LEARNING LANGUAGE–VISION CORRESPONDENCES

by

Michael Jamieson

A thesis submitted in conformity with the requirements for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

Copyright © 2010 by Michael Jamieson
Abstract

Learning Language–Vision Correspondences

Michael Jamieson

Doctor of Philosophy

Graduate Department of Computer Science

University of Toronto

2010

Given an unstructured collection of captioned images of cluttered scenes featuring a variety of objects, our goal is to simultaneously learn the names and appearances of the objects. Only a small fraction of local features within any given image are associated with a particular caption word, and captions may contain irrelevant words not associated with any image object. We propose a novel algorithm that uses the repetition of feature neighborhoods across training images and a measure of correspondence with caption words to learn meaningful feature configurations (representing named objects). We also introduce a graph-based appearance model that captures some of the structure of an object by encoding the spatial relationships among the local visual features. In an iterative procedure we use language (the words) to drive a perceptual grouping process that assembles an appearance model for a named object. We also exploit co-occurrences among appearance models to learn hierarchical appearance models. Results of applying our method to three data sets in a variety of conditions demonstrate that from complex, cluttered, real-world scenes with noisy captions, we can learn both the names and appearances of objects, resulting in a set of models invariant to translation, scale, orientation, occlusion, and minor changes in viewpoint or articulation. These named models, in turn, are used to automatically annotate new, uncaptioned images, thereby facilitating keyword-based image retrieval.
Acknowledgements

During my time at U of T, I was fortunate enough to have a great supervisory duo at my back: Sven Dickinson and Suzanne Stevenson. Even in our first meeting, Sven’s insight and infectious enthusiasm made a deep impression, convincing me to take the first steps on a long and eventful journey. These qualities, together with his apparently inexhaustible supply of big ideas, made him an excellent guide and teacher. Suzanne’s clarity of thought and commitment to scholarly excellence was invaluable on the long uphill stretches. I will always be grateful for their unstinting help and encouragement. I also thank my advisory committee, Allan Jepson and Rich Zemel, for their openness, thoughtful comments and for helping me out of some thorny underbrush. Thanks also go to my external examiner, Larry Davis, for his careful analysis and thought-provoking questions.

During the course of my study, I was fortunate enough to collaborate with several other highly talented individuals. Much of my thesis flows from Sven Wachsmuth’s earlier work as a post-doc at Toronto and he provided unstinting help and support as a co-author on most of my papers. Afsaneh Fazly spent an unreckonable number of hours helping me with my writing and I am privileged to have her as a friend and colleague. Geri Grolinger and especially Yulia Eskin made important contributions to my work, provided abundant support and helped me to understand my field from the other side of supervisory fence. My thanks also to all the other friends and colleagues who made my time at U of T so memorable and rewarding, including Fransisco Estrada, Alex Levenshtein, Palbo Sala, Diego Macrini and Stefan Mathe. Thanks also to office mates and good friends Chris Parisien and Ilya Sutskever for companionship and many intriguing conversations.

Finally, I am grateful to my parents, for supporting and believing in their son, The Eternal Student. It’s your fault for encouraging me in all these learning expeditions from such an early age!
Contents

1 Introduction 1
  1.1 Learning Language–Vision Correspondence ............... 3
  1.2 Language–Vision Correspondence in Context ............. 4
  1.3 The Language–Vision Correspondence Framework .......... 6
  1.4 Chronological development of our system ............... 7
    1.4.1 Unstructured Appearance Models .................. 8
    1.4.2 Structured Appearance Models ................... 9
    1.4.3 Probabilistic Appearance Models ................. 10
    1.4.4 Hierarchical Appearance Models ................. 11
  1.5 Contributions ........................................ 11

2 Related Work 13
  2.1 Language–Vision Correspondence ........................ 13
  2.2 Roots of Correspondence ................................ 15
    2.2.1 Content-Based Image Retrieval ................... 15
    2.2.2 Text-based Methods .............................. 17
  2.3 Correspondence in Context ............................... 19
    2.3.1 Minimizing Supervision ...................................... 21
    2.3.2 Capturing Configurations ............................ 23
    2.3.3 Encompassing Variations in Appearance ............ 25
4.2.1 Image Representation ............................................. 64
4.2.2 Appearance Model Representation .............................. 65
4.2.3 Energy Function .................................................. 66
4.2.4 Model Instance Detection Algorithm ............................. 67
4.3 Using Words to Learn Appearance Models ....................... 68
  4.3.1 A Measure of Word–Model Correspondence .................. 68
  4.3.2 Model Initialization ............................................. 69
  4.3.3 Iterative Improvement ........................................ 70
4.4 Using Models to Annotate New Instances ......................... 71
  4.4.1 Parameter Settings ............................................ 72
  4.4.2 Experiments on Toy Images .................................. 72
  4.4.3 Experiments on Web Data .................................... 73
4.5 Conclusions ......................................................... 78

5 Probabilistic Appearance Models ..................................... 79
  5.0.1 Background ..................................................... 80
  5.0.2 An Overview of Our Approach ................................ 81
5.1 Representing and Matching Objects ................................. 83
  5.1.1 Image Representation ........................................ 84
  5.1.2 Object Appearance Model ...................................... 86
  5.1.3 Detecting Instances of an Object ................................ 87
5.2 Discovering Word–Appearance Associations ....................... 94
  5.2.1 Word–Appearance Correspondence Confidence ................ 94
  5.2.2 Model Initialization ............................................ 96
  5.2.3 Iterative Improvement ........................................ 99
5.3 Evaluation: Annotating Objects in Uncaptioned Images .......... 101
  5.3.1 Experiments on a Controlled Data Set ......................... 102
  5.3.2 Experiments on Web Data Sets ................................ 106
5.3.3 Summary of Results .............................................. 120
5.4 Conclusions ......................................................... 120
5.5 Parameter Settings .................................................. 121

6 Hierarchical Appearance Models ........................................ 124
   6.1 Images, Parts and Multipart Models ................................. 125
   6.2 Discovering Parts .................................................. 126
      6.2.1 Model Initialization through Image Pair Sampling .............. 127
      6.2.2 Part Coverage Objective ...................................... 128
   6.3 Building Multipart Models .......................................... 130
      6.3.1 Detecting Duplicate Parts ...................................... 132
      6.3.2 Locating Part Detections ...................................... 132
      6.3.3 Choosing Initial Multipart Models .............................. 133
      6.3.4 Refinement and Expansion of Multipart Models ................. 133
      6.3.5 Detecting Multipart Models .................................... 134
   6.4 Results ............................................................ 135
      6.4.1 Experiments on the TOYS Data Set .............................. 136
      6.4.2 Experiments on the HOCKEY Data Set ........................... 139
      6.4.3 Experiments on the LANDMARK Data Set ....................... 141
   6.5 Conclusions ......................................................... 144

7 Conclusions and Discussion ............................................ 147
   7.1 Review of contributions ........................................... 147
      7.1.1 Novel Mechanisms in other Contexts ............................ 148
      7.1.2 Robustness to Visual Ambiguity ................................ 148
      7.1.3 Robustness to Unreliable Captions ............................. 149
   7.2 Discussion .......................................................... 150
      7.2.1 Acknowledging Connections Between Models .................... 150
7.2.2 Improving Scalability .................................................. 151
7.2.3 Moving Beyond Local Interest Points ................................. 153
7.2.4 Making Better Use of Language ....................................... 154
7.2.5 Building a Firmer Foundation for Grouping ....................... 155

Bibliography ......................................................................... 157
Chapter 1

Introduction

The world is awash in images. The rise of low-cost imaging, storage and communication technologies have increased our ability to capture imagery far beyond our ability to annotate and understand them using traditional techniques. We can make full use of only a small fraction of available images because manual annotation of new images is prohibitively expensive for commercial databases and overly time-consuming for the home photographer. Still, hundreds of millions of images are meaningfully associated with text in the form of captions or keywords. It is tempting to see these pairings of visual and linguistic representations as a kind of distributed Rosetta Stone from which we may learn to automatically translate between the names of things and their appearance. This situation has spurred a variety of efforts to learn valuable correspondences between the vision and language domains and thus enable image databases to be searched using both image features and keywords that describe their content.

The patterns of co-occurrence of words and visual features in annotated images can provide the evidence needed to establish meaningful links between visual and linguistic representations. However, this approach can only succeed to the extent that the words and visual features correspond to meaningful aspects of what is portrayed in the image. Regardless of the type of image feature used, a word typically refers not to a single feature, but to a configuration of features that form the object of interest. Therefore learning meaningful correspondences
involves a difficult perceptual grouping problem. The problem is particularly acute since any
given image may contain multiple objects or configurations; moreover, the meaningful con-
fignurations may be easily lost among a huge number of irrelevant or accidental groupings of
features. Without substantial bottom-up grouping hints, it is a nearly hopeless task to glean the
meaningful feature configurations from a single image–caption pair.

How might we find meaningful configurations of features in the context of the annotation
problem? Simple frequency of occurrence in the training data can provide a clue. If certain
collections of image features exist more often than can easily be explained by chance, they
may have a common, meaningful source. However, the number of such groupings may still be
prohibitively large. Even if they do arise from some common, recurring source in the image,
such groupings might have no corresponding word on the language side. This suggests a dual
approach: find recurring configurations of visual features which also have a significant level
of co-occurrence with specific words. Considering the modalities of language and vision to-
gether can make the perceptual grouping problem more tractable at the same time as it offers a
solution to the semantic association problem. Our system searches for meaningful feature con-
figurations that appear to correspond to a caption word. From these starting points it iteratively
constructs flexible appearance models that maximize word–model correspondence.

Our approach is best-suited to learning the appearance of objects distinguished by their
structure (_e.g._, logos or landmarks) rather than their color and texture (_e.g._, tigers or blue sky).
By detecting the portions of an object with distinctive structure, we can find whether the object
is present in an image and where (part of) the object appears. Our specific focus is on learning
correspondences between the names and appearances of exemplar objects from relatively noisy
and complex training data rather than attempting to learn the more highly-variable appearance
of object classes from less ambiguous training sets. However, our framework and the structure
of our appearance model are designed to learn to recognize any objects that appear as multiple
parts in a reasonably consistent configuration. We therefore believe that with the right choice
of features, our framework could be adapted to learn the appearance of object classes such as
cars, jets, or motorcycles.

1.1 Learning Language–Vision Correspondence

While many structured appearance models use features designed for object classes, our system uses features that are best suited to learning the appearance of exemplar objects (such as St. Paul’s Cathedral) rather than a broad class of objects (such as cathedrals in general). The world is full of exemplars, and there has been a great deal of work in sorting and annotating exemplar images, such as the method proposed by Simon et al. [75] for organizing collections of related photographs into labeled canonical views. While our current detection method is not as scalable as high-performance exemplar image retrieval systems such as that proposed by Philbin et al. [68], our use of language can improve text-based querying and link together widely different appearances or views of a single exemplar.

The goal of this work is to annotate exemplar objects appearing in images of cluttered scenes, such as the images shown in Figure 1.1(a-c). A typical such image, with hundreds (or even thousands) of local features, contains a huge number of possible feature configurations, most of which are noise or accidental groupings. A complex configuration of features that occurs in many images is unlikely to be an accident, but may still correspond to common elements of the background or other unnamed structures. The only evidence on which to establish a connection between words and configurations of visual features is their co-occurrence across the set of captioned images. The key insight is that this evidence can guide not only the annotation of complex feature configurations, but also the search for meaningful configurations themselves. Accordingly, we have developed a novel algorithm that uses language cues in addition to recurring visual patterns to incrementally learn strong object appearance models from a collection of noisy image–caption pairs (as in Figure 1.1(e-f)). The result of learning is a set of exemplar object appearance models paired with their names, which can be used for annotating similar objects in new (unseen and uncaptioned) images; sample annotations are
CHAPTER 1. INTRODUCTION

shown in Figure 1.1(h-j).

1.2 Language–Vision Correspondence in Context

A number of researchers have studied the problem of automatic image annotation in recent years [2,10,17,18,29,61,70]. Given cluttered images of multiple objects paired with noisy captions, these systems and others can learn meaningful correspondences between caption words and appearance models. Below, we briefly explain how our framework relates to other automatic image annotation systems and sketch our position in the broader object recognition landscape. These issues are explored in greater depth in chapter 2.

In many automatic annotation systems, the main component of the appearance model is a distribution over colors and textures. This kind of representation is a good fit for relatively structureless materials such as grass, sand or water and is relatively robust to grouping or segmentation errors. However, objects such as buildings and bicycles often lack a distinctive color or texture, and require representations that can capture a particular configuration of individually ambiguous parts. Most of these automatic annotation systems do not focus on learning such feature configurations. Often, appearance is modeled as a mixture of features (e.g., [18,61,70]) in which common part configurations are reflected in co-occurrence statistics but without spatial information.

In contrast, the broader object recognition literature contains many methods for grouping individual features into meaningful configurations and even arranging features into hierarchies of parts. For instance, Fergus et al. [34] and Crandall and Huttenlocher [25] look for features and relationships that repeat across a collection of object images in order to learn object appearance models consisting of a distinctive subset of features and their relative positions. While many object recognition techniques can learn appearance models from cluttered backgrounds, most require bounding boxes or training images in which a single object of the desired category occupies a large portion of the image. Unlike most automatic annotation work, few systems
Mats Sundin of the Toronto Maple Leafs misses a scoring chance against Ryan Miller of the Buffalo Sabres.

Toronto Maple Leafs vs. Montreal Canadiens.

Florida’s Olli Jokinen gets bumped by Alexei Ponikarovsky of the Maple Leafs.

Mats Sundin of the Toronto Maple Leafs misses a scoring chance against Ryan Miller of the Buffalo Sabres.

Toronto Maple Leafs vs. Montreal Canadiens.

Florida’s Olli Jokinen gets bumped by Alexei Ponikarovsky of the Maple Leafs.

Figure 1.1: Our method uses captioned images (a-c), each represented by thousands of local interest features (yellow crosses), and learns correspondences between configurations of features and caption words (d-f) to annotate new images (h-j).
are designed to learn from images without implicit or explicit object location information. It is especially rare for object recognition systems to learn from images containing multiple objects and multiple annotation words.

1.3 The Language–Vision Correspondence Framework

Though we have developed a number of different systems for learning word–appearance correspondences, our overall approach has remained relatively constant. Figure 1.2 illustrates the main components of our framework and our basic learning process. The system has two main stages: first, given a collection of image–caption pairs, we learn a set of corresponding appearance model–word pairs which we then use to annotate previously-unseen images. Word–appearance learning is composed of an initialization stage that finds ‘seed’ model–word pairs and an improvement stage that incrementally extends and refines each appearance model in order to maximize correspondence with it’s paired word. Annotation is a straightforward process of detecting appearance models in new images and attaching the corresponding word label. That being said, how we choose the confidence we assign to each labeling strongly affects overall annotation accuracy.

Since our goal is to learn meaningful correspondences between words and appearance, how we evaluate those correspondences has far-reaching effects. Our training data does not indicate where in an image a named object resides, so we must evaluate correspondence based on the distribution of words and appearance across the training image set. Word–appearance correspondence evaluation plays an important filtering role in the initialization process; it is the objective of model improvement and it influences our confidence when annotating new images.

The purpose of model initialization is to sift through the ocean of local features that represent an input image collection and identify recurring visual structure that shows some potential for strong word correspondence. Usually, the system identifies recurring visual structure and then filters the results by evaluating correspondence with words in the captions. The seed ap-
Figure 1.2: A pictorial representation of our system for learning the appearance of named objects and annotating new images in which they appear.

Appearance models produced by the initialization process are quite small (just individual local features in our first implementation) and are usually fairly unreliable and noisy object detectors.

The model improvement process grows small seed models into larger, more precise object detectors with strong word–model correspondence. We improve our models through an incremental search process that greedily tries to maximize correspondence strength. At each improvement iteration, we first propose a modification to the current model (based on the current set of detections and their context), then we detect the new model throughout the training set and evaluate whether the change has increased correspondence. Changes that improve correspondence are accepted, and the improvement cycle continues until none of the proposed changes leads to an improvement or an iteration limit is reached.

1.4 Chronological development of our system

While staying within the overall contours of the framework described above, over time we have replaced every major component of our system in order to improve annotation performance and deal with more challenging collections of captioned images. Most prominent among these
changes is the evolution of our object appearance model from a relatively simple local bag of features to a graph structure that captures the local configuration of features to a hierarchical graph that can integrate local patches of distinctive appearance over an entire object view. Of course, these changes required corresponding development in closely related areas, such as model detection and refinement. In parallel, we made a series of changes to the correspondence measure and our initialization approach in the quest to uncover all available word–appearance correspondences.

The central portion of this document (chapters 3 through 6) details the chronological evolution of our method. Below, we provide a summary of the most significant changes between the different iterations of our system. In order to preserve the readability of each chapter on its own, we have kept in place some explanatory material that may be redundant to someone reading the chapters in sequence. We hope the reader feels free to skim these sections and focus on the changes between each iteration of the system. Though each component of the framework changes, the relationships between components and the approach of iteratively amplifying correspondences between word and appearance, our overall strategy, endures.

1.4.1 Unstructured Appearance Models

Based on [45], chapter 3 describes the initial implementation of our automatic annotation system, referred to as the CVPR system. Here, we use an unstructured local ‘bag of features’ model to represent distinctive patches of appearance. As opposed to global bags of features that represent the distribution of features across the entire image, our models represent a set of features within a local neighborhood. Models start with a single feature and are refined by adding or removing features or adjusting the number of features required for detection. Following the influential work of Duygulu et al. [29], we use a translation model proposed by Brown et al. [15] to discover and measure associations between our appearance models and caption words. We captured a collection of cluttered scenes of children’s toys, ensuring that there were multiple named objects and several distractor words in each caption. Despite the noisy and unstructured
training data, and without the aid of bounding boxes, the system learned to recognize most of the named objects.

### 1.4.2 Structured Appearance Models

Chapter 4, based on [46], describes the next implementation of our approach, called the ICCV system. We address three limitations of our initial CVPR approach. First, in CVPR, our local bag-of-features models did not represent the *configuration* of their component features. By ignoring spatial information, a bag-of-features model can be fairly robust to articulation and perspective distortion but also relatively ambiguous. In our new method, we represent appearance as a graph of features where edges represent spatial relationships. We introduce an energy function to represent the fit between an observed set of features and an appearance model along with a learned per-model energy threshold that determines whether a set of features counts as a detection. Though we retained the overall strategy of incrementally refining our models in order to maximize word–model correspondence, a new appearance model naturally leads to changes in the refinement process. At each training iteration, the system tests either the addition of a new vertex or edge, or adjusts the per-model detection threshold, with changes accepted if they improve word–model correspondence.

The second major change from the initial system is in our approach to measuring word–model correspondence. The translation model of [15] solves for correspondence between all words and all elements of visual appearance in one stroke. In some cases, this highly-integrated approach bears fruit, as when it uses an appearance model’s correspondence with a background word to explain-away a correlation between the appearance and an object name. In other cases, it can be problematic, since it does not allow multiple related words to correspond to the same appearance. In addition, the translation model is computationally expensive, especially if the primary goal is to evaluate the correspondence strength of a single intermediate appearance model with a particular word. Therefore in chapter 4 we replace the translation model with a much faster correspondence score based on comparing the relative likelihoods of word–model
correspondence and independence.

Finally, in chapter 4 we introduce a new model initialization mechanism. The initial single-feature models of Chapter 3 ([45]) are often too ambiguous to be used as a starting point for iterative model refinement. We use an approach proposed by Sivic and Zisserman [78] to find clusters of *neighborhood contexts* that are potentially suitable for model initialization.

In addition to demonstrating the improved performance revised method on the TOYS image collection introduced in Chapter 3, we also successfully learned models from the much larger and more challenging HOCKEY collection of real-world NHL images downloaded from the web.

### 1.4.3 Probabilistic Appearance Models

Chapter 5 is based on a journal-length extension and refinement ([47]) of the material in chapter 4. Though the system described in this chapter, called PAMI, contains some important improvements over the ICCV system of chapter 4, we also describe many issues common to chapter 4 in more detail. Hence there is significant overlap between the two chapters and the reader may wish to focus on one or the other.

In the PAMI system, we replace the thresholded detection energy function of the ICCV system with a probabilistic framework. This has several advantages. First, we are able to explicitly account for background feature likelihood. Second, instead of a binary detection, the probabilistic model generates a detection confidence score in $[0, 1]$ which helps smooth learning. Finally, when confronted with a new image, we can smoothly combine detection confidence and correspondence confidence for better overall annotation performance. In addition, we modified the iterative refinement stage of the system to encourage development of more highly-connected appearance model graphs.

These changes lead to a significant improvement in annotation performance for the TOYS and HOCKEY captioned image sets. In this chapter we also perform some sensitivity analysis of system parameters and introduce a third image collection, LANDMARKS, composed of images
of famous structures throughout the world.

1.4.4 Hierarchical Appearance Models

While the appearance models learned by the PAMI system can be reasonably effective as detectors, they tend to cover only a relatively small, distinctive patch of an object’s appearance. As a whole, the appearance models may have good coverage of the underlying object, but in PAMI they are treated independently. This makes it difficult to determine the spatial extent of an object or to distinguish multiple objects from a single object with multiple distinct parts. In addition, because we do not consider their interactions, appearance models are poorly distributed among the views of an object, with less-popular views exhibiting poor coverage or not represented at all.

