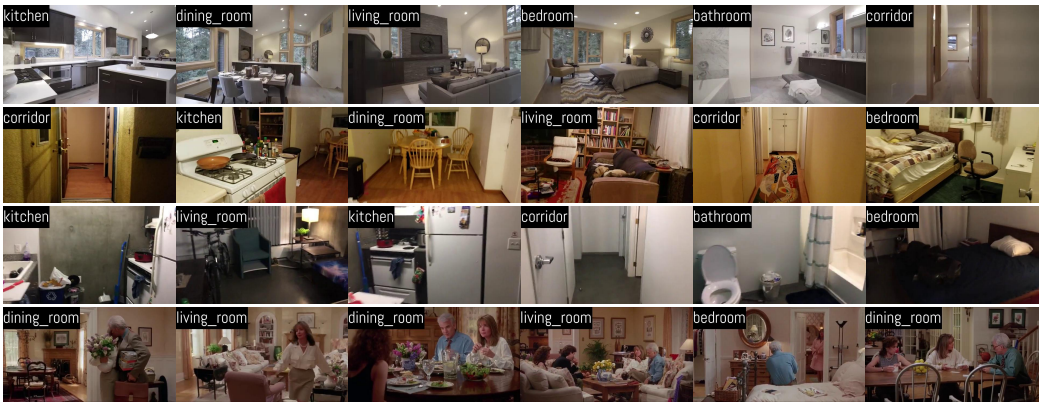


Figure 3.3: Detection Results on Real-World Videos. Top row corresponds to the video of a Real-Estate model house. The next 2 rows are from the houses of the authors and their friends. The bottom row is obtained from a house in the movie *"Father of the Bride"*.

Real-World Scene Recognition

In order to test the robustness of our DEDUCE models, we expand the domain of test cases beyond the aforementioned still image data sets. Figure 3.3 does this by showing the results of scene recognition on real-world data recorded using hand-held cameras. We employ the Φ_{N-best} model for these cases due to its ability to mimic the natural behavior of humans, whereby an initial prediction is made based on the scene attributes, and if unsure, more information related to specific objects is gathered in order to update/re-inforce the initial prediction. The top row corresponds to the tour of a semi-furnished real estate home obtained from YouTube which only has the relevant objects in the scene. Although it is not a sequential tour of the house, it does contain all the rooms. Also, professional photographers captured this video and hence, the image

quality and white-balance of the camera is pretty good. The next-2 rows pose a more challenging case as they correspond to homes currently inhabited by people. We consider two examples of these houses, one which is a standard bungalow residence, while the other being a student apartment. From experience, the bungalow is a much cleaner home, whereas student apartments are prone to presence of cluttered objects and overlapping scene boundaries. Moreover, the videos were recorded by the inhabitants using their cellphone cameras. This inherently brings motion blur into the picture, specially during scene transitions. Finally, the last row depicts the settings of a house from the movie *"Father of the Bride"*. This ensures that our model is robust enough to classify scenes even when the focus of the recording is on people instead of the background settings. All the detection results mentioned in this chapter are available as individual videos in the following link `https://goo.gl/sYyVZ2`.

Semantic Mapping

The experimental setting for semantic mapping involves running our algorithm on a mobile robot platform in two different environments. The platform is a Fetch Mobile Manipulator and Freight Mobile Robot Base by Fetch Robotics*. Figure 1.2 shows the robot performing scene classification in one of the environments.

The semantic maps for the experiments were constructed using Omnimapper [TRC14]. It utilizes the GTsam library in order to optimize a graph of measurements between robot poses along a trajectory, and between robot poses and various landmarks in the environment. The measurements of simple objects like points, lines, and planes are data associated with mapped landmarks using the joint compatibility branch and bound (JCBB) technique [NT01]. The regions for color segmentation are acquired by the Gaussian Region algorithm of [NGRTC10]. However, in [NGRTC10], the map partitions were built through human guidance, whereby the robot was taken on a tour of the space (either by driving the robot manually, or using a person following

*`https://fetchrobotics.com/robotics-platforms/`

behavior) and the respective scene labels were taught to it. This is in contrast to our approach, where the labels are learned from our visual place categorization system. Thus, the robot is itself capable of identifying the scenes without any human guide. We used the Φ_{N-best} model for this task, and retrained the scene classifier to exclude the *Bedroom*, *Dining Room* & *Bathroom* scenes, and instead include *Conference Room* as it is more likely to occur in an academic building environment.

