ABSTRACT

Title of dissertation: CONTEXT MODELS FOR UNDERSTANDING IMAGES AND VIDEOS

Varun K. Nagaraja
Doctor of Philosophy, 2016

Dissertation directed by: Professor Larry S. Davis
Department of Computer Science.

A computer vision system that has to interact in natural language needs to understand the visual appearance of interactions between objects along with the appearance of objects themselves. Relationships between objects are frequently mentioned in queries of tasks like semantic image retrieval, image captioning, visual question answering and natural language object detection. Hence, it is essential to model context between objects for solving these tasks. In the first part of this thesis, we present a technique for detecting an object mentioned in a natural language query. Specifically, we work with referring expressions which are sentences that identify a particular object instance in an image. In many referring expressions, an object is described in relation to another object using prepositions, comparative adjectives, action verbs etc. Our proposed technique can identify both the referred object and the context object mentioned in such expressions.

Context is also useful for incrementally understanding scenes and videos. In the second part of this thesis, we propose techniques for searching for objects in an image and events in a video. Our proposed incremental algorithms use the context from previously explored regions to prioritize the regions to explore next. The advantage of incremental understanding is restricting the amount of computation time and/or resources spent for various detection tasks. Our first proposed technique shows how to learn context in indoor scenes in an implicit manner and use it for searching for objects. The second technique shows how explicitly written context rules of one-on-one basketball can be used to sequentially detect events in a game.

CONTEXT MODELS FOR UNDERSTANDING IMAGES AND VIDEOS

by

Varun K. Nagaraja

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2016

Advisory Committee:
Professor Larry S. Davis, Chair/Advisor
Professor Rama Chellappa
Professor David Jacobs
Professor Hal Daumé III
Professor Tom Goldstein

© Copyright by
Varun K. Nagaraja
2016

Acknowledgments

I would like to express my sincere gratitude to my advisor Prof. Larry Davis. He provided me ample freedom during my Ph.D to explore many problems while providing valuable guidance which saved me from pursuing unfruitful ideas. His feedback has greatly helped me in improving my ability to clearly convey ideas in papers and presentations.

I thank Vlad Morariu for patiently listening to all my crazy ideas, providing insightful comments during our discussions and also motivating me when I got stuck. During the initial phase of my graduate study, I got to work with Wael Abd-Almageed and I am grateful for the opportunity he provided me. I am also thankful to Prof. Hal Daumé, for helping with his code which I used in my work on object searching, and Prof. David Jacobs, for letting me sit in his group meetings during the beginning of my graduate study and also providing advice with an extremely friendly attitude.

I got to spend two summers at Amazon working on the wonderful Echo with Shiv Vitaladevuni and his team. I learned a lot from him on how to methodically solve problems, and got a glimpse of what it takes to build an amazing product.

My graduate student life would not have gone smooth without the help of Jenny, Fatima, Brenda, Sharron and Arlene. They were always quick in helping me with any administrative issue. I thank Prof. Samir Khuller and the members of the CS department for supporting our enthusiasm to create a social environment in the department.

I have been extremely lucky to have lived with amazing roommates - Rajesh, Ashish, Prathyusha, Vivek, Arvind and Puneeth. I can't thank them enough for taking care of me. I also found great friends in the lab and the department - Ejaz, Sravanthi, Bharat, Abhishek, Preeti and Snigdha. They provided invaluable help and feedback on the many problems that I worked on. It was a lot of fun spending time with them while working on course projects, and also outside of school. I am also thankful to Aleks, Kotaro, Christoph, Zamira, Aishwarya, Vikas, Matt, Austin, and Angjoo for all the good times we had playing board games, having thanksgiving dinners, watching movies, and spending nights in DC.

My parents and sister have encouraged me in all of my pursuits and stood by me through difficult times. I am extremely grateful for their love and support.

Table of Contents

List of Figures vi

1 Introduction 1
- 1.1 Object detection using natural language queries 3
- 1.2 Sequential object detection in indoor scenes 3
- 1.3 Sequential event detection in videos 5

2 Modeling Context Between Objects for Referring Expression Understanding 6
- 2.1 Related Work 9
- 2.2 Modeling context between objects 11
- 2.3 Experiments 18
  - 2.3.1 Datasets 18
  - 2.3.2 Implementation details 19
  - 2.3.3 Comparison of different techniques 20
  - 2.3.4 Ablation experiments 23
- 2.4 Conclusions 28

3 Searching for Objects using Structure in Indoor Scenes 29
- 3.1 Related Work 32
- 3.2 Sequential Exploration 33
  - 3.2.1 Data subset selection 38
- 3.3 Experiments and Results 40
  - 3.3.1 Dataset 40
  - 3.3.2 Sequential Exploration 41
- 3.4 Conclusion 43

4 Feedback Loop between High Level Semantics and Low Level Vision 48
- 4.1 Related Work 51
- 4.2 Incremental Inference with Feedback Loop 53
  - 4.2.1 Clusters under closed world assumption 54
  - 4.2.2 Detection scoring function 57
- 4.3 Experiments 61

4.3.1 One-on-One basketball dataset 61
4.3.2 Hypothesizing candidate events 62
4.3.3 Incrementally adding events with feedback loop 63
4.3.4 Effect of initial threshold 67
4.4 Conclusion 68

5 Feature Selection using PLS regression and Optimal Experiment Design 70
5.1 Related Work 72
5.2 Preliminaries 74
5.2.1 Partial Least Squares 74
5.2.2 Notation 77
5.3 Optimal Loadings Technique 77
5.3.1 Optimal Experiment Design for PLS 77
5.3.2 PLS models with Maximum Relevance and Minimum Redundancy 82
5.3.3 Approximation for the D-Optimal Loadings criterion 84
5.4 Analysis of the relationship between $P_\pi P_\pi^\top$ and $\Pi^\top P P^\top \Pi$ 89
5.5 Experiments and Results 93
5.5.1 Datasets 94
5.5.2 Comparison with other Feature Selection Techniques 94
5.6 Conclusion 97

6 Conclusion 99

A Derivation of Scoring Functions for Feedback Loop Inference 101
A.1 Linear Programming Relaxation of the MAP problem 102
A.2 Block Coordinate Descent in the Dual 105
A.3 Upper Bound Score - Proof of Proposition 1 106
A.4 Blind Score - Proof of Proposition 2 109
A.5 Message Update Equations 111

Bibliography 117

List of Figures

2.1 Context between objects is specified using spatial relationships between regions such as "above", "to the right", "to the left" etc. It is also represented using interactions between objects such as "riding", "holding", "sitting", etc. When there are multiple instances of the same type of object, context helps in referring to the appropriate instance of that object . . . 7

2.2 We identify the referred region along with its supporting context region. We start with a set of region proposals in an image and consider pairs of the form {*region, context_region*}. The entire image is also considered as a potential context region. The probability is evaluated using an LSTM which takes as input region CNN features, context region CNN features, bounding box features and an embedding vector for words in the referring expression. All the LSTMs share the same weights. The probability of a referring expression for an individual region is obtained by finding the maximum over its pairs with context regions. The noisy-or function can be used instead of the max function. After pooling over context regions, the top scoring region (along with its context region) is selected as the referred region . . . 13

2.3 Given a set of region proposals in an image, we construct positive and negative bags containing pairs of regions. In this example, the plant in $Region1$ is the referred object. Hence the positive bag consists of pairs of the form ($Region1$,R_i) where R_i is one of the remaining regions. The negative bag consists of pairs of the form (R_i, R_j) where the first region R_i can be any region except $Region1$ and the second region R_j can be any region including $Region1$ . . . 16

2.4 Google RefExp results. We show results from the model trained with positive and negative bag margin. We compare the grounding between using image context only and pooling the context from all regions using noisy-or. A box with dashed line indicates the context region. We first identify the referred region using noisy-or function. The context region is then selected as the one which produces maximum probability with the referred region. The last row shows images with misplaced context regions . . . 25

2.5 Google RefExp failure cases. We observe errors when there is wrong grounding of attributes or when there is incorrect localization of context region . . . 25
2.6 UNC RefExp results from TestB partition. We show results from the model trained with negative bag margin. We observe that our method can identify the referred region even when the context object is not explicitly mentioned . . . 26
2.7 UNC RefExp failure cases from TestA partition. We show results from the model trained with negative bag margin. This partition contains terse referring expressions. Most of the time, the referring expressions do not uniquely identify the people . . . 26
2.8 Spatial likelihood of referred region given a context region. We fix the context region and evaluate the likelihood of the referred object being present in various locations of the image. When the entire image is used as context, the high likelihood regions do not necessarily overlap with the location of the referred region. However when the context region is fixed, the high likelihood regions overlap the referred region 27

3.1 **Searching for a table**. Each step in the above sequence shows exploration of three additional regions in the image. The search strategy learned using our method utilizes the room structure and the presence of other objects in the image to discover the table region much earlier than using the ranked sequence from an object proposal technique. . 31
3.2 **Average Precision (AP) vs. number of processed regions.** A classifier trained for a query class with unary scene context features alone can achieve a significantly high average precision by processing very few regions. Classes like *bed, nightstand* and *sofa* need only 20-25% of the regions when compared to the proposal ranking sequence. A search strategy trained for a query class using both object-object context and scene-context features further improves the performance for classes like *counter, lamp, pillow* and *sofa*. While the plots show sequential processing of all 100 regions, the stopping criterion for practical situations can be chosen based on the number of regions at which we obtain the maximum AP. . . . 44
3.3 **Scene+Objects Context: Comparison of background selection techniques.** We see that the search strategy trained with a background subset selected using determinant maximization performs better or equally well as the strategy trained with a background subset selected randomly. But the main advantage of the determinant maximization based subset selection is the repeatability of experiments unlike the random subset selection. . . . 45

3.4 **Scene Context: Comparison of background selection techniques.** We see that the performance of the classifier trained with a background subset selected using determinant maximization is comparable to that of the classifier trained with a random background subset. But the main advantage of the determinant maximization based subset selection is the repeatability of experiments unlike the random subset selection. 46
3.5 **Search results for different queries.** We compare three strategies - ranked sequence obtained from the region proposal technique (unaware of query class), ranked sequence obtained from a classifier trained for a query class using scene context features alone and sequence produced by a search strategy trained for a query class using both scene context and object-object context features. Red boxes indicate regions labeled as query class, yellow boxes indicate regions other than the query class and blue boxes indicate regions labeled as background. The images show a state in the search sequence of different methods at a certain number of regions processed. We can see that our strategy which uses both scene context and object-object context can locate an object of the query class earlier than the other methods. 47

4.1 The shared nodes between clusters in a partitioning of a Markov network. The set f contains active detections that are currently in the network and $\boldsymbol{x_f}$ are the nodes that are instantiated over only the detections from f. The set of factors $\theta_f(\boldsymbol{x_f})$ is defined over the nodes $\boldsymbol{x_f}$. Similarly, g is the set of detections to be unclamped at an iteration and h is the set of detections that are still clamped to false. 56
4.2 Visualization of the Feedback Loop 60
4.3 PR curves for the newly hypothesized events with continuous confidence measures. The red star shows the operating point of Morariu et al. [1] in their feed-forward approach. 61
4.4 Feedback based scores achieve better solutions with fewer detections; We apply an initial threshold on the Clear events and incrementally add the remaining events using the feedback based scores. We measure the exact MAP value of the Markov network along with the $f1$ score corresponding to the ground truth. The plots start at the same initial value for all the five scoring methods since the initial network contains the same set of events. Our feedback based scores achieve better solutions with fewer detections than the baselines - observation score and random score. 64

4.5 We apply threshold on both the Rebound and Clear events for initial network and then incrementally add both events at every iteration. We still see that the exact score and the upper bound score reach better solutions with fewer detections than the observation score. However, the blind score falls slightly below the observation score since it depends only on the current network and the context in the current network is weak due to fewer events. 65
4.6 Effect of initial threshold for the Rebound event in video 4; The confidence scores for the Clear events are scaled between 0.5 to 1 and the Rebound events between -0.25 to 0.1. We fix the initial threshold for Clear event at 0.75 and vary the threshold for Rebound from -0.25 to 0. We observe that a higher threshold for Rebound event in the initial network decreases the MAP value that is achieved in the first iteration of adding Rebound and Clear events to the initial network. The blind score continues to perform poorly in later iterations at higher initial threshold due to weak context in the initial network. However, the exact score and the upper bound score are still stable with respect to the initial threshold. 68

5.1 Relationship between the original criterion $\log\det[P_\pi P_\pi^\top]$ and the approximate criterion $\log\det[\Pi^\top P P^\top \Pi]$, that are obtained by applying PLS for varying number of features, k, in a subset π. The approximate and original criterions are positively correlated for the real datasets. Hence, by maximizing the approximate criterion we are not too far away from the maximum of the original criterion. 89
5.2 Classification performance with feature subsets: The D-Optimal Loadings criterion performs better than others on the MNIST and the CMU PIE datasets and performs equally well with the mRMR technique on the ORL and the Arcene datasets. It also shows a consistent performance especially when the number of selected features is small. 95
5.3 Feature points selected by D-Optimal Loadings, Regression Coefficients, Relief-F, Fisher Score and mRMR techniques. The features selected by D-Optimal Loadings are well distributed across the significant regions of the image unlike others that tend to get clustered or lie in noisy regions. 97

Chapter 1: Introduction

Many environments possess a pattern in spatial and temporal arrangement of their constituent objects. For example, objects in a room of a house are placed in a characteristic layout, events in a basketball game follow strict rules of the game. The presence of such structure can have an effect on the perception of an individual object independent of its attributes. Such an influence is called the context effect [2–5].

Context has been found to affect the probability of correctly recognizing an object. Palmer et al. [2] performed user studies to evaluate the effect of context and identified that the probability of correctly recognizing an object is higher with appropriate context and lower with inappropriate context. Context has been useful in tasks like semantic segmentation [6–8], object detection [3, 9–15], human pose estimation [16, 17], action recognition [18, 19] and event recognition [1, 20, 21]. In many of these tasks, especially object detection, context mostly acts as an additional source of information. It is used to enhance the confidence of recognition in poor viewing conditions like occlusion, low resolution etc. [3]. However, there are situations where understanding context is essential. As humans and computers begin to interact using natural language, a vision system must be able to visually ground nouns and verbs along with prepositions and comparative adjectives. Examples of such situations arise in tasks like image captioning [22–24], semantic image retrieval [25, 26], visual question answering [27, 28] and natural language object detection [29–32]. In these tasks, we are interested in relationships of objects and high level understanding of a scene and hence context is not just an additional source of information.

Context is also useful in few other ways as an additional source of information. Biederman et al. [33] evaluated the effect of context in searching for an object and found that the less time is taken for locating a target in appropriate context. Torralba [3] also noted that context can be used in "cutting down on the number of object categories, scales, locations and features that need to be considered". Context can help in the task of searching [34–36] where the goal is to locate an object or an event under a restricted budget of time and/or computational resources (Ex.: low power mobile devices). A typical search algorithm works sequentially and can use context from previously explored regions to prioritize the regions to process next. Another use of context is to exploit it as an automatic supervisory signal for learning visual representations in an unsupervised manner [37, 38].

In this thesis we explore the two main uses of context discussed above. We show how context helps in understanding natural language queries for detecting objects in images. We also show how context can be used to search for objects and events under restricted computational budget.

1.1 Object detection using natural language queries

In chapter 2, we propose a technique [39] that integrates context between objects to understand referring expressions. Referring expressions are natural language queries that identify a particular object in an image. Such expressions usually describe an object using properties of the object and relationships of the object with other objects. Our approach uses an LSTM to learn the probability of a referring expression, with input features from a region and a context region. The context regions are discovered using multiple-instance learning (MIL) since annotations for context objects are generally not available for training. We utilize max-margin based MIL objective functions for training the LSTM. We perform experiments on the Google RefExp and UNC RefExp datasets and show that modeling context between objects provides better performance than modeling only object properties. We also qualitatively show that our technique can ground a referring expression to its referred region along with the supporting context region.

1.2 Sequential object detection in indoor scenes

In chapter 3, we propose a search technique [40] for localizing objects in indoor scene images. In situations where we are interested in identifying the location of an object of a particular class, a passive computer vision system would process all the regions in an image to finally output a small region. The low level processes like feature extraction from regions can be computationally expensive and it is wasteful to run the detectors throughout the video/image given that most of their results will be discarded after thresholding. Instead, if we use the structure in the scene, we can search for objects without processing the entire image. Our proposed search technique sequentially processes image regions such that the regions that are more likely to correspond to the query class object are explored earlier. We frame the problem as a Markov decision process and use an imitation learning algorithm to learn a search strategy. Since structure in the scene is very essential to perform an intelligent search, we work with indoor scene images as they contain both scene context and spatial context between objects in the scene.

One of the issues that arises during the training of a search strategy is the availability of a large number of background regions when compared to the number of foreground regions. This makes the training time slow and hence requires selection of a subset of the background regions. We perform data subset selection using ideas from Optimal Experiment Design (OED). Given a linear regression model, the goal of OED is to select samples such that the variance in the regression coefficients is minimized. A smaller variance in the coefficients indicates that the prediction error on the test set is low and hence the linear regression model trained with such a subset does not overfit the training data. We have also applied the theory of OED to feature selection. In chapter 5, we show that the variance of the Partial Least Squares (PLS) regression can be minimized by employing the OED criterions on the loadings covariance matrix obtained from PLS [41]. We also provide an intuitive viewpoint to the technique by deriving the A-optimality version of the Optimal Loadings criterion using the properties of maximum relevance and minimum redundancy for PLS models.

1.3 Sequential event detection in videos

In chapter 4, we propose a feedback based sequential technique [42] to detect events in one-on-one basketball videos. Typically, high level semantic analysis involves constructing a Markov network over the low level detections to encode relationships between them. In complex higher order networks (e.g. Markov Logic Networks), each low level detection can be part of many relationships and the network size grows rapidly as a function of the number of detections. As the network size increases, there is an exponential increase in the amount of computational resources required for instantiating the network and also perform inference. Hence we propose a sequential technique to keep the network size small. The network is initialized with detections above a high confidence threshold and then based on the high level semantics in the initial network, relevant detections are incrementally selected from the remaining ones that are below the threshold. We perform experiments on one-on-one basketball videos that uses Markov Logic Networks to encode the rules of the game. We show three different ways of selecting detections which are based on three scoring functions that bound the increase in the optimal value of the objective function of network, with varying degrees of accuracy and computational cost.

Chapter 2: Modeling Context Between Objects for Referring Expression Understanding

In image retrieval and human-robot interaction, objects are usually queried by their category, attributes, pose, action and their context in the scene [26]. Natural language queries can encode rich information like relationships that distinguish object instances from each other. In a retrieval task that focuses on a particular object in an image, the query is called a *referring expression* [29, 43]. When there is only one instance of an object type in an image, a referring expression provides additional information such as attributes to improve retrieval/localization performance. More importantly, when multiple instances of an object type are present in an image, a referring expression distinguishes the referred object from other instances, thereby helping to localize the correct instance. The task of localizing a region in an image given a referring expression is called the *comprehension task* [31] and its inverse process is the *generation task*. In this work we focus on the comprehension task.

Referring expressions usually mention relationships of an object with other regions along with the properties of the object [44, 45] (See Figure 2.1). Hence, it is important to model relationships between regions for understanding referring expressions. However, the supervision during training typically consists of annotations

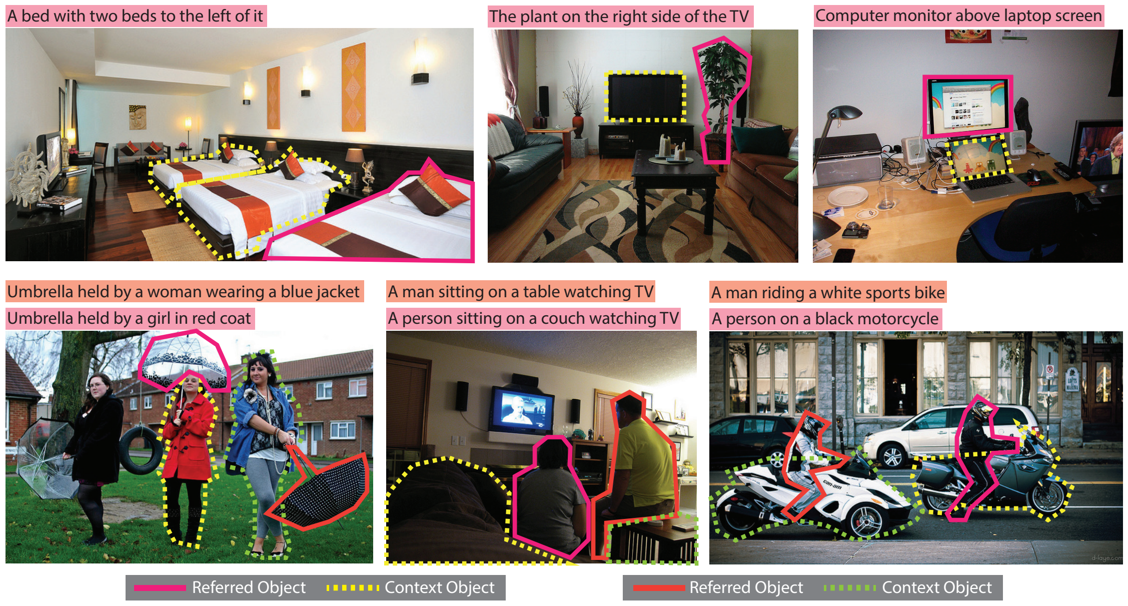

A bed with two beds to the left of it
The plant on the right side of the TV
Computer monitor above laptop screen
Umbrella held by a woman wearing a blue jacket
Umbrella held by a girl in red coat
A man sitting on a table watching TV
A person sitting on a couch watching TV
A man riding a white sports bike
A person on a black motorcycle
Referred Object
Context Object
Referred Object
Context Object

Figure 2.1: Context between objects is specified using spatial relationships between regions such as "above", "to the right", "to the left" etc. It is also represented using interactions between objects such as "riding", "holding", "sitting", etc. When there are multiple instances of the same type of object, context helps in referring to the appropriate instance of that object

of only the referred object. While this might be sufficient for modeling attributes of an object mentioned in a referring expression, it is difficult to model relationships between objects with such limited supervision. Previous work on referring expressions [29, 31, 32] generally ignores modeling relationships between regions. In contrast, we learn to map a referring expression to a region and its supporting context region. Since the bounding box annotations of context objects are not available for training, we learn the relationships in a weakly supervised framework.

We follow the approach of Mao et al. [31] to perform the comprehension task. The probability of a referring expression is measured for different region proposals and the top scoring region is selected as the referred region. The input features in our model are obtained from a {*region, context_region*} pair where the image itself is considered as one of the context regions. The probability of a referring expression for a region can then be pooled over multiple pairs using the max function or the noisy-or function. We use an LSTM [46] for learning probabilities of a referring expression similar to Mao et al. [31]. Since the bounding boxes for context objects are not known during training, we train using a Multiple-Instance Learning (MIL) objective function. The max-margin based LSTM training of Mao et al. [31] is extended to max-margin MIL training for LSTMs. The first formulation is similar to MI-SVM [47] which has only negative bag margin and the second formulation is similar to mi-SVM [47] which has both positive and negative bag margins. Experiments are performed on the Google RefExp dataset [31] and UNC RefExp dataset [48]. Our results show that modeling objects in context for the comprehension task provides better performance than modeling only object properties. We also qualitatively show that our technique can ground the correct context regions for those referring expressions which mention object relationships.

Our contributions are:

- We propose a technique that integrates context between objects to understand referring expressions.
- We demonstrate that training an LSTM by multiple-instance learning is effective when the annotations for context objects are not available.
- We show that modeling context between objects provides better performance than modeling only object properties.

2.1 Related Work

The two tasks of localizing an object given a referring expression and generating a referring expression given an object are closely related. Some image caption generation techniques [49, 50] first learn to ground sentence fragments to image regions and then use the learned association to generate sentences. Since the caption datasets (Flickr30k-original [22], MS-COCO [23]) do not contain the mapping from phrases to object bounding boxes, the visual grounding is learned in a weakly supervised manner. Fang et al. [51] use multiple-instance learning to learn the probability of a region corresponding to different words. However, the associations are learned for individual words and not in context with other words. Karpathy et al. [24] learn a common embedding space for image and sentence with an MIL objective such that a sentence fragment has a high similarity with a single image region. Instead of associating each word to its best region, they use an MRF to encourage neighboring words to associate to common regions.

Attention based models implicitly learn to select or weigh different regions in an image based on the words generated in a caption. Xu et al. [52] propose two types of attention models for caption generation. In their stochastic hard attention model, the attention locations vary for each word and in the deterministic soft attention model, a soft weight is learned for different regions. Neither of these models are well suited for localizing a single region for a referring expression. Rohrbach et al. [53] learn to ground phrases in sentences using a two stage model. In the first stage, an attention model selects an image region and in the second stage, the selected region is trained to predict the original phrase. They evaluate their technique on the Flickr 30k Entities dataset [50] which contains mappings for noun phrases in a sentence to bounding boxes in the corresponding image. The descriptions in this dataset do not always mention a salient object in the image. Many times the descriptions mention groups of objects and the scene at a higher level and hence it becomes challenging to learn object relationships.

Kong et al. [54] learn visual grounding for nouns in descriptions of indoor scenes in a supervised manner. They use an MRF which jointly models scene classification, object detection and grounding to 3D cuboids. Johnson et al. [30] propose an end-to-end neural network that can localize regions in an image and generate descriptions for those regions. Their model is trained with full supervision with region descriptions present in the Visual Genome dataset [55].

Most of the works on referring expressions learn to ground to a single region by modeling object properties and image level context. Rule based approaches to generating referring expressions [56,57] are restricted in the types of properties that can be modeled. Kazemzadeh et al. [29] designed an energy optimization model for generating referring expressions in the form of object attributes. Hu et al. [32] propose an approach with three LSTMs which take in different feature inputs such as region features, image features and word embedding. Mao et al. [31] propose an LSTM based technique that can perform both tasks of referring expression generation and referring expression comprehension. They use a max-margin based training method for the LSTM wherein the probability of a referring expression is high only for the referred region and low for every other region. This type of training significantly improves performance. We extend their max-margin approach to multiple-instance learning based training objectives for the LSTM. Unlike previous work, we model context between objects for comprehending referring expressions.

2.2 Modeling context between objects

Given a referring expression S and an image I, the goal of the comprehension task is to predict the (bounding box of the) region R^* that is being referred to. We adopt the method of Mao et al. [31] and start with a set of region proposals ($\mathcal{C}$) from the image. We learn a model that measures the probability of a region given a referring expression. The maximum scoring region $R^* = \arg\max_{R \in \mathcal{C}} p(R|S, I)$ is then selected as the referred region. Mao et al. [31] rewrite the scoring function as $R^* = \arg\max_{R \in \mathcal{C}} p(S|R, I)$ by applying Bayes' rule and assuming a uniform prior for $p(R|I)$. This implies that comprehension can be accomplished using a model trained to generate sentences for an image region.

Many image and video captioning techniques [49, 58, 59], learn the probability of a sentence given an image or video frame using an LSTM. The input features to the LSTM consist of a word embedding vector and CNN features extracted from the image. The LSTM is trained to maximize the likelihood of observing the words of the caption corresponding to the image or the region. This model is used by Mao et al. [31] as the baseline for referring expression comprehension. Along with the word embedding and region features, they also input CNN features of the entire image and bounding box features to act as context. They further propose a max-margin training method for the LSTM to enforce the probability of a referring expression to be high for the referred region and low for all other regions. For a referring expression S, let $R_n \in \mathcal{C}$ be the true region and $R_i \in \mathcal{C} \setminus R_n$ be a negative region; then the training loss function with a max-margin component is written as

$$J(\theta) = -\sum_{R_i \in \mathcal{C} \setminus R_n} \left\{ \begin{matrix} \log p(S|R_n, I, \theta) \\ \\ -\lambda \max(0, M - \log p(S|R_n, I, \theta) + \log p(S|R_i, I, \theta) \end{matrix} \right\} \quad (2.1)$$

where θ are the parameters of the model, λ is the weight for the margin loss component and M is the margin. The max-margin model has the same architecture as the baseline model but is trained with a different loss function.