In chapter 6, we address the above concerns by introducing multipart models (MPMs) and proposing a new, more effective model initialization scheme. With MPMs, we extend the object appearance model into a hierarchy where each MPM represents a recurring configuration of several local ‘part’ appearance models. This allows us to improve annotation accuracy by integrating evidence from a set of part detections that may be individually ambiguous. MPMs have the same graph structure as lower-level parts and they can be trained using the same mechanism used to learn part models. In addition, we correct the distribution of part models across object views and significantly improve the overall recall of the system by introducing an initialization method based on detecting shared visual structure between selected training image pairs.

1.5 Contributions

We present a system for discovering meaningful correspondences between caption words and visual appearance that makes the following contributions:

- Unlike prior systems for discovering language–vision correspondence, we use word–
appearance co-occurrence not only to learn correspondences for existing visual elements, but to drive perceptual grouping and formation of new structured appearance models.

- We introduce a novel word–appearance correspondence measure based on the relative likelihood of common-source and independent explanations for observed word–appearance co-occurrences. By adjusting the parameters of this measure, we set the desired detection properties of our appearance models.

- We propose a graph-based appearance model that captures distinctive visual structure of objects, is invariant to translation, scale and orientation and robust to occlusion, articulation and small changes in perspective. We further extend these models to form new hierarchical appearance models that improve localization and increase annotation accuracy by integrating appearance information over a larger region of the image.

- We also introduce a novel detection method that can reliably find and annotate new instances of the learned object models in previously unseen (and uncaptioned) test images.

- Employing the above features, we develop a technique for automatically expanding and revising our structured and hierarchical appearance models in order to improve correspondence with caption words.

- Due to the above contributions, we are able to learn meaningful word–appearance pairs from complex, cluttered, real-world scenes with noisy captions, without the use of bounding boxes.
Chapter 2

Related Work

2.1 Language–Vision Correspondence

Over the past decade, low-cost imaging, storage and communication technologies have lead to an explosion in the number of available digital images. This wealth of available data has motivated the search for effective tools to organize large image collections, making them truly accessible. In some cases it is enough to organize collections using low-level characteristics of appearance such as color and texture. This approach, called Content-Based Image Retrieval (CBIR), has been extensively studied (see [79] for a survey) and the results are often striking. However, formulating a query in terms of such low-level characteristics is difficult and unintuitive for most users and the alternative of providing a query image that is close in appearance to the one they desire is often impractical. In any case, how the image looks is usually less important than what the image is about.

An image annotation in the form of a text caption or a set of keywords can provide an effective pointer to an image’s semantic content, but manual annotation of new images in large image collections is prohibitively expensive for commercial databases, and overly time-consuming for the home photographer. However, millions of images that are more-or-less meaningfully associated with text are already available from the web and existing annotated
image collections. It is tempting to see these pairings of visual and linguistic representations as a kind of distributed Rosetta Stone from which we may learn to automatically translate between the names of things and their appearances. Even limited success in this challenging project would support at least partial automatic annotation of new images, enabling search of image databases by both image features and keywords that describe their contents.

The field of language–vision correspondence targets collections of image–caption pairs to discover the meaningful relationships between the linguistic and visual representations of image content. This difficult task is made more difficult because an image–caption pair is typically composed of many visual and linguistic elements with no indication of the proper correspondence between the two. Even though the range and subtlety of the relationships is expected to fall short of human understanding, any uncovered meaningful relationships are valuable levers for smart, large-scale manipulation of image data.

This survey begins by examining some of the origins and context of language–vision correspondence. Section 2.2 describes the similarities and contrasts between language-vision correspondence and the related areas of content-based image retrieval and text-based association methods. The next section surveys the strengths and weaknesses of correspondence relative to the broader field of object recognition. Section 2.4 provides an overview of the various visual representations that have been used in language–vision correspondence papers, and some of the benefits and limitations of these design choices. The core focus of language–vision correspondence has been the statistical relationship between linguistic and visual representations of scene content. Section 2.5 describes the major approaches to this task. The next section covers the equally important question of how a system models relationships within the visual representation of a scene. Finally, section 2.6 summarizes our observations about the field as a whole.
2.2 Roots of Correspondence

The roots of language–vision correspondence lie not just in object recognition, but also in the fields of content-based image retrieval and text-based methods. The following subsections briefly sketch the tools, representations and perspectives that the field has inherited from work in content-based image retrieval and discuss the impact, on correspondence, of association techniques originally developed for use on text.

2.2.1 Content-Based Image Retrieval

In Content-Based Image Retrieval (CBIR) systems, users can search for images based on their visual content. Visual features extracted from images themselves supplement or replace so-called metadata such as timestamps, captions and keywords that may be insufficient to capture the user’s search concept or are simply unavailable. IBM’s QBIC (Query By Image Content) system by Flickner et al. [38] and Blobworld by Carson et al. [22] are prominent examples of CBIR systems. Smeulders et al. [79] provide an extensive survey of the development of CBIR up until the turn of the century.

Early CBIR systems [79] produced some striking results using very simple methods such as measuring the intersection of global color histograms, and this helped stimulate early exploration in the field. Eventually, interest began to focus more on addressing two limitations in CBIR. First, it can be awkward to create an effective query in a CBIR system. Since a CBIR system organizes images based on their visual features, a user must either construct a query in terms of characteristics such as color, texture and shape [38] (which can be unintuitive), or provide a query image similar to the one she wishes to retrieve (which is often impractical). More importantly, for most users, what an image looks like is less important than what an image is about. Smeulders et al. [79] called this lack of congruence between the information that a CBIR system extracts from an image and a user’s interpretation of that data the semantic gap. While closing the semantic gap entirely is a quite daunting proposition (to say the least),
an incremental approach to the project may be feasible. The chasm is not equally wide in all places, so that even simple visual representations can capture some useful concepts.

Language–vision correspondence can be seen in part as a response to these two obstacles to content-based retrieval. Learning some of the relationships between linguistic and visual representations of a scene allows a user to query with language even for images where useful annotations were not originally available. Thus users need not leap the gap themselves in order to communicate with the system. A clearer model of how language and visual representations intersect also simplifies support for mixed visual and word queries.

Some CBIR systems such as Blobworld [22] support such mixed queries, and the method proposed by Cascia et al. [23] presents concepts such as joint clustering of words and appearance and Latent Semantic Indexing that would later become popular approaches for finding links between words and image structures. Indeed, a number of techniques in the area are presented in the context of earlier work on content-based retrieval [2, 33, 48, 55, 61], and the eventual goal is not only annotation, but to enable a more semantic version of CBIR [18, 29].

It is therefore not surprising that many of the most widely-used visual representations and datasets for CBIR are also popular in the correspondence literature. color and texture descriptors (discussed in more detail in section 2.4.1) are the most popular visual features in both areas. Shape is typically represented by a few statistical measures such as elongation or ratio of area to perimeter (e.g., [24,29]) and usually has much lower weight in the feature vector than color and texture components. One significant advantage of color and texture representations over detailed shape models is that they tend to be less affected by grouping errors such as oversegmentation. It is difficult to avoid such errors in a ‘pure’ CBIR setting since the only cues for the correct groupings are local, bottom-up information and repetition of part configurations across an image set. In addition, it is hard to know what groupings may be important to the user until a query is received.

In the word–image correspondence setting, however, there may be less reason to avoid features that require grouping. The availability of language can make grouping more tractable
since it is known at training time that we would like to find groups with strong correspondence
to word labels. Furthermore, word correspondence may help guide grouping decisions, as
proposed in Barnard et al. [3] and Wachsmuth et al. [86]. This issue is discussed in more depth
in section 2.4.2.

Along with visual features, language-vision correspondence has inherited from CBIR some
of the more popular datasets, such as the Corel set of themed photo-CDs [63] (used in [2, 12, 18,
29, 48, 55, 61]). Some of the lessons from content-based retrieval provide grounds for caution
in this area, however. The Corel set is large and different researchers can use different subsets.
Muller et al. [63] describe how small changes to the training and testing image collections can
lead to dramatic improvements in apparent retrieval quality. Even if different algorithms are
compared on exactly the same image set, results can be highly misleading. For instance, the
color histograms of two images of an object taken from the same camera at roughly the same
time can be nearly identical, yet the same approach would fail to detect the same object under
different lighting conditions. From personal experience with a commercial CBIR system, the
perception that a retrieval algorithm is capturing meaningful image properties is often an illu-
sion based on peculiarities of the image set composition. Therefore results of correspondence
systems trained on these image sets must be interpreted cautiously.

2.2.2 Text-based Methods

In order to learn associations between words and visual appearance, many authors have adapted
learning methods originally developed for processing text data. One of the most prominent ex-
amples of such adaption is Duygulu et al.’s [29] use of the IBM translation model developed
by Brown et al. [15] to learn a translation between blob appearance and words. The translation
model is also used by Wachsmuth et al. [86], Barnard et al. [2] and Carbonetto et al. [17]. In an
early paper, Cascia et al. [23] use the Latent Semantic Analysis (LSA) method of Deerwester
et al. [28]. The probabilistic version of LSA (pLSA) introduced by Hoffman [43] has been
employed in [61, 76] to discover latent topics in image collections. Blei et al. [13] first intro-
duced Latent Dirichlet Allocation (LDA) for document modelling and text classification and it has been adapted for language–vision correspondence by Blei et al. [12], Barnard et al. [2] and Fei-Fei and Perona [31]. Finally, He et al. [41] use Lafferty et al.’s [52] Conditional Random Field (CRF) model to help label novel images.

Many of the methods above view an image with associated language elements like a text document written in two languages. Based on the results of the above work, this can be a very useful analogy. However, it is also important not to carry the analogy too far. An association model developed for use on text can bring with it implicit assumptions that may not be valid when applied to images. For instance, the translation model of Brown et al. [15] learns a probabilistic mapping between two fixed ‘vocabularies’ of discrete tokens. This discretization can be problematic even in the original text domain where tokens are usually individual words. This is because the meaningful correspondences between two languages can exist at the level of word configurations, rather than individual words.

While word-based quantization of a text document may imperfectly reflect the underlying semantics, the quantization of visual characteristics is often almost arbitrary. When applying the translation model in [29], Duygulu et al. use K-means clustering in the visual feature space to generate the discrete ‘visual words’ required by the model. This ‘visual vocabulary’ approach moves the problem of discovering the meaningful units of visual representation to a preprocessing stage and blocks the possibility of using word-correspondence as a guide to help find the correct units of representation (as discussed in [86]). Indeed, Barnard et al. [2] observe that this early quantization of the feature space independent of the language information appears to significantly hurt annotation performance. Errors in the discretization can be ameliorated if the association method can learn mixtures over visual words as in [61, 76], but any efficiency gains from early discretization have to be weighed against the amount of noise introduced into the learning process.
2.3 Correspondence in Context

A common strategy in language–vision correspondence is to use an unknown object’s context to help clarify its identity. This section takes a similar approach to bring correspondence itself into focus. We first examine the relationship between language–vision correspondence and the broader field of object recognition. The boundary between language–vision correspondence techniques and object recognition in general is sometimes difficult to discern. This may be because correspondence ultimately is the problem of object recognition, from a slightly different perspective. Barnard et al. [2] invite the reader to see “object recognition as a process by which one uses a huge data set to learn to put image structures and words in correspondence.” Instead of training a fixed number of object models, each with its own image collection, an algorithm should deal with images and associated text as it finds them and learn what correspondences it can. This shift in perspective emphasizes the ability to deal with training data that is both more ambiguous and much larger than normal for object recognition systems.

The emphasis on dealing with correspondence ambiguity in the training set is probably the most consistent difference between work on language-vision correspondence and other approaches to object recognition. Images and captions in large training collections are not usually selected for the purpose of recognition and therefore each image will often be associated with many objects and words. As it is usually too expensive to provide an explicit mapping between these text and image elements, the correspondences between words and image structures in any given image may be quite ambiguous. This is a significant departure from many recognition systems, such as [35, 88], where most training images are centered on a single object of the given class. On the other hand, not all annotation systems focus on resolving this ambiguity. He et al. [41] learn from a small number of per-pixel labeled images, rather than a large number of more loosely annotated images.

While a recognition system is often provided with definitive semantics in the form of consistent object labels, the text corresponding to an image in a large dataset might contain all the ambiguities and inconsistencies of natural language. Therefore the correspondence between
words and meaning may be just as ambiguous within a single image as the correspondence between words and appearance. Discovering correspondences between words and image structures provides an opportunity to reduce uncertainty in both domains: “while images and text are separately ambiguous, jointly they tend not to be” [2]. Algorithms that can unite words and appearances under latent topics (such as [2, 12, 61]) can reduce ambiguity in both parts of the representation. In addition, Barnard and Johnson [6] propose using correspondences between words and visual elements to help disambiguate word sense.

While many object recognition systems learn models for each object independently (e.g., [35, 88]), learning many word–image correspondences together provides an opportunity to exploit the many interactions between different caption words and between different objects in the images to help resolve ambiguity. Many correspondence systems such as [2, 12, 29, 48, 61] take this approach. On the other hand, training for one correspondence at a time, as in [18, 55], can be more efficient and simplify the learning of new correspondences.

Language-vision correspondence systems do tend to learn from larger data sets of 30-60,000 images [2, 18, 55], though these can seem rather small when compared to commercial databases containing millions of images, or the billions of images available on the web. While a much larger training set can provide the opportunity to learn many more meaningful correspondences, the text annotations for such large sets tend to be even more ambiguous and inconsistent. Also, even simple operations on large image collections can require a prohibitive amount of processing and storage resources. We can only speculate as to how well existing approaches would handle web-scale collections.

Also, while most word–image structure correspondence systems can be quite sophisticated in dealing with ambiguity, their visual representations are often surprisingly simple. While object recognition systems such as [25, 35, 88] make use of shape and configurations of parts, spatial information plays a much less prominent role in most correspondence systems. Exceptions to this trend tend to learn from more strongly annotated data [41, 87] or haven’t been demonstrated on large, real-image collections [86]. The ability to distinguish between spa-
tial image structures is probably critical to recognizing many common object classes. Thus there seems to be a need for learning mechanisms that can discover correspondences between words and configurations of visual features, while maintaining the ability to deal with annotation ambiguity and process large image–caption collections. Such models of configurations can be very hard to learn, however, which can undermine their effectiveness. For instance, the method proposed by Li and Wang [55] learns 2D multi-resolution hidden Markov models (2D MHMMs) that could in principle capture fairly complex spatial relationships. The approach proposed by Carneiro et al. [18] uses no spatial information, yet seems to perform better in most cases.

The following sections consider the relationship between language–vision correspondence and the broader object recognition work with respect to three important criteria: the required degree of supervision, the ability to represent configurations of features and robustness to variations in appearance.

2.3.1 Minimizing Supervision

Ideally, an object recognition system learns from widely-available information and does not need the link between concept and appearance to be highlighted through manual augmentation or filtering of training data. Language–vision correspondence systems tend to have an advantage in this important criterion since they are usually designed to learn in a relatively unsupervised fashion.

Why is it important that systems be capable of learning from relatively unstructured collections? First, consistently annotating large image collections by hand is tedious and difficult, especially as the set of relevant concepts expands. Efforts to make the annotation process more reliable and enjoyable, such as the ESP game developed by von Ahn [85] and the LabelMe database by Russel et al. [72] have met with significant success but the image sets they produce are still quite small compared to the number of annotated images on the web. In addition, as noted by Kennedy et al. [50], participants in the ESP game usually are not experts in the
subject of an image and so tend to describe the contents in relatively broad terms in order to achieve consensus. Kennedy et al. [50] demonstrated that much more specific descriptions can be reliably extracted by searching for consensus terms among images that are loosely-labeled but with high visual similarity. The second important reason to minimize supervision is that supervised data can contain many unintended biases. If it is capable of learning with less supervision, an algorithm could be applied to a large online source of annotated images and learn based on the full range of natural variation in appearance.

Object recognition systems vary dramatically in the degree of supervision they require. Some systems start with a detailed semantic map of each image, such a per-pixel object labels as in the methods proposed by Winn [87] and He et al. [41]. More common is to provide the label and location of the object of interest in each image in the form of a bounding box [1,27,32,37,67,74]. This information is particularly helpful when dealing with non-articulated objects viewed always from a particular aspect (e.g. [1,37]), since common parts and features can be expected to occupy approximately the same position in each subimage. Even when no bounding box is provided, training images can be selected so that the object of interest is large and centered in the frame (e.g. [35,90,91]) . In effect, the entire image is an implicit bounding box. Even systems that can learn from much noisier training image collections (e.g. [25,34]) tend to encode a subset of examples where the object of interest is large and centered with a canonical pose.

Language-vision correspondence systems such as [2,4,17,18,29] learn from less-structured input where objects appear in a variety of poses and scales with several annotation words and multiple objects of interest per image. Several methods of this type (e.g. [2,12,29,61]) are also designed to accommodate a certain amount of noise, ambiguity and inconsistent terminology in the labels assigned to each image. Exterior sources of prior knowledge such as Wordnet can also help resolve ambiguities in this type of data set, as in the system proposed by Hoogs et al. [44]. A system that can learn from such ambiguous data is better-suited to learn from the tremendous and growing number of annotated images available online.
Finally, it is worth noting that, over time, new sources of relatively unsupervised image information can become widely available as new technologies are widely adopted. For instance, just as the proliferation of digital cameras, video recorders and media-sharing sites have made a huge amount of annotated imagery available, integration of GPS systems into many of these devices has dramatically increased the availability of geo-tagged images. Several recent system (e.g. [56, 57, 89]) use this information to good effect to quickly cluster images of potential landmarks. Other automatically-acquired information such as temporal order can also dramatically improve the reliability of a recognition system, as in [57, 77, 78].

2.3.2 Capturing Configurations

In many systems designed to learn language-vision correspondences, the main component of the appearance model is a distribution over colors and textures (e.g. [2, 18, 29]), with little emphasis on spatial configuration. Often, appearance is modeled as a mixture of features (e.g., [18, 61, 70, 87]) in which common part configurations are reflected in co-occurrence statistics. However, these models contain no spatial relationships (even proximity) between parts that would allow them to represent true part configurations.

The ability to explicitly represent spatial structure is not always necessary to recognize structured objects. Images of different object classes can often be distinguished based on texture [69], or the distribution of low-level features [26, 87]. A feature distribution can often be more effective than relatively rigid models for object classes such as animals (e.g., Berg and Forsyth [10]) that have a consistent structure but are so variable in articulation and pose that the structure is difficult to discern in 2D images. At the other extreme, when matching particular views of highly-detailed exemplar objects, indexing on unstructured features can dramatically reduce the number of viable contenders [58, 68], sometimes even to the point of a unique match [65]. However, for the large number of recognition tasks for which low-level features are too ambiguous or unreliable, capturing distinctive configurations of features is vital.

One way to represent spatial configurations is to copy the raster organization of the image
itself. With features such as the histogram of oriented gradients (HOG) [27] calculated over
the image in a regular grid, an object detector can be evaluated at each location of interest in
a sliding-window fashion. Scale-invariance can be achieved by generating a feature pyramid,
and features of different scales can be integrated by selectively including multiple levels of the
pyramid in an appearance model (e.g., Lazebnik et al. [53], Felzenszwalb et al. [32]). Viola and
Jones [84] demonstrated how some sliding detectors can be made quite fast by amortizing the
cost of feature extraction across positions and scales and taking a fail-early approach to each
detection.

An alternative to arranging features in a fixed grid (or subsets thereof) is to represent mean-
ingful configurations as a graph, where vertices represent relatively local features and edges
represent spatial relationships. For instance, Fergus et al. [34] learn object appearance models
consisting of a distinctive subset of local interest features and their relative positions, by look-
ing for a subset of features and relationships that repeat across a collection of object images.
Crandall and Huttenlocher [25] use graph models in which vertices are oriented edge templates
rather than local feature detections, and edges represent spatial relationships. This type of rep-
resentation is sufficiently flexible to handle occlusion, minor changes in scale and viewpoint,
and common deformations.

A natural strategy to improve the flexibility and robustness of such models is to organize
the object representation as a part hierarchy (e.g., [14, 30, 51, 66, 91]). The part hierarchy can
be formed by composing low-level features into higher and higher-level parts (e.g., Kokkinos
and Yuille [51]) or by decomposing larger-scale shared structures into recurring parts (e.g.,
Epshtein and Ullman [30]). The composition and learning method of parts at different levels
of the hierarchy may be highly similar (e.g., Bouchard and Triggs [14]) or heterogeneous (e.g.,
Ommer and Buhmann [66]). An interesting extension of the hierarchical approach is to intro-
duce grammar-like composition rules (e.g., Zhu and Mumford [92], Zhu et al. [90]) that can
capture the complex interrelationships between parts that are present in many object classes
and scenes.
2.3.3 Encompassing Variations in Appearance

The success of an object recognition system is determined in large part by the degree to which it can represent distinctive features of the objects of interest while remaining robust to variations in their appearance. A wide variety of factors can dramatically affect an object’s appearance, such as differences in lighting, viewpoint, deformation or articulation of the object itself and within-class variation. Features that are robust and distinctive for one type of object usually rely on assumptions that make them unsuitable for other categories. It is harder than it might seem to evaluate recognition systems with respect to this criteria because each approach makes different trade-offs, achieving robustness to certain types of variation by accepting sensitivity to others.