Figure 3.4a shows the navigation of the robot in the Computer Science and Engineering (CSE) Building. Our system was able to classify the seven regions of the floor map. However, there are some regions detected by Omnimapper using the laser range finder. These are painted in white to denote their invisibility to the camera. The second test environment is the Contexual Robotics Institute (CRI) building, which has a very different floor map in comparison to CSE. The result of the run made here is shown in Figure 3.4b.

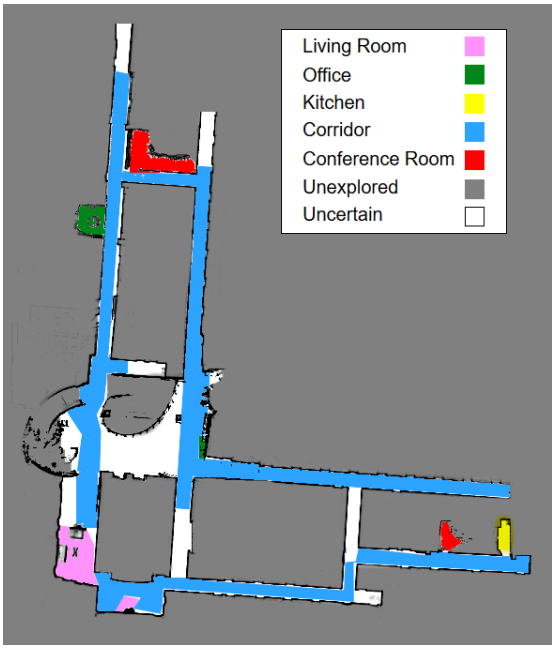
3.4 Conclusion

In this chapter, we considered five different models for performing place categorization, which are derived mainly from two base modules - a scene recognizer, and an object detector. We demonstrated the effectiveness of our algorithms in a series of experiments, ranging from scene recognition in still-image data sets to real-world videos captured from different sources, and finally via the generation of labeled semantic maps using data gathered by multiple mobile robot platforms.

We showed that (i) different models are favorable for different scenes (Table 3.2 and 3.3), and thus the ideal scene recognition system would likely be a combination of these five models, (ii) the proposed methods give successful results on many different types of video recordings, even when they are affected by object clutter, motion blur, and overlapping boundaries and (iii) our models are robust enough to be tested on data gathered by mobile robotic platforms on

multiple building scenarios which are affected by occlusions and poor lighting conditions.

We plan to extend our experimental evaluation to other mobile robots and then flying robots. While we demonstrated the effectiveness of our approach in different indoor scenarios, we believe that the next step is to allow the robot to walk on a tour and label important regions in its environment. This would be useful for navigation in unknown places using a semantic map for robots and humans. Furthermore, our system could be applied to autonomous robots, thus enabling them to assist humans in safety and rescue missions inside a house or a building.



(a) Semantic Map of the CSE building.

(b) Semantic Map of the CRI.

Figure 3.4: Place categorization experiments with mobile robots. Each color represents one of the seven classes of the visual place categorization that our system classified.

Chapter 4

Visual Attention Mechanism for Action Classification

4.1 Background

The study of Optical Flow in Computer Vision has long been an interesting domain, right from its inception. Horn *et al*. [HS81], Lucas *et al*. [LK81] Fleet *et al*. [FW06], and Sun *et al*. [SKS+18] give a background on what Optical Flow is and how it can be determined in a computer vision problem. Convolutional Neural Networks are the primary form of architecture used in most Computer Vision problems these days. Deep Learning essentially involves having a series of convolutional layers, followed by multiple fully-connected layers which enable the step-wise detection of features of importance, finally leading to the higher level-linguistics representations.

Gathering temporal information is of prime importance for video-based classification techniques. Karpathy *et al*. [KTS+14] used this and introduced the concepts of Early Fusion and Late Fusion. Early Fusion combines information over a time window at a pixel-level, while Late Fusion places two separate single-frame networks until the last convolution layer with frames

being separated periodically. Another representation of fusion is the Two-Stream Convolutional Networks [SZ14] used for classifying videos, where the spatial and temporal information form the two steams and are processed separately. Recurrent Neural Networks are useful when we associate a memory with a sequence of data gathered across several frames in a video [GBC16].

Our model uses the above-mentioned concepts and expands them to incorporate attention in a series of frames to classify videos. We tested our model on a labeled dataset of human actions spread over the length of a video. The dataset is custom-made and involves common human actions performed and captured in a short video format.