In the above model, the probability of a referring expression is influenced by the region and only the image as context. However, many referring expressions mention an object in relation to some other object (e.g., "The person next to the table") and hence it is important to incorporate context information from other regions as well. One of the challenges for learning relationships between regions through referring expressions is that the annotations for the context regions are generally not available for training. However, we can treat combinations of regions in an image as bags and use Multiple Instance Learning (MIL) to learn the probability of referring expressions. MIL has been used by image captioning techniques [24, 25, 51] to associate phrases to image regions when the ground-truth mapping is not available.

We learn to map a referring expression to a region and its supporting context

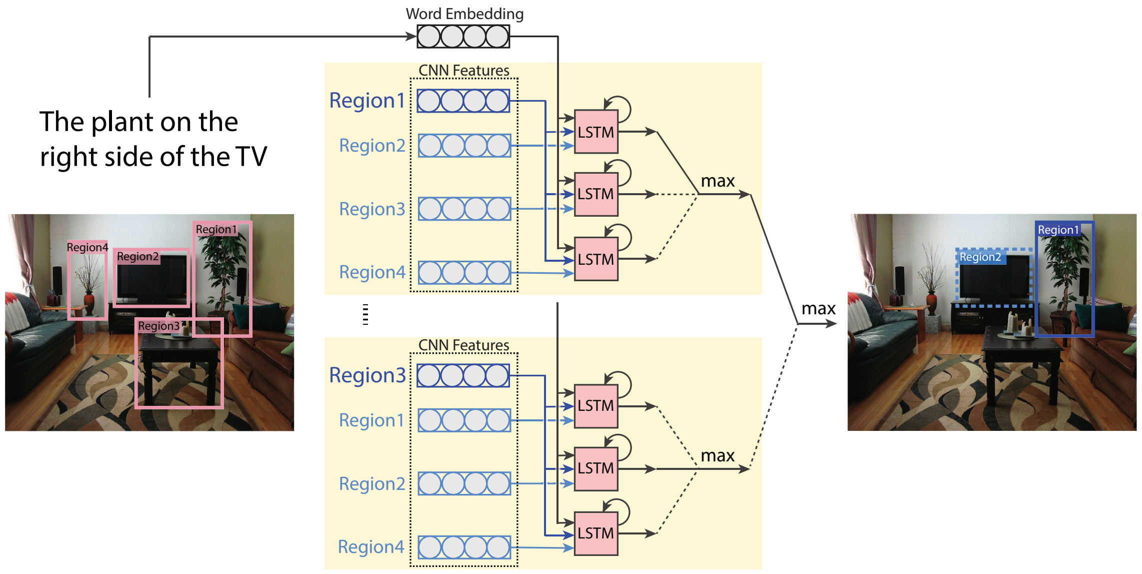


Figure 2.2: We identify the referred region along with its supporting context region. We start with a set of region proposals in an image and consider pairs of the form $\{$*region, context_region*$\}$. The entire image is also considered as a potential context region. The probability is evaluated using an LSTM which takes as input region CNN features, context region CNN features, bounding box features and an embedding vector for words in the referring expression. All the LSTMs share the same weights. The probability of a referring expression for an individual region is obtained by finding the maximum over its pairs with context regions. The noisy-or function can be used instead of the max function. After pooling over context regions, the top scoring region (along with its context region) is selected as the referred region

region. We start with a set of region proposals in an image and consider pairs of the form $\{$*region, context_region*$\}$. The image is included as one of the context regions. The probability of a referring expression is learned for pairs of regions where the input features include visual features and bounding box features for both regions. The probability of an individual region is then obtained by pooling from probabilities of the region's combinations with its potential context regions. After pooling, the top scoring region (along with its context region) is selected as the referred region. Figure 2.2 shows an overview of our system.

Let $\mathcal{C} = \{I, R_1, R_2, \ldots, R_n\}$ be the set of candidate context regions which includes the entire image, I, and other regions generated by the object proposal algorithm. The minimum size of the context region set is one since it always includes I and the model in that case would be equivalent to Mao et al. [31]. We now define the probability of a sentence S given a region R as

$$p(S|R) = \max_{R_i \in \mathcal{C} \setminus R} p(S|R, R_i) \tag{2.2}$$

This implies that the probability of a sentence given a region is defined as the maximum probability obtained by any of the region's combination with a context region. The referred region can now be selected as the top scoring region from the max-pooled probabilities.

$$R^* = \arg\max_{R \in \mathcal{C} \setminus I} \left\{ \max_{R_i \in \mathcal{C} \setminus R} p(S|R, R_i) \right\} \tag{2.3}$$

The noisy-or function can be used instead of the max function in Equation 2.2. Then the referred region is selected as

$$R^* = \arg\max_{R \in \mathcal{C} \setminus I} \left\{ 1 - \prod_{R_i \in \mathcal{C} \setminus R} (1 - p(S|R, R_i)) \right\} \tag{2.4}$$

The noisy-or function can integrate context information from more than one pair of regions and it is more robust to noise than the max function.

We learn the probability function $p(S|R_i, R_j)$ using multiple-instance learning. In our MIL framework, a positive bag for a referring expression consists of pairs of regions of the form (R_t, R_i). The first element in the pair is the region R_t referred to in the expression and the second element is a context region $R_i \in \mathcal{C} \setminus R_t$. A negative bag consists of pairs of regions of the form (R_i, R_j) where $R_i \in \mathcal{C} \setminus R_t$ and $R_j \in \mathcal{C}$. Figure 2.3 shows an example of bags constructed for a sample referring expression.

An LSTM is used to learn the probability of referring expressions and we define multiple-instance learning objective functions for training. Similar to the max-margin training objective defined in Equation 2.1, we apply the max-margin approach of MI-SVM and mi-SVM [47] here to train the LSTM. In MI-SVM, the margin constraint is enforced on all the samples from the negative bag but only on the positive instances from the positive bag. The training loss function with a margin for the negative bag is given by

$$J'(\theta) = -\sum_{\substack{R_i \in \mathcal{C} \setminus R_t,\\ R_j \in \mathcal{C}}} \left\{ \begin{array}{l} \log p(S|R_t, \theta) \\ \\ -\lambda_N \max(0, M - \log p(S|R_t, \theta) + \log p(S|R_i, R_j, \theta) \end{array} \right\} \tag{2.5}$$

The difference between the max-margin Equation 2.1 and Equation 2.5 is that the probability of the referred region is now obtained from Equation 2.2 and the negative samples are not just pairs of regions with the entire image.

The loss function in Equation 2.5 ignores potential negative instances in the positive bag. We can attempt to identify the negative instances and apply a margin to those pairs as well. In mi-SVM, the labels for instances in positive bags are assumed to be latent variables. The goal is to maximize the margin between all positive and negative instances jointly over the latent labels and the discriminant hyperplane. In many referring expressions, there is usually one other object men-

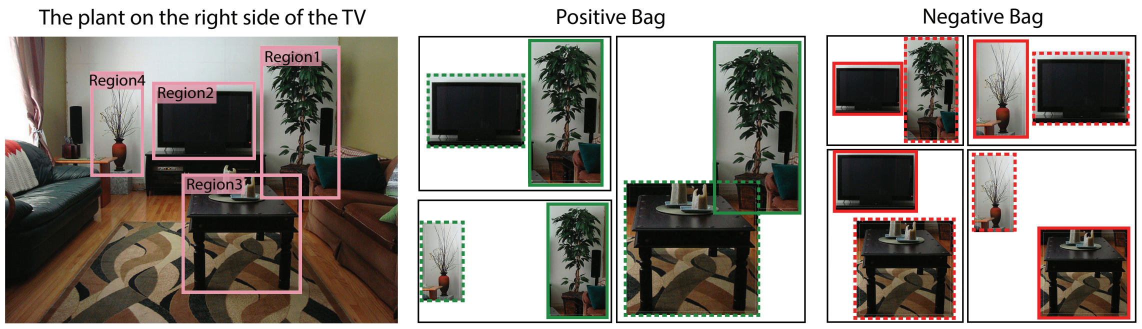


Figure 2.3: Given a set of region proposals in an image, we construct positive and negative bags containing pairs of regions. In this example, the plant in $Region1$ is the referred object. Hence the positive bag consists of pairs of the form $(Region1,R_i)$ where R_i is one of the remaining regions. The negative bag consists of pairs of the form (R_i, R_j) where the first region R_i can be any region except $Region1$ and the second region R_j can be any region including $Region1$

tioned in context. We assume that there is only one positive pair in the positive bag and assign a positive label for the instance with the maximum probability. The remaining pairs in the positive bag are assigned a negative label. Without loss of generality, let (R_t, R_c) be the positive instance from the positive bag. The training loss function with margins for both positive and negative bags is given by,

$$\begin{aligned} J''(\theta) = &- \sum_{\substack{R_i \in \mathcal{C} \setminus R_t, \\ R_j \in \mathcal{C}}} \left\{ \begin{aligned} &\log p(S|R_t, R_c, \theta) \\ &-\lambda_N \max(0, M - \log p(S|R_t, R_c, \theta) + \log p(S|R_i, R_j, \theta) \end{aligned} \right\} \\ &- \sum_{R_k \in \mathcal{C} \setminus R_c} \left\{ \begin{aligned} &\log p(S|R_t, R_c, \theta) \\ &-\lambda_P \max(0, M - \log p(S|R_t, R_c, \theta) + \log p(S|R_t, R_k, \theta) \end{aligned} \right\} \end{aligned} \tag{2.6}$$

In the training algorithm proposed by Andrews et al. [47] for mi-SVM, the latent labels for instances in a positive bag are obtained in an iterative manner. The mi-SVM algorithm iterates over two steps: use the current hyperplane to determine the latent labels, then use the labels to train a new hyperplane. Since neural networks are trained over multiple epochs of the data, the training process is similar to the iterative algorithm used to train mi-SVM. During an epoch, the positive instance (R_t, R_c) in the positive bag is determined as

$$R_c = \underset{R_i \in \mathcal{C} \setminus R_t}{\arg\max}\, p(S|R_t, R_i) \tag{2.7}$$

The parameter θ is updated by applying the loss function in Equation 2.6 with R_c substituted into it. In the following epoch, R_c is updated using the model with updated parameter θ.

The assumption that there is one positive instance in the positive bag holds true when a referring expression uniquely identifies an object and its context object. Such referring expressions are present in the Google RefExp dataset (e.g., “A white truck in front of a yellow truck”). The UNC RefExp dataset contains referring expressions which do not always uniquely refer to an object with its context object (e.g., “Elephant towards the back”). Hence the two different formulations (Equation 2.5 and Equation 2.6) harness different characteristics of referring expressions between the two datasets.

2.3 Experiments

2.3.1 Datasets

We perform experiments on the Google RefExp dataset [31] and the UNC RefExp dataset [48]. Both datasets contain referring expressions for images in the Microsoft COCO dataset [23].

The dataset partition accompanying the current release of Google RefExp dataset was created by randomly selecting 5000 objects for validation and 5000 objects for testing. This type of partitioning results in overlapping images between training, validation and test sets. To avoid any overlap between the partitions, we create our own partition for the training and validation sets. Our training partition contains 23199 images with 67996 objects. Some objects have multiple referring expressions and hence the total number of referring expressions is 85,408. The validation partition contains 2600 images with 7623 objects and 9602 referring expressions. The results of the baseline and max-margin techniques did not differ much between our partition and the Mao et al. [31] partition. However, we perform experiments with our partition since we model context from many regions in an image and that information should not leak into the test stage. We will make our partition publicly available. The test set of this dataset has not been released yet. Hence, we use 4800 referring expressions from the training set for validation.

The UNC RefExp dataset was collected by applying the ReferIt game [29] on MS-COCO images. The training partition contains 16994 images, 42404 objects and 120624 referring expressions. The validation partition contains 1500 images, 3811 objects and 10834 referring expressions. The testing partition contains two splits. TestA partition contains 750 images, 1975 objects and 5657 person-centric referring expressions. TestB partition contains 750 images, 1810 objects and 5095 object-centric referring expressions. While Mao et al. [31] create their own test partition of the UNC RefExp data from a random subset of objects, we work with the partitioning provided by Yu et al. [48].

The evaluation is performed by measuring the Intersection over Union (IoU) ratio between a groundtruth box and the top predicted box for a referring expression. If the IoU >0.5, the prediction is considered a true positive and this is the Precision@1 score. The scores are then averaged over all referring expressions.

2.3.2 Implementation details

Our neural network architecture is the same as Mao et al. [31]. We use an LSTM to learn probabilities of referring expressions. The size of the hidden state vector is 1024. We extract CNN features for a region and its context region using the 16 layer VGGNet [60] pre-trained on the ImageNet dataset. We use the 1000 dimensional features from the last layer (fc8) of VGGNet and fine tune only the last layer while keeping everything else fixed. The CNN features for each region are concatenated with bounding box features of the form $[\frac{x_{min}}{W}, \frac{y_{min}}{H}, \frac{x_{max}}{W}, \frac{y_{max}}{H}, \frac{Area_{bbox}}{Area_{image}}]$ where (W, H) are the width and height of the image. The resulting feature length for both the region and the context region is 2010. We scale the features to lie between

-0.5 and 0.5 before feeding them into the LSTM. The scaling factors are obtained from the training set. We use a vector embedding of size 1024 for the words in a referring expression. The size of the vocabulary is 3489 and 2020 for the Google RefExp and UNC RefExp datasets respectively. The vocabularies are constructed by choosing words that occur at least five times in the training sets.

We implement our system using the Caffe framework [61] with LSTM layer provided by Donahue et al. [58]. We train our network using stochastic gradient descent with a learning rate of 0.01 which is halved every 50,000 iterations. We use a batch size of 16. The word embedding and LSTM layer outputs are regularized using dropout with a ratio of 0.5.

While Mao et al. [31] used proposals from the Multibox [62] technique, we use proposals from the MCG [63] technique. We obtain top 100 proposals for an image using MCG and evaluate scores for the 80 categories in the MS-COCO [23] dataset. We then discard boxes with low values. The category scores are obtained using the 16 layers VGGNet [60] CNN fine-tuned using Fast RCNN [64]. The category scores of proposals are not used during the testing stage by the referring expression model.

2.3.3 Comparison of different techniques

We compare our MIL based techniques with the baseline and max-margin models of Mao et al [31]. The model architecture is the same for all the different variants of training objective functions.

Our implementation of the max-margin technique provided better results than those reported in Mao et al. [31]. We use a margin $M = 0.1$ and margin weight $\lambda = 1$ in the max-margin loss function. The margin is applied on word probabilities in the implementation. For each referring expression and its referred region, we sample 5 "hard MCG negatives" for training, similar to their "hard Multibox negatives". The "hard MCG negatives" are MCG proposals that have the same predicted object category as the referred region. The object category of a proposal is obtained during the proposal filtering process. For our MIL based loss functions, we randomly sample 5 ground-truth proposals as context regions for training. We also sample 5 hard MCG negatives. We use a margin $M = 0.1$ and margin weights $\lambda_N = 1, \lambda_P = 1$ in the MIL based loss functions. During testing, we combine the scores from different context regions using the noisy-or function (Equation 2.4). We sample a maximum of 10 regions for context during the testing stage.

Table 2.1 shows the Precision@1 scores for the different partitions of both datasets. We show results using ground-truth proposals and MCG proposals to observe the behavior of our framework with and without proposal false positives. The results show that our MIL loss functions perform significantly better than the max-margin technique of Mao et al. [31] on the validation partitions of both datasets and the TestB partition of UNC RefExp dataset. The results on the TestA partition show only a small improvement over the max-margin technique and we investigate this further in the ablation experiments.

We observe on the Google RefExp dataset that the MIL loss function with both positive and negative bag margin performs better than the one with negative bag margin only. In this dataset, referring expressions which mention context between

Table 2.1: Precision@1 score of different techniques. The results are obtained using the noisy-or function for pooling context information from multiple pairs. We experiment with both ground-truth (GT) and MCG proposals

Proposals	GT	MCG
Google RefExp - Val		
Max Likelihood [31]	57.5	42.4
Max-Margin [31]	65.7	47.8
Ours, Neg.Bag Margin	**68.4**	49.5
Ours, Pos. & Neg. Bag Mgn.	**68.4**	**50.0**
UNC RefExp - Val		
Max Likelihood [31]	67.5	51.8
Max-Margin [31]	74.4	56.1
Ours, Neg. Bag Margin	**76.9**	57.3
Ours, Pos. & Neg. Bag Mgn.	76.1	**57.4**
UNC RefExp - TestA		
Max Likelihood [31]	65.9	53.2
Max-Margin [31]	74.9	58.4
Ours, Neg. Bag Margin	**75.6**	58.6
Ours, Pos. & Neg. Bag Mgn.	75.0	**58.7**
UNC RefExp -TestB		
Max Likelihood [31]	70.6	50.0
Max-Margin [31]	76.3	55.1
Ours, Neg. Bag Margin	**78.0**	**56.4**
Ours, Pos. & Neg. Bag Mgn.	76.1	56.3

objects usually identify an object and its context object uniquely. Hence there is only one positive instance in the positive bag of region and context region pairs. This property of the referring expressions satisfies the assumption for using the loss function with both positive and negative bag margin.

On the UNC RefExp dataset, we observe that the MIL loss function with negative bag margin performs better or similar to the loss function with both positive and negative bag margin. Unlike the Google RefExp dataset, the referring expressions in the dataset do not always uniquely identify a context object. Many times the context object is not explicitly mentioned in a referring expression e.g., in Figure 2.6b, the elephant in the front is implied to be context but not explicitly mentioned.

Table 2.2: Pooling context in different ways during testing. We compare the performance of pooling context using noisy-or function, max function and also restricting to image as context. The bold values indicate the best performance obtained for the corresponding dataset among all settings

MIL with Negative Bag Margin		
Proposals	GT	MCG
Google RefExp - Val		
Noisy-Or	**68.4**	49.5
Max	66.5	48.6
Image context only	65.9	48.1
UNC RefExp - Val		
Noisy-Or	**76.9**	57.3
Max	75.5	56.5
Image context only	76.4	56.7
UNC RefExp - TestA		
Noisy-Or	75.6	58.6
Max	74.1	57.9
Image context only	**76.2**	58.8
UNC RefExp - TestB		
Noisy-Or	**78.0**	**56.4**
Max	76.8	55.3
Image context only	77.0	55.0

MIL with Pos. & Neg. Bag Margin		
Proposals	GT	MCG
Google RefExp - Val		
Noisy-Or	**68.4**	**50.0**
Max	67.2	49.3
Image context only	67.9	49.3
UNC RefExp - Val		
Noisy-Or	76.1	**57.4**
Max	75.3	56.5
Image context only	76.1	56.6
UNC RefExp - TestA		
Noisy-Or	75.0	58.7
Max	73.4	58.2
Image context only	75.5	**58.9**
UNC RefExp - TestB		
Noisy-Or	77.5	56.3
Max	76.1	55.3
Image context only	76.1	55.0

The assumption of one positive instance in the positive bag does not always hold. Hence, the performance is better using the loss function with negative bag margin only.

2.3.4 Ablation experiments

In Table 2.1, the results for the MIL based methods use the noisy-or function for measuring the probability of a referring expression for a region. The noisy-or function integrates context information from multiple pairs of a regions. We can also use the max function to determine the probability of a referring expression for a region. In this case, the probability for a region is defined as the maximum probability obtained by any of its pairings with other regions. We also experiment with restricting the context region set to include only the image during testing.

The results in Table 2.2 show that noisy-or pooling provides the best performance on all partitions except the UNC RefExp TestA partition. It is also more robust when compared to max pooling, which does not exhibit consistent performance. Our models with just image context perform better than the max-margin model of Mao et al. [31] which also used only image as context. The reason for this improvement is that our MIL based loss functions mine negative samples for context during training. In the max-margin model of Mao et al. [31], the model was trained on negative samples for only the referred region and it was not possible to sample negatives for context.

Figure 2.4 and Figure 2.5 show a few sample results from the Google RefExp dataset. We observe that our model can localize the referred region and its supporting context region. When there is only one instance of an object in an image, the presence of a supporting context region helps in localizing the instance more accurately when compared to using just the image as context. When there are multiple instances of an object type, the supporting context region resolves ambiguity and helps in localizing the correct instance.

The sample results in Figure 2.6 from the TestB partition of the UNC RefExp dataset shows that our method can identify the referred region even when the context object is not explicitly mentioned. Since our method considers pairs of regions, it

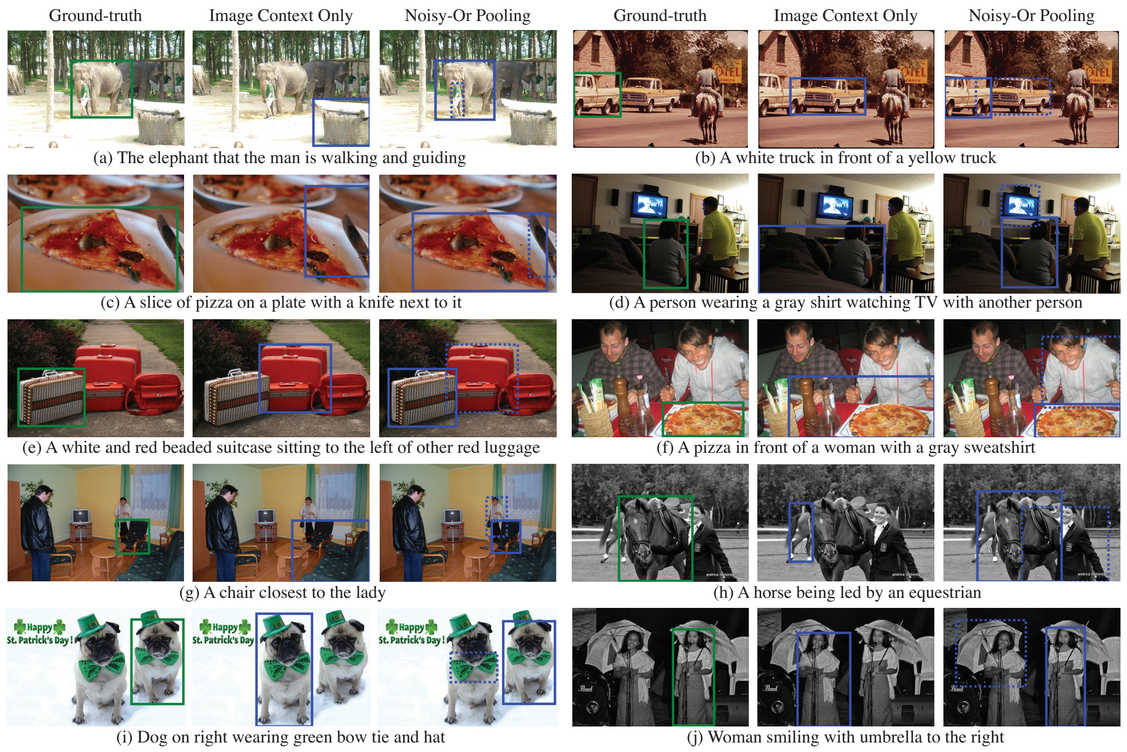

Ground-truth
Image Context Only
Noisy-Or Pooling
Ground-truth
Image Context Only
Noisy-Or Pooling
(a) The elephant that the man is walking and guiding
(b) A white truck in front of a yellow truck
(c) A slice of pizza on a plate with a knife next to it
(d) A person wearing a gray shirt watching TV with another person
(e) A white and red beaded suitcase sitting to the left of other red luggage
(f) A pizza in front of a woman with a gray sweatshirt
(g) A chair closest to the lady
(h) A horse being led by an equestrian
(i) Dog on right wearing green bow tie and hat
(j) Woman smiling with umbrella to the right

Figure 2.4: Google RefExp results. We show results from the model trained with positive and negative bag margin. We compare the grounding between using image context only and pooling the context from all regions using noisy-or. A box with dashed line indicates the context region. We first identify the referred region using noisy-or function. The context region is then selected as the one which produces maximum probability with the referred region. The last row shows images with misplaced context regions

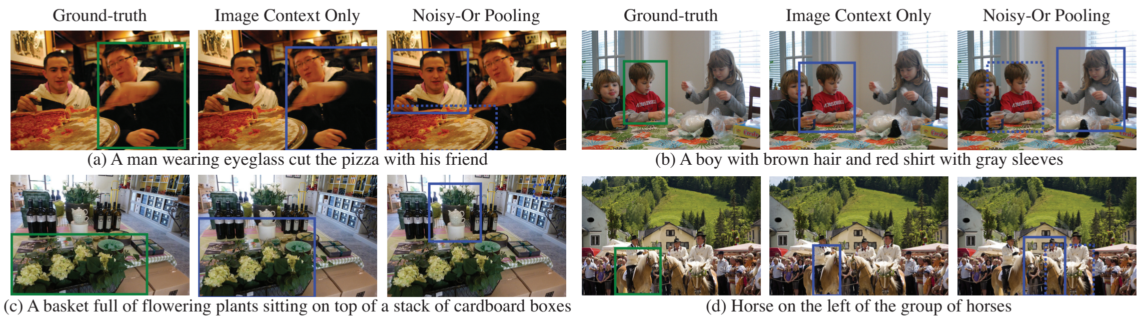

Ground-truth
Image Context Only
Noisy-Or Pooling
Ground-truth
Image Context Only
Noisy-Or Pooling
(a) A man wearing eyeglass cut the pizza with his friend
(b) A boy with brown hair and red shirt with gray sleeves
(c) A basket full of flowering plants sitting on top of a stack of cardboard boxes
(d) Horse on the left of the group of horses

Figure 2.5: Google RefExp failure cases. We observe errors when there is wrong grounding of attributes or when there is incorrect localization of context region

can evaluate the likelihood of a region relative to another region. For example, when there are two instance of the same object on the left, our method can evaluate which

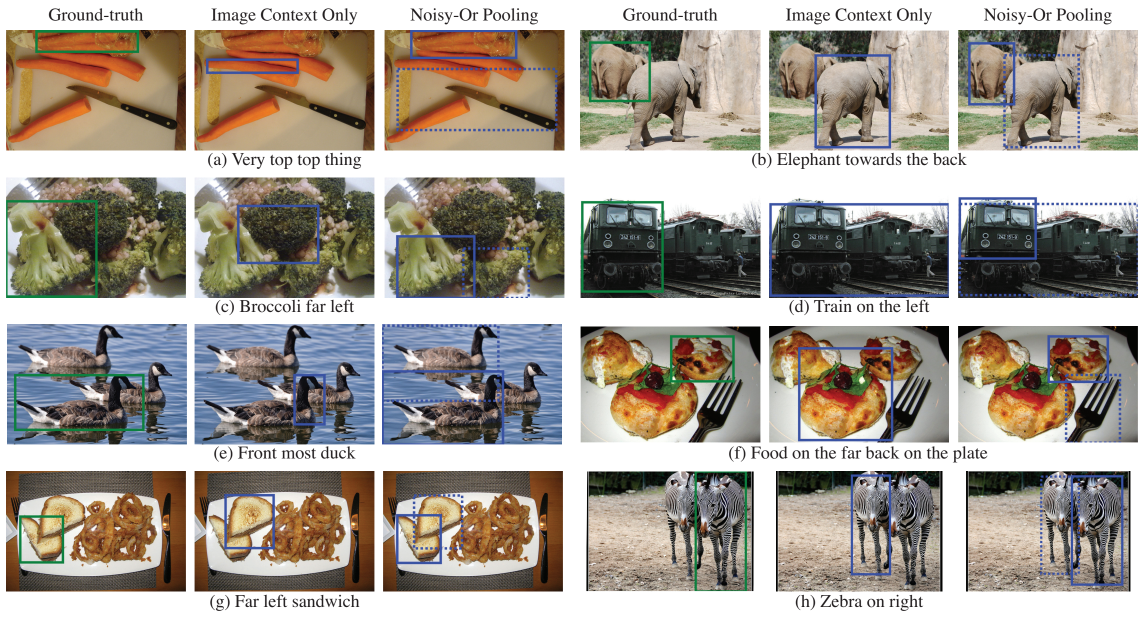


Figure 2.6: UNC RefExp results from TestB partition. We show results from the model trained with negative bag margin. We observe that our method can identify the referred region even when the context object is not explicitly mentioned