For instance, language–vision correspondence systems that rely heavily on color and texture (e.g., [2, 17, 18, 29]), are robust to deformation or articulation, but potentially quite sensitive to changes in lighting and are focused on object classes that lack large variations in surface appearance. Exemplar recognition systems based on local interest points such, as the method proposed by Nister and Stewenius [65] or Philbin et al. [68] (or our own work), can extract very distinctive features over a range of lighting conditions, deformations and poses, but critically assume that the objects of interest are exemplars with (mostly) identical surface appearance. Landmark recognition systems such as Li et al. [56] or Snavely et al. [80] go a step further and exploit the rigidity of their target object category to verify matches based on epipolar geometry. Restricting objects of interest to a particular object class, such as faces (e.g., Berg et al. [9]), is another effective strategy because it allows the algorithm to apply more domain knowledge, focus on the aspects of the object’s appearance that are most robust, and exploit its characteristic structure. Systems designed to recognize several different object classes often trade a degree of robustness to variation in surface appearance for additional sensitivity to pose and viewpoint (e.g., [25, 30, 34, 66]).

Unfortunately, all the above systems are still based to some degree on detecting correspondences in surface appearance. Contour-based features are also sensitive to pose and viewpoint,
but more robust to large changes in surface appearance (provided that the true shape contours are not too lost among appearance edges). Contour-based representations have a long history in view-based object recognition (e.g., [7, 11, 64, 81, 83]) and there has been a significant amount of research in this area in the past several years [1, 36, 37, 67, 74]. In contrast to many popular features, constructing distinctive contours usually requires significant bottom-up grouping. While keypoints can be reasonably distinctive even at the atomic level, the most basic component of a contour is an edgel, which by itself is not distinctive at all. Even groups of edgels that form short, straight or curved line segments are common across almost all images and therefore not particularly helpful by themselves. The rules used to group the nearby contour fragments are crucial and might include Gestalt concepts such as co-termination, co-linearity and symmetry. Alternatively, one might encode the layout of all contours within some context region and decide (through provided spatial correspondence with other objects of the same class) which parts of the context are important and which should be ignored (as in [1]).

2.4 Visual Representations for Correspondence

This section describes the main visual representations used in language–vision correspondence systems. The first subsection describes the common types of features underpinning visual representations in correspondence systems. Subsection 2.4.2 surveys different ways of choosing the image regions over which features are computed. Finally, the different approaches to representing relationships among these image regions are discussed in subsection 2.4.3.

2.4.1 Features

Most language-vision correspondence systems use a feature vector that combines color, texture and shape features. Typically, most of the feature vector dimensions are devoted to color or texture while shape has the fewest [2, 17, 18, 29]. A few methods don’t use color at all, and focus on texture (as in [31, 76]) or shape [86].
color is represented in a variety of color spaces, including RGB, rgS, YUV, HSV, L*a*b* and in one case several color spaces at once [2]. Barnard et al. [3] found that strong correlation among the components of an RGB representation undermined its effectiveness as a descriptor. The distribution of color may be represented by the mean and variance [29] or by a color histogram [61]. Chang et al. [24] model the separate contributions of eleven distinct ‘cultural colors’ plus a twelfth category for unusual colors. One drawback specific to color is that it can vary tremendously with lighting conditions. Barnard and Gabbur [5] found that applying a simple color-constancy preprocessing step could improve correspondence results, but this approach has not been widely adopted.

A variety of techniques has been employed to capture texture, including oriented filter responses [2, 17, 29, 41], wavelet coefficients [24, 55] and discrete cosine transform (DCT) coefficients [18]. SIFT descriptors [58] have also been used to capture local structure in [31, 61, 76]. Simple normalized grayscale patches can also be effective [31].

Shape descriptors are only used by systems that segment each training image according to its content, such as [2, 17, 24, 29, 86]. In most of these systems, the shape features are statistical properties of the region layout such as area as a fraction of the image, the ratio of area to perimeter squared, and moment of inertia. Barnard et al. [3] find that a more detailed region shape description leads to lower correspondence performance in their system, likely due to inconsistent segmentation. Wachsmuth et al. [86] use a shock graph representation for shape and propose to use word–shape correspondence as a guide to correct segmentation errors. However, the effectiveness of this approach has not been evaluated on real images.

One of the more striking aspects of feature design in language-vision correspondence is how little attention it seems to receive. Discussion of the visual features underpinning the learned visual structures is usually confined to a single short paragraph, or in Blei et al. [12], a single sentence: “For each region, we compute a set of real-valued features representing visual properties such as size, position, color, texture and shape.” Yet the choice of low-level features can strongly affect the design and performance of the whole system. Emphasizing
color and texture features insulates the visual representation from perceptual grouping errors, but does this choice circumscribe the class of objects that can be recognized? Intuitively it would appear that recognizing many classes of objects requires an ability to form meaningful groups, but perhaps this grouping need not occur at the feature-extraction level.

### 2.4.2 Parts

While the system proposed by Cascia et al. [23] relies exclusively on global image features, most language-vision correspondence systems divide the image into parts. The three main approaches are to extract features on a fixed grid, over-segmented regions, or at interest-point detections.

Among the surveyed correspondence papers, the most popular approach is to segment the image using an off-the-shelf algorithm, such as normalized cuts [73], and extract features from the larger segments. Chang et al. [24] segment the image according to fixed ‘cultural colors’, then describe color distribution, texture and shape properties for each cultural color block. The major drawback to the segmentation approach is inconsistent segmentation across images with similar content. While in most cases these errors are simply tolerated, Barnard et al. [3] and Wachsmuth et al. [86] propose different techniques for using correspondence information to correct segmentation errors. In [3], adjacent regions may be merged if their distributions over corresponding words are sufficiently similar. This raises the possibility of restoring objects that are often over-segmented using bottom-up techniques, such as a penguin with a black back and a white belly. However, each over-segmented part of the object must be distinctive enough to capture the proper semantics. So, for instance, the white penguin’s belly cannot be too strongly associated with ‘snow’, ‘ice’ or ‘polar’. In a similar vein, Quattoni et al. [70] use the co-occurrence of caption words and visual features to merge together “synonymous” features. This lowers the dimensionality of their bag-of-features image representation and therefore allows image classifiers to be trained with fewer labeled examples. Wachsmuth et al. [86] propose to test merges of adjacent regions and keep those that improve the correspondence between
The second most common technique for dividing an image into regions is a fixed rectangular grid. Most systems that take the fixed grid approach use disjoint regions (as in [62]) rather than overlapping patches [18]. Features may be extracted at a single scale [12] or a fixed collection of scales [55]. None of the systems that take this approach encode shape-based features, though it is feasible to use a shape-context-like descriptor [8]. The grid method is often selected over segmentation for reasons of efficiency, through Carbonetto et al. [17] find that a rectangular-grid version of their algorithm also generates better correspondences than the version using normalized cuts.

Methods employing interest-point detectors have been increasingly popular over the last five years, and some recent correspondence papers have experimented with this technique, including Fei-Fei and Perona [31] and Monay et al. [61]. Interest point detection is also popular among object recognition systems with some correspondence-like properties, such as [34, 76]. While this approach can lead to better scale and translation invariance, the number of interest points detected is highly variable and they may not adequately cover semantically meaningful parts of the image. Fei-Fei and Perona [31] find that a dense rectangular sampling produced stronger correspondence results.

Overall, a fixed division into parts without even considering the underlying image seems to generate the best correspondence results among current systems. Perhaps part of the problem is that grouping is usually considered complete before learning begins. This constrains the search for meaningful parts to strictly bottom-up techniques. An efficient method along the lines of [3] or [86] that can use the correspondence between words and image structures to make better grouping decisions might make perceptual groups a help rather than a hindrance.
2.4.3 Part Relationships

Many types of objects do not appear as a single uniform color and texture, but as a configuration of distinct parts. Even when objects do have a roughly uniform appearance, meaningful relationships with other image elements can provide an important contextual clue to the object’s identity. The ability to represent such meaningful relationships between the parts of an object model, and between an object and its surroundings might significantly affect the types of word–image structure correspondences that can be discovered. Current correspondence systems span a spectrum from almost completely independent parts to models that can represent strong spatial relationships across a range of scales. Most of the surveyed systems, however, model only very weak relationships among parts.

Many systems look for meaningful correspondences between words or concepts and unstructured mixtures of features. For instance, Carneiro et al. [18] propose associating conceptual labels with a hierarchical mixture of Gaussian distributions in the visual feature space. Another popular technique is to model image structure as a bag of discrete visual words, as in [31, 48, 61, 76]. This type of approach can loosely model the co-occurrence of distinct parts of an object and co-occurrence relationships between the object and background. However, there is no constraint that elements of the mixture occur in some particular spatial configuration or even in close proximity. Sivic et al. [76] note that while such bag-of-features representations contain no explicit spatial information, they can still be sensitive to spatial re-ordering if the feature support regions overlap (“randomly shuffling bits of the image around will almost certainly change the bag of words description”).

The translation model correspondence system proposed by Duygulu et al. [29] and described again in Barnard et al. [2] also has very weak part relationships. When annotating a new image, each segmented region is independently associated with a distribution over possible annotation words based only on its extracted feature vector. Therefore context and part relationships are only a factor in annotation for a particular blob when they are included as part of each segment’s internal representation. For instance, Barnard et al. [3] describe a color
context feature that computes the average color adjacent to a region in four directions.

Carbonetto et al. [17] augment the translation system by introducing a Markov random field model. A translation potential describes the compatibility between an image region and a word label, while a spatial context clique potential describes the compatibility of word labels for adjacent regions. Therefore a region with the label ‘bear’ is likely to be adjacent to a region with the label ‘trees’, but unlikely to be adjacent to a region labeled ‘astronaut’. Carbonetto et al. find that taking into account neighbour relationships in this way improved their correspondence results, but tended to over-smooth the labels. A region of sky adjacent to a bird in flight might also be labeled ‘bird’, for instance.

Li and Wang [55] attempt to learn a much more complex two-dimensional multiresolution hidden Markov model (2D MHMM) for each semantic concept. Each image is described at three resolutions organized as a 3-level quadtree of 1, 4 and 16 rectangular blocks. Each block is associated with an observed color and texture feature vector and a hidden discrete state. Each feature vector is independent of the rest of the model, given the state. The state of each block is independent of other states, given its parent, adjacent siblings’ and children’s states. Each state is associated with a Gaussian distribution over feature vectors and a spatial distribution over child states, and the model was trained using the EM algorithm. While the algorithm is capable of generating reasonable results on novel images, overall performance is not usually as strong as the approach proposed by Carneiro et al. [18], which does not include any explicit spatial relationships. This may be due to the large number of model parameters allowing it to overfit the training data.

The method proposed by He et al. [41] adapts the conditional random field (CRF) [52] to model the conditional probability of labels given images. Modeling this conditional distribution as opposed to a full generative model (as in [55]) focuses modeling resources on the task of inferring semantic labels from an image. The multiscale CRF framework has three types of components that operate at the local, regional and global scales. The local neural network classifier calculates a distribution over labels given a feature vector. Regional features represent
the spatial boundaries between object classes while global features encode regularities in the overall layout of object classes. While the system is able to learn complex spatial relationships from a very small number of training images, the results are not comparable with other systems in the survey as the training images have per-pixel labels.

Several papers discuss the importance of taking into account an object’s context in order to achieve better recognition performance [3, 17, 41, 61]. When dealing with highly ambiguous training image labels, however, sometimes it seems that the harder problem is separating an object from its context. For instance, the system proposed by Carneiro et al. [18] makes no attempt to separate the appearance of the trained object in any given image from the appearance of the background. The method relies on observing the object in a variety of contexts to build up probability mass on the actual object appearance. Thus if many of the bear training images show bears in the woods, then part of the ‘bear’ mixture model will be green and leafy. Similarly in [55], there is no difference between parts of the image due to the trained concept and the background. While there is a semantic difference between an object and its context, it does not seem important in the current test sets to encode that distinction in the model.

2.5 Models of Language–Vision Relationships

The core of a language–vision correspondence system is how it models the relationship between words and image structures. This section describes the most common methods, their advantages and drawbacks, and their impact on other parts of the system. Important considerations are whether the system can model relationships among words, how it deals with correspondence ambiguity and whether the framework supports the introduction of complex appearance models.
2.5.1 One Model per Concept

The most straightforward method for associating words and image structures is to train a visual model for each word or concept of interest. This is essentially the default approach in object recognition (e.g., [25, 35, 88]) and it has also been successfully applied in correspondence systems proposed by Carneiro et al. [21] and Li and Wang [55].

The method described in [21] learns a feature-space distribution for each word \( w \) in the annotation vocabulary. The system extracts a dense, unordered set of feature vectors from each image whose annotation includes \( w \), then uses the resulting point set to fit a hierarchical mixture of Gaussian distributions. When annotating a new image, the system extracts feature vectors and assigns the 5 words with the highest posterior probability as the annotation.

In [55], the training set is a collection of 600 Corel photo-CDs, where the theme of each CD is manually assigned a set of descriptive keywords such as “ruin, historical building, landmark”, or “fashion, people, cloth, female”. Since not all the assigned words properly apply to each image within a theme, the annotation data is fairly noisy and ambiguous. A separate generative model is learned from 40 images from each theme. When given a new image, the system finds the learned themes with the highest likelihood of generating the image. Annotation words for the new image are drawn from the theme descriptions, with a preference for words that are unusually common among the high-likelihood themes relative to their prevalence across all themes. Therefore these fixed themes play a similar role to the automatically-discovered latent topics described in subsection 2.5.4.

These approaches do not attempt to resolve the correspondence ambiguity between word labels and image elements in any given image–annotation pair. One system [21] relies on a large number of training examples to accumulate probability mass on the right visual features for each word while another [55] relies on a large number of theme models to raise the profile of the right word. Since words and concepts are trained independently, there is no opportunity to consider possible interactions among scene labels during training. On the other hand, overall annotation quality can be quite competitive, adding a new concept to the vocabulary.
is relatively simple, and the approach allows for complex appearance models when sufficient training data is available.

2.5.2 Joint Distribution of Words and Visual Features

A number of correspondence systems, such as [33, 48, 62], attempt to learn the joint word–feature distribution. The system proposed by Mori et al. [62] divides each training image into rectangular regions and associates each region feature vector with all the words in the image caption. The visual feature space is then clustered, and each cluster accumulates a multinomial word distribution from its member feature vectors. This is essentially the reverse of [21], as Mori et al. [62] calculate a word distribution for each visual category instead of a visual distribution for each word. One potential drawback of this approach is that the system will tend to merge concepts with similar appearance but very different word associations.

By contrast, Feng et al. [33] and Jeon et al. [48] use a non-parametric approach to model the joint word–feature distribution. Each training image is associated with both a visual distribution and a word distribution. The overall joint word–feature distribution is the average of the training image distributions. Feng et al. [33] model each training image’s visual characteristics as a mixture of Gaussian distributions and the annotation word set as a multiple-Bernouli distribution. Jeon et al. [48] use a smoothed multinomial distribution to represent both the distribution over ‘visual words’ (quantized region descriptors) and vocabulary words for each training image.

The major drawback of these non-parametric models is the potentially huge number of model parameters. The space required to store the learned model and the time required to estimate word labels for a previously-unseen image are both linear in the size of the training set. From a research standpoint, it can also be hard to gain insight into what types of word–image relationships the model encodes, as these relationships are scattered across a large number of training image distributions. Finally, it is difficult to see how to incorporate more complex configurational models of appearance in this type of framework.
2.5.3 Translation Model

The translation model is probably the most widely-cited approach to language–vision correspondence. In this view, finding the correct mapping between the linguistic and visual descriptions of a dataset is analogous to learning a translation between two languages. Each image–caption pair is treated like a sentence written in two different languages. The ambiguity in word–image structure correspondence is analogous to ambiguity in word-level alignment. Brown et al. [15] propose a model in which each word in one language (English) is associated with a distribution of words a second language (French). Together, these distributions form a translation table. Each word in a French sentence is a sample from the distribution of one of the words in the corresponding English sentence (one column of the translation table) or else are generated by the implicit NULL word. Brown et al. [15] use the EM algorithm to find the optimal translation table for a given set of French–English sentence pairs.

Duygulu et al. [29] replace the English vocabulary in this scheme with a ‘visual vocabulary’ of quantized region descriptors. Thus, a certain appearance is associated with a particular distribution over annotation words and the EM algorithm can be used to find the optimal translation between appearance and words. The same approach is also used in [2, 3, 17, 86]. One significant advantage of this framework is the potential known word–appearance pairs to ‘explain away’ some of the information in an image, making the remaining associations more clear. So in an image labeled ‘desert’ and ‘sky’, the continuous blue region may be already strongly associated with ‘sky’, potentially strengthening the association between the other parts of the image and the label ‘desert’.

One potential drawback of this approach is the early discretization of the visual domain. Barnard et al. [2] note that making such quantization decisions without language guidance seems to result in weaker correspondence. One possible way of addressing this problem would be to periodically recalculate the translation table based on new language-guided grouping decisions. Supporting configurational appearance models within this framework could lead to efficiency problems, as any incremental improvement to the visual models could change the
visual vocabulary and therefore require a recalculation of the translation table.

2.5.4 Latent Semantic Analysis

The visual features extracted from an image and the caption words associated with it can both be viewed as artifacts of the scene’s content. If the scene contains a tiger, for instance, it is the tiger’s presence that causes both the distinctive striped texture and ‘tiger’, ‘wild’ and ‘predator’ word labels. Latent Semantic Analysis (LSA) in language–vision correspondence attempts to identify the hidden semantic factors in scenes that are the common cause of word labels and visual representations.

LSA, as introduced by Deerwester et al. [28], is a document classification and retrieval technique very closely related to principle components analysis (PCA) in vision. A collection of documents is represented as a matrix where each column is a normalized count-vector of the words observed in a single document. LSA uses singular value decomposition to explain these observations in terms of a set of latent topics, each with its own distribution over words, and per-document set of topic mixing weights. Hoffman [43] introduced an improved version of this approach, called probabilistic LSA (pLSA), that is built on a stronger statistical foundation and often generates more meaningful results.

Cascia et al. [23] use LSA, but only to reduce the dimensionality of their joint annotation–appearance image descriptors. Both Sivic et al. [76] and Monay et al. [61] use pLSA to detect meaningful themes in image collections. However, of the two, only [61] uses joint text and visual descriptions. LSA shares many of the same advantages and limitations of the translation model, including the construction of a discrete visual vocabulary as a preprocessing step. Unlike the translation model, when annotating a new image, the co-occurrence of several distinct visual elements can raise the posterior probability of a particular visual concept. However, the LSA model contains no spatial relations among visual parts and it is not obvious how such features might be incorporated.
2.5.5 Latent Dirichlet Allocation

Latent Dirichlet allocation (LDA) was introduced by Blei et al. [13] as a generative model for text collections. The overall goals are similar to LSA in that it attempts to uncover a meaningful set of topics that can explain statistical structure in a dataset. The primary difference is that LDA captures how the mixture of latent topics is generated. In LDA, a document is generated by first sampling from a global Dirichlet distribution, which provides the document-specific mixture probabilities for latent topics. A variable number of topics are then sampled from this distribution, and each sample generates a word.

In [12], Blei et al. adapt LDA to generate both words and continuous visual features. Here each topic is associated with a Gaussian distribution in feature space. Blei et al. consider two versions of the model: one in which visual and word topics are drawn independently given the document, and one in which words depend on the specific visual topics selected (called correspondence LDA). They found that adding this additional dependence to the model significantly improved correspondence results. Similar models have also been tested in [2, 31].

Again, LDA has much in common with pLSA as a model for word–image structure correspondence. One benefit is that it has been demonstrated using continuous features without pre-computing a quantization of the visual feature space. It is interesting to speculate on whether the Dirichlet distribution could be replaced by a distribution that could model co-occurrence relationships among topics. While in principle each latent topic could be associated with a more complex appearance model, it is not immediately clear how to train such a model.

2.5.6 Hierarchical Aspect Model

In [2], Barnard et al. apply a hierarchical aspect model first proposed by Hoffman [42] to the correspondence problem. In this approach, an image region is associated with a path from the root to a leaf of a tree. Each tree node is associated with a Gaussian distribution over visual features and a multinomial distribution over words. Since nodes closer to the root are shared
among a larger portion of the training set, the intended behavior is that nodes closer to the root encode common words and visual structures (such as ‘sky’) while nodes closer to the leaves encode rarer concepts such as ‘waves’.

Barnard et al. propose and test a larger number of variations to the model, but it is not clear that the extra complexity of the model leads to stronger correspondence results. While the model might be capable of encoding a number of interesting relationships between words, difficulties in training the model seem to impede performance.

2.6 Conclusions

The most interesting aspect of the language–vision correspondence problem is its challenging view of object recognition. The challenge is to create algorithms than can learn from data sets that are both larger and more ambiguous than we are accustomed to. In this view, words are not seen as semantics in themselves, but as nodes in a web of meaningful relationships. How many of these relationships can we bring to light?

The systems surveyed in this chapter have shown that it is possible to uncover at least some of the meaningful correspondences lurking in an ambiguous collection of image–text pairs. While these results are a good beginning, not all types of objects can be distinguished by an unstructured distribution over low-level visual features. To find word–image structure correspondences for new types of objects will likely require new image structure representations.

Object representations in the broader object recognition literature often require some kind of perceptual grouping of parts (e.g. [25, 35, 88]). These systems are able to learn appearance models that are configurations of features. Ideally, a correspondence system could learn such models from less structured and more ambiguous data sets. Admittedly, some of the results in the correspondence literature are not encouraging here. While sometimes adding spatial constraints improves correspondence results, as in [17], systems that mostly ignore shape and spatial relationships, such as [21, 61], are still very competitive.
The under-performance of appearance models that incorporate spatial relationships is in part due to the fact that more complex models are harder to train and more susceptible to overfitting. Another, more subtle, factor is how the grouping decisions that underpin a configurational model are made. Object recognition systems that learn configurations of parts such as [35] often look for recurring structures across the training image set to help decide what groups should be formed. In contrast, correspondence systems that encode shape or spatial structure tend to either use a pre-defined grouping structure [55] or perform grouping for each image independently as a preprocessing step [3]. Correct bottom-up perceptual grouping is very hard, and no grouping is usually better than bad grouping.