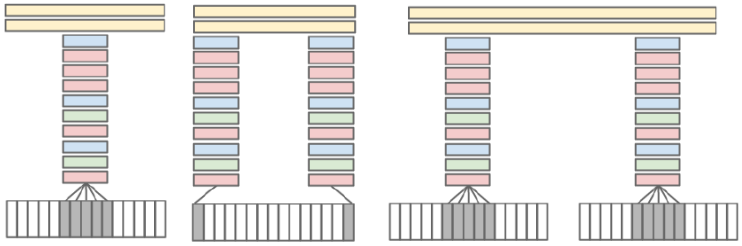


Figure 4.1: From left to right: Comparison of early-fusion, late-fusion, and two-stream methods.

4.2 Proposed Methodology

4.2.1 Camera motion estimation and scene structure in video frames

To show the connection between camera motion and image point motion, we first introduce a general optical flow equation:

$$u(x) = N_x(d\tau + Wx), \tag{4.1}$$

where $x \in \mathbb{R}^3$ is a point on the imaging surface, $u(x) \in \mathbb{R}^3$ is the optical flow vector at that point (always tangent to the imaging surface), d is the inverse distance from the camera principal point to the scene point imaged at x, τ is the camera translation vector, and W is the camera angular velocity. N_x is the linear transformation, a 3×3 matrix, that projects flow on the

unit sphere at x to flow on the true imaging surface.

Lucas-Kanade Method

We calculate the motion of each person via Lucas-Kanade Method[LK81]. This method assumes a constant small velocity between two nearby frames. We want to minimize the sum of squared error between template T (or the first frame) and the second frame I warped back onto the template. We setup the cost function as:

$$\sum_{x,y}[I(W(x,y;p)) - T(x,y)]^2 \tag{4.2}$$

We minimize the equation as a non-linear optimization problem because pixel values $I(x,y)$ are independent of coordinates x,y. Therefore, we optimize the method by descent step Δp:

$$\sum_{x,y}[I(W(x,y;p) + \Delta p) + T(x,y)]^2 \tag{4.3}$$

The update step can be written as

$$p_{t+\Delta} = p_t + \Delta p \tag{4.4}$$

Next we applied the first order term of Taylor series

$$\sum_{x,y}[I(W(x,y;p)) + \nabla I \frac{\partial w}{\partial p}\Delta p - T(x,y)]^2 \tag{4.5}$$

We assume the motion is pure translation.

$$W((x,y;p) + \Delta p)) = \begin{vmatrix} x + p1 \\ y + p2 \end{vmatrix} \tag{4.6}$$

$$\frac{\partial w}{\partial p} = \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} \tag{4.7}$$

Then, we solve for Δp by Gauss-Newton Method where Δp is

$$\Delta p = H^{-1} \sum_{x,y} [\nabla I]^T [T(x,y) - I(W(x,y;p))] \tag{4.8}$$

and the Hessian is

$$H = \sum_{x,y} [\frac{\partial w}{\partial p}]^T [\nabla I \frac{\partial W}{\partial p}] \tag{4.9}$$

$$H = \sum x,y [\nabla I]^T [\nabla I] \tag{4.10}$$

For a clearer visual analysis, Figure 4.2 shows the steps to compute optical flow from a sequence of two pictures. The arrows represent the motion which the object has moved with respect to the camera view.

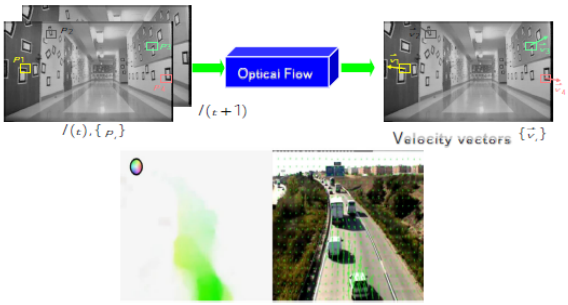
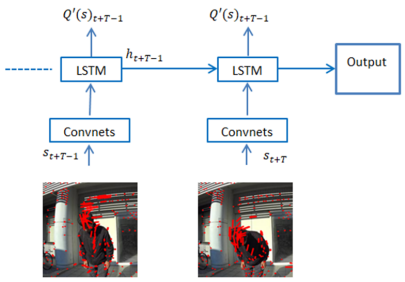


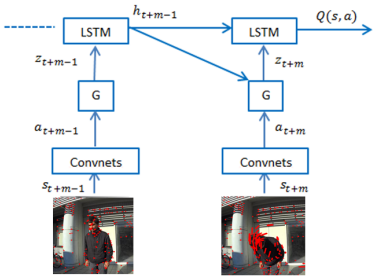
Figure 4.2: Diagram of computing optical Flow via Lucas-Kanade Method. In the above image, optical flow gives the velocity gradients of the pixels which move in position over time. In the lower image, we highlight the portion of the image which is in motion (the vehicles).