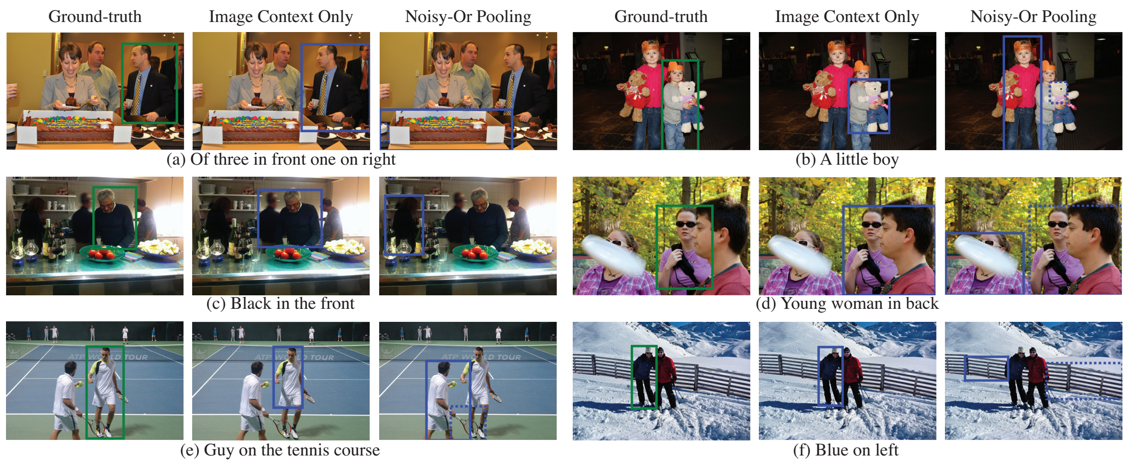
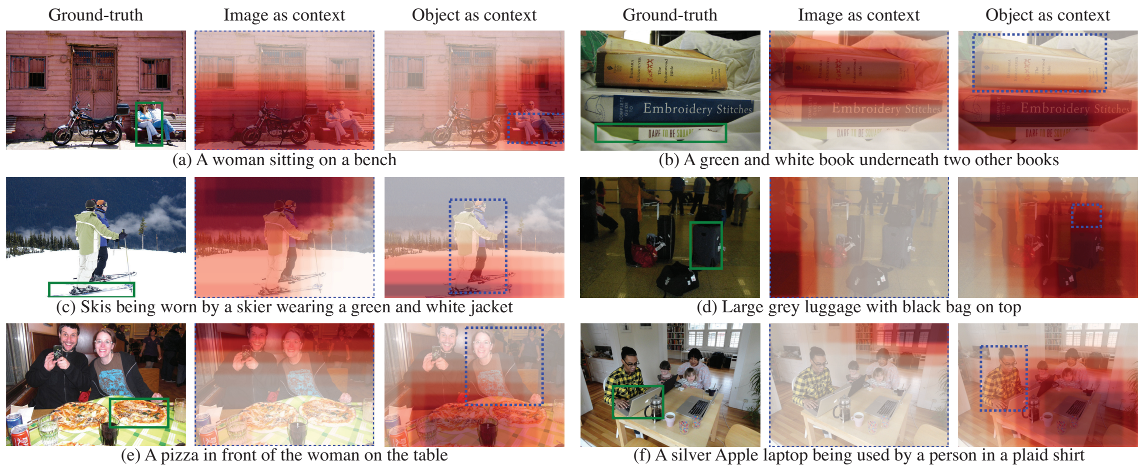


Figure 2.7: UNC RefExp failure cases from TestA partition. We show results from the model trained with negative bag margin. This partition contains terse referring expressions. Most of the time, the referring expressions do not uniquely identify the people

of those two instances is more to the left than the other. On the TestA partition of UNC RefExp dataset, we observe that adding context did not improve performance. Samples from this partition are shown in Figure 2.7. The referring expressions in



(a) A woman sitting on a bench
(b) A green and white book underneath two other books
(c) Skis being worn by a skier wearing a green and white jacket
(d) Large grey luggage with black bag on top
(e) A pizza in front of the woman on the table
(f) A silver Apple laptop being used by a person in a plaid shirt

Figure 2.8: Spatial likelihood of referred region given a context region. We fix the context region and evaluate the likelihood of the referred object being present in various locations of the image. When the entire image is used as context, the high likelihood regions do not necessarily overlap with the location of the referred region. However when the context region is fixed, the high likelihood regions overlap the referred region

this partition deal with people only and are usually terse. They do not always refer to a unique region in the image. We also observe that many referring expressions do not mention that they are referring to a person.

To observe the effect of spatial relationships between objects, we move the referred region to different locations in the image and evaluate the likelihood of the referred region at different locations. Figure 2.8 shows sample heat-maps of the likelihood of a referred object. We first select the entire image as context and observe that the likelihood map is not indicative of the location of the referred object. However, when the relevant context object is selected, the regions of high likelihood overlap with the location of referred object.

2.4 Conclusions

We have proposed a technique that models the probability of a referring expression as a function of a region and a context region. We demonstrate that multiple-instance learning based objective functions can be used for training LSTMs to handle the lack of annotations for context objects. Our two formulations of the training objective functions are conceptually similar to MISVM and mi-SVM [47]. The results on Google RefExp and UNC RefExp dataset show that our technique outperforms the max-margin model of Mao et al. [31]. The qualitative results show that our models can identify a referred region along with its supporting context region.

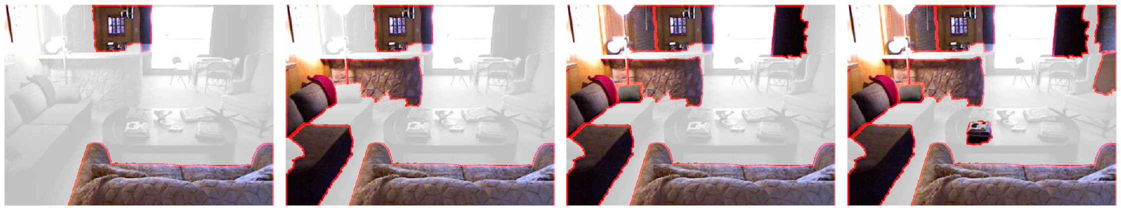
Chapter 3: Searching for Objects using Structure in Indoor Scenes

One of the popular object detection frameworks, RCNN [65], is a pipeline of two main stages: the object proposal stage and the feature extraction/classification stage. Object proposals are image regions that with high probability significantly overlap with an object, irrespective of object class. Features are extracted from object proposals and then a label is predicted. Even with high quality object proposals, the typical number of proposals considered by the feature extraction stage ranges from hundreds to tens of thousands for high resolution imagery.

Consider the situation where a computer vision system needs to identify the presence or location of a particular object in an image. In a passive computer vision system, if we ask a specific question like “Where is the table in this room?”, it would process all the region proposals in the image to detect a table instance. Such a vision system does not exploit the structure in the scene to efficiently process the image. Our goal is to locate objects of interest in an image by processing as few image regions as possible using scene structure. We build on a region proposal module that generates candidate regions and a region classification module that predicts the class label for a region. The generic strategy is to sequentially process image regions such that the regions that are more likely to correspond to the object of interest are explored earlier. At each step, we use the labels of the explored regions and spatial context to predict the likelihood that each unexplored region is an instance of the target class. We then select a few regions with highest likelihood, obtain the class label from the region classification module and add them to the explored set. The process is repeated with the updated set of explored regions.

We frame our sequential exploration problem as a Markov Decision Process (MDP) and use a reinforcement learning technique to learn an optimal search policy. However, it is challenging to manually specify a reward function for the search policy. The true reward function is unknown for our sequential exploration problem since the underlying distribution from which a spatial arrangement of objects in an image is generated is unknown, analogous to a game generated by a hidden emulator [66]. But we have access to an oracle's actions in the individual images. Learning an optimal policy in such situations is known as imitation learning [67] where an oracle predicts the actions it would take at a state and the search policy learns to imitate the oracle and predict similar actions. The oracle in our image exploration problem selects the next set of regions to explore based on the groundtruth labels. We use the DAgger algorithm of Ross *et al.* [68] that trains a classifier as the search policy on a dataset of features extracted at states and actions taken by the oracle (labels), where the states are generated by running the policy iteratively over the training data.

Intelligent search strategies can be learned only in domains that contain sufficient structure in the scenes. Frequently recurring patterns between the constituent objects of a scene are essential to learn powerful strategies and predict exploration


(a) Ranked sequence obtained from an object proposal technique.

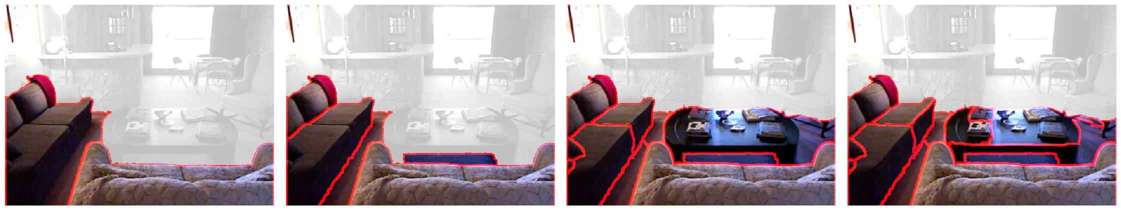

(b) Sequence obtained from a search strategy that uses structure in the scene.

Figure 3.1: **Searching for a table**. Each step in the above sequence shows exploration of three additional regions in the image. The search strategy learned using our method utilizes the room structure and the presence of other objects in the image to discover the table region much earlier than using the ranked sequence from an object proposal technique.

paths with high confidence. Such structures can be found in indoor scenes of houses, stores and buildings. Hence we illustrate our technique on the indoor scene dataset, NYU depth v2 [69]. The other advantage of indoor scenes is the availability of depth data. Gupta *et al.* [70] showed that RCNN [65] trained with depth information greatly improved object detection performance. Apart from improving detection performance, depth information provides spatial context that is highly informative for efficient localization of objects. Our experiments show that given a fixed number of regions that can be processed, our sequential exploration technique provides a better average precision than using a ranked sequence provided by the object proposal technique. Figure 3.1 shows a sequence of regions explored by a strategy trained to detect a table. We compare the search sequence produced by our technique to the ranked sequence provided by the region proposal technique. Our technique is able to utilize the room structure and the presence of other objects in the image to explore the table region much earlier than the object proposal ranking.

3.1 Related Work

Many techniques reduce the number of image windows to limit computation time for object detection. For example, object proposal techniques [71, 72] rank regions in an image based on their likelihood of containing an object. The ranking can be used to prioritize regions for running an object classifier depending on the available computation budget. Such object proposal techniques use only low level image information and do not exploit scene structure.

Some techniques iteratively run the classifier on a few windows and find the next set of windows to be processed based on feedback from the classifier scores and/or spatial context. Lampert *et al.* [73] prune the space of windows using a branch and bound algorithm. Butko and Movellan [74] use a Partially Observable Markov Decision Process to sequentially place a *digital fovea* (a center of fixation) to detect a single target in an image. Neither of these techniques make use of spatial context between objects to improve window selection. Alexe *et al.* [34] use only spatial context to choose a set of windows to be processed. The classifier is run at the end of window set selection and the maximum scoring window is output as the object location. Gonzalez-Garcia *et al.* [35] use both spatial context and the classifier scores of previously explored regions. While their output during the testing stage is a sequence of regions, the training is performed without taking into consideration the states (set of objects explored until a step) encountered in a sequence. Hence, they can only model pairwise constraints between an unexplored region and an object. Our technique models relationships between an unexplored region and all the explored objects. Unlike existing work, we use a framework that allows training and testing using the same procedure, thus reducing the burden of tuning many modules in the system.

The idea of sequentially processing an image by exploiting structure is not just relevant to object localization. Sequential processing has also been explored for video event detection, where running a multitude of detectors at all spatio-temporal scales is very expensive. Amer *et al.* [36] propose an explore-exploit strategy that schedules processes of top-down inference using activity context and bottom-up inference using activity parts. They use a Q-learning algorithm to learn the optimal actions to perform at a state. However, the learning algorithm needs the specification of a reward function which is difficult to obtain in many domains. We use an imitation learning algorithm that alleviates the problem of choosing a reward function.

3.2 Sequential Exploration

The most common formalism for sequential decision making is the Markov Decision Process (MDP). An MDP is characterized by a set of states S, a set of actions A, transition probabilities P and a reward function G (or equivalently a loss function). A policy π is a function that maps states to actions $\pi(s)$. The goal is to find a policy that will maximize a cumulative function of the reward. When the transition probabilities are unknown, reinforcement learning techniques are used to interact with the problem domain and sample the probabilities.

Our problem is to locate objects of a query class (q) by exploring as few image regions as possible. Let R be the set of indices of the regions in the image and t correspond to a step index. Let R_e^t be the set of indices of the explored image regions and $R_u^t = R \setminus R_e^t$ be the set of indices of the unexplored image regions at a step t. To state our problem in the reinforcement learning setting, a state s_t is the set of all the image regions (r) explored until that step.

$$s_t = \{r_i | i \in R_e^t\} \tag{3.1}$$

An action corresponds to selecting the next image region to explore, $a_t = r_j$ where $j \in R_u^t$. The reward function is difficult to specify for our image exploration problem. If we assume that spatial arrangements of objects in images are generated from a hidden distribution similar to games generated by a hidden emulator, a true reward function will allow the policy to learn a predictor that can replicate the behavior of the hidden distribution. For example, if we are searching for a chair, by setting the reward values higher for regions near a table than those far away from it, the policy assigns a greater importance to table proximity feature. Since images contain samples of spatial arrangements from the hidden distribution and the true reward values are unknown, an imitation learning algorithm can be used to learn the optimal policy. In imitation learning [67], rather than specifying a reward function, an oracle demonstrates the action to take and the policy learns to imitate the oracle. For us, the oracle selects the next region to explore based on the groundtruth labels.

Hence, the policy is trained to predict labels similar to the groundtruth.

Imitation learning algorithms usually learn a strategy by training a classifier on the dataset of state features and actions (labels) obtained by sampling sequences produced by an oracle policy. They make an i.i.d assumption about the states encountered during the execution of a learned policy which does not hold for our problem since the policy's prediction affects future states. During the test stage, if the policy encounters a state that was not generated by the oracle policy, it could predict an incorrect action that can lead to compounding of errors. Ross *et al.* [68] propose an imitation learning algorithm called DAgger (Dataset Aggregation) that does not make the i.i.d assumption about the states. DAgger finds a policy $\hat{\pi}$ which minimizes the observed surrogate loss $\ell(s, \pi)$ under its induced distribution of states,

$$\hat{\pi} = \underset{\pi \in \Pi}{\arg\min}\, \mathbb{E}_{s \sim d_\pi}[\ell(s, \pi)] \tag{3.2}$$

where $d_\pi = \frac{1}{T}\sum_{t=1}^{T} d_\pi^t$ is the average distribution of states if we follow policy π for T steps. Since d_π is dependent on the policy π, this is a non-i.i.d supervised learning problem.

In our work, $\ell(s, \pi)$ is the Hamming loss of π with respect to π^* - the oracle policy. At a given state, the policy is penalized for what it predicts for all the regions in the unexplored set. Let

$$\boldsymbol{p} = (p_i)_{i \in R_u^t} \tag{3.3}$$

be a list of the predicted labels where each label $p_i \in \{0, 1\}$ indicates whether the corresponding region is predicted to contain an object of the queried class or not.

The oracle policy produces a list the same length as $\boldsymbol{p}$ with groundtruth labels. The Hamming loss is measured between the list of predicted labels and groundtruth labels. When the policy labels more than one region for exploration, we select the region with the highest belief as the next region to explore.

DAgger trains a single cost sensitive classifier for policy $\hat{\pi}$ that considers features extracted from a state and predicts labels to determine the next action. During training, it starts with an initial classifier and runs through the states, predicting labels for each state. Based on its predictions, it is assigned a loss value at each state. At the end of an iteration, all the features, the predicted labels and the loss values for all states are collected. The aggregate of all the collected datasets until the current iteration is used to train a cost sensitive classifier, which becomes the policy for the next iteration. DAgger is available through a simple interface in the Vowpal Wabbit[1] library. It contains a new programming abstraction proposed by Daumé III *et al.* [75] where a developer writes a single *predict* function that encodes the algorithm for the testing stage and the training is done by making repeated calls to this predict function.

The function SEQ_EXPLORE shown in Algorithm 1 is substituted for the *predict* function in the programming abstraction of Daumé III *et al.* [75]. The input to the algorithm is a list of object proposals and the number of regions that we are allowed to process. We use a modified MCG [70, 71] for region proposal generation and RCNN-depth [70] for region classification. The unary features used for classification are objectness score, proposal rank, mean depth of the region,

[1]https://github.com/JohnLangford/vowpal_wabbit

Algorithm 1 Sequential Exploration

```
1: function SEQ_EXPLORE($obj\_proposals, N$)
2:     $explored\_list \leftarrow \varnothing$
3:     $curr\_regions \leftarrow obj\_proposals[0]$
4:     $unexplored\_regions \leftarrow obj\_proposals[1:$ end$]$
5:     $i \leftarrow 0$
6:     while $i < N$ and $curr\_regions \neq \varnothing$ do
7:         $i \leftarrow i + 1$
8:         $r_{curr} \leftarrow$ POP($curr\_regions$)
9:         PUSH($explored\_list$, $r_{curr}$)
10:        for $r_j \in unexplored\_regions$ do
11:            $score_j =$ CLASSIFY($r_j, explored\_list$)
12:        end for
13:        $next \leftarrow \arg\max_j score_j$
14:        PUSH($curr\_regions, r_{next}$)
15:        REMOVE($unexplored\_regions, r_{next}$)
16:    end while
17:    return $explored\_list$
18: end function
```

```
1: function CLASSIFY($r_j, explored\_list$)
2:     $features \leftarrow \varnothing$
3:     $unary_j \leftarrow$ UNARY_FEATURES($r_j$)
4:     APPEND($features, unary_j$)
5:     NON_MAXIMAL_SUPRESS($explored\_list$)
6:     $pairs \leftarrow \varnothing$
7:     for $r_k \in explored\_list$ do
8:         $label \leftarrow$ QUERY_LABEL($r_j$)
9:         if $label \neq$ bgnd then
10:            $pair_{r_j,r_k} \leftarrow$ PAIR_FEATURES($r_j, r_k$)
11:            APPEND($pairs, pair_{r_j,r_k}$)
12:        end if
13:    end for
14:    $agg\_pair\_features \leftarrow$ AGG_STATS($pairs$)
15:    APPEND($features, agg\_pair\_features$)
16:    $(label, score) \leftarrow$ DAGGER_PREDICT($features$)
17:    if training then
18:        DAGGER_SETLOSS($label$, groundtruth)
19:    end if
20:    return $score$
21: end function
```

mean distance from the back of the room, minimum height from the ground and maximum height from the ground. The pairwise features are 2D area overlap, 2D size ratio, distance between centroids, difference in mean distance from the back of the room, difference in minimum heights from the ground and the difference in maximum heights from the ground. Most of these features were used by Silberman *et al.* [69] for performing support inference. The aggregate feature set is constructed by performing min-pooling for each class and each pairwise feature. For example, one of the aggregate features would be constructed by collecting all the distances between centroids of the current region and the regions of table class, and then taking the minimum of those distances. This feature measures "how far is the closest table (and every other class) from the current region?"

The computational complexity of our algorithm in the worst case scenario of exploring all regions is $O(n^2)$ where we perform classification for every unexplored region at every step after adding one region. However, we do not repeat the classification if a newly explored region is marked as background since it does not change the context features at that step. Hence the number of iterations where we classify the unexplored regions is dependent on the number of foreground regions (k) in the image and the complexity is $O(nk)$. Since there are very few foreground regions in an image, k is usually small.

3.2.1 Data subset selection

Due to the presence of a large number of background regions, the training process can become very slow. Hence we need to select a subset of the background regions such that the training time becomes tractable while maintaining performance. A popular approach to background set collection is hard negative mining, an iterative process where the training data is progressively augmented with the false positive examples produced by the classifier in an iteration. Hard negative mining is a computationally expensive technique which is exacerbated in our case by the already expensive training process for a search strategy. Instead we use a data subset selection technique motivated by the theory of Optimal Experiment Design (OED) [76]. Given a linear regression model, the goal of OED is to select samples such that the variance in the regression coefficients is minimized. A smaller variance in the coefficients indicates that the prediction error on the test set is low and hence the linear regression model trained with such a subset does not overfit the training data. Since DAgger uses a linear classifier to predict actions, we employ an OED criterion to select a subset of the background samples.

Let X be a matrix of n samples with p features. Let Π be a row selection matrix of size $k \times n$. Each row of Π contains a value of one in exactly one column and zeros otherwise. Let Y be the vector of predicted labels. A linear regression model can be written as

$$Y_{k\times 1} = \Pi_{k\times n} X_{n\times p} \beta_{p\times 1} + \epsilon_{k\times 1} \tag{3.4}$$

ϵ is the noise vector with mean zero and variance $\sigma^2 I_k$. In ordinary least squares regression, the prediction error is directly proportional to the variance of the regression coefficients. The variance is given by

$$var(\hat{\beta}) = \sigma^2 (X^T \Pi^T \Pi X)^{-1} \tag{3.5}$$

Optimal Experiment Design suggests many criteria that optimize the eigenvalues of the inverse covariance matrix as a way to minimize the variance in the regression coefficients. The A-optimal criterion minimizes the trace of the inverse covariance matrix and the D-optimal criterion minimizes the determinant of the inverse covariance matrix. The D-optimal criterion [77] is more popular due to the availability of off-the-shelf implementations and also, it simplifies the determinant minimization of an inverse to maximizing the determinant of the covariance matrix. Since we want to select only a subset of the negative samples, we fix the selection variables for the positive samples. We use a row exchange algorithm[2] that iteratively adds and removes rows based on the increments in the determinant. The features we use in the data matrix for subset selection are only the mean centered unary features, since the pairwise features are constructed dynamically and they are difficult to know beforehand.

3.3 Experiments and Results

3.3.1 Dataset

We demonstrate our approach on the NYU depth v2 dataset [69]. We use the RCNN-Depth module of Gupta *et al.* [70] for the region classification. Their region proposal module is a modified Multiscale Combinatorial Grouping (MCG) [71] technique that incorporates depth features. Their feature extraction module is RCNN [65] which includes CNNs fine-tuned on the depth images. The dataset is split into three partitions - 381 images for training, 414 images for validation and 654 images for testing. Since RCNN is trained on the training split, the performance of the detectors on the training set images is extremely good and does not reflect the behavior of the detectors on the test set. Hence we run the detectors on the validation set, obtain groundtruth labels for the detections and this set forms the training set for learning search strategies. The thresholds for the detectors are set based on the best F1 point on the validation set PR curves. We work with 18 categories and do not include the box category as its performance values are very

[2]The row exchange algorithm is available as part of the Statistics Toolbox in MATLAB.

low with an average precision of 1.4%. We consider the top 100 regions obtained from the region proposal module. One of the reasons we use only 100 regions is that as we increase the number of regions, the amount of variation in the background regions increases, making the classification boundaries highly nonlinear given our feature set. Since the number of available positive samples is not sufficiently large, it is difficult to train a nonlinear classifier.

3.3.2 Sequential Exploration

Given a sequence of processed image regions, we measure the performance by the average precision (AP) of object detection performance versus the number of regions processed. Specifically, we measure the average precision at intervals of 10 image regions until we reach 100 image regions. Since our goal is to search for an object of a query class, the sequence of regions produced by our sequential exploration technique is different for different query classes. Figure 3.4 shows average precision as the number of processed regions increases. Each figure compares various sequences produced for a particular query class. The baseline technique we compare with is the rank sequence obtained from the region proposal technique. The sequence is usually rank diversified and not necessarily sorted by objectness scores. Since the region proposal technique is not aware of the query class, it produces only one sequence for an image.

First, we train a classifier with the query class as the label and just the unary scene context features (see Sec. 3.2). Since the scene context features do not change based on the regions explored, we obtain scores from this classifier in a single step. The scores are then used to rank the regions to obtain a sequence. Our results indicate that the scene context features alone can achieve a significantly high average precision by using very few regions - for some classes (Ex: *bed, nightstand, sofa*) almost 20-25% of the regions when compared with the proposal ranking sequence. Next, we perform a sequential search using strategies trained with object-object context features along with the scene context features. The results indicate that for classes like *counter, lamp, pillow* and *sofa*, object-object context improves the average precision over using just the scene context features. While we see improvement in the dresser class as well, the number of test samples are too few to determine the significance of the improvement. Figure 3.5 shows examples of search results for different query classes. The examples show that our strategy which uses both scene context and object-object context can locate objects of the query class earlier than the other methods.

The supplementary material contains plots that compares our technique trained with a randomly selected background subset against our technique trained with the determinant maximization based background subset. The plots show that our determinant maximization based subset selection technique performs better or equally well with the random subset on most of the classes. But the main advantage of our subset selection technique is the repeatability of experiments unlike the one with random subset selection.

Computation time: On a single core of an Intel 4.0GHz processor, it takes only 20ms on average for the search process in an image with 100 region proposals.

The time taken for extracting CNN features is 5ms per region on a GPU. Since our results show that for most of the classes we can achieve a high average precision at around 25 to 50 regions instead of evaluating all 100 regions, the total time taken for feature extraction and search overhead is 0.145s and 0.27s for 25 and 50 regions respectively. This shows that the search overhead is negligible compared to the total time and the reduction in number of regions directly translates to 2 to 4 times speedup in computation time while still achieving a high average precision. The time for context feature extraction is negligible because the necessary information is already extracted by the region proposal module.

3.4 Conclusion

We have proposed a search technique for detecting objects of a particular class in an image by processing as few image regions as possible. The search strategy is framed as a Markov decision process learned using an imitation learning algorithm, which sequentially explores regions based on structure in the scene. Our experiments shows that unary scene context features of regions can alone achieve a significantly high average precision after processing only 20-25% of the regions for classes like *bed, night-stand* and *sofa.* By incorporating object-object context, the performance is further improved for classes like *counter, lamp, pillow* and *sofa.* Our sequential search process adds a negligible overhead when compared to the time spent on extracting CNN features, hence the reduction in number of regions leads directly to a gain in computation speed of the object detection process.