If it is helpful to consider the possible connections between images when making grouping decisions, it may be even more helpful to consider potential correspondences with words. Using word–model correspondence to help guide grouping could focus exploration efforts for potential configurations on the meaningful relationships we wish to uncover.
Chapter 3

Unstructured Appearance Models

Image annotation is recognized as an important means for associating meaning (in the form of caption words or keywords) with an image; it can also be seen as a means for assigning meaning (in the form of visual features) to the caption words (e.g., [4, 29, 71, 86]). The patterns of co-occurrence of words and visual features in annotated images can provide the evidence needed to establish meaningful links between the visual and linguistic representations. However, this approach can only succeed to the extent that the words and visual features correspond to meaningful aspects of what is portrayed in the image.

On the language side, we face three problems. Words in the caption may be noisy (e.g., misspelled), they may be irrelevant (i.e., they don’t refer to objects in the image), or they may, in isolation, not capture the best meaning of the object (e.g., if, for a particular object in an image, “rocket ship” is more appropriate than “rocket” or “ship”). In this work, we will focus on only one of these problems, namely the problem of irrelevant words in the caption. On the vision side, we will use the associations between caption words and image features to overcome all three analogous problems in vision, eliminating unstable (noisy) features from a model, excluding background (irrelevant) features from the model, and grouping individual features belonging to an object into collections that better capture the scope (granularity) of the object.
A cluttered scene may yield hundreds or even thousands of local features, only a small subset of which corresponds to any given object. This perceptual grouping or segmentation problem exists no matter what type of image representation is used: pixels, line segments, local features or regions. Given a set of features extracted from an image, the challenge is to find the meaningful subsets, to ‘carve nature at its joints.’ However, these ‘important’ subsets represent a vanishingly small portion of all possible subsets for any non-trivial representation. Although effective bottom-up grouping heuristics exist for certain classes of features (e.g., Gestalt grouping of lines), today’s popular interest point-based features do not lend themselves to bottom-up grouping. Simply evaluating all possible groupings is not usually feasible.

How might we find meaningful groupings in the context of the annotation problem? Simple frequency of occurrence in the training data can provide a clue. If certain collections of image features exist more often than can easily be explained by chance, they may have a common, meaningful source. However, the number of such groupings may still be prohibitively large. Even if they do arise from some common, recurring source in the image, such groupings might have no corresponding word on the language side. This suggests a dual approach: find co-occurring visual features which also have a significant level of co-occurrence with specific words. Considering the modalities of language and vision together can make the perceptual grouping problem more tractable at the same time as it offers a solution to the semantic association problem.

We begin with a set of captioned training images of cluttered scenes containing multiple objects. On the vision side, each image is processed to yield a set of local SIFT features [58]. On the language side, each image is annotated with a set of nouns which may or may not name objects in the image. Drawing on the probabilistic translation model of [15] (as in [29, 86]), we introduce a novel iterative algorithm for growing candidate associations between individual SIFT features and nouns into more definitive models of object appearance in the form of collections of SIFT features, or compounds. This simultaneous language-driven perceptual grouping and association yields a set of models which is subsequently used to annotate new images.
3.1 Features and Compound Features

In an image annotation system, the choice of feature largely determines the types of object classes that can be reliably detected. Some object types, such as grass, pavement or sky, have no consistent shape, and might be well-described by a region descriptor (e.g., a blob) encoding color or texture. Other objects, such as tables, lamps and clothed people, are defined more by their shape and less by their color and texture, suggesting a structured or parameterized shape model. Still other object classes, such as trees or mountains, exhibit patterns of limited variation in both shape and appearance.

In this work, we adopt the local interest point detector and SIFT feature representation developed by Lowe [58]. We represent an image as a set of local interest points $p_m$, i.e., $I = \{p_m|m = 1 \ldots |I|\}$, detected at $I$’s scale-space maxima. Each point $p_m$ is detected at a particular Cartesian position $x_m$, scale $\lambda_m$ and orientation $\theta_m$. A SIFT feature, $f_m$, encodes the local pattern of intensity changes as a 128-element vector. The feature is invariant to scaling, translation and rotations in the image plane, and is designed to be robust to changes in intensity and contrast, small translation errors, and small rotations in depth.

In adopting SIFT features, we restrict ourselves to object classes which generate interest points in reasonably stable configurations and where the surface appearance is stable for objects within the class. For most purposes, this means exemplar objects or objects manufactured to the same design. In cluttered scenes, SIFT features are plentiful, with thousands detectable in a typical image. This is both a blessing and a curse. It allows us to potentially detect objects that occupy only a small portion of the image, yet the number of visual features far outweighs the number of relevant words in a typical image annotation.

The SIFT feature vectors $f$ are continuous, while our current translation model operates on discrete tokens. We therefore perform vector quantization to replace each feature $f_m$ with a discrete ‘quantized’ feature, $c_m$. Other recently-developed image annotation techniques avoid this quantization step and estimate continuous probability densities over a feature space [21, 33]. Since we are employing SIFT features, which already have high dimensionality, and
wish to find collections or configurations of these features, the continuous description would probably become too unwieldy. Of course, transforming the feature vectors into discrete classes introduces an unavoidable level of quantization noise. It may be possible to reduce the effect of this noise by associating a small weighted set of features with each interest point.

Vector quantization is trained on a set of 300,000 local interest features selected at random from a pool of 2500 stock photo images. We use the K-means algorithm to generate a set \( C \) of 5000 cluster centers, \( C = \{ f_c | c = 1 \ldots |C| \} \), similar to the approach of Sivic and Zisserman [77]. The feature data is whitened before clustering so that the Euclidean distance on the transformed data equals the Mahalanobis distance in the original space:

\[
d(f_m, f_c) = \sqrt{(f_m - f_c)^T \Sigma^{-1} (f_m - f_c)}
\]  

(3.1)

where \( \Sigma \) is the covariance matrix of the interest features. Each detected SIFT feature is replaced with the index of the nearest cluster center:

\[
c_m = \arg \min \limits_c d(f_m, f_c).
\]  

(3.2)

At training time, we also calculate the proportion \( \alpha_v \) of SIFT features in our stock photo collection assigned to each cluster center \( f_c \). There is a great deal of variation in these cluster weights. Again following Sivic and Zisserman [77], we suppress the most common 0.5 percent and the least-common 10 percent of features. This is based on an analogy with text retrieval, where the least common and most common terms are less informative.

In addition to describing each point individually, we also attempt to capture the local spatial configuration of points using neighborhoods that describe the local context of a point. Each point \( p_m \) is associated with a neighborhood \( n_m \) that is the set of its spatially closest neighbors \( p_n \).

Individual SIFT features may not be strong indicators of a particular object or object class. Specific arrangements or structures of local features are more discriminative. In this chapter, we consider features that exist within a local neighborhood. A compound feature \( G \) is essentially a ‘bag’ of local features. Each compound feature is a pair, \( G = \{ C_G, \eta_G \} \), consisting of
a set of quantized visual features $C_G$, and a presence threshold $\eta_G$. A compound is considered present if at least $\eta_G$ distinct quantized features within $C_G$ exist within a ‘local neighborhood’ $n_m$. Specifically, consider an interest point $p_j$ with the corresponding quantized SIFT feature $c_j \in C_G$. We detect the compound $G$ at $p_j$ if at least $\eta_G$ distinct elements of $C_G$ are present in $n_m$. We ignore compound features whose neighborhood would overlap with a previously detected instance of the same compound. Note that each individual feature can also be considered a ‘singleton’ compound of size one. $\Gamma$ is the mixed set of singleton compounds and true, multi-element compounds that together account for all $p_m$ within image $I$. The combinatorics of the problem yields a very large number of possible compound features in an image. However, only a vanishingly small fraction of these correspond to meaningful configurations, and fewer still may correspond to nouns appearing in the captions. This further increases the asymmetry between annotation word counts and visual feature counts.

### 3.2 The Caption-to-Image Translation Model

The translation model serves two purposes. Given a set of images, each of which is associated with a set of visual features and a collection of annotation words, the translation model discovers correspondences between the visual features and the annotations. However, the initial set of visual features, extracted based on image characteristics alone, may not be distinctive enough to pick out the objects that are named in the annotations. Therefore we also use the translation model to help guide a search for better features. An iterative process proposes larger, more distinctive compound features, and uses the correspondence strength in the current translation model to evaluate the goodness of each potential compound. In the following subsections, we discuss each of these aspects of the translation model in turn.
3.2.1 The Basic Translation Model

As in earlier image annotation work (e.g., [29, 86]), we begin with the following translation model: the conditional probability $P(F|E)$, given two sets of symbols $F$ and $E$. (In the formulation by Brown et al. [15] for machine translation, $F$ and $E$ referred to sequences of words in French and English, respectively.) To reduce the number of parameters to be estimated, it is generally assumed that the symbols (words or image features) can be generated independently. Each symbol $F_i$ has an alignment variable $a_i$ from $\{0, \ldots, L\}$, where $L$ is the number of symbols in $E$, that associates $F_i$ with a single symbol, $E_{a_i}$, which may be the “null” symbol, $E_0$.

There is a strong asymmetry in this model, as each symbol in $F$ is associated with a single symbol in $E$ (or with none of those—the null symbol), while each symbol in $E$ can be associated with an arbitrary number of symbols in $F$. In our annotated images, we also have an asymmetry, in that there are typically a very large number of visual features and a relatively small number of caption words. It is much more likely that multiple visual features correspond to a single word than vice versa. We thus treat the set of annotation words $w$ as $E$ and the set of visual features $\Gamma$ as $F$ in the formula above, yielding the following, based on Brown et al.’s Model 1:

$$
P(\Gamma|w) = \frac{\epsilon}{(L + 1)^M} \prod_{j=1}^{M} \sum_{a_j=0}^{L} t(G_j|w_{a_j})
$$

(3.3)

Here $M$ is the number of compound features, $L$ is the number of caption words, $\epsilon$ is a constant, and $t(G_j|w_{a_j})$ is an element of the translation table $t$ defining the distribution over compound visual features for each word. As in Brown et al., we use EM to find the $t$ that maximizes the probability of the training data.

3.2.2 Dealing with “Noisy” Features

The goal is to learn stable associations between words and image features. One of the drawbacks of the above model is that it is devised for a situation—translation between two languages—
in which most elements in one representation (the source language) are aligned with an element in the other representation (the target language). The possibility of alignment with the null word exists, but most words are expected to align with an actual word. However, this is generally not the case in aligning visual features and caption words, and is especially not the case with SIFT features. There are thousands of SIFT features in any given image, and only those that are a stable part of the appearance of a named object have any counterpart in the caption text. This motivates a larger role for the null symbol, to serve as a “default” alignment for the many SIFT features which correspond to objects or surfaces that are not named among the annotation words, or that are transient, unstable aspects of a named object. We want to ensure that such “chance” features are not linked to actual caption words, by increasing the likelihood that they align to the null word.

One issue is that our small pool of labeled training images is not broad enough for us to determine the distribution of the types of background features in images more generally. To address this, we use the counts of singleton features over our pool of 2500 stock images (see Section 3.1) to estimate the prior probability distribution over individual features. The prior likelihood of any compound feature is calculated assuming singleton features are placed independently. We then add a dummy entry to the translation training data which includes a set of visual feature counts (both singleton and compound) in proportion to their prior likelihoods (effectively estimating their occurrence by chance). The dummy entry has no associated caption words, entailing that a strong association is established between the “chance” features and the null word. Then, features with high prior likelihood or which appear rarely in the training images are more likely to align with the null symbol (i.e., to be considered background).

Since the background or noisy features typically comprise more of the image than features from the objects of interest, the translation table is normalized to give the null word a higher alignment probability than the actual caption words. In our experiments, visual features are ten times more likely \textit{a priori} to align with the null word than with a caption word.
3.2.3 The Search for Compound Features

A cornerstone of our framework is the use of the associations between annotation words and visual features to guide the process of grouping visual features into meaningful collections that serve as better indicators of the objects named by the words. This is achieved by initially learning the translation probabilities on singleton visual features, then iteratively trying out potential collections of the existing features. These potential compound features are evaluated with respect to their improvement to the translation probability for predicting the annotation word under consideration. A potential feature that leads to an improvement is adopted, and the translation model is iteratively re-learned.

We use a simple greedy algorithm. We start by initializing the set of features that will be considered. For each of the $W$ annotation words $w_i$ (excluding the null word), we choose the $n_{seed} = 20$ singleton visual features $c$ which have the highest $P(w_i|c)$ and occur more than $n_{min} = 8$ times in the training data. These features form $\mathcal{G} = \{G_{ij}|i = 1 \ldots W, j = 1 \ldots n_{seed}\}$, the basis for the parallel, independent growth of compound features. Initially, all $G_{ij}$ are singleton features, while on successive iterations, they may be singleton or compound features.

Each iteration of the search algorithm considers all elements of $\mathcal{G}$. If $G_{ij}$ is the current feature, we try one previously untested modification to produce a new compound feature $G^*_{ij}$ for the associated word $w_i$. The compound $G^*_{ij}$ replaces its constituent features wherever they appear together in the training images. If $G^*_{ij}$ occurs at least $n_{min}$ times, we update the translation model for the altered data to get $P(w_i|G^*_{ij})$. If this is greater than $P(w_i|G_{ij})$, then $G^*_{ij}$ replaces $G_{ij}$ in $\mathcal{G}$. Either way, we “undo” the changes to the training data and continue the iterative process with the next feature in $\mathcal{G}$.

The possible modifications for producing potential compounds include removing any one element from the current compound, and adding any element that co-occurs (in the same neighborhood) with the compound in the training data. For each removal/addition operation, we test both a ‘changed’ and a ‘stable’ version where the target number $\eta_{ij}$ of constituent features is
Figure 3.1: To annotate an image (a), we first extract SIFT features and identify local neighbourhoods (b). If any compounds with training scores above the threshold are detected in the image, we add the associated word to the annotations (c).

decreased/increased by one, respectively, in the former, and unchanged in the latter.

In theory, we could run the search until there are no more potential changes in \( G \). In practice, this is too time-consuming, and we limit the outer loop of the search to 200 iterations.

### 3.3 Annotation

Once we have generated a set of learned compound features \( C \) from the training set, we employ a simple technique to annotate test images. Given a confidence threshold, \( \Delta \), we remove from \( G \) all learned compound features \( G_{ij} \) where \( P(w_i|G_{ij}) < \Delta \).

Figure 3.1 summarizes the annotation process. On loading a new test image, the algorithm extracts a set of SIFT features and their associated local neighborhood structure. If the image contains one or more instances of a compound \( G_{ij} \) in \( G \), we label the image with the corresponding word, \( w_i \). Labels are binary, as multiple compound detections do not necessarily indicate multiple instances of the object.
3.4 Experiments

3.4.1 Data Set

We tested the algorithm on a set of real object images, in this case 228 images of arrangements of children’s toys. The original color photographs were converted to intensity images with a resolution of 800x600. The images contain views of the 10 named toys shown in Figure 3.2. Within the set of 228 images used for training and testing, objects are never shown in isolation. Most images contain 3 or 4 toy objects, though there are a handful of examples of up to 8 objects. The objects are not arranged in any consistent pose and many are partially occluded. Illumination was either direct sunlight, indirect natural light or from the camera’s integrated flash. The images were collected against approximately 15 different backgrounds of varying complexity.

The pool of 228 images was randomly divided into a training set of 128 and a test set of 100. Each training image was annotated with the unique keyword for each object of interest shown and between 2 and 5 other keywords uniformly drawn from a pool of distractor labels. Figure 3.3 displays some example images from the training set and their associated annotations. Note that the objects of interest never appear individually and the training data contains no information as to the position or pose of the labeled objects.

3.4.2 Results

We ran the compound feature search technique on the training set with all compounds set to a fixed neighborhood size of $|n_m| = 100$. Figure 3.4 illustrates some example instances in the training set of the single best compound feature for the each of the ‘rocket’, ‘ernie’, ‘horse’ and ‘bug’ objects. Locations of SIFT features that form the detected compound are marked with yellow circles while a blue rectangular region indicates the approximate extent of the local neighborhood.

Note that the spatial configuration of component features sometimes varies across detec-
Figure 3.2: Images of the individual toy objects, shown in isolation. All images used for training and testing contain multiple named objects.
rocket, bug, cash, 
floor, toys 
bongos, drum, rocket, 
dino, bus, wall, deck

Figure 3.3: Example training images and their associated labels. Approximately half the labels (e.g., ‘wall’, ‘tile’, ‘floor’) serve as distractors for the relevant labels such as ‘rocket’, ‘bug’ and ‘cash’.
tions of the same compound. In some cases, a compound may even match more than one location on the same object. This degree of flexibility allows the compound to compensate for noise in the feature extraction process and to match the object across orientation changes and occlusions. However, such a pliable configuration definition is also more likely to generate false detections.

Precision is the portion of detected annotations that are correct, while recall is the proportion of correct annotations that are properly detected. The precision-recall curves in Figures 3.5 and 3.6 represent the output with the confidence threshold $\Delta$ ranging from 1 to 0. Figure 3.5 demonstrates that the feature compounds learned on the training set are more distinctive and a stronger basis for annotation than the individual features. On the whole, single SIFT features do not perform substantially better than chance using this simple annotation scheme.

The fact that the curves are not monotonic indicates that confidence on the training set is not always a good predictor of performance on the test set. In fact, a few compounds with a moderately high confidence score in training appear to encode patterns that do not fall on the object of interest. However, most compounds that result from such coincidental correlations in the training data have low confidence scores.

Figure 3.6 shows the precision recall curves broken down by individual object. Results are fairly good for several objects, notably the ‘rocket’ and ‘cash’ exemplars. Given the variety of object poses and prevalence of partial occlusions, even the stronger compounds learned in training do not detect every instance of the object.

For some objects, most notably ‘dino’, the system fails to find any useful identifying compounds. This is because none of the 20 original most highly-correlated seed features for the word ‘dino’ actually fall on the object of interest. Each is a relatively rare noise feature that happens to correlate with ‘dino’ more than any other label. This illustrates an important drawback of the current implementation: the object must have a feature that is somewhat distinctive in itself. It is likely that ‘dino’ displays reasonably stable common configurations of interest points, but if there is no single interest point that is relatively exclusive to the object of interest
Figure 3.4: Example detections of compound features associated with the labels (a) ‘rocket’, (b) ‘ernie’, (c) ‘horse’ and (d) ‘bug’, respectively.
Figure 3.5: Precision vs. recall for single features versus compound features on the test set. Single features are not as distinctive.

then noise features may dominate the seed set. To combat this effect, the algorithm could consider more potential starting points. In some cases, it might be necessary to start the language iteration above the single-feature level.

Table 3.1 contains precision and recall results for two values of the confidence threshold, $\Delta$. A high confidence threshold generally leads to relatively high precision in the annotation, though two compounds associated with the ‘bongos’ and ‘franklin’ labels have anomalously high probabilities, leading to relatively low precision on the test set for these two labels.

Finally, Figure 3.7 displays the annotation results using $\Delta = 0.85$ for a few randomly-selected test images. The results indicate that the proposed method is capable of correctly assigning highly distinctive features to names of real objects even though the training set contains no instances of the object or object name in isolation.
Figure 3.6: Precision-recall curves arranged into objects with better responses (a) versus weaker responses (b). The search process found fewer high-confidence compounds for the objects listed in (b).
Table 3.1: Precision-Recall results for two confidence thresholds.

<table>
<thead>
<tr>
<th>Label</th>
<th>$\Delta = 0.9$</th>
<th>$\Delta = 0.8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>bongos</td>
<td>0.42</td>
<td>0.14</td>
</tr>
<tr>
<td>franklin</td>
<td>0.77</td>
<td>0.70</td>
</tr>
<tr>
<td>drum</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>ernie</td>
<td>1.00</td>
<td>0.31</td>
</tr>
<tr>
<td>rocket</td>
<td>0.97</td>
<td>0.70</td>
</tr>
<tr>
<td>bug</td>
<td>1.00</td>
<td>0.16</td>
</tr>
<tr>
<td>cash</td>
<td>1.00</td>
<td>0.41</td>
</tr>
<tr>
<td>dino</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>bus</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>horse</td>
<td>0.92</td>
<td>0.39</td>
</tr>
<tr>
<td>Overall</td>
<td>0.87</td>
<td>0.27</td>
</tr>
</tbody>
</table>
Figure 3.7: Randomly-selected test images with detected labels for $\Delta = 0.85$. 
3.5 Discussion and Future Work

The method we have described can find groupings of SIFT features that are distinctive to individual objects and at the same time associate these objects with appropriate words from the image annotations. The system can detect these compounds and word correlations even though the features of interest themselves provide no grouping hint and always appear intermixed with features from other objects and complex backgrounds.

The system was implemented as a relatively simple vehicle to explore language-assisted grouping. As a practical image annotation system there are many directions for improvement. For instance, we would like to employ insights from computational linguistics to both construct a richer translation model and to allow us to move beyond individual words to groupings of words (compound nouns, collocations, or modifier-noun relations) that correspond to distinct visual patterns.