4.2.2 Neural Networks

In this section, we describe our proposed neural network model for video classification. We present two different models i.e., Recurrent Convolutional Neural Networks (R-CNN: Figure 4.3a) and Recurrent Attentive Convolutional Neural Networks (R-ACNN: Figure 4.3b). R-ACNN extends R-CNN by adding the neural attention module. Attention model enables a neural network to focus on certain parts of an input image instead of processing it entirely at a fine scale. This region selection reduces the number of training parameters as well as the computational operations. Beside computational benefits, attention models also provides information about where a neural network is looking at, while taking any decision. Rest of the section is divided up according to the different neural models of R-CNN and R-ACNN i.e., 1) Convnets; 2) Long Short-term Memory (LSTM) network; and 3) Attention network (G) (used by R-ACNN). The flow of information between the different modules of R-CNN and R-ACNN can be seen in Figure 4.3a and Figure 4.3b.



(a) Recurrent Convolutional Neural Network



(b) Recurrent Attentive Convolutional Neural Network

Figure 4.3: Neural network architectures used

ConvNets

The ConvNets take preprocessed visual frame as input in each time-step and transforms it into L feature vectors, each of which provides D-dimensional representation of a part of an input image i.e, $a_t = \{\mathbf{a}_t^1, \cdots, \mathbf{a}_t^L\}, \mathbf{a}_t^l \in \mathbb{R}^D$. This feature vector is either taken as an input by the linear

layer in case of R-CNN or by attention network in case of R-ACNN to generate an annotation vector $z \in \mathbb{R}^D$.

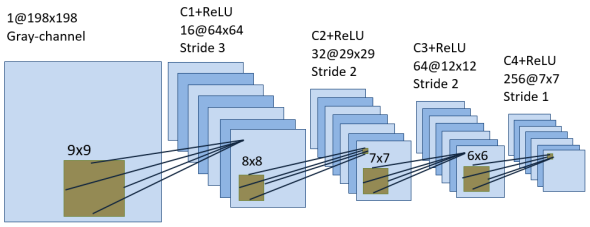


Figure 4.4: The structure of the Convolutional network encoder.

LSTM

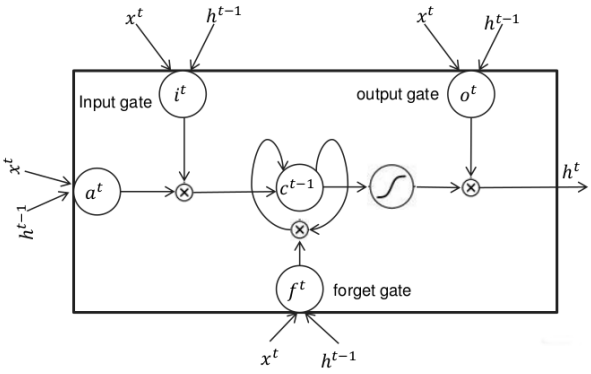


Figure 4.5: The structure of a long short-term memory (LSTM) network.

We employ the following implementation of LSTM network. It is based on the structure shown in Figure 4.5.

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} M \begin{pmatrix} h_{t-1} \\ z_t \end{pmatrix} \tag{4.11}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{4.12}$$

$$h_t = o_t \odot \tanh(c_t) \tag{4.13}$$

where i_t, f_t, o_t, c_t, and h_t correspond to the input, forget, output, memory and hidden state of the LSTM respectively. The activation functions for the input, forget, and output gate is given by the sigmoid function σ, while for the hidden state, it is the hyperbolic tangent *tanh* function. Let d be the dimensionality of all LSTM states and matrix $M : \mathbb{R}^a \rightarrow \mathbb{R}^b$, in equation 4.11, is an affine transformation of trainable parameters with dimension $a = d + D$ and $b = 4d$. As shown in equation 4.13, the LSTM network takes the annotation vector $z_t \in \mathbb{R}^D$, previous hidden state h_{t-1}, and the previous memory state c_{t-1} as an input in order to produce the next hidden state h_t. This hidden state h_t is given to the attention network G (in case of R-ACNN) and to the linear output layer for generating the annotation vector z_{t+1} at next time step and for providing the output prediction value for each of the legal classes, respectively.