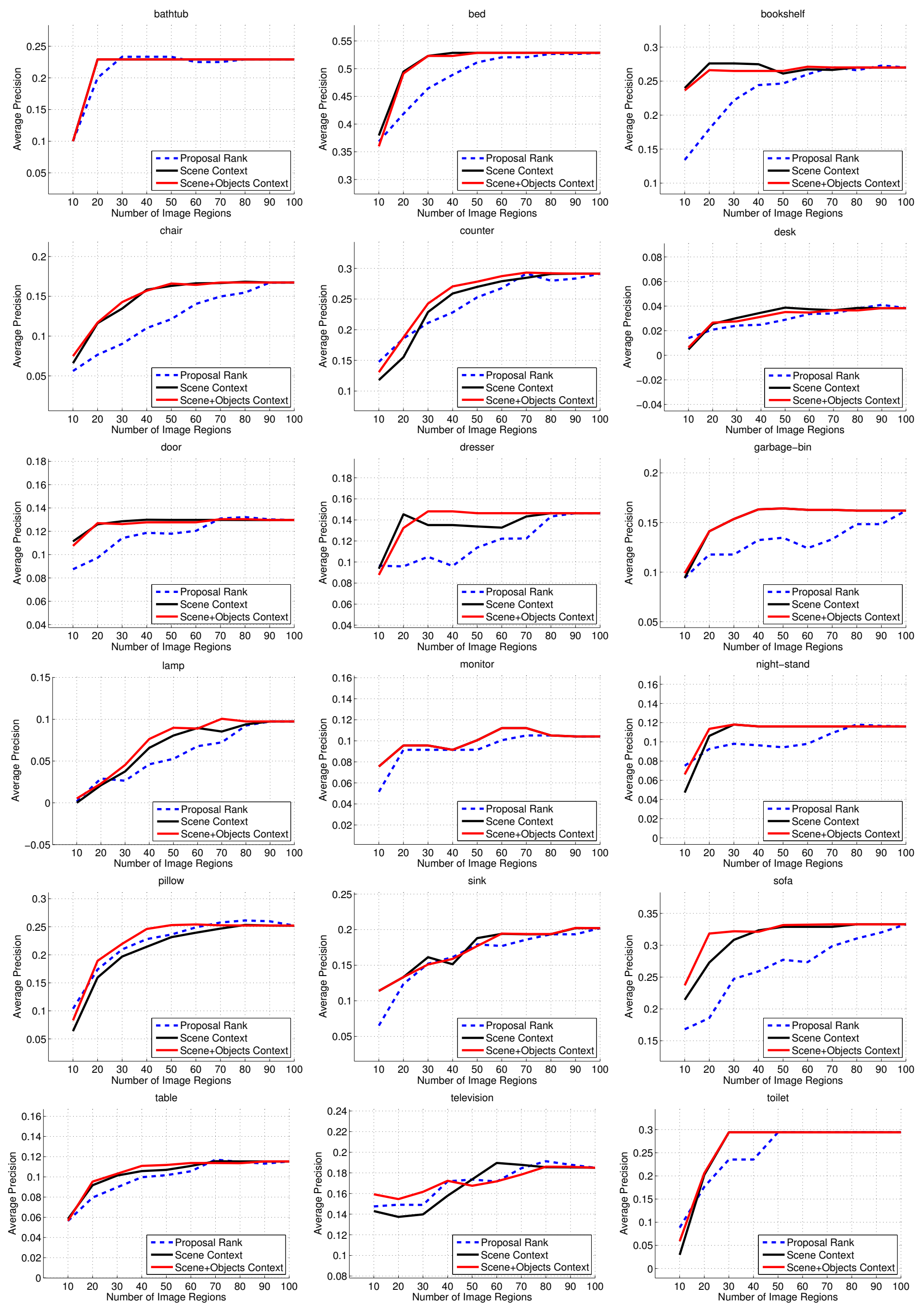


Figure 3.2: **Average Precision (AP) vs. number of processed regions.** A classifier trained for a query class with unary scene context features alone can achieve a significantly high average precision by processing very few regions. Classes like *bed, nightstand* and *sofa* need only 20-25% of the regions when compared to the proposal ranking sequence. A search strategy trained for a query class using both object-object context and scene-context features further improves the performance for classes like *counter, lamp, pillow* and *sofa*. While the plots show sequential processing of all 100 regions, the stopping criterion for practical situations can be chosen based on the number of regions at which we obtain the maximum AP.

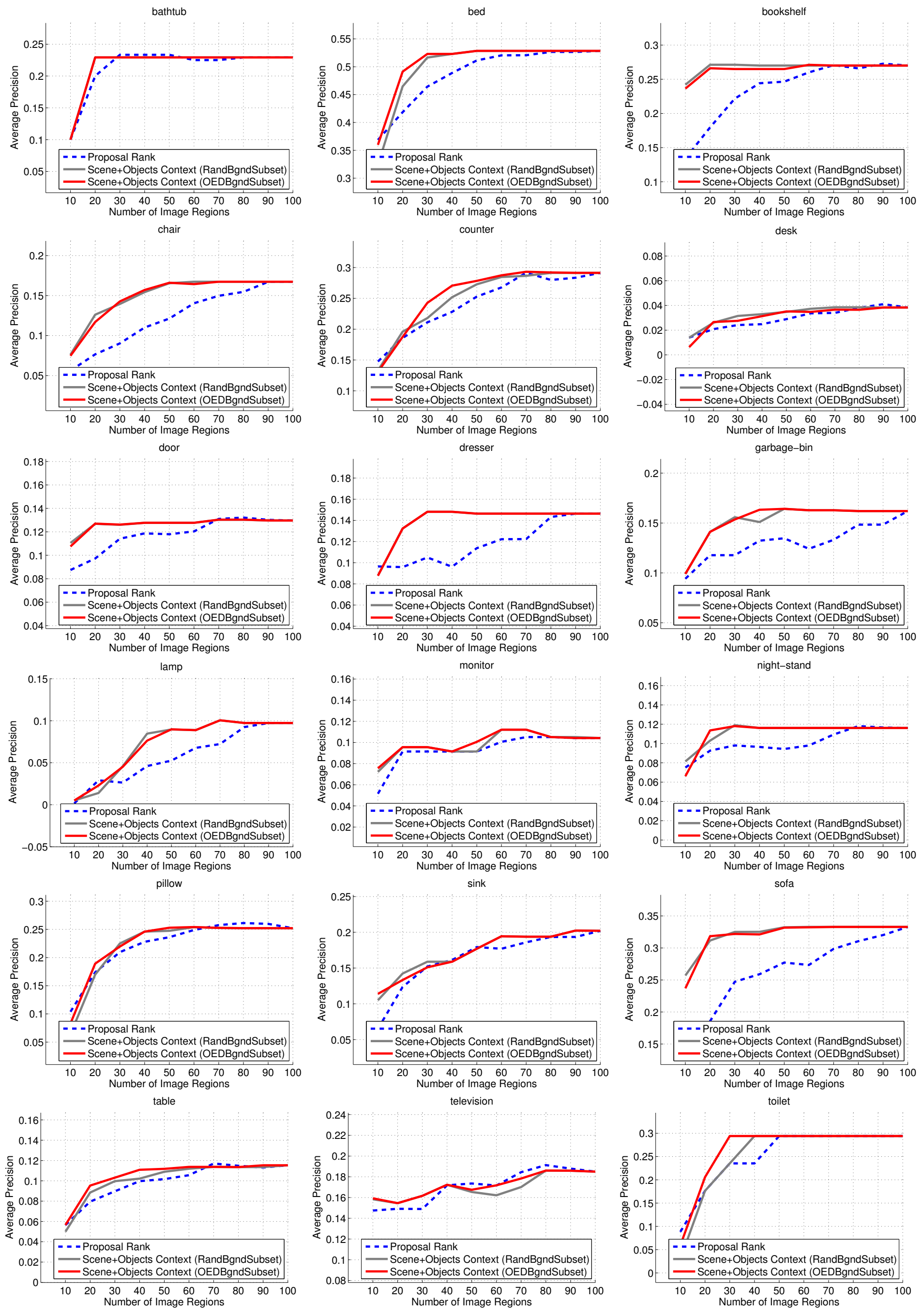


Figure 3.3: **Scene+Objects Context: Comparison of background selection techniques.** We see that the search strategy trained with a background subset selected using determinant maximization performs better or equally well as the strategy trained with a background subset selected randomly. But the main advantage of the determinant maximization based subset selection is the repeatability of experiments unlike the random subset selection.

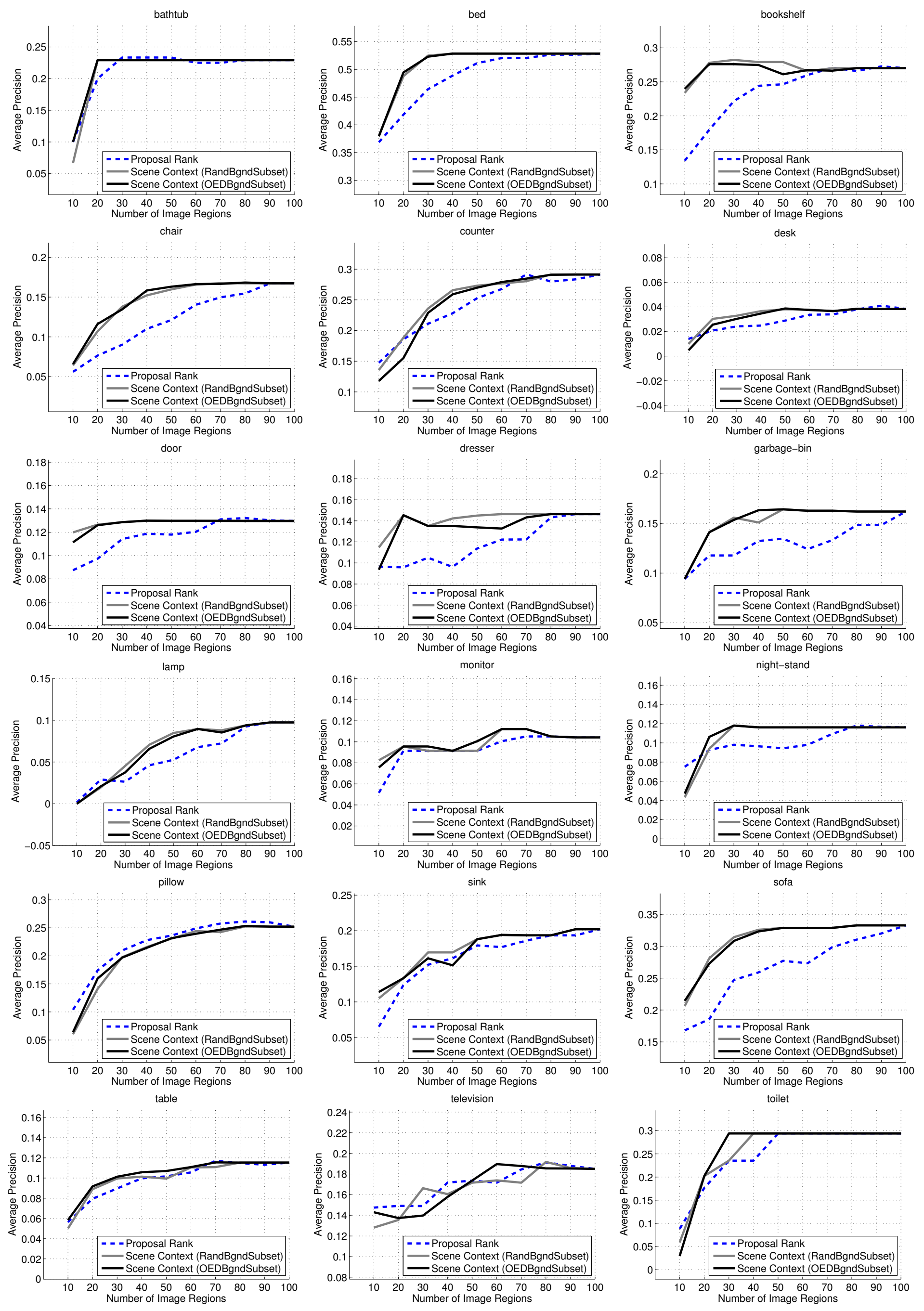
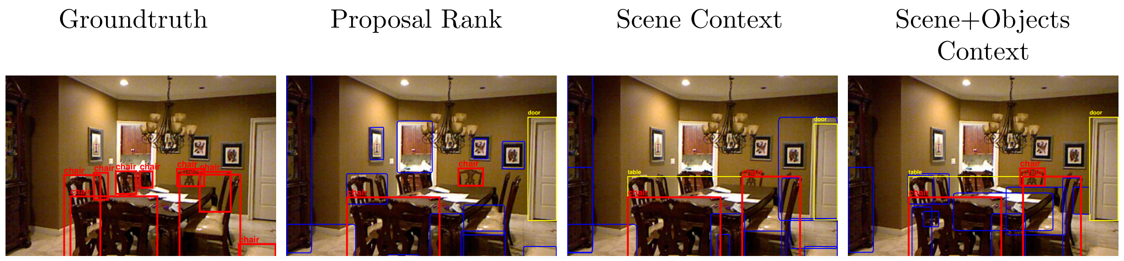
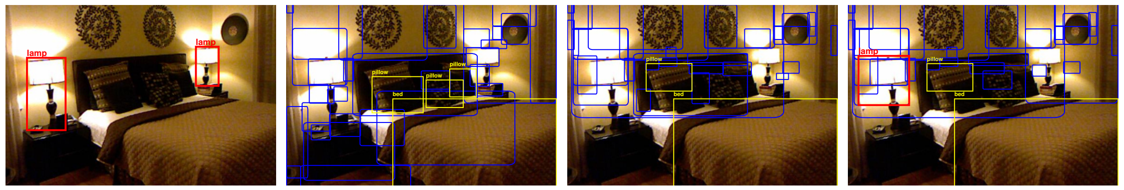


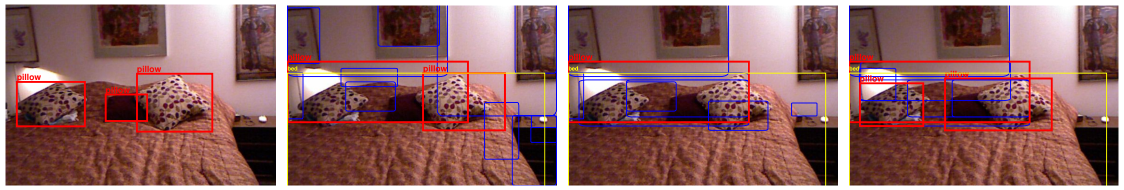
Figure 3.4: **Scene Context: Comparison of background selection techniques.** We see that the performance of the classifier trained with a background subset selected using determinant maximization is comparable to that of the classifier trained with a random background subset. But the main advantage of the determinant maximization based subset selection is the repeatability of experiments unlike the random subset selection.



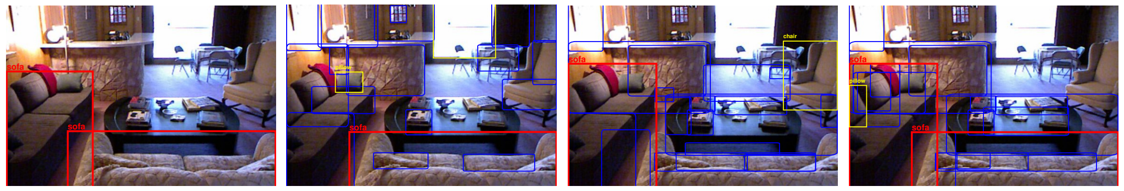
(a) Searching for chair. Number of regions processed = 15



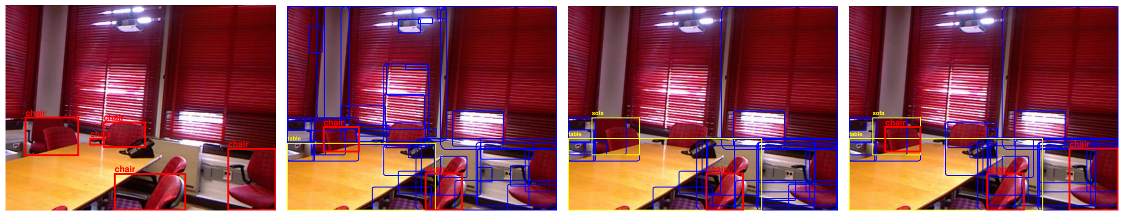
(b) Searching for lamp. Number of regions processed = 35



(c) Searching for pillow. Number of regions processed = 15



(d) Searching for sofa. Number of regions processed = 20



(e) Searching for chair. Number of regions processed = 45

Figure 3.5: **Search results for different queries.** We compare three strategies - ranked sequence obtained from the region proposal technique (unaware of query class), ranked sequence obtained from a classifier trained for a query class using scene context features alone and sequence produced by a search strategy trained for a query class using both scene context and object-object context features. Red boxes indicate regions labeled as query class, yellow boxes indicate regions other than the query class and blue boxes indicate regions labeled as background. The images show a state in the search sequence of different methods at a certain number of regions processed. We can see that our strategy which uses both scene context and object-object context can locate an object of the query class earlier than the other methods.

Chapter 4: Feedback Loop between High Level Semantics and Low Level Vision

Computer vision systems are generally designed as feed-forward systems where low level detectors are cascaded with high level semantic analysis. Low level detectors for objects, tracks or short activities usually produce a confidence measure along with the detections. The confidence measures can sometimes be noisy and hence a multitude of false detections are fed in to subsequent analysis stages. To avoid these false detections, it is common practice to discard some detections that are below a particular confidence threshold. Unfortunately, it is difficult to reliably select a threshold a priori given a particular task. The threshold is generally selected to achieve a "reasonable" trade-off between detector precision and recall, since it is generally not possible to find all true detections (high recall) without also hallucinating false alarms (low precision).

High level analysis integrates multiple low level detections together using semantics to discard false detections rather than simply thresholding detector scores. This typically involves constructing a Markov network over the detections, where contextual relationships corresponding to high level knowledge about the image or video are encoded as factors over combinations of detections [1, 12, 14, 20, 21]. A detection usually corresponds to one or more nodes in the network and relationships between detections correspond to a factor. In Markov networks of high order, each detection can be part of exponentially many instantiations of a factor and the network size grows rapidly as a function of the number of detections. The problem is further exacerbated by the inference process, whose computational cost is related exponentially to the network complexity. When many detections are hypothesized at low precision, the size of the Markov network becomes unnecessarily high since the inference process sets most of the detections to false.

We tackle the problem of keeping the network size small by incrementally adding only those detections that are most likely to be inferred as true while the rest of them are kept false. We achieve this by adding a feedback loop between the high level and low level stages, where the high level semantics guides the selection of relevant low level detections. There are several advantages to this feedback loop. First, it can locally adjust the thresholds for low level detectors based on the neighboring context. Second, it keeps the network size small and the inference procedure tractable. And third, we can potentially save computation by selectively running the low level procedures like feature extraction and classification only when needed.

The goal of our feedback based incremental technique is to perform inference and obtain the optimal solution of the objective function corresponding to the *full network* (the network obtained when we include all the detections) by unclamping only the relevant detections. We start with detections above a high confidence threshold and clamp the remaining detections to false based on the closed world assumption, the assumption that what is not known to be true is false. We then incrementally select from the remaining detections below the threshold to add to the network. Our proposed feedback loop involves a principled mechanism by which we identify the detections that are most likely to improve the objective function. Motivated by cluster pursuit algorithms [78] for inference, we derive three scoring functions that bound the increase in the objective function with varying degrees of accuracy and computational cost. The first score function yields the exact increase in the objective function, but it requires that the detector has been run everywhere and that inference can be performed exactly; the second bounds the change in the objective function, relaxing the inference requirements; the third provides an even looser bound, but it is least computationally intensive and does not require the low level detector to have processed the candidate detections (which is why we call it the *Blind Score*).

We perform experiments on an event recognition task using one-on-one basketball videos. Morariu and Davis [1] used Markov Logic Networks (MLNs) on this dataset to detect events like Shot Made, Shot Missed, Rebound etc. The inputs are a set of event intervals hypothesized from low level detectors like the tracks of objects. Using the feedback loop technique we show that we can successfully select the most relevant event intervals that were earlier discarded due to thresholding. The experiments show that our score functions can reach the optimal value in fewer iterations with smaller network sizes when compared with using just the low level confidence measures.

4.1 Related Work

While many inference techniques work in an incremental fashion to tackle the complexity issues, they do not necessarily behave as a feedback loop and hence do not present with the advantages mentioned earlier. We mention few works here that iteratively add detections while performing inference. In a scene segmentation task, Kumar and Koller [79] hypothesize a set of regions in an image through multiple bottom-up over-segmentations and exploit the high level energy function to iteratively select input regions that are relevant for the task. Zhu et al. [80] use the greedy forward search technique of Desai et al. [81] for inference in their event recognition system. The inference algorithm of Desai et al. first sets the output label for the inputs to the background class. Each input is then scored based on the change in the objective function if it were allowed to be labelled as a non-background class. The top scoring inputs are then iteratively added until convergence. Our feedback loop technique is based on the same idea of greedily reaching the MAP value as quickly as possible but we provide a principled mechanism to performing inference in higher order networks. Also we do not use it just as an incremental technique, but extract more insight from the high level semantics to save computation for the low level module. An interesting characteristic of our feedback technique is that we can potentially run low level processes only when required during the inference.

Apart from the advantages of keeping the inference tractable, a feedback loop can also be useful in other ways. Sun et al. [13] apply a feedback loop for object detection with geometrical context. They jointly infer about the location of an object, the 3D layout of the scene and the geometrical relationships between the object and the 3D layout. The speciality of their feedback loop is that the object detector module adaptively improves its accuracy in the confidence measures of detections based on the feedback from the scene layout.

The idea of incrementally building a network can be approached in principled ways, including Cutting Plane Inference (CPI) and Cluster Pursuit Algorithms. Many inference problems can be cast as an Integer Linear Program (ILP) which is well suited for CPI. CPI employs an iterative process where the ILP is kept small by adding only the most violated constraints. However, CPI cannot be used for our feedback loop technique where we need to selectively set some detections to false. Sontag et al. [78] propose a cluster pursuit algorithm, an alternative formulation that incrementally adds cliques of variables (called *clusters*) and optimizes the dual function, an objective function obtained through Lagrangian relaxation that is an upper bound on the original (or *primal*) objective function. Their score function for clusters is an approximation to the decrease in the dual value of the objective function after adding a cluster, which is derived from the message passing updates of Globerson and Jaakkola [82]. We use this idea of cluster pursuit algorithm and derive a feedback technique for higher order Markov networks. Our scoring functions use the dual value to calculate approximations for the increase in the primal MAP value after adding a particular cluster.

4.2 Incremental Inference with Feedback Loop

We consider Markov networks defined over binary nodes $\boldsymbol{x} = \{x_1, \dots, x_n\}$ with factors $\theta_c(\boldsymbol{x}_c)$ defined over cliques of nodes $\boldsymbol{x}_c$ such that $c_1, \dots, c_k \subset \{1, \dots, n\}$. The Maximum A Posteriori (MAP) problem is defined as finding an assignment $\boldsymbol{x}^*$ that maximizes the function

$$\Phi(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{c \in \mathcal{C}} \theta_c(\boldsymbol{x}_c) \tag{4.1}$$

The nodes x_i are instantiated over candidate detections that are hypothesized by low level detectors. For example, they can be object detections obtained from running single-object detectors. The detector confidence scores output along with the detections are used as unary factors for the nodes. The factors θ_c that involve more than one detection represent the relationships between the detections. For example, they can be spatial relationships like the placement of an object on top of other objects. We obtain a MAP solution by performing inference, that will ultimately label the hypothesized detections as true positives or false positives.

In Markov networks of high order, every newly added detection can become combinatorially linked to other detections through the higher order factors. When many detections are hypothesized at low precision, the size of the Markov network becomes exponentially large and the inference process becomes computationally expensive even though many of the detections are going to be inferred as false.

The goal of our incremental approach for inference is to maximize the function in (4.1) while keeping the network size small. We achieve this by unclamping only those detections that are most likely to be labeled as true by the inference. The rest of the detections are clamped to false, and while they always participate in the objective function over the iterations, they are excluded from the network during inference. We first perform inference with an initial network constructed from high confidence detections while the rest are clamped to false. We then calculate scores for the remaining detections based on the initial network. The scores measure the change in the MAP value after adding a detection to the current network. These scores are equivalent to locally adding an offset to the low level detector confidences, based on the feedback, so that the detections appear above the threshold. Another way to interpret this is that the thresholds get locally modified to select the detections that are below the threshold. We then unclamp a selected number of top detections and the process is repeated. When the incremental procedure is stopped, the MAP solution to the current network provides the true/false labels to the active detections and the remaining set of detections are labeled as false.

4.2.1 Clusters under closed world assumption

We show that incrementally unclamping detections is equivalent to adding clusters of factors. First we partition the Markov network into three clusters as shown in Figure (4.1). Let f be the set of active detections that are currently in to the network and $\boldsymbol{x_f}$ be the nodes that are instantiated over only the detections from f. The factor θ_f is defined over just the nodes $\boldsymbol{x_f}$. Let g be the set of one or more detections that is to be unclamped in a given iteration and $\boldsymbol{x_g}$ be the nodes instantiated over at least one detection from g and any other detections from f. The factor θ_g is defined over nodes $\boldsymbol{x_g}$ and other nodes from $\boldsymbol{x_f}$ that it shares with θ_f. Let h be the remaining set of detections and $\boldsymbol{x_h}$ be the nodes that are grounded over at least one detection from h and any other detections from $f \cup g$. The factor θ_h is defined over $\boldsymbol{x_h}$ and the other shared nodes with θ_f and θ_g. The overall objective function expressed as a sum of these clusters is

$$\begin{aligned}\Phi(\boldsymbol{x}) = \theta_f(x_{f1}, x_{f2}, x_{f3}, x_{f4}) + \theta_g(x_{g1}, x_{g2}, x_{f2}, x_{f3}) \\ + \theta_h(x_{h1}, x_{g2}, x_{f3}, x_{f4})\end{aligned} \tag{4.2}$$

Under the closed world assumption, any detection that is not included in the Markov network due to thresholding is assumed to be false. To satisfy this condition during the incremental process, we need to repartition the objective function (4.2). During every iteration of the process, we have a Markov network that includes a set f of active detections. The remaining detections from g and h are not yet added and hence the nodes instantiated over these detections must be clamped to false. The associated factors are projected on to the current network after setting the nodes of the excluded detections to false. The resulting objective function is

$$\begin{aligned}\Phi_{\mathrm{cur}}(\boldsymbol{x}_{\mathrm{cur}}) = \theta_f(x_{f1}, x_{f2}, x_{f3}, x_{f4}) + \theta_g(x_{g1} = 0, x_{g2} = 0, x_{f2}, x_{f3}) \\ + \theta_h(x_{h1} = 0, x_{g2} = 0, x_{f3}, x_{f4})\end{aligned} \tag{4.3}$$

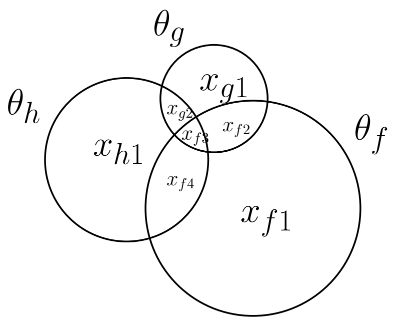


Figure 4.1: The shared nodes between clusters in a partitioning of a Markov network. The set f contains active detections that are currently in the network and $\boldsymbol{x_f}$ are the nodes that are instantiated over only the detections from f. The set of factors $\theta_f(\boldsymbol{x_f})$ is defined over the nodes $\boldsymbol{x_f}$. Similarly, g is the set of detections to be unclamped at an iteration and h is the set of detections that are still clamped to false.

To calculate a score for the set of detections in g, we need the objective function to include these detections in the active set while all other remaining detections from h are still clamped to false. This gives rise to the objective function

$$\begin{aligned}\Phi'(\boldsymbol{x}) = \theta_f(x_{f1}, x_{f2}, x_{f3}, x_{f4}) + \theta_g(x_{g1}, x_{g2}, x_{f2}, x_{f3}) \\ + \theta_h(x_{h1} = 0, x_{g2}, x_{f3}, x_{f4})\end{aligned} \tag{4.4}$$

Hence, the cluster of factors that need to be added to the current network during an iteration is given by

$$\Phi_{\text{new}}(\boldsymbol{x}_{\text{new}}) = \Phi' - \Phi_{\text{cur}}(x_{\text{cur}}) \tag{4.5}$$

$$\begin{aligned}= \theta_g(x_{g1}, x_{g2}, x_{f2}, x_{f3}) - \theta_g(x_{g1} = 0, x_{g2} = 0, x_{f2}, x_{f3}) \\ - \theta_h(x_{h1} = 0, x_{g2} = 0, x_{f3}, x_{f4}) + \theta_h(x_{h1} = 0, x_{g2}, x_{f3}, x_{f4})\end{aligned} \tag{4.6}$$

We now propose three score functions that measure the change in the MAP value after adding the cluster $\Phi_{\text{new}}(x_{\text{new}})$ to $\Phi_{\text{cur}}(x_{\text{cur}})$, with varying degrees of accuracy and computational cost.

4.2.2 Detection scoring function

We define the score for a detection based on the change in the MAP value after adding the detection to the current network. If we are adding the detection in g, the score is given by

$$score(g)_{exact} = \Delta\Phi = \max\left[\Phi_{\text{cur}}(x_{\text{cur}}) + \Phi_{\text{new}}(x_{\text{new}})\right] - \max\left[\Phi_{\text{cur}}(x_{\text{cur}})\right] \qquad (4.7)$$

We also propose an upper bound to the exact score - $score(g)_{upper}$, that is derived based on the ideas of cluster pursuit algorithm of Sontag et al. [78]. We first obtain a dual of the MAP problem through Lagrangian relaxation. The MAP problem is now equivalent to minimizing the dual objective function since the dual value is an upper bound on the primal MAP value. We then use the message passing algorithm of Globerson et al. [82] to obtain the message update equations for the dual variables. Similar to Sontag et al. [78], we obtain an approximation to the new dual value after adding a cluster to the current network, by performing one iteration of message passing. Since the dual value is an upper bound on the primal MAP value, the new decreased dual value gives an upper bound for the exact score.