On the vision side, our current compound features, though simple and flexible, work over a limited range of scales and have very weak spatial constraints. An approach based on pairwise connections, such as that proposed by Carneiro and Jepson [19], could better model entire flexible or articulated objects while achieving much tighter spatial configuration constraints.

In both the visual and language domains, these more constrained compounds are more difficult to discover by brute force. We must induce them by exploiting patterns of co-occurrence both within and between the language and vision domains. For instance, a strong correlation in the training data between two or three SIFT features might provide a promising starting point to grow models of objects with no individually distinctive features.

Though SIFT features provide a strong basis for detecting unique objects, they are less ideal for detecting object categories or general settings. However, the grouping problem persists for most types of visual features and most forms of annotation, and many of the mechanisms for addressing the grouping problem for SIFT features are in fact quite general.

Any system for finding meaningful correlations between images and words also faces the problem of finding the level of description at which meaningful correlations exist. We expect
that, in general, many correlations will exist not between individual words and individual features or regions, but between groups of words and collections of features. Patterns of correlation between the vision and language aspects of a data set provide important cues to meaningful grouping in both domains.
Chapter 4

Structured Appearance Models

Manual annotation of new images in large image collections is prohibitively expensive for commercial databases, and overly time-consuming for the home photographer. However, low-cost imaging, storage and communication technologies have already made accessible millions of images that are meaningfully associated with text in the form of captions or keywords. It is tempting to see these pairings of visual and linguistic representations as a kind of distributed Rosetta Stone from which we may learn to automatically translate between the names of things and their appearances. Even limited success in this challenging project would support at least partial automatic annotation of new images, enabling search of image databases by both image features and keywords that describe their contents.

Any such endeavor faces the daunting challenge of the perceptual grouping problem. Regardless of the type of image feature used, a word typically refers not to a single feature, but to a configuration of features that form the object of interest. The problem is particularly acute since any given image may contain multiple objects or configurations; moreover, the meaningful configurations may be easily lost among a huge number of irrelevant or accidental groupings of features. Without substantial bottom-up grouping hints, it is a nearly hopeless task to glean the meaningful feature configurations from a single image–caption pair. Given a collection of images, however, one can look for patterns of features that appear much more of-
ten than expected by chance. Usually, though, only a fraction of these recurring configurations correspond to salient objects that are referred to by words in the captions.

Some of our previous work has shown that correspondences between image features and caption words can be used to guide the search for meaningful feature configurations [45, 86]. Building on this idea, we introduce a new word–appearance correspondence measure that uses language cues in addition to recurring visual patterns to incrementally construct strong object appearance models. In contrast to [86], our appearance models are composed of easily-extractable local features and can therefore detect exemplar objects in highly cluttered real-world images. Our new models capture more of the structured appearance of an object than the simple ‘bag of features’ models used in [45] while remaining robust to changes in scale and orientation, as well as to minor deformations. We demonstrate improved results on the set of images used in [45] and also discover meaningful word–appearance pairs in a larger and more challenging set of captioned hockey images.

4.1 Learning to Annotate Exemplar Objects

The goal of this work is to annotate exemplar objects appearing in images of cluttered scenes. A typical such image, with more than a thousand local features, contains a huge number of possible feature configurations, most of which are noise or accidental groupings. A complex configuration of features that occurs in many images is unlikely to be an accident, but may still correspond to common elements of the background or other unnamed structures. The only evidence on which to establish a connection between words and configurations of visual features is their co-occurrence across the set of captioned images. The key insight of [45] and [86] is that this evidence can guide not only the annotation of complex feature configurations, but also the search for meaningful configurations themselves.

Accordingly, we look for recurring configurations of features that also strongly co-occur with certain words, hence simultaneously finding objects and annotating them. To illustrate,
New York Islanders’ defenseman Alexei Zhitnik mashes Vancouver Canucks’ right wing Todd Bertuzzi into the glass.

Figure 4.1: From a set of image–caption pairs, each containing hundreds of local features (gray crosses), our algorithm has discovered an association between a team name (shown in red) and its logo (red features and green relationships in a yellow box).

Figure 4.1 shows a sample image containing hundreds of local features, paired with a caption containing many irrelevant words. Our algorithm has learned both an appearance model (here representing the team’s logo), as well as its association with a caption word (here the team name Islanders). To begin, the learning algorithm finds a set of simple recurring object models, evaluates how strongly they correspond to each caption word, and accordingly chooses a set of good ‘seed’ appearance models for each word. Each seed model is then iteratively expanded (if possible) into a more reliable object detector, guided at each step by the change in its strength of association with the corresponding caption word.

We use a learning framework that allows a one-to-many relationship between words and
appearance models. It is thus not necessary that a single model capture object appearance from all possible viewpoints. Moreover, since we deal with exemplar objects, our method need not handle the changes in texture and structural detail that are possible within a class of objects. In order to serve as a robust object detector, however, it is important that the appearance model representation be invariant to reasonable changes in lighting, scale, orientation, articulation and deformation. The representation must also be reliably detectable, in order to avoid false annotations. We use local interest features to represent small patches of appearance and use the pairwise spatial relationships between the patches to construct a connected graph model for the object. The model that is built this way is a reliable descriptor of the object appearance and at the same time flexible enough to handle common deformations. Details of our choices for the representation of images and objects are described in Section 4.2.

The initial stage of our learning algorithm provides a set of seed appearance models to use as starting points for the detection of objects mentioned in the captions. The most straightforward starting points are singleton features, but their relationship to an object may be too tenuous to provide effective guidance for building strong object models. At the same time, trying all possible configurations of even a small number of features as seeds is impractical. Instead, our initialization method generates structured seed models by looking at recurring neighborhoods of features that also co-occur with particular caption words.

The initial seed models are fed into an iterative improvement stage, which expands them into appearance models that cover a larger portion of the object. In previous work, the guidance for the iterative improvement of an initial model is provided through a probabilistic translation method [45, 86]. However, it is expensive to relearn all of the translation probabilities every time a new configuration of features is formed. Here, we use a simpler and more efficient measure of correspondence between a caption word and an appearance model. The measure reflects the amount of evidence, available in a set of training images, that the word and the model are generated from a common underlying source object. Section 4.3 elaborates on the correspondence measure, as well as the initial and the iterative improvement stages that draw
on this measure.

4.2 Image and Object Representations

4.2.1 Image Representation

We represent an image as a set $I$ of interest points $p_m$, i.e., $I = \{p_m | m = 1 \ldots |I| \}$. These points are detected using Lowe’s SIFT method [58], which defines a point in terms of its Cartesian position $x_m$, scale $\sigma_m$ and orientation $\theta_m$. In addition to these, for each interest point we also extract a feature vector $f_m$ that encodes a portion of the image surrounding the point. Since $f_m$ is extracted relative to the spatial coordinates of $p_m$, it is invariant to changes in position, scale and orientation. For this, we use the PCA-SIFT feature encoding developed in [49] because it allows for fast feature comparison and low memory requirements. This feature encoding is reasonably robust to lighting changes, minor deformations and changes in perspective. Since individual features capture small, independent patches of object appearance, the overall representation is robust to occlusion and articulation.

In addition to the continuous feature vector $f_m$, we also use a quantized descriptor $c_m$ for each image point, in order to quickly scan for potentially matching features. Following [78], we use K-means to generate a set of cluster centers from a set of features randomly selected from our training images. The index of the cluster center closest to $f_m$ is the descriptor $c_m$ associated with $p_m$.

Each point $p_m$ is also associated with a neighborhood $n_m$ that is the set of spatial neighbors of the point. The distance measure used to construct a neighborhood is:

$$\Delta x_{ij} = \frac{\|x_i - x_j\|}{\sqrt{\sigma_i^2 + \sigma_j^2}} \quad (4.1)$$

This normalized distance measure makes neighborhoods more robust to changes in scale, as newly-introduced fine-scale points are less likely to push coarse-scale points out of the neighborhood when the scale of the object increases.
To summarize, each point $p_m$ in an image representation $I$ is a 6-tuple of the form $(f_m, x_m, \sigma_m, \theta_m, c_m, n_m)$.

### 4.2.2 Appearance Model Representation

We represent an appearance model using a graph $G = (V, E)$. Each vertex $v_i = (f_i, c_i)$ is composed of a continuous feature vector $f_i$, and a cluster index vector $c_i$ containing indexes for the $|c_i|$ nearest cluster centers to $f_i$. Associating each model vertex with a set of clusters allows for fast comparison of features during model detection while minimizing the effects of quantization noise. Note that model vertices, unlike image points, do not include spatial information because the appearance model must be invariant to scale, translation and rotation. Instead, each edge in $G$ encodes a spatial relationship between vertices in four parts: $e_{ij} = (\Delta x_{ij}, \Delta \sigma_{ij}, \Delta \phi_{ij}, \Delta \phi_{ji})$, where $\Delta x_{ij}$ is the relative distance between $v_i$ and $v_j$, $\Delta \sigma_{ij}$ is the relative scale difference between them, $\Delta \phi_{ij}$ is the relative heading from $v_i$ to $v_j$, and $\Delta \phi_{ji}$ is the heading in the opposite direction. These relationships are taken from Carneiro and Jepson [19], and are calculated as in (4.1) above, and (4.2) and (4.3) below:

\[
\Delta \sigma_{ij} = \frac{\sigma_i - \sigma_j}{\sqrt{\sigma_i^2 + \sigma_j^2}} \tag{4.2}
\]

\[
\Delta \phi_{ij} = \Delta \theta (\tan^{-1}(x_i - x_j) - \theta_i) \tag{4.3}
\]

where $\Delta \theta(.) \in [-\pi, +\pi]$ denotes the principle angle.

An observed instance of a model is a set of vertex–point associations, $O = \{(v_i, p_m) | v_i \in V, p_m \in I\}$, where $O$ defines a one-to-one mapping between a subset of the model vertices and some local interest points in an image. Though each model is intended to robustly describe an object’s appearance, no observed instance of an object is expected to fit its model exactly. Deformations, noise and changes in perspective can distort the features encoded at local interest points and the spatial relationships between them. Also, a model may be only partially observed, with vertices occluded or lost due to inconsistent detection of points.
4.2.3 Energy Function

We introduce an energy function, $H(G, I, O)$, that measures how well the observed instance $O$ in image representation $I$ matches the object appearance model $G$. The function is defined so that observed configurations of points that closely fit the model have low energy, whereas configurations that are very different from the model have high energy. The energy function has two components, looking at vertices and edges, respectively:

$$ H(G, I, O) = \sum_{(v, p_m) \in O} h_V(v_i, p_m) + \sum_{e_{ij} \in O} h_E(e_{ij}, p_m, p_n) $$

where $e_{ij}$ is in $O$ if both $(v_i, p_m) \in O$ and $(v_j, p_n) \in O$.

$h_V(v_i, p_m)$ measures the fit between a model vertex $v_i$ and the observed image point $p_m$, and is calculated as:

$$ h_V(v_i, p_m) = \delta_V + \alpha_f \|f_i - f_m\|^2 $$(4.5)

where $\delta_V$ is the maximum energy reward for observing a vertex, and $\alpha_f$ is the rate the reward decays with feature dissimilarity.

The component $h_E$ measures how well spatial relationships between observed points fit expectations set by the model. It is calculated as:

$$ h_E(e_{ij}, p_m, p_n) = \delta_E + \alpha_\sigma (\Delta \sigma_{ij} - \Delta \sigma_{mn})^2 + \alpha_x (\Delta x_{ij} - \Delta x_{mn})^2 + \alpha_\phi (\Delta \phi_{ij} - \Delta \phi_{mn})^2 $$(4.6)

where $\delta_E$ is the maximum energy reward for matching the expected spatial relationship between vertices, and $\alpha_\sigma$, $\alpha_x$ and $\alpha_\phi$ control the rates of reward decay. In our experiments, these parameters are set as follows: $\delta_V = -8$, $\delta_E = -5$, $\alpha_f = 0.16$, $\alpha_\sigma = 25$, $\alpha_x = 0.25$ and $\alpha_\phi = 2.5$. By experimenting with several pairs of training images, we determine thresholds for spatial and feature variation that capture most corresponding interest point pairs while excluding most false positives. The $\delta_V$ and $\delta_E$ parameters roughly reflect the log probability of two random point pairs falling within their respective thresholds while the $\alpha$ values are adjusted so that $h_V < 0$ and $h_E < 0$ within the allowed variational range.
4.2.4 Model Instance Detection Algorithm

To detect an instance of an object model, we need to find a low-energy association of its vertices to points in an image. Given that a typical image can contain thousands of interest points, determining the optimal associations is potentially quite expensive. We thus propose a greedy heuristic that efficiently searches the space of possible associations for a nearly-optimal solution. Since individual vertices of a model may be unobserved, our detection heuristic allows for a connected model to be instantiated as disconnected components. To reduce the probability of false detections, the search for disconnected parts is confined to the neighborhood of observed vertices, and isolated singleton points are ignored. That is, in a valid model instance, each observed vertex shares a model edge with at least one other observed vertex. Also, only those observations $O$ with $H(G, I, O)$ below a learned model-specific threshold, $\Delta_H$, are considered valid instances for annotation.

Algorithm 1 uses a greedy heuristic to detect instances of an appearance model $G$ within the image representation $I$. The actual implementation can detect more than one instance of $G$ within $I$ by suppressing observed image points.

Algorithm 1 Detects instances of $G$ in $I$

FindModelInstance($G$, $I$)

1. Find the set of potential vertex–point associations $A = \{(v_i, p_m)\}$, where $c_m \in c$ and $h_V(v_i, p_m) < 0$.
2. Find the set $L$ of potential links $((v_i, p_m), (v_j, p_n))$ between elements of $A$, where $p_n \in n_m$ and $h_E(e_{ij}, p_m, p_n) < 0$.
3. Set the initial instance $O$ to the pair $\{(v_i, p_m), (v_j, p_n)\}$, such that the link $((v_i, p_m), (v_j, p_n)) \in L$ and $H(G, I, O)$ is minimum.
4. Remove $(v_i, p_m)$ from $A$ if either $v_i$ or $p_m$ are part of another vertex–point association $\in O$.
5. Remove $((v_i, p_m), (v_j, p_n))$ from $L$ if neither end is in $A$.
6. Let $A_{adj}$ be the subset of $A$ that shares an edge in $L$ with $O$.
7. If $A_{adj}$ contains associations that could decrease $H(G, I, O)$, add to $O$ the association with greatest decrease in $H$, and go to step 4.
8. Let $L_{neigh}$ be the subset of $L$ within the union of the neighborhoods of observed points in $O$.
9. If $L_{neigh}$ contains observed links that could decrease $H(G, I, O)$, add to $O$ the pair of associations with the link that produces the greatest decrease in $H$, and go to step 4.
10. Return $O$. 
4.3 Using Words to Learn Appearance Models

Here, we propose an unsupervised learning algorithm that constructs structured appearance models for the salient objects appearing in a set of training image–caption pairs. Salient objects are those that appear in many images, and are often referred to in the captions. Because each image contains many features of non-salient objects, and the caption contains words irrelevant to the displayed objects, the algorithm has to discover which image features and words are salient. The algorithm learns object models through discovering strong correspondences between configurations of visual features and caption words. The output is a set of appearance models, each associated with a caption word.

4.3.1 A Measure of Word–Model Correspondence

We seek pairs of words and appearance models that are representations of the same object in different modalities (linguistic and visual). We assume that both the word and the appearance model are present in an image because the object is present. We thus define and use a measure of confidence that a given appearance model is a reliable detector for the object referred to by a word.

Consider a set of $k$ captioned images. The occurrence pattern of a word $w$ in the captions of these images may be represented as a binary vector $r_w = \{r_{wi}|i = 1, \ldots, k\}$. Similarly, we can represent the occurrence of a model $G$ with another binary vector, $q_G = \{q_{Gi}|i = 1, \ldots, k\}$. It is always possible that the two patterns of occurrence are independent (the null hypothesis or $H_0$). Alternatively, they may have been derived from a hidden common source object (the common-source hypothesis or $H_C$). According to $H_C$, some fraction of image–caption pairs contain a hidden source $s$, which may emit the word $w$ and/or the appearance model $G$. We define the correspondence between $w$ and $G$ as the log-likelihood ratio of generating the observed patterns, $r_w$ and $q_G$, under $H_C$ and $H_0$: 
\[ \text{Corr}(w, G) = \log \frac{P(r_w, q_G|H_C)}{P(r_w, q_G|H_0)} \quad (4.7) \]
\[ P(r_w, q_G|H_C) = \prod_i \sum_{s_i} P(s_i) P(r_{wi}|s_i) P(q_{Gi}|s_i) \quad (4.8) \]
\[ P(r_w, q_G|H_0) = \prod_i P(r_{wi}) P(q_{Gi}) \quad (4.9) \]

where \( s_i \in \{0, 1\} \) represents the presence of the common source in image–caption pair \( i \).

\( \text{Corr}(w, G) \) reflects the degree to which the common-source hypothesis explains the observed patterns of occurrence for a word \( w \) and a model \( G \). To get the likelihood of observed data under \( H_C \), we need to estimate the parameters \( P(s_i), P(r_{wi}|s_i), \) and \( P(q_{Gi}|s_i) \). \( P(r_{wi}|s_i) \) and \( P(q_{Gi}|s_i = 0) \) are given fixed values according to assumptions we make about the training images, which we elaborate in Section 4.4.1. \( P(s_i) \) and \( P(q_{Gi}|s_i = 1) \) are given maximum likelihood estimates (MLEs) determined using expectation maximization over the training data. The MLEs for parameters under \( H_0 \) are simply the observed probabilities for word and model occurrence.

### 4.3.2 Model Initialization

The goal of the model initialization stage is to quickly find a set of seed appearance models that are fruitful starting points for building strong object detectors. Later, the seed object models are iteratively expanded and refined into larger and more distinctive appearance models using image captions as a guide. The existence of good starting points strongly affects the final outcome of the iterative improvement process. Nonetheless, a balance must be reached between time spent searching for good seeds and time spent in the improvement stage refining the seeds. Even searching the space of small models for good starting points is not a trivial task.

We thus look for good seed models using the more tractable neighborhood patterns introduced by Sivic and Zisserman [78]. A neighborhood pattern \( p_m \) is a vector containing the quantized descriptors \( c \) of a point \( p_m \) and all of its neighbors \( n_m \). We use the clustering method described in [78] to find commonly recurring neighborhoods across the training images.
To find distinctive starting points that also correspond to named objects, the initialization module finds associations between a caption word $w$ and a cluster of similar neighborhoods $\mathcal{N} = \{p_m\}$, using $\text{Corr}(w, \mathcal{N})$ in Equation 4.7 above.\(^1\) Neighborhoods within a cluster $\mathcal{N}$ that strongly correspond to a word $w$ are likely to spatially overlap the object referred to by $w$. We thus construct seed appearance models potentially corresponding to $w$, using recurring points and their spatial relationships within $\mathcal{N}$. Since the simplest detectable appearance model is a single pair of vertices, we identify neighboring pairs of points with consistent spatial relationships that have appeared in the elements of $\mathcal{N}$ in more than one distinct image. The pair with the lowest summed energy (Equation 4.4) across all detections is adopted as a seed appearance model.

### 4.3.3 Iterative Improvement

The improvement stage iteratively makes changes to a seed object model, guided by the correspondence between caption words and models. More specifically, the improvement algorithm starts with a seed model for a given word, makes a simple modification to this model (e.g., adds a new vertex), and detects instances of the new model in the training images. The new model is accepted as a better object detector under either of two conditions: it has a stronger correspondence with the word (according to Equation 4.7), or it has the same correspondence, but a lower total detected energy (according to Equation 4.4). On alternating iterations, the algorithm randomly tries to either adjust the energy threshold, or expand the model by adding a vertex or an edge. For the former, the threshold is increased (decreased) by a small percentage ($\pm 25\%$), in order to leave out false detections or bring in true instances. When expanding the model, candidates for addition include: (i) points that consistently appear in the neighborhoods of the currently-detected vertices—to be added as new vertices; and (ii) consistently appearing pairwise spatial relationships between the currently-detected vertices—to be added as new edges.

\(^{1}\)For this, we replace $q_{G'}$ in (4.7) with $q_{\mathcal{N}'}$ indicating the occurrences of $\mathcal{N}'$ in the training images.
The algorithm first tries the addition that would result in the greatest decrease in total energy over the currently detected instances. This gives priority to neighboring points that are more common and have the most consistent spatial relationships with detected model vertices. Since adding a vertex to the model also adds an edge, it is often associated with a greater decrease in energy than adding a new internal edge. Consequently, the algorithm tends to favor expanding the scope of the model over adding spatial constraints. Nonetheless, if a relationship between vertices consistently appears in many detected instances, it will have priority over a point that appears in few instances.

The initial two-vertex model often is not large enough to encode the visual structure of the initial neighborhood cluster $N$ and so may occur in many other distracting contexts. In order to give the model a chance to grow to better-characterize its seed context, the first few iterations of the improvement stage are confined to search for the model in the starting cluster of neighborhoods, $N$, rather than the complete training set.

### 4.4 Using Models to Annotate New Instances

For evaluation, we use the models our algorithm has learned from training image–caption pairs to detect and annotate new instances of objects in previously unseen (and uncaptioned) test images. For detection of new instances, we use the same algorithm we use in learning. To annotate a detected instance of an object, we use the word associated with the learned object model.