Attention network

The attention network (for R-ACNN) generates a dynamic representation, called annotation vector z_t, of the corresponding parts of an input image at time t. The attention mechanism ϕ, a multilayer perceptron, takes a D-dimensional L feature vectors a_t and a previous hidden state h_{t-1} of the LSTM network as an input for computing the positive weights β_t^l for each location l. The weights β_t^l are computed as follow:

$$\beta_t^l = \frac{\exp(\alpha_t^l)}{\sum_{k=1}^{L} \exp(\alpha_t^k)}, \tag{4.14}$$

where

$$\alpha_t^l = \phi(a_t^l, h_{t-1}), \tag{4.15}$$

The annotation vector z_t is given by

$$z_t = \sum_{l=1}^{L} \beta_t^l a_t^l. \tag{4.16}$$

This annotation vector is used by LSTM for computing next hidden state.

There are two types of attention networks [XBK+15] in the literature: the soft and hard attention network. The attention network used in our method is the soft attention network and unlike hard attention network, it is fully differentiable and deterministic.

4.3 Technical Approach

This section outlines the implementation details of our project. The algorithms were implemented in Pytorch. The system used for training our model has 3.40GHz octa-core Intel Core i7 processor with 32 GB RAM and NVIDIA GeForce GTX 1080 GPU. The rest of the section explains various modules of the project.

4.3.1 Data Collection and Pre-processing

The collected dataset comprises of 5 classes that we use in daily life communication. Each gesture is depicted in Figure 4.6, along with a representative caption. Moreover, because we wanted to ensure that the network is invariant to irrelevant factors such as background or lighting and sensitive only with respect to gesture, we collected videos in 7 different locations, outdoors and indoors. Furthermore, participants were recorded with three cameras, positioned at different angles and heights. As the gestures involve motion to be clearly distinguishable, we decided to include the temporal information by giving the network a series of consecutive frames, instead of a single image. From each gesture clip, we randomly sampled 5 frames from 4 consecutive

Figure 4.6: Examples of input sequences for each class. From top to bottom, the classes are “bye”, ”don’t know”, “konnichiwa”, “quiet”, “high five”.

frame intervals. Around 160 samples were obtained per gesture class. Afterwards the images are rescaled and cropped in the center area such that their dimensions decrease to 210×210. After being transformed to YUV input, frames go through the process of computing optical flow [HS81]. Finally, the images were globally normalized, such that the mean of all pixel values was 0 and the standard deviation 1.

4.3.2 Model Architecture and hyper-parameters

This section provides the details of the architecture of our R-CNN and R-ACNN model. The convolutional neural network consists of four convolution layers each of which is followed by a non-linear rectifier function. The input dimension to the CNN is $2 \times 210 \times 210$. The convolution layer 1, 2, 3 and 4 convolves 16 filters of 9×9, 32 filters of 8×8, 64 filters of 7×7 and 256 filter of 6×6, respectively. The stride of the convolution layers 1, 2, 3 and 4 are 3, 2, 2 and 1 respectively. The CNN outputs 256 feature maps of dimension 7×7. The output feature maps from CNN are given to the attention network which takes 49 vectors each of size 256. To be consistent with attention network, the LSTM network also has 256 units. To generate prediction-values for the five classes, the output of the LSTM is transformed to five units through a linear layer preceded by non-linear rectifier unit. Finally, the cost function is negative log-likelihood and the learning rate was kept at 10^{-3}.

4.4 Results and Discussion

The dataset consists of 780 videos in total, with 624 (or 80%) for training and 156 (or 20%) for testing. Five models, including Early Fusion, Late Fusion, R-CNN, Two-stream, and R-ACNN, have been implemented. Where applicable, optical flow have been added to the dataset. The results of these experiments are shown in Table 4.1.

The results suggest that the best models are our R-CNN and R-ACNN with optical flow,

Table 4.1: The accuracy for each of the five models, with and without optical flow.

Models	Grayscale Accuracy	Optical Flow Accuracy
Early Fusion	42.8%	–
Late Fusion	57.3%	86.7%
R-CNN	59.0%	**98.2**%
Two Stream	87.3%	87.3%
R-ACNN	57.8%	**97.4**%

having accuracy above 97%. The next best models, two-stream and late fusion, have only been able to achieve 87% accuracy. Other methods, especially the ones that do not use optical flow, could not achieve an accuracy higher than 60%.

Figure 4.7: The attention masks generated from training the recurrent attentive convolutional neural network (R-ACNN). The regions with higher attention are marked with brighter color.