Proposition 1 (Upper Bound Score)**.** *An upper bound on the change in the MAP* *value (4.7) after adding a cluster is given by*

$$\Delta\Phi \leq score(g)_{upper} \tag{4.8}$$

$$= \frac{1}{|s|}\sum_{i\in s}\max_{x_i}\left(\max_{x_{cur\setminus i}}\Phi_{cur}(x_{cur}) + \max_{x_{new\setminus i}}\Phi_{new}(x_{new})\right) - \max\Phi_{cur}(x_{cur}) \tag{4.9}$$

where s is the set of nodes in the intersection of the sets x_{cur} and x_{new}.

The proof can be found in Appendix A. The first term in the upper bound score is equivalent to averaging the MAP values obtained by enforcing same assignment for one shared node at a time. The upper bound score can be calculated efficiently using an inference algorithm that calculates max-marginals with only a little computation overhead (eg. dynamic graph cuts [83]) and hence can avoid performing repeated inference to calculate the exact score.

We derive another approximation to the score function called the *Blind Score* since it is dependent only on the max-marginals of the current network and does not involve the max-marginals of the new cluster to be added. It is obtained as a lower bound to the upper bound score (not the exact score).

Proposition 2 (Blind Score)**.** *A lower bound to the upper bound score (A.15) is given by*

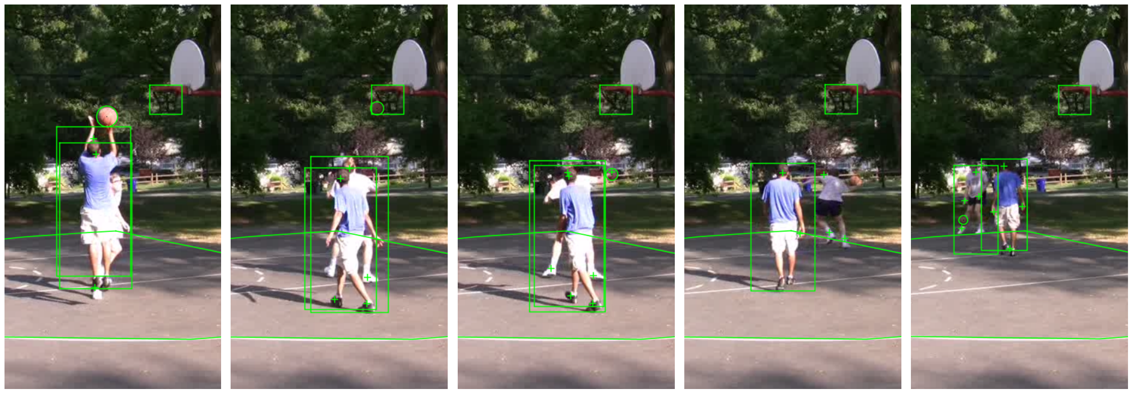
$$score(g)_{upper} \geq score(g)_{blind} \tag{4.10}$$

$$= \frac{-1}{|s|}\sum_{i\in s}\left|\max_{x_i=0,x_{cur\setminus i}}\Phi_{cur}(x_{cur}) - \max_{x_i=1,x_{cur\setminus i}}\Phi_{cur}(x_{cur})\right| \tag{4.11}$$

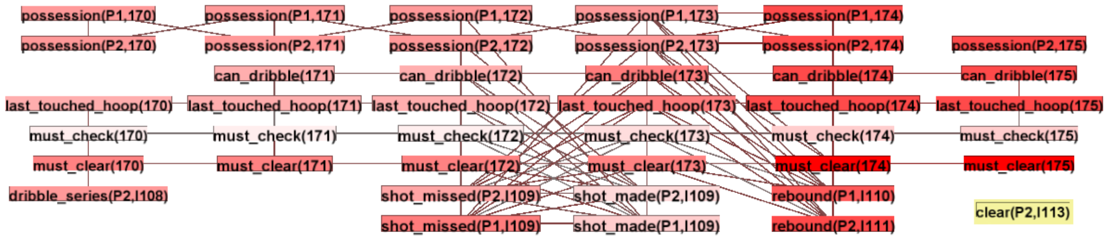
where s is the set of nodes in the intersection of the sets x_{cur} and x_{new}.

The proof can be found in Appendix A. This score measures the average of the difference in max-marginals of the shared nodes. It indicates the susceptibility of the shared nodes in the current network to change their values when a new cluster is added. The score is low if the absolute difference in the max-marginals of the shared variables is high. This indicates that the current network has low uncertainty (or strong belief) in the assigned values to the shared variables. Similarly the score is high if the absolute difference in the max-marginals is low. This indicates that the network has high uncertainty in the assignments to the shared variables and that is where we need more evidence/observations.

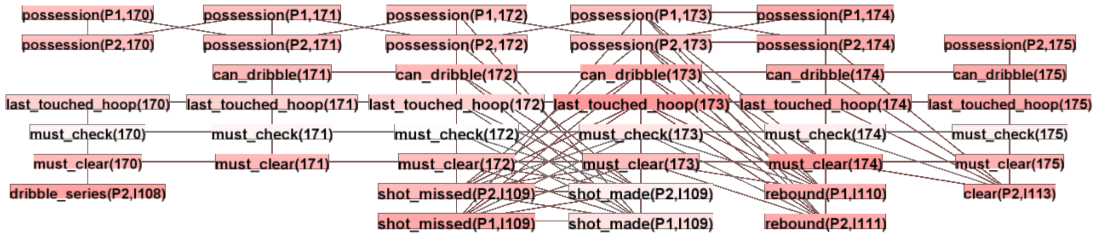
Since the blind score is independent of max-marginals of the new cluster, it does not need the confidence score of a detection which is usually used as a unary potential in the new cluster. This can save computation for the low level detectors by avoiding expensive procedures like feature extraction and classification throughout an image/video and instead run them only when it is needed by the inference. However, the blind score needs to know the shared variables (s) between the new cluster and the current network. This corresponds to determining the locations where the detector would be run and these are usually easy to obtain for sliding-window approaches. For example, to perform 3D object detection, Lin et al. [14] first generate candidate cuboids without object class information which fixes the structure of their network and hence tells us the shared variables for any cluster. They then extract features for generating unary potentials and use it in a contextual model to assign class labels to the hypothesized cuboids. If we use the blind score during the inference, we can potentially save computation by not extracting features



(a) A sequence of events which shows a shot being missed by Player1 and the rebound received by Player2. When Player2 is clearing the ball, the track goes missing for a while and hence the confidence measure for that clear event is low.



(b) Applying an initial threshold for Clear events does not include the highlighted Clear event. However the corresponding Shot Missed event by Player1 is included in the network. The absolute difference in the max-marginals represents *certainty* of a node assignment and hence the negative of that difference represents *uncertainty*. Here, darker colors indicate high uncertainty. When the Clear event is missing, the network is highly uncertain right after the Shot Missed event.



(c) The node assignments become more certain after adding the missing Clear event.

Figure 4.2: Visualization of the Feedback Loop

for cuboids that are likely to be labeled as false. Figure (4.2) illustrates our feedback loop technique using an example from the basketball dataset of Morariu et al. [1].

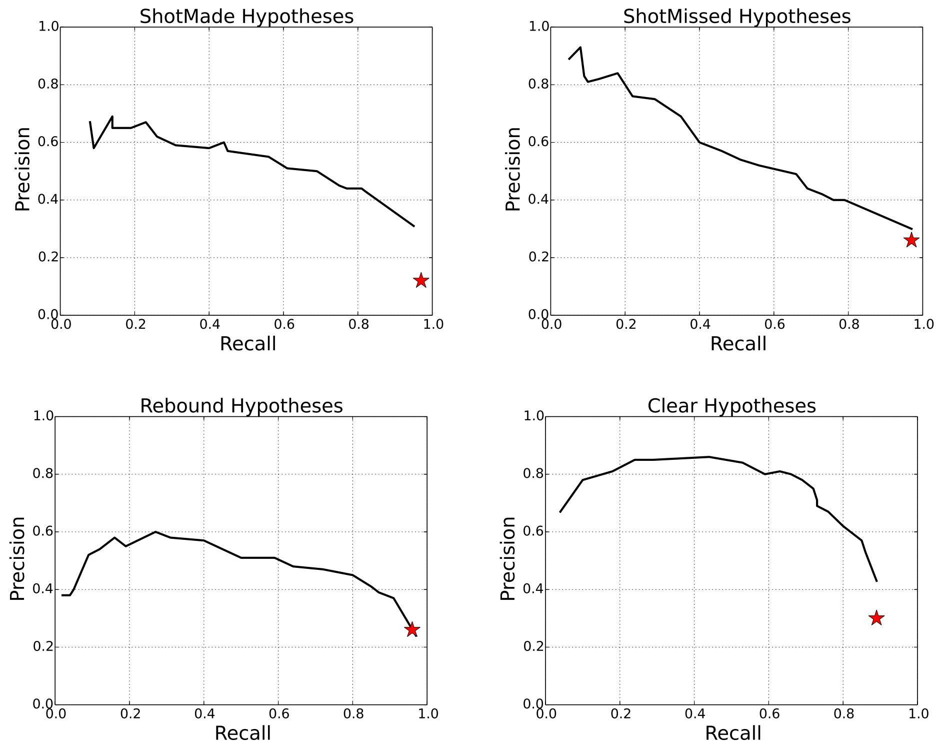


Figure 4.3: PR curves for the newly hypothesized events with continuous confidence measures. The red star shows the operating point of Morariu et al. [1] in their feed-forward approach.

4.3 Experiments

4.3.1 One-on-One basketball dataset

The one-on-one basketball dataset used by Morariu et al. [1] contains tracks of players and ball along with court annotations for seven basketball videos. The are eight events of interest: Check, Out Of Bounds, Shot Made, Shot Missed, Rebound, Clear, Dribble and Steal. They use a Markov Logic Network (MLN) [84] to represent high level rules of the game which interrelates the various events. The inputs to the MLN are candidate events hypothesized by low level detectors which use the tracks of players and the ball.

4.3.2 Hypothesizing candidate events

In the MLN used by Morariu et al. [1], each event was hypothesized with just two discrete confidence values. However, continuous confidence measures are required for the events to better tie them to reality. We hypothesize a new set of candidates with continuous confidence measures for the Shot Made, Shot Missed, Rebound and Clear events and copied the other events (Check, Dribble, Out Of Bounds, Steal) from their dataset. The confidences are obtained based on observations like ball near a player, ball seen inside the hoop, player being inside the two point area, etc. The PR curves of the event hypotheses is shown in Figure (4.3). Since our modified observation model introduces higher uncertainty in event interval endpoints, we also make few minor modifications to the original MLN to make it robust to the overlapping endpoints of different event intervals.

We first test the importance of continuous confidences in the feed-forward setting by feeding in all the hypothesized intervals to the MLN without thresholding. The confidence measures are used as unary potentials for event predicates in the MLN. Inference is then performed to obtain a MAP assignment for the ground MLN, which labels the candidate events as true or false based on the high level context of the game. The results are shown in Table (4.1). We see that the confidence measures play a significant role in improving the event recognition performance.

We have implemented our system as an extension of Alchemy [85], a software package for probabilistic logic inference. The MAP problem for MLNs is framed as an Integer Linear Program (ILP) [86] and we integrated our system with the Gurobi

	Morariu et al. [1]			Ours		
	P	R	F1	P	R	F1
Check	0.84	0.89	0.87	**0.86**	**0.90**	**0.90**
Clear	**0.86**	0.61	0.71	0.81	**0.82**	**0.82**
Dribble	**0.81**	0.75	0.78	0.79	**0.82**	**0.80**
OutOfBounds	**0.88**	**0.66**	**0.75**	0.80	0.62	0.70
Rebound	0.62	0.72	0.67	**0.82**	**0.84**	**0.83**
ShotMade	0.64	0.86	0.73	**0.87**	**0.87**	**0.87**
ShotMissed	0.67	0.79	0.72	**0.81**	**0.85**	**0.83**
Steal	0.08	**0.50**	**0.13**	**0.25**	0.25	0.12
Overall	0.72	0.75	0.74	**0.81**	**0.83**	**0.82**

Table 4.1: Comparison of MLN Recognition Performance using all the hypothesized intervals without thresholding. We can see that the continuous confidence measures for input events play a significant role in improving the performance.

ILP solver [87] for performing inference.

4.3.3 Incrementally adding events with feedback loop

We demonstrate the feedback loop technique by incrementally adding one type of event, the Clear event. The confidence values for the Clear event are scaled between 0.5 and 1. We initialize the network with all the event intervals except for Clear which is thresholded at 0.75. We then run four iterations of the feedback loop and in each iteration, we add a certain number of top ranking Clear events from the remaining set. There are five different kinds of scores that we experiment with: $score(g)_{exact}$, $score(g)_{upper}$, $score(g)_{blind}$, observation score and random score. The observation and random scores are baseline approaches to incrementally adding constants without using a feedback loop. The observation score is the confidence measure that comes from the low level detectors. By adding constants based on their observation score, we are effectively reducing the threshold uniformly throughout the video. The random score is basically selecting a certain number of Clear events

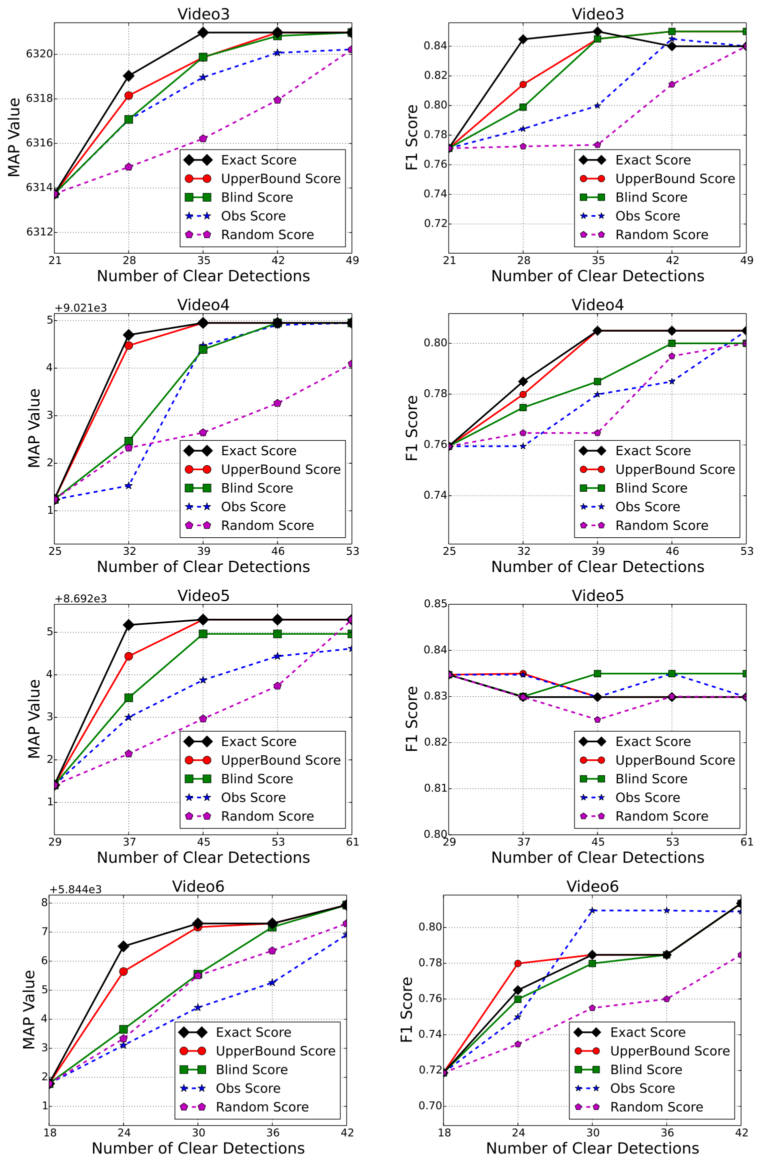


Figure 4.4: Feedback based scores achieve better solutions with fewer detections; We apply an initial threshold on the Clear events and incrementally add the remaining events using the feedback based scores. We measure the exact MAP value of the Markov network along with the $f1$ score corresponding to the ground truth. The plots start at the same initial value for all the five scoring methods since the initial network contains the same set of events. Our feedback based scores achieve better solutions with fewer detections than the baselines - observation score and random score.

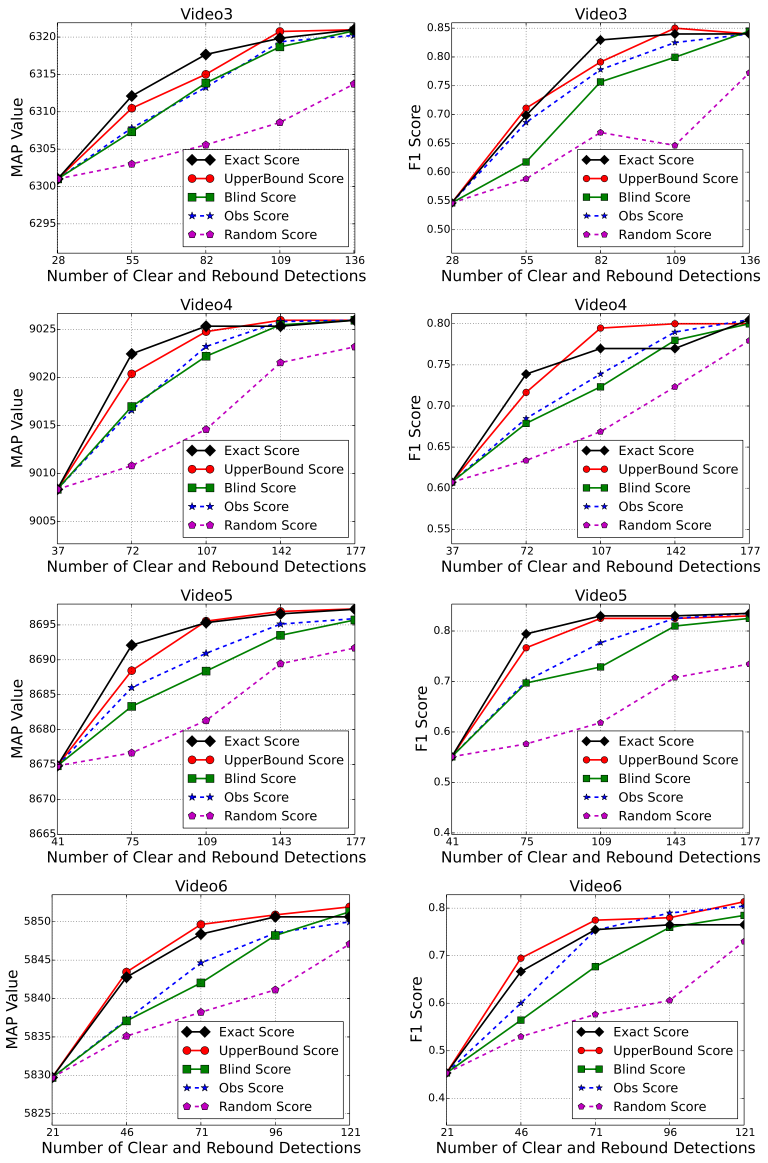


Figure 4.5: We apply threshold on both the Rebound and Clear events for initial network and then incrementally add both events at every iteration. We still see that the exact score and the upper bound score reach better solutions with fewer detections than the observation score. However, the blind score falls slightly below the observation score since it depends only on the current network and the context in the current network is weak due to fewer events.

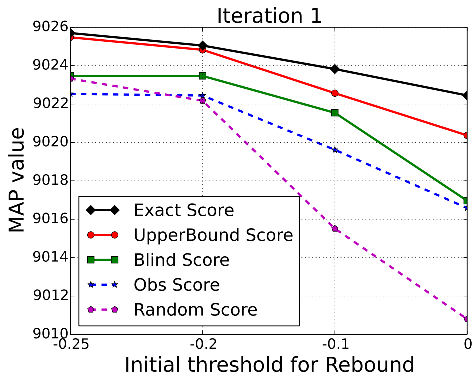
randomly and adding them without looking at either the confidence measures or the context in the main network.

The results are shown in Figure (4.4). Among the seven videos from the dataset, four of the them are large enough to add intervals in an iterative manner. We show the plots of MAP value and also the $f1$ scores against the number of Clear detections in the current network. The plots start at the same initial value for all the five scoring methods since the initial network contains the same set of detections. The goal of our feedback technique is to reach the final MAP value in few iterations by adding only the relevant detections while keeping the rest of them false. The MAP values increase faster with all of our three feedback based score functions when compared to the observation score. The exact score is the quickest followed by the upper bound score and then the blind score. The plots of $f1$ scores also show that we can reach the best possible value with fewer detections using feedback based score functions implying that they select the most relevant events from the missing ones. We observe that the blind score performs well when compared with the observation score. This indicates that the context in the main network has a huge impact on what needs to be added to improve the MAP value.

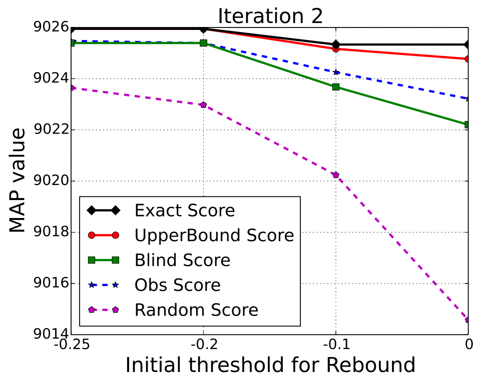
We also experiment with jointly thresholding the Rebound event along with the Clear event. The Rebound events are scaled between -0.25 to 0.1 and we choose a threshold of 0 for the initial network. The Clear events are scaled between 0.5 to 1 and we choose a threshold of 0.75. We then proceed to iteratively add the remaining Rebound and Clear events. The results in Figure (4.5) show that the exact score and upper bound score can reach the best possible MAP value and $f1$ score by adding fewer detections. However the plot for blind score falls below that of the observation score. By increasing the threshold on the Rebound event, the strength of context in the main network is weakened and hence the blind score which is dependent on just the current network starts to perform poorly.

4.3.4 Effect of initial threshold

To observe the effect of initial threshold, we experimented with four different initial thresholds for the Rebound event. Like before, the Rebound events are scaled between -0.25 to 0.1 and the Clear events are scaled between 0.5 to 1. We choose a threshold of 0.75 for Clear events and vary the initial threshold for Rebound events starting from the lowest, which is -0.25 (includes all the Rebound events) and increase up to the value 0 which is high enough to weaken the context. As the initial threshold is increased for the Rebound events, the initial network becomes sparse weakening the context in the initial network. Figure (4.6a) shows that a higher threshold decreases the MAP value achieved in the first iteration of adding events to initial network. The blind score is affected the most since it is dependent only on the current network. It continues to perform poorly in later iterations (Figure (4.6b)) at higher initial threshold for the Rebound event. Hence, it is important to select a reasonably high threshold that allows enough number of events in the initial network without increasing the network size.



(a) First iteration of adding Rebound and Clear events



(b) Second iteration of adding Rebound and Clear events

Figure 4.6: Effect of initial threshold for the Rebound event in video 4; The confidence scores for the Clear events are scaled between 0.5 to 1 and the Rebound events between -0.25 to 0.1. We fix the initial threshold for Clear event at 0.75 and vary the threshold for Rebound from -0.25 to 0. We observe that a higher threshold for Rebound event in the initial network decreases the MAP value that is achieved in the first iteration of adding Rebound and Clear events to the initial network. The blind score continues to perform poorly in later iterations at higher initial threshold due to weak context in the initial network. However, the exact score and the upper bound score are still stable with respect to the initial threshold.

4.4 Conclusion

We propose a computational framework for a feedback loop between high level semantics and low level detectors in a computer vision system, where we use the information in the high level model to select relevant detections from a set of candidate hypotheses. We start with high confidence detections and then iteratively add only those detections to the model that are most likely to be labeled as true by the high level model. This helps us keep the model size small especially in the presence of many noisy detections. We develop the framework for higher order Markov networks and propose three feedback based scoring functions to rank the detections. We show through our experiments on an event recognition system that the feedback loop can construct smaller networks with fewer detections and still achieve the best possible performance.

Chapter 5: Feature Selection using PLS regression and Optimal Experiment Design

Datasets with a large number of features are prevalent in many fields like Computer Vision, Bioinformatics and Chemometrics. These large datasets pose analytical and computational challenges, and the problem is even worse for high dimensional cases where the number of features is much greater than the number of samples. A feature selection process reduces the dimensionality of the data by identifying a subset of the original features that captures the maximum amount of information from the data. The advantages of feature selection are improving the generalization capability of models, reduce computation time and provide a better understanding of the interaction among features [88].

Among supervised feature selection techniques, ranking by regression coefficients is one of the simplest ways to select features. Partial Least Squares (PLS) [89,90] is a widely used regression technique for high dimensional datasets. It is extensively used for wavelength selection in Chemometrics and gene selection in Computational Biology [91,92] as they typically present with high dimensional datasets. The features are usually selected by ranking them according to the value of their PLS regression coefficients or other relevance measures. The caveat of this procedure is that it doesn't jointly look at the features and is susceptible to selecting redundant features. Similar to ℓ_1 and ℓ_2 norm penalized regression techniques, penalized techniques for PLS [93, 94] are one of the approaches to perform feature selection with PLS. The penalized regression techniques enforce sparsity in the regression coefficients along with the minimization of model variance.

The other approach to minimizing the variance of the regression model is to apply the theory of Optimal Experiment Design (OED) [76] and its optimality criterions to PLS regression. The three most commonly used optimality criterions are A-optimality, D-optimality and E-optimality which respectively minimize the trace, determinant and maximum eigenvalue of the covariance matrix of regression coefficients. Optimal Experiment Design has been used for sample selection problems like sensor selection and Active Learning [95]. The optimality criterions are not specific to sample selection and can also be used to measure the optimality of models with different sets of features. Hence we use these criterions with PLS to develop a supervised feature selection technique. We show that an optimal feature subset can be selected by applying these criterions to the loadings covariance matrix obtained from PLS.

We first decompose the prediction error of PLS regression into its bias, variance and noise components. We then apply the OED criterions to the covariance matrix of regression coefficients to derive the A-optimality and D-optimality versions of the Optimal Loadings criterion. We also show that the A-Optimal Loadings criterion can be obtained by explicitly incorporating the property of maximum relevance as maximizing energy content in the loadings matrix. The minimum redundancy property is incorporated as minimizing the condition number of loadings matrix. However, solving the Optimal Loadings criterions is computationally challenging as it is dependent on different PLS models for evaluating different feature subsets. Hence we propose an approximate D-Optimal Loadings criterion that is based on a single loadings covariance matrix obtained with the entire set of features. We also obtain a mathematical relationship between the approximate and the original D-Optimal Loadings criterion and use it to qualitatively justify the approximation.

The advantage of the Optimal Loadings criterions is that the features are evaluated as subsets rather than individual features and hence can simultaneously measure redundancy along with relevance of features. This advantage is clearly evident in our experiments when the number of selected features is small. In our experiments we implement the D-Optimal Loadings criterion that maximizes the determinant of the loadings covariance matrix. Experiments on four datasets indicate that the D-Optimal Loadings criterion performs consistently better than the standard feature selection techniques, in terms of classification accuracies obtained with feature subsets.

5.1 Related Work

Feature selection techniques can be classified [88] into individual feature ranking methods and feature subset evaluation methods. The individual feature ranking methods use relevance measures to sort the features in a rank order. Fisher Score [96] and ReliefF [97] are two techniques that belong to the ranking methods. Features can also be ranked based on regression coefficients and other informative vectors like Variable Influence on Projection (VIP) [98]. Although these methods have a computational advantage, they fail in the presence of redundant features as the minimum redundancy property needs to be measured by jointly looking at the features. A popular technique that incorporates both the relevance and redundancy properties is the minimum redundancy and maximum relevance (mRMR) framework [99,100]. It involves an objective function that is based on Information Theoretic measures and uses incremental search techniques to find the feature subsets. The computational challenge in the original mRMR framework is the estimation of mutual information when the number of samples is small and also when the data is continuous. However, a kernel based dependency measure like the Hilbert Schmidt Independence Criterion (HSIC) can be used instead of the mutual information measure. The HSIC has been used as a measure of feature dependence by L.Song et al. [101].