In Section 4.4.1 we explain our choices for the parameters of our algorithms. We then compare results with our previous work [45] by looking into the performance of our learning and annotation methods on a data set of 228 real images of toys in Section 4.4.2. Finally, we present the results of applying our methods to a larger and more challenging set of 1240 real-world images from the web in Section 4.4.3.
4.4.1 Parameter Settings

For our image representation described in Section 4.2.1, we use a cluster set of size 4000 to generate the quantized descriptors $c_m$ associated with each image point. We set the neighborhood size $|n_m|$ to 50 since this was empirically found to be an appropriate tradeoff between having distinctiveness and locality in a neighborhood. Recall from Section 4.2.2 that in an appearance model, each vertex is associated with a vector of neighboring cluster centers, $c_i$; we set $|c_i| = 10$ to minimize the chance of missing a matching feature due to quantization noise.

We set the parameters of the word–model correspondence measure, $\text{Corr}(w, G)$, according to the following assumptions: Reflecting high confidence in the captions of training images, $P(r_{wi} = 1|s_i = 1) = 0.95$ and $P(r_{wi} = 1|s_i = 0) = 0.05$. During model learning, $P(q_{Gi} = 1|s_i = 0) = 0.01$ in order to decrease the likelihood of false detections. When evaluating correspondences between words and neighborhoods to find good seed models, we set this target false-positive rate higher at 0.05, as we do not want to overlook potentially-strong seed models.

At the initialization stage, all word–model pairs with $\text{Corr}(w, G) > 0$ are selected as seed models, further modified in up to 100 stages of iterative improvement. For annotation of new instances, we only use learned models $G$ that are considered reliable according to our correspondence measure ($\text{Corr}(w, G) \geq 10$ for some word $w$). Since we consider precision more important than recall for annotation, we set the energy threshold during annotation to twice the threshold learned during training. The value of this threshold could be determined by a user, depending on whether they desire more detections (higher recall) or fewer detections with higher confidence (higher precision).

4.4.2 Experiments on Toy Images

We use the 128 training and 100 test images described in [45]. Each image in this data set contains 3 or 4 toy objects (out of a pool of 10), arranged in different poses, and partially
occluded in many cases. Each training image is paired with a manually-generated caption that contains names of all objects in the image plus a few distractor names.

Figure 4.2 shows the precision–recall curves of both the new detection method and that of [45] on the test images. Our new method consistently demonstrates substantially higher annotation precision for equivalent recall; in addition, we learn strong models for 3 of the objects for which the original method of [45] was unsuccessful. These results confirm our hypothesis that we can build more distinctive object models by finding recurring spatial relationships among the recurring image points. Improved performance is also due to the choice of better starting points for model construction. Moreover, our simpler correspondence measure speeds up the training phase by a factor of 10 (∼1–2 hours for the new method vs. ∼22 hours for that of [45]).

4.4.3 Experiments on Web Data

The web data set includes images of National Hockey League (NHL) players and games, with associated captions, downloaded from various web sites, and randomly divided into 850 training and 390 test image–caption pairs. About a third of the captions are full sentence descriptions (as in Figure 4.1, page 62), whereas the remainder simply name the two teams involved in
the game (e.g., ‘Maple Leafs vs Senators’). Most images are on-ice shots and display multiple players in a variety of poses and scales.

We automatically process captions of the training images, removing capitalization, punctuation, and plural indicators, and dropping words that occur in less than 1% of the captions. Since NHL teams are referred to by both their team name (e.g., ‘Bruins’) and their city name (e.g., ‘Boston’), we treat an occurrence of either as an instance of the team name. This link between team names and city names is the only prior knowledge available to our learning algorithm. The final vocabulary extracted from the training captions contains 105 words, of which only 30 are team designations. Note that the algorithm has only these caption words and the contents of the images to guide its search for meaningful word–appearance pairs.

From the training image–caption pairs, our algorithm learns one or more strong appearance models (i.e., those with $\text{Corr} \geq 10$) for 9 team names (out of 30). The strength of learned models is highly influenced by the number of times the object appears in the training set. The 9 teams for which strong models are learned are on average mentioned in 108 captions, whereas the other 21 teams are on average mentioned in 34 captions. In addition, a team does not have to be visible in an image in order to be mentioned in the caption. Thus, in most cases, fewer instances of an object are accessible to the learning algorithm. For instance, the team name Vancouver Canucks is mentioned in 52 training captions but players only appear in 20 images and the logo is only visible in 14 of these.

Of the 9 teams with strong learned models, 8 were detected in the test images. Figure 4.3 shows a test image with a detection of a learned appearance model associated with the Toronto Maple Leafs. Figure 4.4 shows several other detections of learned appearance models (and their associated team names) on test images. Table 4.1 gives the precision and recall of detections of each of the 8 teams, calculated based on whether the model predicted the presence of the correct word in the corresponding test caption. (Note that we use the test captions only for evaluation.) Test image annotation generally has high precision but low recall. This reflects the fact that teams mentioned in the captions are not always visible and that a hockey player has
a highly variable appearance depending on viewing angle and pose. A model that captures the appearance of the front logo will not help annotate a view of a player from the side.

Figure 4.5 illustrates several interesting types of model detections. Part (a) demonstrates that the learned appearance models can be detected across a wide range of scales. In (b), the detected Buffalo Sabres shoulder-patch model is an example of multiple distinct models being associated with a single word (cf. Figure 4.4(d)). The model detection on the left of (c) shows the appearance of an advertisement along the boards of the Minnesota Wild arena, learned as a second model for this team name. (d) shows a detection of the Minnesota Wild logo and several background detections associated with the word ‘vs’. Due to the probabilistic nature of our algorithm, it learns strong associations between background features in images, such as parts the net, and function words, such as vs and the. These types of detections can be easily avoided by ignoring words that occur frequently across many captions, which are less distinctive.
Figure 4.4: Sample detections of team logos in test images.
Table 4.1: Precision and recall of detection for 8 team names in test images, and the number of test captions each name appears in.

<table>
<thead>
<tr>
<th>Name</th>
<th>Precision</th>
<th>Recall</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tampa Bay Lightning</td>
<td>0.92</td>
<td>0.73</td>
<td>48</td>
</tr>
<tr>
<td>Maple Leafs</td>
<td>1.00</td>
<td>0.15</td>
<td>80</td>
</tr>
<tr>
<td>Minnesota Wild</td>
<td>1.00</td>
<td>0.22</td>
<td>51</td>
</tr>
<tr>
<td>New York Islanders</td>
<td>1.00</td>
<td>0.20</td>
<td>20</td>
</tr>
<tr>
<td>Buffalo Sabres</td>
<td>1.00</td>
<td>0.14</td>
<td>22</td>
</tr>
<tr>
<td>Chicago Blackhawks</td>
<td>0.67</td>
<td>0.16</td>
<td>37</td>
</tr>
<tr>
<td>Dallas Stars</td>
<td>1.00</td>
<td>0.08</td>
<td>50</td>
</tr>
<tr>
<td>Ottawa Senators</td>
<td>1.00</td>
<td>0.05</td>
<td>20</td>
</tr>
</tbody>
</table>

Figure 4.5: Some interesting detections in test images.
4.5 Conclusions

We have proposed an unsupervised method that uses language both to discover salient objects and to construct distinctive appearance models for them, from cluttered images paired with noisy captions. The algorithm simultaneously learns appropriate names for these object models from the captions. We have devised a novel appearance model that captures the common structure among instances of an object by using pairs of points together with their spatial relationships as the basic distinctive portions of an object. We have also introduced a novel detection method that can be reliably used to find and annotate new instances of the learned object models in previously unseen (and uncaptioned) test images. Given the abundance of existing images paired with text descriptions, such methods can be very useful for the automatic annotation of new or uncaptioned images, and hence can help in the organization of image databases, as well as in content-based image search.
Chapter 5

Probabilistic Appearance Models

Manual annotation of new images in large image collections is prohibitively expensive for commercial databases, and overly time-consuming for the home photographer. However, low-cost imaging, storage and communication technologies have already made accessible millions of images that are meaningfully associated with text in the form of captions or keywords. It is tempting to see these pairings of visual and linguistic representations as a kind of distributed Rosetta Stone from which we may learn to automatically translate between the names of things and their appearances. Even limited success in this challenging project would support at least partial automatic annotation of new images, enabling search of image databases by both image features and keywords that describe their contents.

Any such endeavor faces the daunting challenge of the perceptual grouping problem. Regardless of the type of image feature used, a word typically refers not to a single feature, but to a configuration of features that form the object of interest. The problem is particularly acute since any given image may contain multiple objects or configurations; moreover, the meaningful configurations may be easily lost among a huge number of irrelevant or accidental groupings of features. Without substantial bottom-up grouping hints, it is a nearly hopeless task to glean the meaningful feature configurations from a single image–caption pair. Given a collection of images, however, one can look for patterns of features that appear much more of-
ten than expected by chance. Usually, though, only a fraction of these recurring configurations correspond to salient objects that are referred to by words in the captions. Our system searches for meaningful feature configurations that appear to correspond to a caption word. From these starting points it iteratively constructs flexible appearance models that maximize word–model correspondence.

Our approach is best-suited to learning the appearance of objects distinguished by their structure (e.g., logos or landmarks) rather than their color and texture (e.g., tigers or blue sky). By detecting the portions of an object with distinctive structure, we can find whether the object is present in an image and where (part of) the object appears, but we do not determine its full extent. Therefore our system is appropriate for annotation but only limited localization. Our specific focus is on learning correspondences between the names and appearances of exemplar objects from relatively noisy and complex training data rather than attempting to learn the more highly-variable appearance of object classes from less ambiguous training sets. However, our framework and the structure of our appearance model are designed to learn to recognize any objects that appear as multiple parts in a reasonably consistent configuration. We therefore believe that with the right choice of features, our framework could be adapted to learn the appearance of object classes such as cars, jets, or motorcycles.

5.0.1 Background

While many structured appearance models use features designed for object classes, our system uses features that are best suited to learning the appearance of exemplar objects (such as St. Paul’s Cathedral) rather than a broad class of objects (such as cathedrals in general). The world is full of exemplars, and there has been a great deal of work in sorting and annotating exemplar images, such as the method proposed by Simon et al. [75] for organizing collections of related photographs into labeled canonical views. While our current detection method is not as scalable as high-performance exemplar image retrieval systems such as that proposed by Philbin et al. [68], our use of language can improve text-based querying and link together
widely different appearances or views of a single exemplar.

The proposed learning algorithm here is an extension and a refinement of the algorithms presented in our previous work, [45, 46]. In [45] we represented appearance as an unstructured local collection of features and used a translation model to find correspondences between words and appearance. In [46] we added spatial relationships to the appearance model and introduced a more direct word–model correspondence measure. Here, we introduce a novel unified framework for evaluating both the goodness of a detection and the appropriateness of associating the detection with a caption word. We have also modified the improvement mechanism to learn more spatial relationships between features and present extensive new evaluation and analysis.

5.0.2 An Overview of Our Approach

The goal of this work is to annotate exemplar objects appearing in images of cluttered scenes, such as the images shown in Figure 5.1(a). A typical such image, with hundreds (or even thousands) of local features, contains a huge number of possible feature configurations, most of which are noise or accidental groupings. A complex configuration of features that occurs in many images is unlikely to be an accident, but may still correspond to common elements of the background or other unnamed structures. The only evidence on which to establish a connection between words and configurations of visual features is their co-occurrence across the set of captioned images. The key insight is that this evidence can guide not only the annotation of complex feature configurations, but also the search for meaningful configurations themselves. Accordingly, we have developed a novel algorithm that uses language cues in addition to recurring visual patterns to incrementally learn strong object appearance models from a collection of noisy image–caption pairs (as in Figure 5.1(a-c)). The result of learning is a set of exemplar object appearance models paired with their names, which can be used for annotating similar objects in new (unseen and uncaptioned) images; a sample annotation is shown in Figure 5.1(b).
Figure 5.1: (a-c) A sample input image–caption collection, where each image contains hundreds of local (SIFT) features (yellow crosses). From the input training collection, associations between structured subsets of local features and particular nouns are learned. (d) A sample output of our system, where the object (the Maple Leafs logo) is detected (shown with red features and green relationships in a yellow box), and annotated with its name (“Maple Leafs”). The annotation is performed using a word–appearance association discovered from the training image–caption collection.
5.1 Representing and Matching Objects

Our learning framework allows a one-to-many relationship between words and appearance models. It is thus not necessary that a single model capture object appearance from all possible viewpoints. Moreover, since we deal with exemplar objects, our method need not handle
the changes in texture and structural detail that are possible within a class of objects. In order to serve as a robust object detector, however, it is important that the appearance model representation be invariant to reasonable changes in lighting, scale, orientation, articulation and deformation. The representation must also be reliably detectable, in order to avoid false annotations.

We represent our images using easily-extractable local interest features that can be used reliably to detect exemplar objects in highly cluttered scenes. We represent small patches of an instance of an object using such local features, and capture its structure using the pairwise spatial relationships between the patches. From the object instances, we then construct an abstract object appearance model in the form of a graph, by modeling the recurrent local features as vertices and the recurrent spatial relationships between pairs of local features as edges. The model that is built this way is a reliable descriptor of the object appearance and at the same time flexible enough to handle common deformations. Details of our choices for the representation of images and objects are given in Sections 5.1.1 and 5.1.2, respectively. Given an object model (in the form of a graph), we need a method for detecting instances of the object in an image—that is, to find a matching between model vertices and local image features. Our detection algorithm is presented in detail in Section 5.1.3.

5.1.1 Image Representation

We represent an image as a set $I$ of local interest points $p_m$, i.e., $I = \{p_m | m = 1 \ldots |I|\}$, referred to hereafter as points or image points. These points are detected using Lowe’s SIFT method [58], which defines a point $p_m$ in terms of its Cartesian position $x_m$, scale $\lambda_m$ and orientation $\theta_m$. In addition to spatial parameters, for each point we also extract a feature vector $f_m$ that encodes a portion of the image surrounding the point. Since $f_m$ is extracted relative to the spatial coordinates of $p_m$, it is invariant to changes in position, scale and orientation. While our approach is not dependent on a particular point detection method or feature encoding, we use the PCA-SIFT feature encoding developed by Ke and Sukthankar [49] because it allows
for fast feature comparison and low memory requirements. This feature encoding is reasonably robust to lighting changes, minor deformations and changes in perspective. Since individual features capture small, independent patches of object appearance, the overall representation is robust to occlusion and articulation.

The continuous feature vector $f_m$ is supplemented by a quantized descriptor $c_m$ for each image point, in order to support the ability to quickly scan for potentially matching features. Following Sivic and Zisserman [78], we use the K-means algorithm to generate a set of cluster centers, $C = \{f_c | c = 1 \ldots |C|\}$, from a set of features randomly selected from a stock image collection. The index of the cluster center closest to $f_m$ is used as the descriptor $c_m$ associated with $p_m$.

In addition to describing each point individually, we also attempt to capture the local spatial configuration of points using neighborhoods that describe the local context of a point. Each point $p_m$ is associated with a neighborhood $n_m$ that is the set of its spatially closest neighbors $p_n$, according to the $\Delta x_{mn}$ distance measure taken from Carneiro and Jepson [20]:

$$\Delta x_{mn} = \frac{\|x_m - x_n\|}{\min(\lambda_m, \lambda_n)} \quad (5.1)$$

This normalized distance measure makes neighborhoods more robust to changes in scale, as newly-introduced fine-scale points are less likely to push coarse-scale points out of the neighborhood when the scale of an object increases.

To summarize, each image is represented as a set of points, $I = \{p_m | m = 1 \ldots |I|\}$, in which $p_m$ is a 6-tuple of the form $(f_m, x_m, \lambda_m, \theta_m, c_m, n_m)$. In addition, a vector of transformation-invariant spatial relationships $r_{mn}$ is defined between each pair of neighboring points, $p_m$ and $p_n$, including the relative distance between the two points ($\Delta x_{mn}$), the relative scale difference between them ($\Delta \lambda_{mn}$), the relative heading from $p_m$ to $p_n$ ($\Delta \phi_{mn}$), and the relative heading in the opposite direction ($\Delta \phi_{nm}$). That is, $r_{mn} = (\Delta x_{mn}, \Delta \lambda_{mn}, \Delta \phi_{mn}, \Delta \phi_{nm})$, where the spatial relationships are taken from Carneiro and Jepson [20], and are calculated as in Equations (5.1)
above, and (5.2) and (5.3) below:

\[
\Delta \lambda_{mn} = \frac{\lambda_m - \lambda_n}{\min(\lambda_m, \lambda_n)}
\]

(5.2)

\[
\Delta \phi_{mn} = \Delta \phi(\tan^{-1}(x_m - x_n) - \theta_m)
\]

(5.3)

where \(\Delta \phi(.) \in [-\pi, +\pi]\) denotes the principle angle. The spatial relationships are not part of the stored image representation, but are calculated on demand when object appearance models are being built (see Section 5.1.2) or detected (see Section 5.1.3).

\section*{5.1.2 Object Appearance Model}

An object model describes the distinctive appearance of an object as a particular set of local features that have a more-or-less structured arrangement. We represent this structured configuration of features as a graph \(G = (V, E)\). Each vertex \(v_i \in V\) is composed of a continuous feature vector \(f_i\), and a cluster index vector \(c_i\) containing indices for the \(|c_i|\) nearest cluster centers to \(f_i\), \(i.e., v_i = (f_i, c_i)\). Associating each model vertex with a set of clusters allows for fast comparison of features during model detection while minimizing the effects of quantization noise. Note that model vertices, unlike image points, do not include spatial information (\(i.e.,\) position, orientation, and scale), because the model must be invariant to translation, rotation, and scale. Each edge \(e_{ij} \in E\) encodes the expected spatial relationship between two vertices \(v_i\) and \(v_j\), in four parts: \(e_{ij} = (\Delta x_{ij}, \Delta \lambda_{ij}, \Delta \phi_{ij}, \Delta \phi_{ji})\) (as defined in Equations (5.1)–(5.3) above).

We assume objects are spatially coherent, and hence two nearby points on an object are expected to be related non-accidentally, \(i.e.,\) through a geometric relation. Thus in our framework only a connected graph is considered to be a valid object appearance model; edges are allowed between any pair of vertices (and thus models are not restricted to trees). Models that can encode all the non-accidental relationships between pairs of nearby image points are generally more distinctive, and more robust to occlusion and inconsistent point detection, than models that are restricted to trees.
The parameters of a model’s vertices (\(f_i\) and \(c_i\)) as well as those of the edges (\(\Delta x_{ij}\), \(\Delta \lambda_{ij}\), \(\Delta \phi_{ij}\), and \(\Delta \phi_{ji}\)) are calculated from the corresponding parameters of the images (points and the spatial relationships between them) through an iterative process of model construction and detection (see Sections 5.1.3 and 5.2).

### 5.1.3 Detecting Instances of an Object

We use a heuristic algorithm that searches for high-confidence instances of an appearance model in an image. Though each model is intended to robustly describe an object’s appearance, no observed instance of an object is expected to fit its model exactly. Deformation, noise, and changes in perspective can distort the features encoded at points and/or the spatial relationships between them. Our detection algorithm should thus be capable of finding partial matches of a model (representing some visible part of an object). At the same time, the algorithm should distinguish a partial match which has only a few observed vertices from an accidental collection of background elements. Our algorithm thus assigns, to each observed instance of a model, a confidence score that meets the above requirements, thereby determining how well the instance fits the model. Below we first explain our proposed detection confidence score, and then present the details of our detection algorithm.

**Detection Confidence**

An observed instance \(O\) of an object appearance model \(G\) is a set of vertex–point associations, \(O = \{(v_i, p_m) | v_i \in V, p_m \in I\}\), where \(O\) defines a one-to-one mapping between a subset of the model vertices and some subset of points in an image. Our detection confidence score defines the goodness of fit between a model \(G\) and an observation \(O\) as the likelihood of \(O\) being a true instance of \(G\) and not a chance configuration of background features.

Though each model is intended to robustly describe an object’s appearance, no observed instance of an object is expected to fit its model exactly. Deformations, noise, and changes in perspective can distort the features encoded at local interest points and/or the spatial relation-
ships between them. Our score thus should not require the detection of all the model vertices. Alternatively, if only a few vertices survive and they are less distinctive, it may be difficult to confidently distinguish an instance of the model (representing some part of a real object) from an accidental collection of background elements.

Let $H_G$ be the hypothesis that $O$ is a true instance of the appearance model $G$, while $H_B$ is the competing hypothesis that $O$ is a chance assortment of background features. The detection confidence is:

$$
\text{Conf}_{\text{detect}}(O, G) = \frac{p(O|H_G)P(H_G)}{p(O|H_G)P(H_G) + p(O|H_B)P(H_B)}
$$

where $p(O|H_B)$ is the background likelihood of $O$, and $p(O|H_G)$ is the model likelihood of $O$. We use $P(.)$ to indicate probability functions and $p(.)$ to indicate probability density functions. Detection confidence of $G$ can be viewed as equivalent to $P(H_G|O)$ in a simplified world where $H_G$ and $H_B$ are the only possible explanations for the observed points $O$. Ideally, $\text{Conf}_{\text{detect}}(O, G)$ would take into account other hypotheses such as partial occlusion of $G$ or textured backgrounds. However, such elaborations would be computationally expensive, so we use the following, relatively simple approach.

Equation 5.4 can be rewritten as:

$$
\text{Conf}_{\text{detect}}(O, G) = \frac{p(O|H_G)P(H_G)}{p(O|H_G)P(H_G) + p(O|H_B)P(H_B)}
$$

Thus the detection confidence can be calculated from the prior likelihood ratio, $P(H_G)/P(H_B)$, and the observation likelihood ratio, $p(O|H_G)/p(O|H_B)$. The prior likelihood ratio is set to a fixed value, empirically determined from experiments on a held-out subset of the training data (see the Appendix). Next, we explain how we estimate the observation likelihood ratio from a set of training image–caption pairs.