We observe that using optical flow significantly boosts accuracy in late fusion, R-CNN and R-ACNN models. The two-stream model does not appear to be able to effectively use the optical flow. In contrast, the R-CNN and R-ACNN models get the most significant boost in accuracy by using optical flow. Curiously, R-ACNN does not perform better than R-CNN in our experiments. In fact, R-ACNN has slightly lower accuracy than R-CNN, regardless of the use of optical flow. We believe this difference in accuracy is due to the fact that attentive model focuses

only on a small region of the input, where as the non-attentive model uses the entire input. If the attention is in the wrong region, then the classification becomes much less likely to be correct.

What, then, is the advantage of using an attentive model such as R-ACNN over R-CNN? An answer to this question is that the attention masks which the attentive model produces may provide intuitive ways to interpret the trained models. Figure 4.7 shows the attention masks generated by the R-ACNN. The attention masks are produced by performing a convolution of a Gaussian blur filter with the annotation vector z_t, which is reshaped and rescaled to match the input frame size, and then adding it on top of the input image with a non-zero alpha channel.

Figure 4.8: The attention masks for two training examples at epochs 0, 2, 5, 20, and 65 demonstrate the convergence of attention masks to a local region.

We observe that in all examples except for "shh", the attention masks focus on "high-activity regions", ie the regions where significant motion is present. For the "konnichiwa" action, the region is the upper body; for the "don't know", "wave" and "high five" actions, the elbow. For the "shh" action, the model identifies the region of interest outside the body. We believe that these highlighted regions demonstrate the regions where the model finds the most "telling" about the action in question, similar to PCA identifying the most significant factors of a model. The attention masks are especially useful for video and image data, since they allow even non-

technical persons to peek into the network's mechanism. This might be important to consider in designing service robots, as users may be more willing to interact with a robot that displays its attention.

How are the attention masks formed by training? Figure 4.8 shows the evolution of two attention masks over training epochs. In both examples, the attention masks converge to a local region. Before training, the attention is dispersed across large areas in the images. With training, the attention gradually collapses into a few prominent regions (usually two) before one region eventually dominates. The examples suggest that the model is the healthiest around epoch 20, as the model at this point has formed a few regions of interest while not yet assigning all trust to a single region. This might prevent over-fitting, to which the relatively small size of our dataset is prone.

Chapter 5

Conclusion and Future Work

This thesis presents a detailed analysis into the study of classifiers, with a focus on what information they can leverage in order to solve a problem. In Chapter 2, this is done through the concept of bias in action recognition, which illustrates the fact that deep learning models often achieve better than average performance, simply by exploiting a set of representations which can be quite different from the ground-truth. New metrics like Fairness score and Bias skew have been introduced, which can be used to assess the bias of an action recognition dataset. A future direction of work in this domain is to utilize the relationship between fairness score and bias to classify a dataset in terms of its classification difficulty. Another possible extension to this work is to evaluate the bias preference of state of the art algorithms. While it is likely that modern algorithms tend to incorporate multiple representations, it is still interesting to see the distribution of these representations. Finally, a new criterion for selecting classes for the dataset manipulation experiment is being thought of, which takes the ratio of the two biases instead of their relative performance.

In Chapter 3, the idea of classification is extended into place categorization, which is an important topic in robotics. The set of algorithms proposed have been proven to be robust to a number of different domains, ranging from simple 2D images, to real world videos captured

via different hand-held cameras and a mobile robot platform. For the next step, the idea is to extend this model in order to do place categorization in rural households, where objects might not be as representative of their surrounding. We also want to add this module to motion planning algorithms in order to solve the task-and-motion planning problems.

Finally, Chapter 4 of the thesis mentions the addition of a visual attention mechanism, combined with optical flow guidance to a two-stream action recognition network. The results show significant improvements over the base algorithm. An extension to this work currently being planned of is to predict an action ahead of time, given the first few frames. This is of importance in autonomous driving scenarios where it is essential to predict the intent of people in an intersection as to whether they want to enter the cross-walk or not.

This thesis is currently being prepared for submission for publication of the material. Pal, Anwesan - The thesis author was the primary investigator and author of this material.

Bibliography

[BBPW04] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision*, pages 25–36. Springer, 2004.

[BGS+05] Moshe Blank, Lena Gorelick, Eli Shechtman, Michal Irani, and Ronen Basri. Actions as space-time shapes. In *The Tenth IEEE International Conference on Computer Vision (ICCV'05)*, pages 1395–1402, 2005.

[BJL+16] Richard Bormann, Florian Jordan, Wenzhe Li, Joshua Hampp, and Martin Hägele. Room segmentation: Survey, implementation, and analysis. In *ICRA*, 2016.