In the presence of high dimensionality, ordinary least squares regression fails due to the singularity of the feature covariance matrix. Hence regularized linear regression, usually with ℓ_1 and/or ℓ_2 penalization [102,103], is employed to obtain a biased model with smaller variance. Partial Least Squares (PLS) regression [89,90] is a commonly used technique for handling high dimensional datasets. It provides two viewpoints to the modeling process - as a regression technique and as a feature extraction technique. While it can extract information in a latent space of few dimensions, the sparsity of the features needs to be explicitly incorporated into the PLS formulation for feature selection. In the Sparse PLS of K-A Lê Cao et al. [93], ℓ_1 penalization is applied to the loading vectors in the PLS-SVD formulation to integrate feature selection into the modeling process. The Sparse PLS of H.Chun and S.Keles [94] uses both the ℓ_1 and ℓ_2 penalization like that of Elastic Nets in the PLS formulation.

In Ordinary Least Squares regression, under uniform noise assumption, the covariance matrix of the regression coefficients is independent of the response variable. This property is used to apply the Optimal Experiment Design [76] to unsupervised feature selection. The Laplacian Score technique [104] is a ranking based algorithm for unsupervised feature selection that has been extended [105] with OED and shown to perform better than the original ranking based algorithm. While both the penalization and the OED approaches have been studied for ordinary least squares regression, only the penalization methods have been tried with PLS. Our work explores the application of the OED criterions to PLS regression.

5.2 Preliminaries

5.2.1 Partial Least Squares

Partial Least Squares is a simultaneous feature extraction and regression technique, well suited for high dimensional problems where the number of samples is much lesser than the number of features ($n \ll p$). The linear PLS model can be expressed as

$$X = TP^\top + X_{res} \tag{5.1}$$

$$Y = UQ^\top + Y_{res} \tag{5.2}$$

where $X_{n\times p}$ is the feature matrix, $Y_{n\times q}$ is the matrix of response variables or class labels, $T_{n\times d}$ is called the X-scores, $P_{p\times d}$ is X-loadings, $U_{n\times d}$ is Y-scores, $Q_{q\times d}$ is Y-loadings, X_{res} and Y_{res} are the residuals. The data in X and Y are assumed to be mean-centered. X-scores and Y-scores are the projections of n samples onto a d-dimensional orthogonal subspace. The X-scores are obtained by a linear combination of the variables in X with the weights W^* as shown in Eqn. (5.3).

$$T = XW^* \tag{5.3}$$

The inner relation between X-scores and Y-scores is a linear regression model [89] and hence X-scores are called predictors of Y-scores. If B is the regression coefficient for the inner relation between the scores, we can write

$$U = TB \tag{5.4}$$

Substituting the above Eqn. (5.4) in Eqn. (5.2) we get

$$Y = TBQ^\top + Y_{res} \tag{5.5}$$

$$= T\tilde{B} + Y_{res} \tag{5.6}$$

where $\tilde{B} = BQ^\top$. The least squares estimate of $\tilde{B}$ is then given by

$$\hat{B} = (T^\top T)^{-1}T^\top Y \tag{5.7}$$

Hence PLS can be expressed in a linear regression form as,

$$\hat{Y} = T\hat{B} = T(T^\top T)^{-1}T^\top Y \tag{5.8}$$

For a detailed explanation of the PLS technique, we guide the readers to refer [89,90].

The two most popular algorithms to obtain the PLS model are NIPALS [90] and SIMPLS [106]. SIMPLS provides weights W^* which can be combined directly with X where as NIPALS provides weights W that act on the residuals Z_a obtained by deflating X at every component a. The relationship between the two is given by [90],

$$W^* = W(P^\top W)^{-1} \tag{5.9}$$

Here we consider the case of a single response variable $Y_{n\times 1}$ and use the equations from the NIPALS algorithm to obtain the PLS model. However we consider a small variation, where we normalize the scores instead of the loadings. At every iteration for the component a, we have

$$w_a = \frac{Z_a^\top Y}{\sqrt{Y^\top Z_a Z_a^\top Y}} \tag{5.10}$$

$$t_a = \frac{Z_a w_a}{\sqrt{w_a^\top Z_a^\top Z_a w_a}} \tag{5.11}$$

$$p_a = Z_a^\top t_a \tag{5.12}$$

$$Z_{a+1} = Z_a - t_a p_a^\top \tag{5.13}$$

where $Z_1 = X$. The weights and scores form an orthonormal set i.e. $w_i^\top w_j = 0$ and

$t_i^\top t_j = 0$ for $i \neq j$.

5.2.2 Notation

Let π denote a subset of feature indices from the set $\{1, 2, 3, \ldots, p\}$ containing exactly k elements. The feature subset matrix X_π is expressed as

$$X_\pi = X_{(n\times p)}\Pi_{(p\times k)} \tag{5.14}$$

where Π is a column selection matrix that selects k out of p features. Each of the k columns of Π contains a single entry of one at a row indexed by an element in π and zeros elsewhere. Any parameter of a model built with a subset of features is represented by a subscript π.

5.3 Optimal Loadings Technique

5.3.1 Optimal Experiment Design for PLS

Consider a linear regression model

$$Y = X\beta + \epsilon \tag{5.15}$$

where $Y_{n\times 1}$ is the response vector, $X_{n\times p}$ is the feature matrix, $\beta_{p\times 1}$ is the regression coefficient vector and $\epsilon_{n\times 1}$ is the noise vector with mean zero and covariance $\sigma^2 I_n$. The noise for different observations are assumed to be independent of each other.

The Partial Least Squares estimate of the regression coefficients can be obtained by substituting for T_π from Eqn. (5.3) in Eqn. (5.8).

$$\hat{\beta}_\pi = \Pi W_\pi^* (T_\pi^\top T_\pi)^{-1} T_\pi^\top Y = \Pi W_\pi^* T_\pi^\top Y \tag{5.16}$$

By substituting for Y from Eqn. (5.15) in the above Eqn. (5.16), we find that the mean of the PLS estimate is given by

$$E[\hat{\beta}_\pi] = \Pi W_\pi^* T_\pi^\top X\beta + \Pi W_\pi^* T_\pi^\top E[\epsilon] \tag{5.17}$$

$$= \Pi W_\pi^* T_\pi^\top X\beta \tag{5.18}$$

where in Eqn. (5.17) we have assumed that $\Pi W_\pi^* T_\pi^\top$ and ϵ are negligibly correlated. This is possible when the Signal to Noise Ratio is high and hence the deviation in the PLS model with respect to noise is negligible. The covariance of $\hat{\beta}_\pi$ is given by

$$cov(\hat{\beta}_\pi) = E\left[(\hat{\beta}_\pi - E[\hat{\beta}_\pi])(\hat{\beta}_\pi - E[\hat{\beta}_\pi])^\top\right] \tag{5.19}$$

$$= E[\hat{\beta}_\pi \hat{\beta}_\pi^\top] - E[\hat{\beta}_\pi] E[\hat{\beta}_\pi^\top] \tag{5.20}$$

$$= \sigma^2 \Pi W_\pi^* W_\pi^{*\top} \Pi^\top \tag{5.21}$$

For a new sample (x, y) such that $y = x^\top \beta + e$ and $\hat{y} = x^\top \hat{\beta}_\pi$, the mean squared prediction error of PLS can be decomposed into its bias, variance and noise components.

$$E[(y-\hat{y})^2] \tag{5.22}$$

$$= x^\top E[(\beta-\hat{\beta}_\pi)(\beta-\hat{\beta}_\pi)^\top]x+\sigma^2 \tag{5.23}$$

$$= Bias^2 + x^\top \left(\sigma^2 \Pi W_\pi^* W_\pi^{*\top} \Pi^\top\right) x + \sigma^2 \tag{5.24}$$

where

$$Bias^2 = x^\top (I_p - \Pi W_\pi^* T_\pi^\top X)\beta\beta^\top (I_p - X^\top T_\pi W_\pi^{*\top} \Pi^\top) x \tag{5.25}$$

Since the squared prediction error is directly proportional to $cov(\hat{\beta}_\pi)$, the prediction error can be minimized by minimizing the covariance of PLS regression coefficients. Also, in high dimensional datasets, reducing the model variance helps avoid overfitting to the data. The theory of Optimal Experiment Design proposes to minimize this covariance by optimizing the eigenvalues of $\Pi W_\pi^* W_\pi^{*\top} \Pi^\top$ through various criterions.

Lemma 1. *The matrices $W_\pi^* W_\pi^{*\top}$ and $(P_\pi P_\pi^\top)^\dagger$ have the same non-zero eigenvalues, where $\dagger$ represents the Moore-Penrose inverse.*

Proof. By substituting for W_π^* from Eqn. (5.9), we get

$$\begin{aligned}
&\text{eigval}(W_\pi^* W_\pi^{*\top}) && (5.26)\\
&= \text{eigval}\left[W_\pi (P_\pi^\top W_\pi)^{-1} \left(P_\pi^\top W_\pi)^{-1}\right)^\top W_\pi^\top\right] && (5.27)\\
&= \text{eigval}\left[W_\pi W_\pi^\top (P_\pi P_\pi^\top)^\dagger W_\pi W_\pi^\top\right] && (5.28)\\
&= \text{eigval}\left[(P_\pi P_\pi^\top)^\dagger\right] && (5.29)
\end{aligned}$$

where eigval() refers to the eigenvalues of a matrix. Eqn. (5.28) can be regarded as a similarity transformation since W_π is orthonormal. The rank of these matrices is equal to the number of latent components (d) extracted. □

Using the above Lemma 1 and applying the A-optimality criterion to the covariance matrix of PLS coefficients in Eqn. (5.21) we get,

$$\arg\min_{\Pi} \ \text{trace}\left[\Pi (P_\pi P_\pi^\top)^\dagger \Pi^\top\right] \tag{5.30}$$

We can drop the pre and post multiplication by Π as it is only padding zeros to change the size of the matrix, $(P_\pi P_\pi^\top)^\dagger$, from $k \times k$ to $p \times p$. For a fixed number of selected features, k, the A-optimal criterion can be rewritten as

Definition 1 (A-Optimal Loadings criterion)**.** *The A-optimality version of Optimal Loadings criterion is given by*

$$\arg\min_{\Pi} \ \text{trace}\left[(P_\pi P_\pi^\top)^\dagger\right] \tag{5.31}$$

We could also apply the D-optimality or E-optimality criterion which minimize the determinant or the maximum eigenvalue respectively, instead of the trace in Eqn. (5.31). Among these optimality criterions, the D-optimality criterion is the most popular due to availability of off-the-shelf algorithms in convex optimization toolboxes and row exchange algorithms. It also simplifies the determinant minimization of an inverse to maximizing the determinant of the matrix itself. The D-optimality version of the criterion (5.31) is given by

$$\arg\min_{\Pi} \ \det{}^{\dagger}\left[(P_\pi P_\pi^\top)^\dagger\right] \tag{5.32}$$

which is equivalent to

Definition 2 (D-Optimal Loadings criterion)**.** *The D-optimality version of Optimal Loadings criterion is given by*

$$\arg\max_{\Pi} \ \det{}^{\dagger}\left(P_\pi P_\pi^\top\right) \tag{5.33}$$

where $\det{}^{\dagger}()$ *represents pseudo-determinant which is a product of non-zero eigenvalues of the matrix.*

The actual determinant is substituted by a pseudo determinant as the criterion involves a rank deficient matrix.

5.3.2 PLS models with Maximum Relevance and Minimum Redundancy

The A-Optimal Loadings criterion (5.31) can also be obtained by applying the requirements of maximum relevance and minimum redundancy for feature subsets. The following derivation provides an intuitive viewpoint to the same criterion that is obtained from the theory of Optimal Experiment Design.

The reconstruction error in a feature extraction technique measures the difference between the original energy content in all the features and the amount captured by the latent components. While it is our goal to obtain features that best explain a response variable, the structure in data should also be preserved. By substituting for p_a from Eqn. (5.12) in Eqn. (5.13), we get

$$Z_{a+1} = [I - t_a t_a^\top] Z_a = [I - \sum_{i=1}^{a} t_i t_i^\top] X \tag{5.34}$$

The reconstruction error can also be viewed as the residuals that cannot be explained by the PLS model. Hence we can use Eqn. (5.34) to express the error in a form similar to that of reconstruction error for PCA.

$$\begin{aligned} error^2 = ||X_{res}||_2^2 &= \left|\left|X - TT^\top X\right|\right|_2^2 && (5.35)\\ &= \text{trace}\left[X^\top X - X^\top TT^\top X\right] && (5.36)\\ &= \text{trace}\left[X^\top X\right] - \text{trace}\left[PP^\top\right] && (5.37)\end{aligned}$$

where Eqn. (5.37) is obtained by substituting for X from Eqn. (5.1) in the second term and making use of the fact that the scores T are orthogonal to the residuals X_{res}. The reconstruction error reduces with increase in the number of components extracted. But for a fixed number of components d, the error is minimum when the trace $\left[PP^\top\right]$ is maximum. Therefore we start by defining the feature selection criterion as

$$\arg\max_{\Pi}\ \text{trace}\left[P_\pi P_\pi^\top\right] \tag{5.38}$$

It should be noted that the reconstruction error in itself is not considered in criterion (5.38). This criterion tries to select the feature subset that contains the maximum energy content (measured by Frobenius norm) in the PLS model after feature selection.

The criterion (5.38) is also directly proportional to covariance between the features X and the response variable Y. This can be seen by substituting for p_π from Eqn. (5.12) in criterion (5.38) and then expanding up to w_π in Eqn. (5.10). We get

$$\text{trace}[P_\pi P_\pi^\top] = \sum_{a=1}^{d} \frac{Y^\top \left(Z_a Z_a^\top\right)^3 Y}{Y^\top \left(Z_a Z_a^\top\right)^2 Y} \tag{5.39}$$

Since PLS extracts components such that the covariance between features and response variable and the covariance between features itself are simultaneously maximized, the criterion (5.38) simultaneously satisfies the relevance property towards the response variable and the latent information in features.

However, the trace criterion (5.38) does not measure the redundancy property and hence we incorporate condition number of $P_\pi^\top$ to measure the linear dependence of columns/features. Since we want to minimize the condition number, the criterion (5.38) can be rewritten as

$$\arg\max_{\Pi} \left(\frac{\operatorname{trace}\left[P_\pi P_\pi^\top\right]}{\left(\kappa\left(P_\pi^\top\right)\right)^2} \right) \tag{5.40}$$

The condition number in Frobenius norm is defined as

$$\left(\kappa\left(P_\pi^\top\right)\right)^2 = \operatorname{trace}\left[P_\pi P_\pi^\top\right] . \operatorname{trace}\left[(P_\pi P_\pi^\top)^\dagger\right] \tag{5.41}$$

We now substitute for κ in criterion (5.40) to obtain

$$\arg\min_{\Pi} \ \operatorname{trace}\left[(P_\pi P_\pi^\top)^\dagger\right] \tag{5.42}$$

This is the same A-Optimal Loadings criterion (5.31) obtained earlier by applying the Optimal Experiment Design to Partial Least Squares regression.

5.3.3 Approximation for the D-Optimal Loadings criterion

In our experiments we choose to implement the D-optimality version of Optimal Loadings criterion as it simplifies the minimization of determinant of inverse matrix to the maximization of determinant itself. The availability of off-the-shelf algorithms for determinant maximization is another advantage of using the D-optimality criterion.

The loadings in criterion (5.33) is dependent on π and is infeasible to construct a PLS model every time a subset of features is to be evaluated. This would defeat the purpose of a feature selection technique. Hence we try to express the criterion in terms of loadings obtained with all features. From Eqn. (5.1), we have

$$X_\pi^\top X_\pi = P_\pi P_\pi^\top + X_{res\,\pi}^\top X_{res\,\pi} \tag{5.43}$$

$$\Pi^\top X^\top X \Pi = \Pi^\top P P^\top \Pi + \Pi^\top X_{res}^\top X_{res} \Pi \tag{5.44}$$

The right hand terms of the above Eqns. (5.43) and (5.44) can be equated (Eqn. (5.14)) to obtain

$$P_\pi P_\pi^\top = \Pi^\top P P^\top \Pi + \Delta_\pi \tag{5.45}$$

where Δ_π is a symmetric matrix given by

$$\Delta_\pi = \left[\Pi^\top X_{res}^\top X_{res} \Pi - X_{res\,\pi}^\top X_{res\,\pi}\right] \tag{5.46}$$

Since we use the D-optimality criterion for feature selection, we discuss the relationship between the determinants of $P_\pi P_\pi^\top$ and $\Pi^\top P P^\top \Pi$. The singularity of these matrices presents difficulties in quantifying their behavior. Therefore we obtain the relationship between the determinants of regularized matrices $(P_\pi P_\pi^\top + I)$ and $(\Pi^\top P P^\top \Pi + I)$.

Theorem 1. *The relationship between the determinants of* $(\Pi^\top P P^\top \Pi + I)$ *and* $(P_\pi P_\pi^\top + I)$ *is given by,*

$$\det(P_\pi P_\pi^\top + I) = \det(M + \Lambda M \Sigma^{-1}) \det(\Pi^\top P P^\top \Pi + I) \tag{5.47}$$

where M is a unitary matrix, Λ is a diagonal matrix of real eigenvalues of Δ_π and Σ is a diagonal matrix of positive eigenvalues of $(\Pi^\top PP^\top\Pi + I)$.

Proof. Let the two symmetric, positive semi-definite matrices $P_\pi P_\pi^\top$ and $\Pi^\top PP^\top\Pi$, each be of rank d and size $k \times k$, with the relationship between them as

$$P_\pi P_\pi^\top = \Pi^\top PP^\top\Pi + \Delta_\pi \tag{5.48}$$

where Δ_π is a symmetric matrix given by

$$\Delta_\pi = \left[\Pi^\top X_{res}^\top X_{res}\Pi - X_{res\,\pi}^\top X_{res\,\pi}\right] \tag{5.49}$$

We make use of Sherman-Morrison-Woodbury formula [107] for expressing the determinant of sum of matrices.

$$\det(P_\pi P_\pi^\top + I) = \det(\Pi^\top PP^\top\Pi + I + \Delta_\pi) \tag{5.50}$$

$$= \det(\Pi^\top PP^\top\Pi + I + U\Lambda U^\top) \tag{5.51}$$

$$= \det(I + \Lambda U^\top(\Pi^\top PP^\top\Pi + I)^{-1}U)\ \det(\Pi^\top PP^\top\Pi + I) \tag{5.52}$$

$$= \det(I + \Lambda U^\top V\Sigma^{-1}V^\top U)\ \det(\Pi^\top PP^\top\Pi + I) \tag{5.53}$$

$$= \det(I + \Lambda M\Sigma^{-1}M^\top)\ \det(\Pi^\top PP^\top\Pi + I) \tag{5.54}$$

$$= \det(M + \Lambda M\Sigma^{-1})\ \det(\Pi^\top PP^\top\Pi + I) \tag{5.55}$$

where we have applied Eigen-decomposition on Δ_π and $(\Pi^\top PP^\top\Pi + I)$. Σ and Λ are diagonal matrices containing non-negative eigenvalues (σ) of $(\Pi^\top PP^\top\Pi + I)$

and real eigenvalues (λ) of Δ_π, respectively. M is a unitary matrix obtained as a product of two other unitary matrices U and V. □

The two determinants are highly correlated when the condition number of $(M + \Lambda M \Sigma^{-1})$ is small. The condition number of a matrix measures the asymptotic worst case of the amount of perturbation that can be produced by the matrix when multiplied with other matrices. The eigenvalues in Σ and Λ are indicators of the energy content in structured data and noise respectively, where noise is any structure that cannot be explained by the first d components of the PLS model. The theoretical and empirical observations (found in the supplementary material) suggest that the condition number is small when the variance in noise is low and levels of noise are far away from that of structure in data. Therefore under the assumption of high Signal to Noise Ratio, we can ignore Δ_π and substitute for $P_\pi P_\pi^\top$ from Eqn. (5.45) in criterion (5.33). The approximate feature selection criterion is given by,

$$\arg\max_{\Pi} \ \det{}^{\dagger}(\Pi^\top P P^\top \Pi) \tag{5.56}$$

The number of components in $\Pi^\top P P^\top \Pi$ and $P_\pi P_\pi^\top$ must be equal to compare the information between the two matrices. The number of components in PLS regression determines the bias and variance of the model. It is usually chosen such that the cross-validation error of PLS regression is minimum.

The experiments and discussion in the following sections use the D-optimality criterion for feature selection. D-optimal designs are usually generated by employing row exchange algorithms [77, 108]. These algorithms add or delete rows, starting from a non-singular set, in order to increase the determinant. The algorithm iterates until the increment in determinant becomes lesser than some fixed threshold or the number of iterations reach a maximum value. However, it is not guaranteed that the iterations will converge to the global maximum value. One of the first exchange algorithms was developed by V.V.Fedorov and several modifications have been proposed to improve the computational performance [108]. The traditional D-optimal experiment design differs from the feature selection problem, as it allows duplicate samples. Hence the standard exchange algorithms need to be tweaked to avoid duplicates for feature selection.

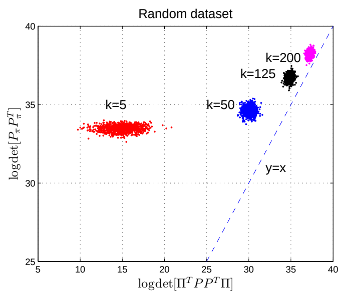
Since the D-optimality criterion involves maximization of the determinant, it can also be treated as a convex optimization problem [109]. The integer constraints $\pi_i \in \{0, 1\}$ need to be relaxed to $\pi_i \in [0, 1]$.

$$\text{minimize} \quad -\log\det\left[\sum_{i=1}^{p} \pi_i P_i P_i^\top\right] \tag{5.57}$$

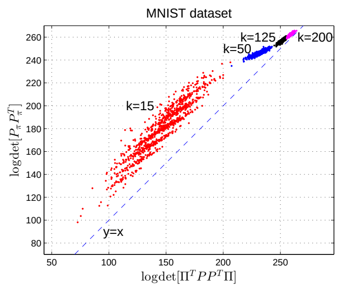
$$\text{subject to} \quad \sum_{i=1}^{p} \pi_i = k \tag{5.58}$$

$$0 \leq \pi_i \leq 1,\ i = 1, \ldots p \tag{5.59}$$

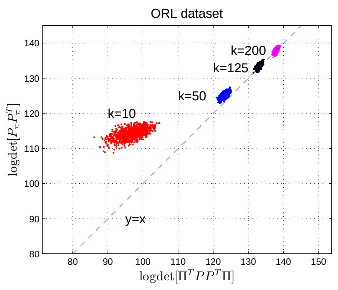
It can be seen that the solution to the original problem in Criterion (5.56) is a feasible solution to the above relaxed problem. Usually we obtain a discrete solution by considering the k largest values of π_i, which can lead to a sub-optimal solution to the original problem. The $\log\det$ criterion is an objective function available with popular SDP solvers [110]. One of the disadvantages of the convex optimization



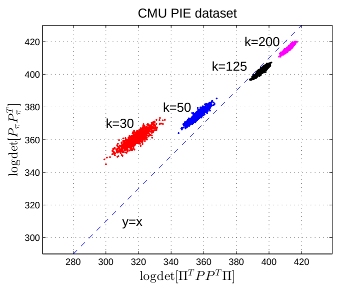
(a) Random data



(b) MNIST dataset



(c) ORL dataset



(d) CMU PIE dataset

Figure 5.1: Relationship between the original criterion $\log\det[P_\pi P_\pi^\top]$ and the approximate criterion $\log\det[\Pi^\top P P^\top \Pi]$, that are obtained by applying PLS for varying number of features, k, in a subset π. The approximate and original criterions are positively correlated for the real datasets. Hence, by maximizing the approximate criterion we are not too far away from the maximum of the original criterion.

methods is that they store the entire convex hull of features, which is difficult to handle for large loadings matrix due to memory restrictions.

5.4 Analysis of the relationship between $P_\pi P_\pi^\top$ and $\Pi^\top P P^\top \Pi$

We can obtain an upper bound for the relationship (5.47) by finding the largest singular value of $\det(M + \Lambda M \Sigma^{-1})$. The spectral norm measures the largest singular value of a matrix. Using few of the properties of norms we can write

$$||M + \Lambda M \Sigma^{-1}||_2 \leq ||M||_2 + ||\Lambda||_2 ||M||_2 ||\Sigma^{-1}||_2 \tag{5.60}$$

$$= 1 + \frac{\lambda_{max}}{\sigma_{min}} \tag{5.61}$$

where $\lambda_{max} = \max_i |\lambda_i|$ and $\sigma_{min} = \min_i \sigma_i$. Therefore the upper bound for relationship (5.47) is given by,

$$\det(P_\pi P_\pi^\top + I) \leq \left(1 + \frac{\lambda_{max}}{\sigma_{min}}\right)^k \det(\Pi^\top P P^\top \Pi + I) \tag{5.62}$$

To determine the lower bound, we will need to find the smallest singular value of $\det(M + \Lambda M \Sigma^{-1})$. However the only safe bound that can be obtained is that the smallest singular value is greater than zero as the determinants on both sides of the relationship need to be positive. Nevertheless, a qualitative discussion can be provided by estimating the smallest singular value by a lower bound for the norms of the columns. We will first express the matrix $M + \Lambda M \Sigma^{-1}$ as,

$$M + \Lambda M \Sigma^{-1} = \tag{5.63}$$

$$\begin{pmatrix} (1+\frac{\lambda_1}{\sigma_1})m_{11} & (1+\frac{\lambda_1}{\sigma_2})m_{12} & \dots & (1+\frac{\lambda_1}{\sigma_k})m_{1k} \\ (1+\frac{\lambda_2}{\sigma_1})m_{21} & (1+\frac{\lambda_2}{\sigma_2})m_{22} & \dots & (1+\frac{\lambda_2}{\sigma_k})m_{2k} \\ \vdots & & \ddots & \\ (1+\frac{\lambda_k}{\sigma_1})m_{k1} & (1+\frac{\lambda_k}{\sigma_2})m_{k2} & \dots & (1+\frac{\lambda_k}{\sigma_k})m_{kk} \end{pmatrix} \tag{5.64}$$

The norm of a column is given by

$$\beta_i = \sqrt{\sum_{j=1}^{k} \left(1 + \frac{\lambda_j}{\sigma_i}\right)^2 m_{ji}^2} \tag{5.65}$$

Since λ_j can be negative, the lower bound is dependent on ratio between λ_j and σ_i. Therefore we just let l be the column that minimizes the column norms and calculate a $\lambda_{min,l}$ such that

$$\lambda_{min,l} = \arg\min_{\lambda_j} \left|1 + \frac{\lambda_j}{\sigma_l}\right| \tag{5.66}$$

Then a lower bound for the norms of columns is given by

$$\beta_i \geq \left|1 + \frac{\lambda_{min,l}}{\sigma_l}\right| \tag{5.67}$$

and an approximate lower bound on the determinant can be written as

$$\det(P_\pi P_\pi^\top + I) \gtrsim \left|1 + \frac{\lambda_{min,l}}{\sigma_l}\right|^k \det(\Pi^\top P P^\top \Pi + I) \tag{5.68}$$

Combining the two bounds in (5.62) and (5.68) we get

$$\left|1 + \frac{\lambda_{min,l}}{\sigma_l}\right|^k \lesssim \frac{\det(P_\pi P_\pi^\top + I)}{\det(\Pi^\top P P^\top \Pi + I)} \leq \left(1 + \frac{\lambda_{max}}{\sigma_{min}}\right)^k \tag{5.69}$$

In most practical situations the bounds in inequality (5.69) are much tighter. The number of non-zero eigenvalues of Δ_π are usually few and hence the exponential factor of k is also low.