The background likelihood of an observation $O$ (i.e., $p(O|H_B)$) is independent of the model $G$ and depends only on the features $f_m$ of the observed points and their spatial relationships $r_{mn}$. In other words, $p(O|H_B)$ can be estimated from the background feature probability,
\( p(f_m|H_B) \), and the background distribution of spatial relationships, \( p(r_{mn}|H_B) \). \( p(f_m|H_B) \) reflects how common a feature is across a set of stock images with a wide variety of objects, while \( p(r_{mn}|H_B) \) reflects the distribution of relationships observed between neighboring points across the stock image set.

According to the background hypothesis, all point feature vectors \( f_m \) are i.i.d. (independent and identically distributed). While some local features represent one of a few very common visual patterns, other local feature values are quite rare, and therefore more distinctive. A Gaussian mixture model (GMM) allows us to approximate such a structured background feature distribution:

\[
p(f_m|H_B) = \sum_{c=1}^{|C|} \omega_c \cdot n(f_m|f_c, \sigma_c)
\]

where \( \omega_c \) is a weight term (\( \sum \omega_c = 1 \)) and \( n(f_m|f_c, \sigma_c) \) is a multivariate normal distribution with mean \( f_c \) and diagonal covariance values \( \sigma_c^2 \). Each mean \( f_c \) is a member of the cluster centroid set \( C \), found using K-means as explained in Section 5.1.1 above. Given these fixed means, the weights \( \omega_c \) and standard deviations \( \sigma_c \) are determined using the EM algorithm on the same (large) set of stock images that we use to find clusters in \( C \). This approach provides a smoother background feature distribution than using the statistics of the K-means clusters directly and is less computationally expensive than full GMM clustering.

According to the background hypothesis, all spatial relationship vectors \( r_{mn} \) between neighboring points are also i.i.d. Since histograms of relative distance (\( \Delta x_{mn} \)) and relative scale (\( \Delta \lambda_{mn} \)) are unimodal in the stock set, we model them with a normal distribution:

\[
p(\Delta x_{mn}|H_B) = n(\Delta x_{mn}|\mu_{xB}, \sigma_{xB})
\]
\[
p(\Delta \lambda_{mn}|H_B) = n(\Delta \lambda_{mn}|\mu_{\lambda B}, \sigma_{\lambda B})
\]

where the means (\( \mu_{xB} \) and \( \mu_{\lambda B} \)) and standard deviations (\( \sigma_{xB} \) and \( \sigma_{\lambda B} \)) are the sample statistics for pairs of neighboring points in the stock image set. For the two relative heading terms (\( \Delta \phi_{mn}, \Delta \phi_{nm} \)) we did not observe any tendency to a particular value in the stock image set, hence we model them as uniformly distributed.
Calculation of the model likelihood of an observation, \( p(O|H_G) \), is more complex, since the components of the appearance model are not identically distributed. In order to account for model vertices that are occluded or lost due to inconsistent interest point detection, we assume each vertex \( v_i \in V \) is observed with probability \( \alpha_v \).\(^1\) In matching an image point \( p_m \) to a model vertex \( v_i \), we do not require the feature vector \( f_m \) of \( p_m \) to be precisely equal to \( f_i \), but we assume that \( f_m \) is normally distributed with mean \( f_i \) and standard deviation \( \sigma_f \). This vertex feature deviation \( \sigma_f \) is approximately equal to the background’s mean cluster variance \( \sigma_c \). While a graph vertex \( v_i \in V \) defines the expected feature vector of a model observation, a graph edge \( e_{ij} \in E \) defines the expected spatial relationships between certain pairs of observed points. If \( O \) contains observations \( (v_i, p_m) \) and \( (v_j, p_n) \) and \( e_{ij} \in E \), then the elements of the spatial relationship \( r_{mn} \) are independent and normally distributed with means \( (\Delta x_{ij}, \Delta \lambda_{ij}, \Delta \phi_{ij}, \Delta \phi_{ji}) \) and standard deviations \( (\sigma_x, \sigma_\lambda, \sigma_\phi, \sigma_\phi) \). If the model does not specify a relationship between the vertices \( (e_{ij} \notin E) \), then the elements of \( r_{mn} \) are distributed according to the background hypothesis.

From the above formulations for the background and model likelihoods of an observation, we can calculate the observation likelihood ratio as:

\[
\frac{p(O|H_G)}{p(O|H_B)} = \prod_{v_i \notin O} (1 - \alpha_v) \prod_{(v_i, p_m) \in O} \alpha_v \frac{p(f_m|f_i)}{p(f_m|H_B)} \prod_{e_{ij} \in O} \frac{p(r_{mn}|e_{ij})}{p(r_{mn}|H_B)} \tag{5.9}
\]

where \( p(f_m|f_i) \) and \( p(r_{mn}|e_{ij}) \) reflect how well the observed points and their spatial relationships match the values expected by the appearance model \( G \) (as explained in the paragraph above). We consider \( e_{ij} \) to be in \( O \) if both \( (v_i, p_m) \in O \) and \( (v_j, p_n) \in O \). Note that the observation likelihood ratio takes into account the size of the observed and unobserved portions of the model, the background likelihood of the observed features and spatial relationships, as well as how well the observed points and spatial relationships fit \( G \).

\(^1\)More accurately, we treat \( \alpha_v \) not as the empirically observed probability of model vertex survival, but rather as what we would like it to be (i.e., as a user-defined parameter to guide detection). We chose \( \alpha_v = 0.5 \) because it is a reasonable compromise between a high \( \alpha_v \), which requires that almost all elements of a model be reproduced, and a low \( \alpha_v \), where a great majority of the model vertices can be absent with little impact on detection confidence.
One drawback of equation 5.9 is that it will over-count the contribution of matching spatial configuration for graphs with many edges. This is because the relative spatial position of a vertex with more than one edge is over-determined. In effect, the parameters of coincident edges are not independent, but are treated as such in order to simplify the calculation. This strengthens the detection confidence of highly-connected appearance models, so that a relative small portion of the model can have high detection confidence if the image points happen to have the right configuration.

**Detection Algorithm**

To detect an instance of an object model in an image, we need to find a high-confidence association between the model vertices and image points. Given that a typical image contains thousands of points, determining the optimal association is potentially quite expensive. We thus propose a greedy heuristic that efficiently searches the space of possible associations for a nearly-optimal solution. Individual vertices of a model may be unobserved (e.g., due to occlusion), and therefore some edges in the model may also not be instantiated.

Our detection heuristic thus allows for a connected model to be instantiated as disconnected components. To reduce the probability of false detections, the search for disconnected parts is confined to the neighborhood of observed vertices, and isolated singleton points are ignored. That is, in a valid model instance, each observed vertex shares a link with at least one other observed vertex. Also, our detection algorithm only reports those observations $O$ with a detection confidence greater than a threshold, i.e., $\text{Conf}_{\text{detect}}(O, G) \geq T_d$. We set $T_d$ to be quite low so that potentially meaningful detections are not overlooked.

Figure 5.3 and algorithm 2 present the greedy heuristic that detects instances of an appearance model $G$ within the image representation $I$. The actual implementation can detect more than one instance of $G$ within $I$ by suppressing points in previously detected instances. The first step is to find the set $\mathcal{A}$ of potential associations between image points and model vertices, that is, all vertex–point pairs whose match $p(f_m | f_i)$ is higher than expected by chance. The al-
Figure 5.3: A model and a set of points (a) may have many potential vertex-point associations $A$ and potential edges $L$ (b). As the set of observed correspondences, $O$, expands, incompatible associations are pruned (c). Finally, there are no more available correspondences (d).
algorithm then calculates the set of potential model edges $\mathcal{L}$ linking these vertex–point pairs. The set of observed correspondences, $O$, is assembled across several iterations by greedily adding the vertex–point pairs from $\mathcal{A}$ that maximize the detection confidence, $\text{Conf}_{\text{detect}}(O,G)$. Each time $O$ is expanded, elements of $\mathcal{A}$ and $\mathcal{L}$ that are incompatible with the current observed set are pruned away. The greedy expansion of $O$ continues until there are no available correspondences that could increase $\text{Conf}_{\text{detect}}(O,G)$.

**Algorithm 2** Detects instances of $G$ in $I$

FindModelInstance($G,I$)

1. Find the set $\mathcal{A}$ of all potential vertex–point associations:

   $$\mathcal{A} = \{(v_i,p_m) | c_m \in c_i, \ p(f_m | f_i) > p(f_m|H_B)\}$$

2. Find the set $\mathcal{L}$ of all potential links between elements of $\mathcal{A}$ ($\mathbf{n}_m$ is the set of neighbours of $p_m$):

   $$\mathcal{L} = \{((v_i,p_m),(v_j,p_n)) | p_n \in \mathbf{n}_m, \ p(r_{mn}|e_{ij}) > p(r_{mn}|H_B)\}$$

3. Set the initial instance $O$ to the pair $\{(v_i,p_m),(v_j,p_n)\}$, such that the link $((v_i,p_m),(v_j,p_n)) \in \mathcal{L}$ and $\text{Conf}_{\text{detect}}(O,G)$ is maximum.

4. Prune incompatible associations: Remove $(v_i,p_m)$ from $\mathcal{A}$ if either $v_i$ or $p_m$ are part of another vertex–point association $\in O$.

5. Prune incompatible links: Remove $((v_i,p_m),(v_j,p_n))$ from $\mathcal{L}$ if either end was removed from $\mathcal{A}$.

6. Let $\mathcal{A}_{\text{adj}}$ be the subset of $\mathcal{A}$ that is linked through $\mathcal{L}$ with an element of $O$.

7. If $\mathcal{A}_{\text{adj}}$ contains associations that could increase $\text{Conf}_{\text{detect}}(O,G)$, add to $O$ the association that leads to the greatest increase, and go to step 4.

8. Let $\mathcal{L}_{\text{neigh}}$ be the subset of $\mathcal{L}$ within the union of the neighborhoods of observed points in $O$.

9. If $\mathcal{L}_{\text{neigh}}$ contains observed links that could increase $\text{Conf}_{\text{detect}}(O,G)$, add to $O$ the pair of associations with the link that produces the greatest increase, and go to step 4.

10. Return $(O,\text{Conf}_{\text{detect}}(O,G))$. 
5.2 Discovering Word–Appearance Associations

We propose an unsupervised learning algorithm that builds structured appearance models for the salient objects appearing in a set of training image–caption pairs. Salient objects are those that appear in many images, and are often referred to in the captions. Because each image contains many features of non-salient objects, and each caption may contain words irrelevant to the displayed objects, the algorithm has to discover which image features and words are salient. The algorithm learns object models through discovering strong correspondences between configurations of visual features and caption words. The output is a set of appearance models, each associated with a caption word, which can be used for the annotation of new images.

The learning algorithm has two stages. First, an initialization stage determines a structured seed model for each caption word, by finding recurring neighborhoods of features that also co-occur with the word. Second, an improvement stage iteratively expands each initial seed model into an appearance model that covers a larger portion of the object of interest, and at the same time is more strongly associated with the corresponding caption word. The two stages of learning use a novel measure of correspondence between a caption word and an appearance model, which is explained in Section 5.2.1. We then explain the initialization and improvement stages of the learning algorithm in Sections 5.2.2 and 5.2.3, respectively.

5.2.1 Word–Appearance Correspondence Confidence

Here, we explain the word–appearance correspondence score used by our learning algorithm. The learning algorithm seeks pairs of words and appearance models that are representations of the same object in different modalities (linguistic and visual). We assume that both the word and the model instance are present in an image because the object is present. We thus define the correspondence score as a measure of confidence that a given appearance model is a reliable detector for the object referred to by a word. In other words, the correspondence score reflects the amount of evidence, available in a set of training images, that a word and an object model...
are generated from a common underlying source object.

Consider a set of \( k \) (training) captioned images. We represent the occurrence pattern of a word \( w \) in the captions of these images as a binary vector \( r_w = \{ r_{wi} | i = 1, \ldots, k \} \). Similarly, we use a binary vector, \( q_G = \{ q_{Gi} | i = 1, \ldots, k \} \) to indicate, for each training image, whether it contains at least one true observation of model \( G \). However, even if we detect model \( G \) in the \( i^{th} \) training image, we cannot be certain that this is a true observation of \( G \) \( (q_{Gi} = 1) \) instead of a random assortment of background features \( (q_{Gi} = 0) \). Therefore, we treat \( q_{Gi} \) as a hidden variable and associate it with an observed value, \( o_{Gi} \in [0, 1] \), that reflects the likelihood of model \( G \) being present in image \( i \). We set \( o_{Gi} \) to the maximum of the detection confidence scores, \( \text{Conf}_{\text{detect}}(O, G) \), over all the detected instances of a given object model \( G \) in a given image \( i \). By using the maximum detection confidence, we focus on learning appearance models that are individually distinctive, potentially allowing us to recognize an object even if large portions are occluded.

It is always possible that the word occurrence pattern, \( r_w \), and the observed confidence pattern, \( o_G = \{ o_{Gi} | i = 1, \ldots, k \} \), are independent (the null hypothesis or \( H_0 \)). Alternatively, instances of the word \( w \) and model \( G \) may both derive from a hidden common source object (the common-source hypothesis or \( H_C \)). According to \( H_C \), some fraction of image–caption pairs contain a hidden source \( s \), which may emit the word \( w \) and/or the appearance model \( G \). The existence of the appearance model in turn influences our observed confidence values, \( o_{Gi} \). We define the correspondence between \( w \) and \( G \) as the likelihood \( P(H_C | r_w, o_G) \), with the important simplification that we assume \( H_C \) and \( H_0 \) are the only possible explanations. This excludes possibilities such as indirection connection between \( G \) and \( w \), or a dependency between the two terms that is too slight to form the basis for a word–appearance pair. Equation (5.10) below presents the simplified scenario:

\[
\text{Conf}_{\text{corr}}(G, w) = \frac{P(H_C | r_w, o_G)}{p(r_w, o_G | H_C)P(H_C) + p(r_w, o_G | H_0)P(H_0)}.
\]
where:

\[ p(r_w, o_G | H_C) = \prod_i \sum_{s_i} P(s_i) P(r_w | s_i) p(o_G | s_i) \tag{5.11} \]

\[ p(r_w, o_G | H_0) = \prod_i P(r_w) p(o_G) \tag{5.12} \]

where \( s_i \in \{0, 1\} \) represents the presence of the common source in image-caption pair \( i \). To calculate the likelihoods of the observed confidence values under the two competing hypotheses \((p(o_G | s_i) \text{ and } p(o_G))\) we marginalize over the unobserved variable \( q_{Gi} \):

\[ p(o_G | s_i) = p(o_G | q_{Gi} = 1) P(q_{Gi} = 1 | s_i) \]

\[ + p(o_G | q_{Gi} = 0) P(q_{Gi} = 0 | s_i) \tag{5.13} \]

\[ p(o_G) = p(o_G | q_{Gi} = 1) P(q_{Gi} = 1) \]

\[ + p(o_G | q_{Gi} = 0) P(q_{Gi} = 0) \tag{5.14} \]

where we choose \( p(o_G | q_{Gi} = 1) = 2o_G \) (proportional to the detection confidence ). Similarly, we choose \( p(o_G | q_{Gi} = 0) = 2(1 - o_G) \). The intuition is that \( o_G \) is usually high when the model \( G \) is present in image \( i \), and is low when the model is not present.

To get the likelihood of observed data under \( H_C \), defined in Equations (5.11) and (5.13), we also need to estimate the parameters \( P(s_i) \), \( P(r_w | s_i) \), and \( P(q_{Gi} | s_i) \). \( P(r_w | s_i) \) and \( P(q_{Gi} | s_i = 0) \) are given fixed values according to assumptions we make about the training images, which we elaborate on in the Appendix. \( P(s_i) \) and \( P(q_{Gi} | s_i = 1) \) are given maximum likelihood estimates (MLEs) determined using expectation maximization over the training data. The MLEs for parameters under \( H_0 \) are more straightforward. \( P(r_w) \) in Equation (5.12) is the observed probability for word occurrence in the training data while \( P(q_{Gi}) \) is the inferred probability of model occurrence: \( \sum_i o_{Gi} / k \).

### 5.2.2 Model Initialization

The goal of the model initialization stage is to quickly find for each word a set of seed appearance models that are fruitful starting points for building strong object detectors. Later, the seed
object models are iteratively expanded and refined into larger and more distinctive appearance models using image captions as a guide. The existence of good starting points strongly affects the final outcome of the iterative improvement process. Nonetheless, a balance must be reached between time spent searching for good seeds and time spent in the improvement stage refining the seeds.

The most straightforward starting points are singleton features, but their relationship to an object may be too tenuous to provide effective guidance for building strong object models [45]. At the same time, trying all possible configurations of even a small number of features as seeds is impractical. The neighborhood pattern introduced by Sivic and Zisserman [78] roughly describes the appearance of a point’s local context as a bag of features. The neighborhood pattern is more distinctive than a singleton but less complex than our configurational models. Our initialization module uses neighborhood patterns with potentially meaningful word correspondences to help construct seed appearance models.

Recall that each point $p_m$ in an image has associated with it a vector of neighboring points $n_m$. The neighborhood pattern $\eta_m$ is a sparse binary vector that denotes which quantized feature descriptors $c$ are present in $p_m$’s neighborhood (including $c_m$). Thus $\eta_m$ roughly captures the types of feature vectors present within the small portion of the image centered at $p_m$. Two neighborhood patterns $\eta_m$ and $\eta_l$ are considered similar ($\eta_m \approx \eta_l$) if they have at least $t_\eta$ quantized feature descriptors in common. We use a modified version of the two-stage clustering method described in [78] to identify clusters of similar neighborhood patterns in the training images. The first stage identifies potential cluster centers with relatively low similarity threshold ($t_\eta = 6$) but requires that the quantized descriptor of the central feature of the neighborhoods match. The second pass that greedily forms the clusters has a higher similarity threshold ($t_\eta = 8$) but does not require a matching central descriptor.

Each resulting neighborhood cluster $N_m$ is a set of neighborhoods with patterns similar to $\eta_m$ ($N_m = \{n_l | \eta_l \approx \eta_m\}$). Each neighborhood cluster represents a weak, unstructured appearance context that occurs multiple times across the training image collection. We represent
the occurrence pattern of a given neighborhood cluster \( N \) in the training images as a binary vector \( q_N = \{ q_{N,i} | i = 1, \ldots, k \} \), where \( q_{N,i} = 1 \) if image \( i \) contains a member of \( N \), and zero otherwise.

The initialization module measures the association between each caption word \( w \) and neighborhood cluster \( N \), using the correspondence confidence score of Equation (5.10) above, i.e., \( \text{Conf}_{\text{corr}}(N, w) \). We assume that occurrences of a neighborhood cluster \( N \) that has a better-than-chance correspondence to a word \( w \) may spatially overlap the object referred to by \( w \). We therefore select for each word \( w \) the 20 neighborhood clusters \( N \) with the strongest correspondence score and attempt to extract from each cluster a seed appearance model with the best possible correspondence with \( w \). Since the simplest detectable and structured appearance model is a single pair of linked vertices, we search for two-vertex seed appearance models \( G \) that could explain the presence of frequently-occurring point pairs in \( N \) and that also have strong correspondence with \( w \).

Two neighboring points are considered to be a pair in a neighborhood cluster \( N \) if at least one member of the pair is in \( N \). We extract groups of “similar” pairs in \( N \), i.e., those that share the same appearance cluster pairs \((c_m, c_n)\). Each such group \( P \) is viewed as a set of observations for a potential two-vertex appearance model \( G \). For each \( P \) with members appearing in more than one image, we propose up to 20 distinct appearance models whose features and spatial relationships are randomly drawn (without replacement) from the pairs in \( P \). For each model \( G \), we calculate a score that reflects how well \( P \) supports model \( G \), by treating the \( K \) pairs in the group as observations \( O_k \):

\[
\text{Supp}(P, G) = \sum_{k=1, \ldots, K} p(O_k|H_G) \tag{5.15}
\]

We keep the model with the highest \( \text{Supp}(P, G) \) for each group of pairs. Then, for as many as 50 models from different pair groups with the highest support, we calculate \( \text{Conf}_{\text{corr}}(G, w) \).

The model with the highest correspondence confidence is selected as word \( w \)’s seed model for

\[\text{In our calculation of } \text{Conf}_{\text{corr}}(N, w), \text{ we replace } q_G \text{ with } q_N \text{ indicating the occurrences of } N \text{ in the training images.}\]
the current neighborhood cluster $\mathcal{N}$. Each starting seed model is therefore initialized from a different neighborhood cluster. While it is possible that different seed models may converge on the same appearance during the following iterative improvement stage, ideally the set of learned appearance models will cover different distinctive parts of the object and also provide coverage from a variety of viewpoints.

### 5.2.3 Iterative Improvement

The improvement stage iteratively makes simple changes to the seed object models found at the initialization stage, guided by the correspondence between caption words and models. More specifically, the improvement algorithm starts with a seed model $G$ for a given word $w$, makes a simple modification to this model (e.g., adds a new vertex), and detects instances of the new model $G'$ in the training images (using the detection algorithm presented in Section 5.1.3). The new model is accepted as a better object detector and replaces its predecessor if it has a higher correspondence score with $w$, i.e., if $\text{Conf}_{\text{corr}}(G', w) > \text{Conf}_{\text{corr}}(G, w)$. In other words, the improvement algorithm performs a greedy search through the space of appearance models to find a reliable object detector for any given word.