[BKMG09] M. Begum, F. Karray, G. K. I. Mann, and R. G. Gosine. Re-mapping of visual saliency in overt attention: A particle filter approach for robotic systems. In *2008 IEEE International Conference on Robotics and Biomimetics*, pages 425–430, 2009.

[BMG06] M. Begum, G. K. I. Mann, and R. G. Gosine. A biologically inspired bayesian model of visual attention for humanoid robots. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 587–592, 2006.

[BMGK08] Momotaz Begum, George K. I. Mann, Ray G. Gosine, and Fakhri Karray. Object- and space-based visual attention: An integrated framework for autonomous robots. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 301–306, 2008.

[CGGS12] Dan Ciresan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[CLS15] Guilhem Ch'eron, Ivan Laptev, and Cordelia Schmid. P-CNN: Pose-based CNN Features for Action Recognition. In *ICCV*, 2015.

[CSWS17] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299, 2017.

[CV18] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, pages 6154–6162, 2018.

[CZ17] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017.

[DDS+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[DT05] 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.

[EKJ07] S. Ekvall, D. Kragic, and P. Jensfelt. Object detection and mapping for service robot tasks. *Robotica*, 25(2):175–187, 2007.

[EKSR10] P. Espinace, T. Kollar, A. Soto, and N. Roy. Indoor scene recognition through object detection. In *ICRA*, 2010.

[FCNL12] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2012.

[FET12] Ehsan Fazl-Ersi and John K Tsotsos. Histogram of oriented uniform patterns for robust place recognition and categorization. *IJRR*, 2012.

[FPWZ18] Christoph Feichtenhofer, Axel Pinz, Richard P Wildes, and Andrew Zisserman. What have we learned from deep representations for action recognition? In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7844–7853, 2018.

[FW06] David J. Fleet and Y. Weiss. Optical flow estimation. In *Handbook of Mathematical Models in Computer Vision*, 2006.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[GDDM14] 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.

[GGM15] Georgia Gkioxari, Ross Girshick, and Jitendra Malik. Contextual action recognition with r*cnn. In *International Conference on Computer Vision (ICCV)*, pages 1080–1088, 2015.

[GKM+17] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo bax, and Roland Memisevic. The" something something" video database for learning and evaluating visual common sense. In *International Conference on Computer Vision (ICCV)*, 2017.

[HGDG17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017.

[HS81] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[IMS+17] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[Jam50] William James. *The principles of Phsychology*. Dover Publications, 1950.

[JKJ+13] Allison Janoch, Sergey Karayev, Yangqing Jia, Jonathan T Barron, Mario Fritz, Kate Saenko, and Trevor Darrell. A category-level 3d object dataset: Putting the kinect to work. In *Consumer depth cameras for computer vision*, pages 141–165. Springer, 2013.

[KCS+17] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[KG15] Ioannis Kostavelis and Antonios Gasteratos. Semantic mapping for mobile robotics tasks. *Robot. Auton. Syst.*, 66(C), April 2015.

[KH09] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[KJSS13] Hilde Kuehne, Hueihan Jhuang, Rainer Stiefelhagen, and Thomas Serre. Hmdb51: A large video database for human motion recognition. In *High Performance Computing in Science and Engineering 12*, pages 571–582. Springer, 2013.

[KL51] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[KR09] T. Kollar and N. Roy. Utilizing object-object and object-scene context when planning to find things. In *ICRA*, pages 4116–4121, 2009.

[KRA+18] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, and Vittorio Ferrari. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv:1811.00982*, 2018.

[KSH12] 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.

[KTS+14] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[LAE+16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LC18] Ren C. Luo and Michael Chiou. Hierarchical semantic mapping using convolutional neural networks for intelligent service robotics. *IEEE Access*, 6:61287–61294, 2018.

[LGG+17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017.

[LK81] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 674–679, 1981.

[LKW+16] Yiyi Liao, Sarath Kodagoda, Yue Wang, Lei Shi, and Yong Liu. Understand scene categories by objects: A semantic regularized scene classifier using convolutional neural networks. In *ICRA*. IEEE, 2016.

[LKW+17] Yiyi Liao, Sarath Kodagoda, Yue Wang, Lei Shi, and Yong Liu. Place classification with a graph regularized deep neural network. *IEEE Transactions on Cognitive and Developmental Systems*, 9(4), 2017.

[LLV18] Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 513–528, 2018.

[LMB+14] 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 *European conference on computer vision*, pages 740–755. Springer, 2014.