Figure 5.1 shows a quantitative relationship between the log determinants of $P_\pi P_\pi^\top$ and $\Pi^\top P P^\top \Pi$ for random data and three of the datasets (ORL, MNIST, CMU PIE) used in our experiments. The random data is of size 100×1000. The point clouds are generated by observing the determinant values for randomly selected subsets of size k. We can see that for the MNIST (Figure 5.1(b)), ORL (Figure 5.1(c)) and CMU PIE (Figure 5.1(d)) datasets, the point clouds are very narrow and the behavior of two matrices are almost positively correlated.

We can use the bounds in inequality (5.69) to qualitatively describe the situations when the point clouds in Figure 5.1 will be narrow so that the determinants are positively correlated. The point clouds are narrower when the ratio between the bounds is close to one. Minimizing this ratio is equivalent to minimizing the condition number of the matrix $(W + \Lambda W \Sigma^{-1})$ which is the ratio of its largest singular value to its smallest singular value. The largest singular value of $(W + \Lambda W \Sigma^{-1})$ is $1 + \lambda_{max}$ since the minimum of σ_i is one. The condition number is then given by,

$$\kappa \simeq \frac{1 + \lambda_{max}}{\left|1 + \frac{\lambda_{min,l}}{\sigma_l}\right|} \tag{5.70}$$

The eigenvalues (λ) of Δ_π are usually few large positive values coming mostly due to $\Pi^\top X_{res}^\top X_{res} \Pi$ and few negative values coming due to $-(X_{res\,\pi}^\top X_{res\,\pi})$. These eigenvalues are indicators of the information content in the noise, where noise is any structure that cannot be explained by the first d components of the PLS model. The eigenvalues in Σ are indicators of information content in the structured data.

When all the λ_j are positive, the approximate lower bound in inequality (5.67) is $\left(1 + \frac{\lambda_{min}}{\sigma_{max}}\right)$ where $\lambda_{min} = \min_j \lambda_j$. In this case, the condition number is low when the ratio between λ_{max} and λ_{min} is low i.e. the variance in noise is low. When there are negative eigenvalues, $\lambda_{min,l}$ is satisfied by an eigenvalue whose absolute value is close to σ_l. In such a situation, the condition number is low when λ_i are farther from σ_i i.e. the levels of noise and structured data are separated. Therefore the approximation gets better as the variance in noise gets lower and noise levels are farther away from that of structured data. In many real datasets, linear regression can provide good models and hence in such situations by maximizing $\det(\Pi^\top P P^\top \Pi + I)$, we are not too far away from the maximum of $\det(P_\pi P_\pi^\top + I)$.

5.5 Experiments and Results

To evaluate the performance of our feature selection criterion, we test it in a classification framework where feature selection is treated as a preprocessing filter that produces the indices, π, of the selected feature subset. The feature subset is then used to obtain low dimensional subspaces using PLS. The classification is performed using a Linear Discriminant Classifier in the low-dimensional projection subspace. In a cross-validation setting, the test data is separated from training data for both feature selection and classifier training. This experimental setup is used in order to avoid any overoptimistic performance results obtained when evaluating feature selection using the entire data, as reported in [111].

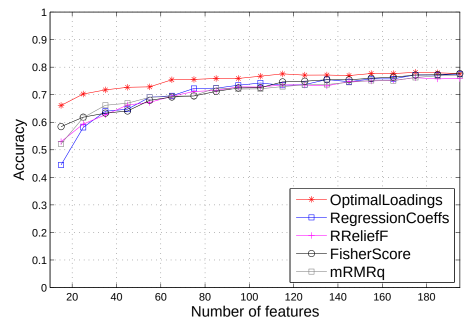
5.5.1 Datasets

The experiments are performed on four datasets - two of them are face image datasets, one is a handwritten digit dataset and the last one is a mass-spectrometric dataset of cancerous and normal tissues. For all of the three image datasets, pixel values are used as features and no feature extraction is performed. The first dataset is a subset[1] [112] of the MNIST handwritten digits. This dataset contains 200 images each for 10 different digit classes, producing a dataset of size 2000×784. The second one is a subset of the AT&T ORL face image database. The dataset consists of face images for 10 subjects with 10 images for each subject with pose variations, which produces a dataset of size 100×10304. The third dataset is a subset of the CMU PIE database that contains face images of 10 different people in a fixed frontal pose (Pose 27) with light and illumination changes. There are 49 images per person, hence producing a dataset of size 490×4096. The fourth is the Arcene dataset from the NIPS Feature Selection Challenge. It contains training and validation sets each of size 100×10000. There are two classes in this dataset.

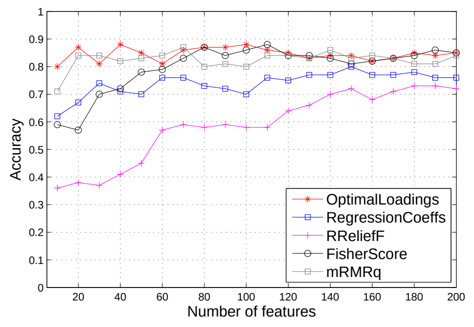
5.5.2 Comparison with other Feature Selection Techniques

We evaluate the performance of D-Optimal Loadings criterion along with other supervised feature selection techniques such as ranking by regression coefficients, Fisher Score [96], RRelief-F [97] and mRMR [100]. For the D-Optimal Loadings criterion, the number of components is chosen based on the minimization of cross

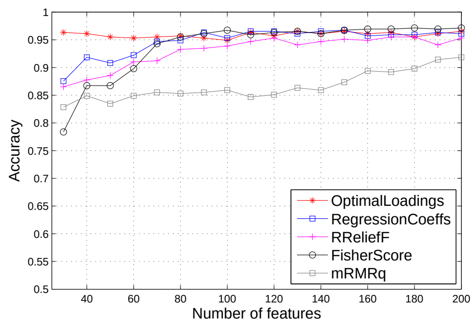
[1] `http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html`



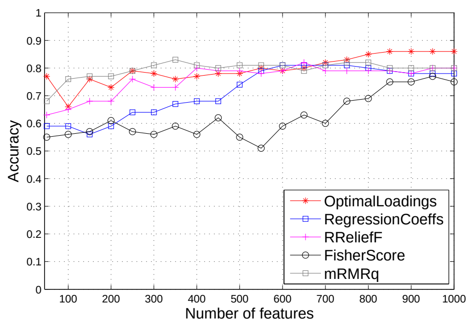
(a) MNIST dataset



(b) ORL dataset



(c) CMU PIE dataset



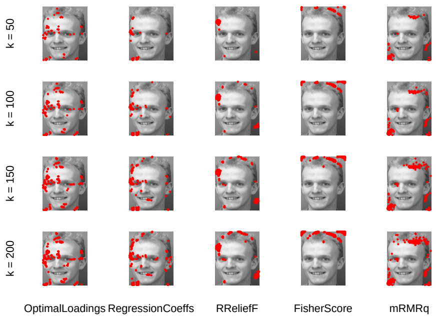
(d) Arcene dataset

Figure 5.2: Classification performance with feature subsets: The D-Optimal Loadings criterion performs better than others on the MNIST and the CMU PIE datasets and performs equally well with the mRMR technique on the ORL and the Arcene datasets. It also shows a consistent performance especially when the number of selected features is small.

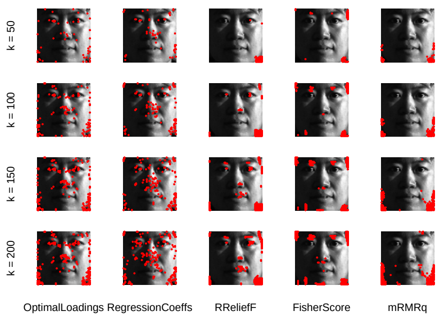
validation error of PLS regression and the determinant maximization is performed using a tweaked version of the row exchange algorithm available in MATLAB Statistics Toolbox. The same PLS model is used to obtain the regression coefficients and the top features are selected based on the absolute value of their coefficient. We use the regression version of Relief-F as it showed better performance than the classification version. In RRelief-F, the neighborhood and number of samples for quality estimation are set to 10 and 100 respectively. Finally for the mRMR technique we use the Mutual Information Quotient scheme since it is shown to perform better than the MI Difference scheme. Here we do not discretize the data any further.

We compare the performance using classification accuracies obtained using a Linear Discriminant Classifier. We prefer to use a simple linear classifier so as to avoid tuning the new parameters introduced by nonlinear classifiers. Since the number of selected features can be greater than the number of samples, the classifier is trained in a PLS subspace to avoid over-fitting. The feature subset is used to construct a subspace whose dimensions are again selected based on least cross-validation error for PLS regression. This happens to be same as that used for the D-Optimal Loadings criterion. Given the number of components as d, the experiments are conducted for varying sizes of the feature subset. During the test phase, we select the feature subset from test data, find projections using weights from training phase and then classify using the trained model.

We found that the cross validation error of PLS regression stabilizes at around 10, 15, 30 and 20 components for the ORL, MNIST, CMU PIE and Arcene datasets respectively. Using these number of components, we perform a 20-fold cross-validation experiment for the ORL dataset and 10 fold cross-validation for MNIST and CMU PIE datasets. A larger number of folds is used for the ORL dataset due to smaller number of samples. For the Arcene dataset, the validation set is used as the test set and entire training set is used for training. Figure 5.2 shows the classification accuracies obtained with D-Optimal Loadings, Regression coefficients, Fisher score, Relief-F and mRMR for the four datasets. The D-Optimal Loadings criterion outperforms other techniques on the MNIST and the CMU PIE datasets and performs equally well with the mRMR technique on the ORL and the Arcene datasets. The D-Optimal Loadings technique can very well handle the situation when the number



(a) Sample ORL image



(b) Sample CMU PIE image

Figure 5.3: Feature points selected by D-Optimal Loadings, Regression Coefficients, Relief-F, Fisher Score and mRMR techniques. The features selected by D-Optimal Loadings are well distributed across the significant regions of the image unlike others that tend to get clustered or lie in noisy regions.

of selected features is small. We see that the Fisher score and Relief-F are generally worse performing for smaller number of features since they do not handle redundancy among features. In Figure 5.3 the feature points selected by the five techniques are shown overlaid on sample images from two of the datasets. The features selected by D-Optimal Loadings are well distributed across the significant regions of the image unlike others that tend to get clustered or lie in the noisy regions.

5.6 Conclusion

Our work explores the application of the theory Optimal Experiment Design (OED) to Partial Least Squares (PLS) regression. We use the OED to derive the A-Optimal Loadings and D-Optimal Loadings feature selection criterions with the goal of minimizing the variance of the PLS regression model. We specifically use an approximation of the D-Optimal Loadings criterion that maximizes the determinant of loadings covariance matrix to select an optimal feature subset. The availability of off-the-shelf row exchange algorithms and convex optimization methods for determinant maximization hastens the feature selection stage in a pattern analysis problem. One of the important characteristics of the Optimal Loadings criterions is that they are based on optimization of eigenvalues which is necessarily evaluated at a subset level. We also provide insight into the technique by deriving the A-Optimal Loadings criterion by using just the properties of maximum relevance and minimum redundancy for feature subsets. The results from our experiments with four datasets indicate that the D-Optimal Loadings criterion selects better feature subsets when compared to other techniques such as mRMR and Relief-F. Apart from classification accuracies, the locations of feature points on these images also indicate that it selects non-redundant features from the significant regions of the image.

Chapter 6: Conclusion

In this thesis, we have presented techniques that model context for improving accuracy and reducing computational requirements of object detection and event detection tasks.

We showed that modeling context is essential for detecting objects mentioned in referring expressions. The main challenge addressed by our technique is the lack of annotations of context objects for training. Our proposed technique learns the contextual relationships between objects in a weakly supervised manner by relying on the annotations available for the referred object. We demonstrated that modeling context provides better performance than models trained on object properties only.

We have also shown the benefits of context as a guide to sequentially processing images and videos. We used structure in scenes and activities to incrementally process images and videos while saving computation resources. In the case of detecting objects in indoor scenes, we learned search strategies using imitation learning which did not involve explicit encoding of spatial relationships between objects. We also proposed a feedback based incremental algorithm to detect events in one-on-one basketball videos. Our sequential technique used basketball rules to construct relationships between detected events and then used it to search for other missing events. The results from both the domains showed that intelligent searching using context can reach the best accuracy possible by processing very few regions when compared to naive methods that do not use context.

Our work in this thesis provides a few directions for future work.

- Referring expressions can be parsed as a tree of objects, attributes and relationships. Such a hierarchy can be imposed on visual grounding as well to better model the referring expressions.

- Datasets like Visual Genome [55] contain annotations of objects and relationships in detail which can be used to learn context models with full supervision. It would be worthy to see if a model learned with weak supervision like ours and a model learned with full supervision can be combined. Even fully annotated datasets may benefit from additional (larger scale) data that is not explicitly annotated.

- In an application where there is a dialogue between humans and robots to perform tasks, information is received by the robot in an incremental fashion. For example, a robot might be asked to fetch a box via a referring expression. If the robot is still confused, it receives additional information in the referring expression for disambiguation. This task could be solved by incrementally constructing the query and knowledge about the environment to understand referring expressions.

Appendix A: Derivation of Scoring Functions for Feedback Loop Inference

Before deriving the score functions, we first formulate the MAP inference problem in binary Markov networks as an Integer Linear Program (ILP) following the work of Globerson and Jaakkola [82]. The integer variables in the ILP are then relaxed to continuous values giving us a relaxed linear program. We then obtain the dual of this relaxed linear program and show a block co-ordinate descent strategy that can be used to solve the dual through a "message passing algorithm". However we do not solve the the inference problem in the dual space. We only use the message update equations for the dual variables to obtain our score functions.

Sontag et al. [78] use these message update equations to rank clusters of variables in their cluster pursuit algorithm which incrementally adds clusters of variables to the objective function and solves the MAP problem in the dual space. They rank the clusters using a score function that measures the decrease in the dual value of the objective function when a cluster is added. We also derive our score functions similar to their approach by using the message update equations.

While the derivations in [78, 113] are provided for pairwise graphical models, we derive them for general networks of any order.

A.1 Linear Programming Relaxation of the MAP problem

Let $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$ be a set of binary variables and $\mathcal{C} = \{c : c \subset (1, 2, \ldots n)\}$ be a set of clusters. Consider a function $\Phi(\boldsymbol{x}; \boldsymbol{\theta})$ defined as a sum of the functions $\theta_c(x_c)$ defined over the clusters of variables. The goal of Maximum A Posteriori assignment (MAP) is to find an assignment that maximizes the function $\Phi(\boldsymbol{x}; \boldsymbol{\theta})$.

$$\arg\max_{\boldsymbol{x}} \Phi(\boldsymbol{x}; \boldsymbol{\theta}) = \arg\max_{\boldsymbol{x}} \sum_{c \in \mathcal{C}} \theta_c(x_c) \tag{A.1}$$

Let $\mathcal{S} = \{c \cap c' : c, c' \in \mathcal{C}, c \cap c' \neq \emptyset\}$ be the set of intersections between clusters and $\mathcal{S}(c) = \{s \in \mathcal{S} : s \subseteq c\}$ be the set of overlap sets for cluster c. The above problem can be reformulated as an integer program by introducing indicator variables $\mu_c(x_c)$ for each cluster, $\mu_s(x_s)$ for each intersection set between clusters and $\mu_i(x_i)$ for each variable.

$$\underset{\boldsymbol{\mu}}{\text{maximize}} \quad \sum_{c \in \mathcal{C}} \sum_{x_c} \mu_c(x_c) \theta_c(x_c) \tag{A.2}$$

$$\text{subject to} \quad \mu_c(x_c) \in \{0, 1\} \quad \forall c \in \mathcal{C} \tag{A.3}$$

$$\sum_{x_i} \mu_i(x_i) = 1 \quad \forall i \in \{1, \ldots, n\} \tag{A.4}$$

$$\sum_{x_{s \setminus i}} \mu_s(x_s) = \mu_i(x_i) \quad \forall s \in \mathcal{S}, i \in s \tag{A.5}$$

$$\sum_{x_{c \setminus s}} \mu_c(x_c) = \mu_s(x_s) \quad \forall c \in \mathcal{C}, s \in \mathcal{S}(c) \tag{A.6}$$

The constraint in Equation (A.6) enforces that the cluster indicator variables must be consistent with the intersection set indicator variable and the constraint in Equation (A.5) enforces the consistency of an individual variable with all the intersection sets that it is part of. The set of constraints on $\boldsymbol{\mu}$ denoted as $\mathcal{M}_L(\mathcal{C})$ is known as the marginal polytope. This problem is completely equivalent to the original problem A.1 and is hence as hard as the original problem. In many cases, this is NP-Hard and hence we obtain a linear programming relaxation by allowing the indicator variables to take on non-integer values i.e. replace the constraints as $\mu_c(x_c) \in [0, 1]$. The optimum of the relaxed problem gives an upper bound on the MAP value.

We will now find the dual problem of the relaxed linear program. Let $\lambda_{cs}(x_s)$ and $\lambda_{si}(x_i)$ be the dual variables corresponding to each of the constraints in Equation (A.6) and Equation (A.5) respectively. The constraint in Equation (A.4) will be kept implicit and used to simplify the Lagrangian later.

The Lagrangian is given by

$$\begin{aligned} L(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \sum_{c \in \mathcal{C}} \sum_{x_c} \mu_c(x_c)\theta_c(x_c) + \sum_{c \in \mathcal{C}} \sum_{s \in \mathcal{S}(c)} \sum_{x_s} \lambda_{cs}(x_s) \left[\mu_s(x_s) - \sum_{x_{c \setminus s}} \mu_c(x_c) \right] \\ + \sum_{s \in \mathcal{S}} \sum_{i \in s} \sum_{x_i} \lambda_{si}(x_i) \left[\mu_i(x_i) - \sum_{x_{s \setminus i}} \mu_s(x_s) \right] \end{aligned} \tag{A.7}$$

After rearranging the terms to group by common indicator variables, we get

$$
\begin{aligned}
L(\boldsymbol{\mu}, \boldsymbol{\lambda}) = \sum_{c \in \mathcal{C}} \sum_{x_c} \mu_c(x_c) &\left[\theta_c(x_c) - \sum_{s \in \mathcal{S}(c)} \lambda_{cs}(x_s)\right] \\
+ \sum_{s \in \mathcal{S}} \sum_{x_s} \mu_s(x_s) &\left[\sum_{c : s \in \mathcal{S}(c)} \lambda_{cs}(x_s) - \sum_{i \in s} \lambda_{si}(x_i)\right] \\
+ \sum_{i} \sum_{x_i} \mu_i(x_i) &\left[\sum_{s : i \in s} \lambda_{si}(x_i)\right] \qquad \text{(A.8)}
\end{aligned}
$$

We can now analytically maximize with respect to $\boldsymbol{\mu} \geq 0$ and the implicit constraint in Equation (A.4) to obtain the dual objective function,

$$
\begin{aligned}
J(\boldsymbol{\lambda}) &= \max_{\boldsymbol{\mu}} L(\boldsymbol{\mu}, \boldsymbol{\lambda}) \\
&= \sum_{c \in \mathcal{C}} \max_{x_c} \left[\theta_c(x_c) - \sum_{s \in \mathcal{S}(c)} \lambda_{cs}(x_s)\right] \\
&\quad + \sum_{s \in \mathcal{S}} \max_{x_s} \left[\sum_{c : s \in \mathcal{S}(c)} \lambda_{cs}(x_s) - \sum_{i \in s} \lambda_{si}(x_i)\right] + \sum_{i} \max_{x_i} \left[\sum_{s : i \in s} \lambda_{si}(x_i)\right] \qquad \text{(A.9)}
\end{aligned}
$$

The unconstrained dual program is now just

$$
\underset{\boldsymbol{\lambda}}{\text{minimize}} \quad J(\boldsymbol{\lambda}) \qquad \text{(A.10)}
$$

The above dual formulation is a simple extension of the technique adopted by D. Sontag [114] where they derive the dual of the LP relaxation for pairwise potentials. Another dual formulation, with constraints, can also be obtained by following the method of Globerson and Jaakkola [82].

A.2 Block Coordinate Descent in the Dual

A block coordinate descent strategy can be used to minimize the dual objective. At every iteration, the dual variables $\lambda_{cs}(x_s)$ are updated for one cluster while the rest are kept fixed. Similarly the dual variables $\lambda_{si}(x_i)$ are updated for one intersection set at a time while the rest are kept fixed. The update messages for the dual variables are given below.

From a cluster to one of its intersection sets:

$$\begin{aligned}\lambda_{cs}(x_s) = -\lambda_s^{-c}(x_s) - \sum_{i\in s}\lambda_{si}(x_i) \\ + \frac{1}{|\mathcal{S}(c)|}\max_{x_{c\setminus s}}\left[\theta_c(x_c) + \sum_{\hat{s}\in\mathcal{S}(c)}\lambda_{\hat{s}}^{-c}(x_{\hat{s}}) - \sum_{\hat{s}\in\mathcal{S}(c)}\sum_{i\in\hat{s}}\lambda_{\hat{s}i}(x_i)\right]\end{aligned} \tag{A.11}$$

where

$$\lambda_s^{-c}(x_s) = \sum_{\hat{c}\neq c:s\in\mathcal{S}(\hat{c})}\lambda_{\hat{c}s}(x_s) \tag{A.12}$$

From an intersection set to one of its variables:

$$\lambda_{si}(x_i) = -\lambda_i^{-s}(x_i) + \frac{1}{|s|}\max_{x_{s\setminus i}}\left[\sum_{c:s\in\mathcal{S}(c)}\lambda_{cs}(x_s) + \sum_{\hat{i}\in s}\lambda_{\hat{i}}^{-s}(x_{\hat{i}})\right] \tag{A.13}$$

where

$$\lambda_i^{-s}(x_i) = \sum_{\hat{s}\neq s:i\in\hat{s}}\lambda_{\hat{s}i}(x_i) \tag{A.14}$$

The derivation of the update messages can be found in Section A.5.

A.3 Upper Bound Score - Proof of Proposition 1

Proposition 3 (Upper Bound Score)**.** *An upper bound on the change in the MAP value after adding a cluster is given by*

$$\Delta\Phi \leq \frac{1}{|s|}\sum_{i\in s}\max_{x_i}\left(\max_{x_{cur\setminus i}}\Phi_{cur}(x_{cur}) + \max_{x_{new\setminus i}}\Phi_{new}(x_{new})\right) - \max\Phi_{cur}(x_{cur}) \quad \text{(A.15)}$$

where s is the set of nodes in the intersection of the sets x_{cur} and x_{new}.

Proof. In the block coordinate descent algorithm, during each iteration of the minimization procedure, the dual variables $\lambda_{cs}(x_s)$ are updated for one cluster c and all its intersection sets $s \in \mathcal{S}(c)$ while the rest are kept fixed. Similarly the dual variables $\lambda_{si}(x_i)$ are updated for one intersection set s and all the variables in this set $(i \in s)$ while the rest are kept fixed.

We calculate the scores for one cluster at a time while setting the dual variables for other clusters to zero. Let $\theta_f(x_f)$ be the cluster of potential functions of the *current* network and $\theta_g(x_g)$ be the cluster of potential functions of the *new* cluster. We start with all the dual variables set to zero. Since we consider only two clusters f and g, the number of intersection sets is just one i.e. $|\mathcal{S}(f)| = |\mathcal{S}(g)| = 1$. The first update (Equation A.11) is performed to the dual variable $\lambda_{fs}(x_s)$ while setting the rest of the dual variables to zero.

$$\lambda_{fs}(x_s) = \max_{x_{f\setminus s}}[\theta_f(x_f)] \quad \text{(A.16)}$$

This is followed by an update to the dual variable $\lambda_{gs}(x_s)$ given by

$$\lambda_{gs}(x_s) = -\lambda_{fs}(x_s) + \max_{x_{g \setminus s}}[\theta_g(x_g) + \lambda_{fs}(x_s)] = \max_{x_{g \setminus s}} \theta_g(x_g) \tag{A.17}$$

Finally we update (Equation A.13) the dual variables $\lambda_{si}(x_i)$

$$\lambda_{si}(x_i) = \frac{1}{|s|} \max_{x_{s \setminus i}} [\lambda_{fs}(x_s) + \lambda_{gs}(x_s)] \tag{A.18}$$

We now measure the value of the dual objective function before and after updating the dual variables. The dual objective function (Equation A.9) in our case is given by

$$\begin{aligned} J = &\max_{x_f} [\theta_f(x_f) - \lambda_{fs}(x_s)] + \max_{x_g} [\theta_g(x_g) - \lambda_{gs}(x_s)] \\ &+ \max_{x_s} \left[\lambda_{fs}(x_s) + \lambda_{gs}(x_s) - \sum_{i \in s} \lambda_{si}(x_i) \right] \\ &+ \sum_{i \in s} \max_{x_i} [\lambda_{si}(x_i)] \end{aligned} \tag{A.19}$$

When we initialize the dual variables to zero, the value of the dual objective function is

$$J^{(0)} = \max_{x_f} \theta_f(x_f) + \max_{x_g} \theta_g(x_g) \tag{A.20}$$

After performing one update for the dual variables $\lambda_{fs}(x_s)$ (Equation A.16) and

$\lambda_{gs}(x_s)$ (Equation A.17), we can see that

$$\max_{x_f}\left[\theta_f(x_f) - \lambda_{fs}(x_s)\right] = \max_{x_f}\left[\theta_f(x_f) - \max_{x_{f\setminus s}}[\theta_f(x_f)]\right] \leq 0 \tag{A.21}$$

$$\max_{x_g}\left[\theta_g(x_g) - \lambda_{gs}(x_s)\right] = \max_{x_g}\left[\theta_g(x_g) - \max_{x_{g\setminus s}}[\theta_g(x_g)]\right] \leq 0 \tag{A.22}$$

Also substituting for $\lambda_{si}(x_i)$ from Equation (A.18) gives us

$$\begin{aligned}&\max_{x_s}\left[\lambda_{fs}(x_s) + \lambda_{gs}(x_s) - \sum_{i\in s}\lambda_{si}(x_i)\right]\\ &= \max_{x_s}\left[\lambda_{fs}(x_s) + \lambda_{gs}(x_s) - \sum_{i\in s}\frac{1}{|s|}\max_{x_{s\setminus i}}\left[\lambda_{fs}(x_s) + \lambda_{gs}(x_s)\right]\right] \leq 0\end{aligned} \tag{A.23}$$

Hence the dual value after performing one update of the dual variables is

$$J^{(1)} \leq \sum_{i\in s}\max_{x_i}[\lambda_{si}(x_i)] \tag{A.24}$$

To avoid performing costly max-marginalization over the intersection set s to calculate $\lambda_{fs}(x_s)$ and $\lambda_{gs}(x_s)$, we can approximate $\lambda_{si}(x_i)$ as follows

$$\lambda_{si}(x_i) = \frac{1}{|s|}\max_{x_{s\setminus i}}\left[\lambda_{fs}(x_s) + \lambda_{gs}(x_s)\right] \tag{A.25}$$

$$\leq \frac{1}{|s|}\left(\max_{x_{s\setminus i}}\lambda_{fs}(x_s) + \max_{x_{s\setminus i}}\lambda_{gs}(x_s)\right) \tag{A.26}$$

$$= \frac{1}{|s|}\left(\max_{x_{f\setminus i}}\theta_f(x_f) + \max_{x_{g\setminus i}}\theta_g(x_g)\right) \tag{A.27}$$

We still need to perform max-marginalization, but only over one variable at a time.