[MBCR18] Massimiliano Mancini, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Robust place categorization with deep domain generalization. *IEEE RA-L*, 2018.

[MBG+18] Farzaneh Mahdisoltani, Guillaume Berger, Waseem Gharbieh, David Fleet, and Roland Memisevic. Fine-grained video classification and captioning. *arXiv preprint arXiv:1804.09235*, 2018.

[MCKA17] Chih-Yao Ma, Min-Hung Chen, Zsolt Kira, and Ghassan AlRegib. TS-LSTM and temporal-inception: Exploiting spatiotemporal dynamics for activity recognition. *CoRR*, abs/1703.10667, 2017.

[MH08] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[MLS09] Marcin Marszałek, Ivan Laptev, and Cordelia Schmid. Actions in context. In *IEEE Conference on Computer Vision & Pattern Recognition*, 2009.

[NGRTC10] C Nieto-Granda, J G Rogers III, A J B Trevor, and H I Christensen. Semantic Map Partitioning in Indoor Environments Using Regional Analysis. In *IROS*, Taiwan, October 2010. IEEE.

[NT01] José Neira and Juan D Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Transactions on robotics and automation*, 17(6):890–897, 2001.

[OT01] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175, 2001.

[PMC08] A. Pronobis, O. M. Mozos, and B. Caputo. Svm-based discriminative accumulation scheme for place recognition. In *ICRA*, 2008.

[QT09] Ariadna Quattoni and Antonio B. Torralba. Recognizing indoor scenes. In *CVPR*, pages 413–420, 2009.

[RDGF16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[RDS+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C.Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[RF17] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017.

[RF18] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[RHGS15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[SBS+18] Niko Sänderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter Corke. The limits and potentials of deep learning for robotics. *IJRR*, 2018.

[SC18] Pierre Stock and Moustapha Cisse. Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases. In *European Conference on Computer Vision (ECCV)*, pages 498–512, 2018.

[SDM+16] Niko Sünderhauf, Feras Dayoub, Sean McMahon, Ben Talbot, Ruth Schulz, Peter Corke, Gordon Wyeth, Ben Upcroft, and Michael Milford. Place categorization and semantic mapping on a mobile robot. In *ICRA*, 2016.

[SHKF12] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[SI07] C. Siagian and L. Itti. Rapid biologically-inspired scene classification using features shared with visual attention. *T. PAMI*, 29(2):300–312, 2007.

[SKS+18] Shuyang Sun, Zhanghui Kuang, Lu Sheng, Wanli Ouyang, and Wei Zhang. Optical flow guided feature: A fast and robust motion representation for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1390–1399, 2018.

[SLC04] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[SLX15] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, pages 567–576, 2015.

[SMTA18] Hao Sun, Zehui Meng, Pey Yuen Tao, and Marcelo H Ang. Scene recognition and object detection in a unified convolutional neural network on a mobile manipulator. In *ICRA*, pages 1–5. IEEE, 2018.

[SVW+16] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, 2016.

[SZ14] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[SZS12] 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.

[TAS14] Siyu Tang, Mykhaylo Andriluka, and Bernt Schiele. Detection and tracking of occluded people. *International Journal of Computer Vision*, 110(1):58–69, 2014.

[TE11] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1521–1528, 2011.

[TMFR03] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin. Context-based vision system for place and object recognition. *ICCV*, 1:273, 2003.

[TPCT17] Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. In *Domain Adaptation in Computer Vision Applications*, pages 37–55. Springer, 2017.

[TRC14] Alexander JB Trevor, John G Rogers, and Henrik I Christensen. Omnimapper: A modular multimodal mapping framework. In *ICRA*. IEEE, 2014.

[TSY17] Zhehang Tong, Dianxi Shi, and Shaowu Yang. Sceneslam: A slam framework combined with scene detection. In *ROBIO*, 2017.

[WCR09] J. Wu, H. I. Christensen, and J. M. Rehg. Visual place categorization: Problem, dataset, and algorithm. In *IROS*, 2009.

[WXWQ15] Limin Wang, Yuanjun Xiong, Zhe Wang, and Yu Qiao. Towards good practices for very deep two-stream convnets. *CoRR*, abs/1507.02159, 2015.

[XBK+15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pages 2048–2057, 2015.

[XOT13] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, 2013.

[YW12] Hao Yang and Jianxin Wu. Object templates for visual place categorization. In *ACCV*, pages 470–483, 2012.

[ZK16] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[ZLK+17] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[ZPB07] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint Pattern Recognition Symposium*, pages 214–223. Springer, 2007.