This gives us a new upper bound on the dual value

$$J^{(1)} \leq \frac{1}{|s|} \sum_{i \in s} \max_{x_i} \left[\max_{x_{f \setminus i}} \theta_f(x_f) + \max_{x_{g \setminus i}} \theta_g(x_g) \right] \tag{A.28}$$

Since the dual value is an upper bound on the primal MAP value, we have

$$\max_x [\theta_f(x_f) + \theta_g(x_g)] \leq \frac{1}{|s|} \sum_{i \in s} \max_{x_i} \left[\max_{x_{f \setminus i}} \theta_f(x_f) + \max_{x_{g \setminus i}} \theta_g(x_g) \right] \tag{A.29}$$

Substituting for cluster $\theta_f(x_f)$ as $\Phi_{\text{cur}}(x_{\text{cur}})$ and $\theta_g(x_g)$ as $\Phi_{\text{new}}(x_{\text{new}})$ we can write an upper bound for the change in the primal MAP value after adding a cluster as

$$\Delta\Phi \leq \frac{1}{|s|} \sum_{i \in s} \max_{x_i} \left(\max_{x_{\text{cur} \setminus i}} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new} \setminus i}} \Phi_{\text{new}}(x_{\text{new}}) \right) - \max \Phi_{\text{cur}}(x_{\text{cur}}) \tag{A.30}$$

□

A.4 Blind Score - Proof of Proposition 2

Proposition 4 (Blind Score)**.** *A lower bound to the upper bound score (A.15) is given by*

$$\begin{aligned} score(g)_{upper} &\geq score(g)_{blind} && \text{(A.31)} \\ &= \frac{-1}{|s|} \sum_{i \in s} \left| \max_{x_i=0, x_{cur \setminus i}} \Phi_{cur}(x_{cur}) - \max_{x_i=1, x_{cur \setminus i}} \Phi_{cur}(x_{cur}) \right| && \text{(A.32)} \end{aligned}$$

where s is the set of nodes in the intersection of the sets x_{cur} and x_{new}.

Proof.

$$score(g)_{upper}$$

$$= \frac{1}{|s|} \sum_{i \in s} \max_{x_i} \left(\max_{x_{\text{cur}\setminus i}} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new}\setminus i}} \Phi_{\text{new}}(x_{\text{new}}) \right) - \max \Phi_{\text{cur}}(x_{\text{cur}}) \tag{A.33}$$

$$= \frac{1}{|s|} \sum_{i \in s} \max \left\{ \begin{array}{c} \left(\max_{x_{\text{cur}\setminus i}, x_i = 0} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new}\setminus i}, x_i = 0} \Phi_{\text{new}}(x_{\text{new}}) \right), \\ \left(\max_{x_{\text{cur}\setminus i}, x_i = 1} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new}\setminus i}, x_i = 1} \Phi_{\text{new}}(x_{\text{new}}) \right) \end{array} \right\}$$

$$- \max \Phi_{\text{cur}}(x_{\text{cur}}) \tag{A.34}$$

$$= \frac{1}{|s|} \sum_{i \in s} \delta_i \tag{A.35}$$

where

$$\delta_i = \max \left\{ \left(\max_{x_{\text{cur}\setminus i}, x_i = 1} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new}\setminus i}, x_i = 1} \Phi_{\text{new}}(x_{\text{new}}) - \max \Phi_{\text{cur}}(x_{\text{cur}}) \right), \right.$$
$$\left. \left(\max_{x_{\text{cur}\setminus i}, x_i = 0} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new}\setminus i}, x_i = 0} \Phi_{\text{new}}(x_{\text{new}}) - \max \Phi_{\text{cur}}(x_{\text{cur}}) \right) \right\} \tag{A.36}$$

Let us assume that the assignment to some $x_i = 1$ in $\max \Phi_{\text{cur}}(x_{\text{cur}})$. Then δ_i becomes

$$\delta_i = \max \left\{ \max_{x_{\text{new}\setminus i}, x_i = 1} \Phi_{\text{new}}(x_{\text{new}}), \right.$$
$$\left. \left(\max_{x_{\text{cur}\setminus i}, x_i = 0} \Phi_{\text{cur}}(x_{\text{cur}}) + \max_{x_{\text{new}\setminus i}, x_i = 0} \Phi_{\text{new}}(x_{\text{new}}) - \max_{x_{\text{cur}\setminus i}, x_i = 1} \Phi_{\text{cur}}(x_{\text{cur}}) \right) \right\} \tag{A.37}$$

We now assume that $\max_{x_{\text{new}\setminus i}, x_i = 0} \Phi_{\text{new}}(x_{\text{new}}) \geq 0$, which can be enforced by adding a positive offset to $\Phi_{\text{new}}(x_{\text{new}})$. A lower bound for δ_i is then

$$\delta_i \geq \left(\max_{x_{\text{cur}\setminus i}, x_i=0} \Phi_{\text{cur}}(x_{\text{cur}}) - \max_{x_{\text{cur}\setminus i}, x_i=1} \Phi_{\text{cur}}(x_{\text{cur}}) \right) \tag{A.38}$$

Since the maximizing assignment to $\Phi_{\text{cur}}(x_{\text{cur}})$ had $x_i = 1$, any other assignment with $x_i = 0$ must be less than the maxima. Hence,

$$\delta_i \geq - \left| \max_{x_{\text{cur}\setminus i}, x_i=0} \Phi_{\text{cur}}(x_{\text{cur}}) - \max_{x_{\text{cur}\setminus i}, x_i=1} \Phi_{\text{cur}}(x_{\text{cur}}) \right| \tag{A.39}$$

A similar argument can be made if the assignment to an $x_i = 0$. Hence we can put all the δ_i together to obtain a lower bound on the upper bound score

$$\begin{aligned} score(g)_{upper} &= \frac{1}{|s|} \sum_{i \in s} \delta_i & \text{(A.40)} \\ &\geq \frac{-1}{|s|} \sum_{i \in s} \left| \max_{x_{\text{cur}\setminus i}, x_i=0} \Phi_{\text{cur}}(x_{\text{cur}}) - \max_{x_{\text{cur}\setminus i}, x_i=1} \Phi_{\text{cur}}(x_{\text{cur}}) \right| & \text{(A.41)} \end{aligned}$$

□

A.5 Message Update Equations

Theorem 2. *The message update in Equation (A.11) for the dual variable $\lambda_{cs}(x_s)$ corresponds to block co-ordinate descent on the dual objective $J(\boldsymbol{\lambda})$.*

Proof. The proof follows from the ideas in the derivation of the optimality of the MPLP update from [113]. It shows that the value of the dual objective function reaches the minima in the variable λ_{cs} after performing a single update to it.

Consider fixing all $\lambda_{cs}(x_s)$ except for one cluster c. The part of the objective function that is dependent on the free variables is given by

$$\bar{J}(\boldsymbol{\lambda}) = \max_{x_c}\left[\theta_c(x_c) - \sum_{s\in\mathcal{S}(c)}\lambda_{cs}(x_s)\right] \\ + \sum_{s\in\mathcal{S}(c)}\max_{x_s}\left[\sum_{c:s\in\mathcal{S}(c)}\lambda_{cs}(x_s) - \sum_{i\in s}\lambda_{si}(x_i)\right] \tag{A.42}$$

Let

$$\lambda_s^{-c}(x_s) = \sum_{\hat{c}\neq c:s\in\mathcal{S}(\hat{c})}\lambda_{\hat{c}s}(x_s) \tag{A.43}$$

then $\bar{J}(\boldsymbol{\lambda})$ can be rewritten as

$$\bar{J}(\boldsymbol{\lambda}) = \max_{x_c}\left[\theta_c(x_c) - \sum_{s\in\mathcal{S}(c)}\lambda_{cs}(x_s)\right] \\ + \sum_{s\in\mathcal{S}(c)}\max_{x_s}\left[\lambda_{cs}(x_s) + \lambda_s^{-c}(x_s) - \sum_{i\in s}\lambda_{si}(x_i)\right] \tag{A.44}$$

$$= A_c(x_c) + \sum_{s\in\mathcal{S}(c)} A_s(x_s) \tag{A.45}$$

The lower bound on $\bar{J}(\boldsymbol{\lambda})$ is given by

$$\bar{J}(\boldsymbol{\lambda}) \geq \max_{x_c}\Bigg(\left[\theta_c(x_c) - \sum_{s\in\mathcal{S}(c)}\lambda_{cs}(x_s)\right] \\ + \sum_{s\in\mathcal{S}(c)}\left[\lambda_{cs}(x_s) + \lambda_s^{-c}(x_s) - \sum_{i\in s}\lambda_{si}(x_i)\right]\Bigg) \tag{A.46}$$

$$= \max_{x_c}\left(\theta_c(x_c) + \sum_{s\in\mathcal{S}(c)}\lambda_s^{-c}(x_s) - \sum_{s\in\mathcal{S}(c)}\sum_{i\in s}\lambda_{si}(x_i)\right) = B \tag{A.47}$$

If we apply the update messages in Equation (A.11) to $A_c(x_c)$, we get

$$A_c(x_c) = \max_{x_c} \left[\theta_c(x_c) - \sum_{s \in \mathcal{S}(c)} \lambda_{cs}(x_s) \right] \tag{A.48}$$

$$= \max_{x_c} \Bigg[\theta_c(x_c) + \sum_{s \in \mathcal{S}(c)} \lambda_s^{-c}(x_s) + \sum_{s \in \mathcal{S}(c)} \sum_{i \in s} \lambda_{si}(x_i)$$

$$- \frac{1}{|\mathcal{S}(c)|} \sum_{s \in \mathcal{S}(c)} \max_{x_{c \setminus s}} \left[\theta_c(x_c) + \sum_{\hat{s} \in \mathcal{S}(c)} \lambda_{\hat{s}}^{-c}(x_{\hat{s}}) - \sum_{\hat{s} \in \mathcal{S}(c)} \sum_{i \in \hat{s}} \lambda_{\hat{s}i}(x_i) \right] \Bigg] \tag{A.49}$$

$$\leq \max_{x_c} \left[\theta_c(x_c) + \sum_{s \in \mathcal{S}(c)} \lambda_s^{-c}(x_s) + \sum_{s \in \mathcal{S}(c)} \sum_{i \in s} \lambda_{si}(x_i) \right]$$

$$- \frac{1}{|\mathcal{S}(c)|} \sum_{s \in \mathcal{S}(c)} \max_{x_c} \left[\theta_c(x_c) + \sum_{\hat{s} \in \mathcal{S}(c)} \lambda_{\hat{s}}^{-c}(x_{\hat{s}}) - \sum_{\hat{s} \in \mathcal{S}(c)} \sum_{i \in \hat{s}} \lambda_{\hat{s}i}(x_i) \right] \tag{A.50}$$

$$= 0 \tag{A.51}$$

Similarly by applying the update to $A_x(x_s)$, we get

$$A_s(x_s) = \max_{x_s} \left[\lambda_s^{-c}(x_s) - \sum_{i \in s} \lambda_{si}(x_i) + \lambda_{cs}(x_s) \right] \tag{A.52}$$

$$= \max_{x_s} \Bigg[\lambda_s^{-c}(x_s) - \sum_{i \in s} \lambda_{si}(x_i) - \lambda_s^{-c}(x_s) - \sum_{i \in s} \lambda_{si}(x_i)$$

$$+ \frac{1}{|\mathcal{S}(c)|} \max_{x_{c \setminus s}} \left[\theta_c(x_c) + \sum_{\hat{s} \in \mathcal{S}(c)} \lambda_{\hat{s}}^{-c}(x_{\hat{s}}) - \sum_{\hat{s} \in \mathcal{S}(c)} \sum_{i \in \hat{s}} \lambda_{\hat{s}i}(x_i) \right] \Bigg] \tag{A.53}$$

$$= \frac{1}{|\mathcal{S}(c)|} \max_{x_c} \left[\theta_c(x_c) + \sum_{\hat{s} \in \mathcal{S}(c)} \lambda_{\hat{s}}^{-c}(x_{\hat{s}}) - \sum_{\hat{s} \in \mathcal{S}(c)} \sum_{i \in \hat{s}} \lambda_{\hat{s}i}(x_i) \right] \tag{A.54}$$

$$= \frac{B}{|\mathcal{S}(c)|} \tag{A.55}$$

Therefore

$$\bar{J}(\boldsymbol{\lambda}) = A_c(x_c) + \sum_{s \in \mathcal{S}(c)} A_s(x_s) \leq B \tag{A.56}$$

whereas we earlier showed that B is the lower bound on $\bar{J}(\boldsymbol{\lambda})$. Hence $\bar{J}(\boldsymbol{\lambda}) = B$ which implies that the update equation does indeed minimize the dual objective in the coordinates $\lambda_{cs}(x_s)$. □

Theorem 3. *The message update in Equation (A.13) for the dual variable $\lambda_{si}(x_i)$ corresponds to block co-ordinate descent on the dual objective $J(\boldsymbol{\lambda})$.*

Proof. Consider fixing all $\lambda_{si}(x_i)$ except for one intersection set s. The part of the objective function that is dependent on the free variables is given by

$$\begin{aligned}\bar{J}(\boldsymbol{\lambda}) &= \max_{x_s}\left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) - \sum_{i\in s}\lambda_{si}(x_i)\right] + \sum_{i\in s}\max_{x_i}\left[\sum_{s:i\in s}\lambda_{si}(x_i)\right] \qquad (A.57)\\ &= A_s(x_s) + \sum_{i\in s} A_i(x_i) \qquad (A.58)\end{aligned}$$

Let

$$\lambda_i^{-s}(x_i) = \sum_{\hat{s}\neq s: i\in\hat{s}} \lambda_{\hat{s}i}(x_i) \tag{A.59}$$

The lower bound on $\bar{J}(\boldsymbol{\lambda})$ is given by

$$\bar{J}(\boldsymbol{\lambda}) \geq \max_{x_s}\left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{i\in s}\lambda_i^{-s}(x_i)\right] = B \tag{A.60}$$

When we apply the update in Equation (A.13) to $A_s(x_s)$ we get,

$$A_s(x_s) = \max_{x_s} \left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) - \sum_{i\in s} \lambda_{si}(x_i) \right] \tag{A.61}$$

$$\begin{aligned} &= \max_{x_s} \Bigg[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{i\in s} \lambda_i^{-s}(x_i) \\ &\qquad - \frac{1}{|s|} \sum_{i\in s} \max_{x_{s\setminus i}} \left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{\hat{i}\in s} \lambda_{\hat{i}}^{-s}(x_{\hat{i}}) \right] \Bigg] \end{aligned} \tag{A.62}$$

$$\begin{aligned} &\leq \max_{x_s} \left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{i\in s} \lambda_i^{-s}(x_i) \right] \\ &\qquad - \frac{1}{|s|} \sum_{i\in s} \max_{x_s} \left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{\hat{i}\in s} \lambda_{\hat{i}}^{-s}(x_{\hat{i}}) \right] \end{aligned} \tag{A.63}$$

$$= 0 \tag{A.64}$$

Similarly by applying the update to $A_i(x_i)$, we get

$$A_i(x_i) = \max_{x_i} \left[\lambda_{si}(x_i) + \lambda_i^{-s}(x_i) \right] \tag{A.65}$$

$$= \max_{x_i} \left[\frac{1}{|s|} \max_{x_{s\setminus i}} \left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{\hat{i}\in s} \lambda_{\hat{i}}^{-s}(x_{\hat{i}}) \right] \right] \tag{A.66}$$

$$= \frac{1}{|s|} \max_{x_s} \left[\sum_{c:s\in\mathcal{S}(c)} \lambda_{cs}(x_s) + \sum_{\hat{i}\in s} \lambda_{\hat{i}}^{-s}(x_{\hat{i}}) \right] \tag{A.67}$$

$$= \frac{B}{|s|} \tag{A.68}$$

Therefore we get

$$\bar{J}(\boldsymbol{\lambda}) = A_s(x_s) + \sum_{i \in s} A_i(x_i) \leq B \tag{A.69}$$

But we showed that $\bar{J}(\boldsymbol{\lambda}) \geq B$. Hence $\bar{J}(\boldsymbol{\lambda}) = B$ and the update equation minimizes the dual objective in the coordinates $\lambda_{si}(x_i)$. $\square$

Bibliography

[1] VI Morariu and LS Davis. Multi-agent Event Recognition in Structured Scenarios. In *CVPR*, 2011.

[2] Stephen E Palmer. The effects of contextual scenes on the identification of objects. *Memory & Cognition*, 3:519–526, 1975.

[3] Antonio Torralba. Contextual priming for object detection. *International Journal of Computer Vision*, 2003.

[4] Aude Oliva and Antonio Torralba. The role of context in object recognition. *Trends in cognitive sciences*, 2007.

[5] Carolina Galleguillos and Serge Belongie. Context based object categorization: A critical survey. *CVIU*, 2010.

[6] Josep M Gonfaus, Xavier Boix, Joost Van de Weijer, Andrew D Bagdanov, Joan Serrat, and Jordi Gonzalez. Harmony potentials for joint classification and segmentation. In *CVPR*, 2010.

[7] Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011.

[8] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014.

[9] Antonio Torralba, Kevin P Murphy, and William T Freeman. Contextual models for object detection using boosted random fields. In *NIPS*, 2004.

[10] Andrew Rabinovich, Andrea Vedaldi, Carolina Galleguillos, Eric Wiewiora, and Serge Belongie. Objects in context. In *ICCV*, 2007.

[11] Santosh K Divvala, Derek Hoiem, James H Hays, Alexei A Efros, and Martial Hebert. An empirical study of context in object detection. In *CVPR*, 2009.

[12] MJ Choi, Antonio Torralba, and AS Willsky. A Tree-based Context Model for Object Recognition. *PAMI*, 34:240–52, 2012.

[13] Min Sun, Sid Yingze Bao, and Silvio Savarese. Object Detection using Geometrical Context Feedback. *IJCV*, August 2012.

[14] Dahua Lin, S Fidler, and Raquel Urtasun. Holistic Scene Understanding for 3D Object Detection with RGBD Cameras. In *ICCV*, 2013.

[15] Yukun Zhu, Raquel Urtasun, Ruslan Salakhutdinov, and Sanja Fidler. segdeepm: Exploiting segmentation and context in deep neural networks for object detection. In *CVPR*, 2015.

[16] Bangpeng Yao and Li Fei-Fei. Modeling mutual context of object and human pose in human-object interaction activities. In *CVPR*, 2010.

[17] Leonid Pishchulin, Mykhaylo Andriluka, Peter Gehler, and Bernt Schiele. Strong appearance and expressive spatial models for human pose estimation. In *ICCV*, 2013.

[18] Abhinav Gupta, Aniruddha Kembhavi, and Larry S Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[19] G. Gkioxari, R. Girshick, and J. Malik. Contextual action recognition with r*cnn. 2015.

[20] S Tran and L Davis. Event Modeling and Recognition using Markov Logic Networks. In *ECCV*, 2008.

[21] William Brendel, Alan Fern, and Sinisa Todorovic. Probabilistic Event Logic for Interval-based Event Recognition. In *CVPR*, 2011.

[22] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2014.

[23] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, 2014.

[24] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015.

[25] Andrej Karpathy, Armand Joulin, and Fei-Fei Li. Deep fragment embeddings for bidirectional image sentence mapping. In *NIPS*, 2014.

[26] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *CVPR*, 2015.

[27] Licheng Yu, Eunbyung Park, Alexander C. Berg, and Tamara L. Berg. Visual madlibs: Fill in the blank description generation and question answering. In *ICCV*, 2015.

[28] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, 2015.

[29] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. Referitgame: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014.

[30] Justin Johnson, Andrej Karpathy, and Fei-Fei Li. Densecap: Fully convolutional localization networks for dense captioning. In *CVPR*, 2016.

[31] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *CVPR*, 2016.

[32] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *CVPR*, 2016.

[33] Irving Biederman, Arnold L Glass, and E Webb Stacy. Searching for objects in real-world scenes. *Journal of Experimental Psychology*, 97(1):22, 1973.

[34] Bogdan Alexe, Nicolas Heess, YW Teh, and Vittorio Ferrari. Searching for objects driven by context. *NIPS*, 2012.

[35] Abel Gonzalez-Garcia, Alexander Vezhnevets, and Vittorio Ferrari. An active search strategy for efficient object detection. In *CVPR*, 2015.

[36] Mohamed R Amer, Dan Xie, Mingtian Zhao, Sinisa Todorovic, and Song-Chun Zhu. Cost-Sensitive Top-down / Bottom-up Inference for Multiscale Activity Recognition. *ECCV*, 2012.

[37] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015.

[38] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[39] Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Modeling context between objects for understanding referring expressions. In *ECCV*, 2016.

[40] Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Searching for objects using structure in indoor scenes. In *BMVC*, 2015.

[41] Varun K Nagaraja and Wael Abd-Almageed. Feature selection using partial least squares regression and optimal experiment design. In *IJCNN*, 2015.

[42] Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Feedback Loop between High Level Semantics and Low Level Vision. In *ECCV Workshops*, 2014.

[43] Emiel Krahmer and Kees van Deemter. Computational generation of referring expressions: A survey. *Computational Linguistics*, 38(1):173–218, 2012.

[44] Margaret Mitchell, Kees van Deemter, and Ehud Reiter. Natural reference to objects in a visual domain. In *INLG*, 2010.

[45] Jette Viethen and Robert Dale. The use of spatial relations in referring expression generation. In *INLG*, 2008.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997.

[47] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *NIPS*, 2003.

[48] Licheng Yu, Patric Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *ECCV*, 2016.

[49] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015.

[50] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, 2015.

[51] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *CVPR*, 2015.

[52] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

[53] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. In *ECCV*, 2016.

[54] Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. What are you talking about? text-to-image coreference. In *CVPR*, 2014.

[55] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2016.

[56] Margaret Mitchell, Kees Van Deemter, and Ehud Reiter. Two approaches for generating size modifiers. In *European Workshop on Natural Language Generation*, 2011.

[57] Nicholas FitzGerald, Yoav Artzi, and Luke S Zettlemoyer. Learning distributions over logical forms for referring expression generation. In *EMNLP*, 2013.

[58] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[59] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence - video to text. In *ICCV*, 2015.

[60] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[61] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[62] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014.

[63] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014.

[64] Ross Girshick. Fast R-CNN. In *ICCV*, 2015.

[65] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[66] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei a Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[67] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, 2004.

[68] Stéphane Ross, Geoffrey J Gordon, and J Andrew Bagnell. A Reduction of Imitation Learning and Structured Prediction. In *AISTATS*, 2011.

[69] Nathan Silberman, Pushmeet Kohli, Derek Hoiem, and Rob Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*, 2012.

[70] Saurabh Gupta, Ross Girshick, P Arbeláez, and J Malik. Learning Rich Features from RGB-D Images for Object Detection and Segmentation. In *ECCV*, 2014.

[71] Pablo Arbelaez, Jordi Pont-Tuset, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale Combinatorial Grouping. In *CVPR*, 2014.

[72] C. Lawrence Zitnick and Piotr Dollar. Edge Boxes : Locating Object Proposals from Edges. *ECCV*, 2014.

[73] Christoph H Lampert, Matthew B Blaschko, and Thomas Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[74] Nicholas J. Butko and Javier R. Movellan. Optimal scanning for faster object detection. *CVPR*, 2009.

[75] Hal Daumé III, John Langford, and Stephane Ross. Efficient programmable learning to search. *arXiv preprint arXiv:1406.1837*, 2014.

[76] Friedrich Pukelsheim. *Optimal Design of Experiments*, volume 50. Society for Industrial and Applied Mathematics, 2006.

[77] R. C. St. John and N. R. Draper. D-Optimality for Regression Designs: A Review. *Technometrics*, 17(1):15, February 1975.

[78] David Sontag, Talya Meltzer, and Amir Globerson. Tightening LP Relaxations for MAP using Message Passing. In *UAI*, 2008.

[79] M. Pawan Kumar and Daphne Koller. Efficiently Selecting Regions for Scene Understanding. In *CVPR*, 2010.

[80] Yingying Zhu, N Nayak, and A Roy Chowdhury. Context-Aware Activity Recognition and Anomaly Detection in Video. In *CVPR*, 2013.

[81] Chaitanya Desai, Deva Ramanan, and Charless Fowlkes. Discriminative Models for Multi-Class Object Layout. In *ICCV*, 2009.

[82] Amir Globerson and Tommi Jaakkola. Fixing Max-Product: Convergent Message Passing Algorithms for MAP LP-Relaxations. In *NIPS*, 2007.

[83] P Kohli and PHS Torr. Measuring Uncertainty in Graph Cut Solutions - Efficiently Computing Min-Marginal Energies Using Dynamic Graph Cuts. In *ECCV*, 2006.

[84] Matthew Richardson and Pedro Domingos. Markov Logic Networks. *Machine Learning*, January 2006.

[85] S Kok, M Sumner, M Richardson, and P Singla. The Alchemy System for Statistical Relational, 2009.

[86] Jan Noessner, Mathias Niepert, and H Stuckenschmidt. RockIt: Exploiting Parallelism and Symmetry for MAP Inference in Statistical Relational Models. In *AAAI Workshop: Statistical Relational Artificial Intelligence.*, 2013.

[87] Gurobi-Optimization-Inc. Gurobi Optimizer Reference Manual, 2013.

[88] Isabelle Guyon and Andre Elisseefi. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3(7-8):1157–1182, October 2003.

[89] P Geladi. Partial least-squares regression: a tutorial. *Analytica Chimica Acta*, 185(1):1–17, 1986.

[90] S Wold. PLS-regression: a basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 58(2):109–130, October 2001.

[91] DV Nguyen and DM Rocke. Tumor classification by partial least squares using microarray gene expression data. *Bioinformatics*, 18(1):39–50, 2002.

[92] Anne-Laure Boulesteix and Korbinian Strimmer. Partial least squares: a versatile tool for the analysis of high-dimensional genomic data. *Briefings in Bioinformatics*, 8(1):32–44, January 2007.

[93] Kim-Anh Lê Cao, Debra Rossouw, Christèle Robert-Granié, and Philippe Besse. A sparse PLS for variable selection when integrating omics data. *Statistical Applications in Genetics and Molecular Biology*, 7(1):Article 35, January 2008.

[94] Hyonho Chun and Sündüz Keles. Sparse partial least squares regression for simultaneous dimension reduction and variable selection. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 72(1):3–25, January 2010.

[95] Xiaofei He. Laplacian Regularized D-optimal Design for active learning and its application to image retrieval. *IEEE Transactions on Image Processing*, 19(1):254–63, January 2010.

[96] R.O. Duda, P.E. Hart, and D.G. Stork. Pattern Classification and Scene Analysis 2nd ed. 1995.

[97] M. Robnik-Sikonja and I. Kononenko. Theoretical and empirical analysis of ReliefF and RReliefF. *Machine learning*, 53(1):23–69, 2003.

[98] Reinaldo F. Teófilo, João Paulo a. Martins, and Márcia M. C. Ferreira. Sorting variables by using informative vectors as a strategy for feature selection in multivariate regression. *Journal of Chemometrics*, 23(1):32–48, January 2009.

[99] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of Bioinformatics and Computational Biology*, pages 1–8, 2005.

[100] Hanchuan Peng, Fuhui Long, and Chris Ding. Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–38, August 2005.

[101] Le Song, Alex Smola, Arthur Gretton, Justin Bedo, and Karsten Borgwardt. Feature selection via dependence maximization. *The Journal of Machine Learning Research*, 13(1):1393–1434, 2012.

[102] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[103] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, April 2005.

[104] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems*, volume 18, page 507, 2006.

[105] Xiaofei He, Ming Ji, Chiyuan Zhang, and Hujun Bao. A Variance Minimization Criterion to Feature Selection using Laplacian Regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2013–2025, March 2011.

[106] Sijmen De Jong. SIMPLS: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18(3):251–263, 1993.

[107] F. Giannessi, P.M. Pardalos, and Tamas Rapcsak. Optimization Theory: Recent Developments from Matrahaza. pages 124–125, 2002.

[108] RD Cook. A comparison of algorithms for constructing exact D-optimal designs. *Technometrics*, 22(3):315–324, 1980.

[109] Lieven Vandenberghe, Stephen Boyd, and Shao-Po Wu. Determinant Maximization with Linear Matrix Inequality Constraints. *SIAM Journal on Matrix Analysis and Applications*, 19(2):499, 1998.

[110] R.H. Tütüncü, K.C. Toh, and M.J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical programming*, 95(2):189–217, 2003.

[111] Pawel Smialowski, Dmitrij Frishman, and Stefan Kramer. Pitfalls of supervised feature selection. *Bioinformatics (Oxford, England)*, 26(3):440–3, February 2010.

[112] Deng Cai, Xiaofei He, and Yuxiao Hu. Learning a spatially smooth subspace for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2007.

[113] David Sontag, Amir Globerson, and Tommi Jaakkola. *Introduction to Dual Decomposition for Inference*. 2011.

[114] David Alexander Sontag. *Approximate Inference in Graphical Models using LP Relaxations*. PhD thesis, MIT, 2010.