At each iteration, the algorithm tries to expand the current model by adding a new vertex and linking it with one of the existing vertices. Vertex candidates are drawn from points that fall within the neighborhood of the detected instances of the current model. To ensure a strong correspondence between the growing model and its associated word, only model detections that occur within images with the desired caption word $w$ are considered for this purpose. Consider a detected point $p_m$ that corresponds to a model vertex $v_i$. Each point $p_n$ that is in the neighborhood of $p_m$ but not part of the detection is a candidate to form a new vertex $v_j$. The corresponding edge candidate, $e_{ij}$, inherits the spatial relationship vector $r_{mn}$ between the two points $p_m$ and $p_n$. As in the initialization stage, from the observations and their neighboring points, we form groups $\mathcal{P}$ each containing observations of a potential one-vertex extension to the current model. The observations are pairs $(p_m, p_n)$ with their first points being observations
of the same existing vertex \( v_i \), and their second points sharing the same cluster index \( c_n \). For each pair group \( \mathcal{P} \), we propose up to 20 two-vertex incremental models \( \Delta G = (\{v_i, v_j\}, e_{ij}) \) that bridge the gap between the existing model and the new vertex. We keep the incremental model with the highest \( \text{Supp}(\mathcal{P}, \Delta G) \). The incremental models from different pair groups form a queue of potential (vertex, edge) additions, prioritized by their degree of support.

An augmented model, \( G' \), is constructed by removing the top incremental model \( \Delta G \) from the queue and incorporating it into the existing model \( (G' = G \cup \Delta G) \). If the augmented model does not improve the correspondence score with \( w \), then the change is rejected and the next iteration begins with the next candidate \( \Delta G \) to be tested. If the augmented model does improve the correspondence score, the change is accepted, and the algorithm attempts to establish additional edges between the existing vertices and the new vertex. The edge candidates are prioritized based on their support among detections of the new model \( G' \), as this reflects the number of times the two end point vertices have been observed together, as well as the consistency of their spatial relationship over those observations. New edges are tested sequentially and those that improve the correspondence score are added. Generally, if the model vertices have very consistent spatial relationships across the current detections, the model will tend to accept many edges. If the underlying object is more deformable or viewed from a variety of perspectives, fewer edges are likely to be accepted.

Once a new vertex is added and connected to the model, and additional edges are either accepted or rejected, a new iteration begins with the new model as the starting point. If none of the proposed model extensions are accepted, the model \( G \) paired with the caption word \( w \) is added to the set of discovered word–appearance associations to be used for future annotation of new images.
5.3 Evaluation: Annotating Objects in Uncaptioned Images

In previous sections, we have presented the components of our learning framework for discovering object models and their associated names from a set of training image–caption pairs. Ultimately, we want to use the discovered model–word pairs to detect and annotate new instances of the objects in unseen and uncaptioned images. For detection of new instances of a given object model, we use the detection algorithm presented in Section 5.1.3. Our confidence that a word \( w \) is appropriate to annotate a detected object \( O \) in an image depends on two factors: (i) our confidence that the observed instance \( O \) is a true instance of the given model \( G \) and not a chance configuration of background features—\( \text{i.e.} \), the detection confidence \( \text{Conf}_{\text{detect}}(O, G) \); and (ii) our confidence that the appearance model \( G \) and the word \( w \) represent the same object—\( \text{i.e.} \), the correspondence confidence \( \text{Conf}_{\text{corr}}(G, w) \). We can thus associate an annotation confidence to every object instance that is to be labeled in a new image. The annotation confidence is defined as the product of the detection confidence and the correspondence confidence, as in:

\[
\text{Conf}_{\text{annotate}}(O, w, G) = \text{Conf}_{\text{detect}}(O, G) \times \text{Conf}_{\text{corr}}(G, w) \tag{5.16}
\]

We annotate a new instance of an object only if the annotation confidence is greater than a threshold, \( \text{i.e.} \), if \( \text{Conf}_{\text{annotate}}(O, w, G) > T_a \). The value of \( T_a \) could be determined by a user, depending on whether they desire more detections (higher recall) or fewer detections with higher confidence (higher precision). In all experiments reported here, we set the threshold very high, \( \text{i.e.}, T_a = 0.95 \).

The following sections present the results of applying our learning, detection, and annotation algorithms to two types of data sets: a small set of real images of toys that we captured and annotated ourselves (Section 5.3.1) and two larger and more challenging sets of real-world images of sports scenes and landmarks, respectively, downloaded from the web (Section 5.3.2). Our choices for the parameters involved in the confidence scores and the algorithms are given in the Appendix.
5.3.1 Experiments on a Controlled Data Set

Here, we report the performance of our method applied to a set of 228 images of arrangements of children’s toys, generated under controlled conditions. The data set was first used in our experiments presented in an earlier paper [45]. Throughout this chapter, we refer to this data set as the TOYS data set. The original color photographs are converted to intensity images with a resolution of 800x600. Most images contain 3 or 4 toy objects out of a pool of 10, though there are a handful of examples of up to 8 objects. The objects are not arranged in any consistent pose and many are partially occluded. The images are collected against approximately 15 different backgrounds of varying complexity. The pool of 228 images is randomly divided into a training set of 128 and a test set of 100. Each training image is annotated with the unique keyword for each object of interest shown and between 2 and 5 other keywords uniformly drawn from a pool of distractor labels. Note that the objects of interest never appear individually and the training data contains no information as to the position or pose of the labeled objects.

Figure 5.4 displays some example images from the test set and the associated annotations produced by our system. False annotations are written in italics and missed annotations in parentheses. While there is no direct limit on the size of learned appearance models, they tend to cover small, distinctive patches of the objects. In many cases, the size of learned models is limited by the availability of sufficient repeatable visual structure. Some objects with large areas of sufficient detail, such as the ‘Cash’ object and the two books ‘Franklin’ and ‘Rocket’, can support larger models (as in Figure 5.4(d)). Our method learns multiple appearance models for many of the objects, such as the two models shown in Figure 5.4(b) for ‘Bongos’. It is thus possible to detect an object even when a distinctive part of it is occluded. The system can sometimes detect planar objects such as the ‘Rocket’ book under significant perspective distortion (as in Figure 5.4(c)).

Our earlier work ([45, 46]) has shown promising results on the TOYS data set, while using simpler appearance models and/or less efficient learning or detection algorithms. To evaluate the improvement in annotation performance, we performed a single training run for our method.
Figure 5.4: Sample detections of objects in the TOYS test set. False detections are shown in italics while missed detections are placed in parentheses.
on the same 128 training images used in previous work and compared results on the 100-image test set. Figure 5.5 shows the precision–recall curves for four systems: the current system (with and without spatial relationships), the ICCV system presented in [46], and the CVPR system described in [45]. To implement a system without spatial relations, we remove all spatial contributions to the detection confidence measure, $Conf_{detect}(O, G)$. Therefore, edge position and connectivity play no role in the detection mechanism. The only remaining spatial constraint in the resulting bag-of-features model is that each vertex added to a detection must fall within the neighborhood of a vertex that is already part of the detection (a constraint of the underlying detection algorithm). Our new system achieves the high annotation precision of ICCV with about 15\% higher overall recall due to our new detection confidence measure. Training without spatial relationships generates approximately a 12\% penalty in recall, indicating that more distinctive object models can be constructed by finding recurring spatial relationships among image points. While the CVPR system and the ‘No Spatial’ variant of our current system both represent appearance as an unstructured collection of local features, a variety of changes in the initialization mechanism, the representation of individual features and the detection and correspondence measures greatly improve overall performance.\footnote{The improved precision of the ICCV system over the CVPR system is due in part to the addition of spatial relationships to the appearance models, as well as to improvements in the correspondence confidence measure and initialization method.}

Table 5.1 shows the per-object precision and recall values of our current system. In this and subsequent tables, the Frequency column shows the number of captions within the test set that contain at least one instance of the corresponding word. All of the precision and recall values we report are based on word occurrence in the captions of the test set; if the system does not detect a word that appears in the caption, that instance is counted as a false positive, even if the named object does not actually appear in the image. The method performs best on objects that have large, roughly planar surfaces and distinctive structural details, such as ‘Rocket’, ‘Franklin’, ‘Drum’ and ‘Horse’. The only instances of these objects that cannot be detected with very high precision either are highly occluded or are viewed from an atypical
Figure 5.5: A comparison of precision–recall curves over the TOYS test set, for four systems: current, current without spatial relationships, ICCV [46], and CVPR [45]. CVPR used a local bag-of-features models that were initialized with singleton features. ICCV added spatial relationships and neighborhood initialization. The current system adds detection and annotation confidence scores and builds models with more spatial relationships.

angle (such as edge-on for a book). Our system has the greatest difficulty with the ‘Ernie’ and ‘Dino’ objects, perhaps because they lack fine-scale surface details and distinctive textures. For instance, the striped pattern of the shirt of the ‘Ernie’ doll is somewhat distinctive within the image set, but the face lacks sufficient keypoints to build a reliable model. The ‘Dino’ object is particularly difficult, as specular reflections significantly alter the local-feature description of its surface appearance depending on perspective and lighting.

Precision and recall indicate whether the system is detecting objects in the correct images, but not how often the models are detected on the objects themselves. To evaluate the locational accuracy of the detections we manually defined bounding boxes for all named objects in the test image set (this data was not available to the system). A detection was considered accurate if all of the detected model vertices were located within the correct bounding box. Overall, 98.8% of above-threshold detections on the TOYS data set are completely within the boundaries defined for the object. The lowest accuracy was for the ‘Franklin’ object, on which 5.2% of detections were partially outside the object bounds.
**Table 5.1**: Performance results on the TOYS test set; $T_a = 0.95$.  

### 5.3.2 Experiments on Web Data Sets

This section reports the performance of our system on two larger and more challenging sets of images downloaded from the web. The first set, which we refer to as the HOCKEY data set, includes 2526 images of National Hockey League (NHL) players and games, with associated captions, downloaded from a variety of sports websites. The second set, which we refer to as the LANDMARK data set, contains 3258 images of 27 well-known buildings and locations, with associated tags, downloaded from the Flickr website\(^4\). Due to space considerations, our analysis focuses mainly on the results on the HOCKEY data set (5.3.2 through 5.3.2), though most of the same phenomena also appear in the LANDMARK results (5.3.2).

**Annotation Performance on the HOCKEY Data Set**

The HOCKEY set contains examples of all 30 NHL teams and is divided into 2026 training and 500 test image–caption pairs. About two-thirds of the captions are full sentence descriptions, whereas the remainder simply name the two teams involved in the game (see Figure 5.1, page 82 for examples of each type). We automatically process captions of the training images,

\(^4\)www.flickr.com
removing capitalization, punctuation, and plural indicators, and dropping words that occur in less than 1% of the captions. Captions of the test images are only used for evaluation purposes.

Most images are on-ice shots and display multiple players in a variety of poses and scales. We thus expect our system to learn distinctive appearance models for the players of each team, and to discover meaningful associations between the models and the corresponding team names. A team’s logo is perhaps the most distinctive appearance model that our system could learn for the team. In addition, there may be other visual appearances that unambiguously identify a particular team, such as shoulder patches or sock patterns. Note that we do not incorporate any prior knowledge into our system about which objects or words are of interest. The system is expected to learn these from the pairings of the images and captions in the training data. The only information we provide to our system is a link between a team’s name (e.g., Bruins) and its city name (e.g., Boston) because most NHL teams are referred to by both names. Our system thus treats the two words (team name and city name) as the same word when learning model–word associations. The final vocabulary extracted from the training captions contains 237 words, of which only 30 are team designations. As we will see later, our system learns appearance models for many of these words, including team names as well as other words.

We experiment with different degrees of spatial constraints imposed by our detection algorithm, in order to see how that affects the learning and annotation performance of our system. Our detection confidence score, presented in Section 5.1.3, has a set of parameters corresponding to the model spatial relationship variances. We set each of these to a fraction $1/\Gamma$ of the corresponding background variance, where $\Gamma$ is the spatial tolerance parameter that determines the amount of spatial constraints required by the detection algorithm to consider an observation as a true instance of a given model.

High values of $\Gamma$ thus imply a narrow acceptability range for spatial relationships between any two connected vertices of a model, resulting in tighter spatial constraints when detecting instances of the model. In contrast, a low value of $\Gamma$ translates into looser spatial constraints.
Figure 5.6: Precision–recall curves of our current system on test HOCKEY images, over a wide range of spatial settings. Although the system can detect objects even without spatial constraints (the ‘No Spatial’ case), the use of moderate spatial constraints (when $\Gamma = 10$) offers the best performance.

in detection. Figure 5.6 shows the overall precision–recall curves for various settings of the spatial tolerance parameter $\Gamma$, including a version in which the spatial relationships between points do not contribute to the detection confidence function (No Spatial). We use the notation $\Gamma_i$ to refer to an implementation of the system where $\Gamma$ is given the value $i$.

The curves indicate that the implementation that ignores spatial factors (No Spatial) and $\Gamma_1$ (where model spatial variances are identical to background spatial variances) are roughly equivalent. Remaining differences are due to the fact that $\Gamma_1$ can learn some edge connection structure while the ‘No Spatial’ model cannot. Very strong spatial constraints, as in $\Gamma_{100}$, may result in brittle model detections, but still the system is capable of producing reasonably good results (e.g., substantially better than a bag-of-features model; see Figure 5.6). Nonetheless, results confirm that moderate spatial constraints are generally more effective, as with $\Gamma_{10}$. One might expect that stronger spatial constraints (higher values of $\Gamma$) would always lead to higher precision. This is not necessarily the case, because even a configuration of relatively few observed vertices may have high detection confidence if their spatial relationships conform to
the model’s tight constraints. Moreover, many of the false annotations on test images are not the result of incorrect model detection at annotation time, but are due to learning a spurious word–appearance correspondence. Finally, a model that requires a rigid spatial configuration among its components may not grow to the size of a model that is more accommodating. The resulting smaller models may be less distinctive, even though each edge is more precise in capturing a particular spatial configuration.

Figure 5.7: Precision–recall curves for our current system are relatively unaffected by modest changes in the spatial tolerance parameter $\Gamma$.

To analyze the sensitivity of our method to the precise value of the spatial tolerance parameter $\Gamma$, we perform experiments with a smaller range of values for this parameter. Figure 5.7 shows the precision–recall curves for $\Gamma$ set to 5, 10, and 20. The results confirm that our method is not sensitive to small changes in the value of $\Gamma$. This might indicate that the iterative improvement process can optimize appearance models to compensate for different spatial tolerance values within a reasonable range. It is also possible that the models themselves are effective across a range of $\Gamma$ values, such that, for example, system performance would not be adversely affected if different values of $\Gamma$ were used for training and annotation.
CHAPTER 5. PROBABILISTIC APPEARANCE MODELS

An Analysis of the Annotation Results for the HOCKEY Set

Our results presented above show that $\Gamma_{10}$ has the best overall performance. We thus focus on $\Gamma_{10}$, analyzing various aspects of its performance. Table 5.2 shows the annotation performance of $\Gamma_{10}$ on the test images, focusing on the team names only. The system has high-confidence detections for 23 of the 30 teams. (There are 2 additional teams for which the system learns high-confidence models, but does not detect them in the test images.) Results show that the annotation of the test images generally has high precision but low recall. The low recall is partly because of our choice of a high annotation threshold, but also due to the fact that the captions that we use as ground truth often mention teams that are not visible in the image. In addition, a hockey player has a highly variable appearance depending on viewing angle and pose. A model that captures the appearance of the front logo will not help annotate a view of a player from the side.

An important factor that determines whether the system can learn high-confidence models for a team, and detect instances of it in the test images, is the number of training examples that contain both the team’s name and some appearance of it. It is hard for the system to learn a high-confidence model for a team if the team’s logo appears in a small number of training images. Even if the system learns a model for such a team, the model may be too specific to be useful for detecting new instances (which may differ in viewpoint, for example). On average, teams detected in the test set are mentioned in the captions of 114 training images, while teams with no test set detections are mentioned in the captions of 57 training images. The system detects only two teams (the Bruins and the Canadiens) with fewer than 60 caption-mentions, and fails to detect only one team (the Flyers) with more than 60 caption-mentions in the training set.

Note also that a team name’s mention in the caption of an image does not necessarily mean that the corresponding object (e.g., the team’s logo) appears in the image. After analyzing 15 of the teams in Table 5.2, we found that only 30–40% of the training images that mention a team’s name also contain a visible (less than half obscured) instance of one of the team’s
<table>
<thead>
<tr>
<th>Name</th>
<th>Precision</th>
<th>Recall</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tampa Bay Lightning</td>
<td>1.00</td>
<td>0.51</td>
<td>49</td>
</tr>
<tr>
<td>Los Angeles Kings</td>
<td>1.00</td>
<td>0.33</td>
<td>36</td>
</tr>
<tr>
<td>New York Islanders</td>
<td>1.00</td>
<td>0.32</td>
<td>60</td>
</tr>
<tr>
<td>Calgary Flames</td>
<td>1.00</td>
<td>0.31</td>
<td>26</td>
</tr>
<tr>
<td>Dallas Stars</td>
<td>1.00</td>
<td>0.31</td>
<td>42</td>
</tr>
<tr>
<td>Minnesota Wild</td>
<td>1.00</td>
<td>0.26</td>
<td>35</td>
</tr>
<tr>
<td>Chicago Blackhawks</td>
<td>1.00</td>
<td>0.24</td>
<td>35</td>
</tr>
<tr>
<td>New York Rangers</td>
<td>1.00</td>
<td>0.21</td>
<td>42</td>
</tr>
<tr>
<td>Buffalo Sabres</td>
<td>1.00</td>
<td>0.19</td>
<td>32</td>
</tr>
<tr>
<td>Carolina Hurricane</td>
<td>1.00</td>
<td>0.17</td>
<td>30</td>
</tr>
<tr>
<td>Nashville Predators</td>
<td>1.00</td>
<td>0.10</td>
<td>20</td>
</tr>
<tr>
<td>Colorado Avalanche</td>
<td>1.00</td>
<td>0.09</td>
<td>23</td>
</tr>
<tr>
<td>Toronto Maple Leafs</td>
<td>0.96</td>
<td>0.30</td>
<td>73</td>
</tr>
<tr>
<td>Ottawa Senators</td>
<td>0.90</td>
<td>0.16</td>
<td>58</td>
</tr>
<tr>
<td>Pittsburg Penguins</td>
<td>0.89</td>
<td>0.28</td>
<td>29</td>
</tr>
<tr>
<td>Atlanta Thrashers</td>
<td>0.86</td>
<td>0.17</td>
<td>35</td>
</tr>
<tr>
<td>New Jersey Devils</td>
<td>0.85</td>
<td>0.19</td>
<td>59</td>
</tr>
<tr>
<td>Detroit Red Wings</td>
<td>0.83</td>
<td>0.24</td>
<td>42</td>
</tr>
<tr>
<td>San Jose Sharks</td>
<td>0.83</td>
<td>0.22</td>
<td>23</td>
</tr>
<tr>
<td>Florida Panthers</td>
<td>0.83</td>
<td>0.20</td>
<td>25</td>
</tr>
<tr>
<td>Montreal Canadiens</td>
<td>0.80</td>
<td>0.17</td>
<td>23</td>
</tr>
<tr>
<td>Vancouver Canucks</td>
<td>0.57</td>
<td>0.10</td>
<td>40</td>
</tr>
<tr>
<td>Boston Bruins</td>
<td>0.44</td>
<td>0.24</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 5.2: Individual precision and recall values for 23 of the team names (of 30) detected with high confidence in the HOCKEY test set; $\Gamma = 10$ and $T_a = 0.95$. 
logos. Therefore, a team with fewer than 60 training examples will almost always have fewer than 24 usable instances of the team logo. In addition, in some cases these instances will display various versions of the team’s logo. For example, images of the Maple Leafs regularly show three different versions of their logo. The training set contains four completely different Buffalo Sabres chest logos, and many of these are placed on different backgrounds for home and away jerseys. As we see later in Figure 5.8(i), the system does not learn the new Sabres logo, but it does have a strong model for the older version of the logo displayed by the fan in the background.

For teams detected with relatively high recall, the system tends to learn separate models for each variation of the chest logo and often additional models for other distinctive parts of a player’s uniform, such as shoulder patches or sock patterns. In some cases, the system learns several high-confidence models for the same logo. Some of these are redundant models that have nearly identical detection patterns, while in other cases the models describe the logo in different modes of perspective distortion. Teams with lower recall tend to only have one or two high-confidence appearance models describing a single logo variation. The training restriction of only 20 seed appearance models for each named object is perhaps ill-advised, given that multiple appearance models are helpful and only about 1 in 6 seed models leads to a high-confidence object appearance model.

The visual properties of a team’s logo also affect whether the system learns an association between its appearance and the team’s name. For example, whereas the Philadelphia Flyers are mentioned in the captions of 168 training images, their primary logo lacks detailed texture and so attracts relatively few interest points. This may explain why the system did not learn a reliable detector for this logo.

**Sample Annotations from the HOCKEY Set**

Figure 5.8 shows some example annotations in the test images for $\Gamma_{10}$. As expected, team logos tend to be the most useful patches of appearance for recognizing the teams. The system is often
Figure 5.8: Sample annotations of our system in the hockey test images. The system automatically discovers chest logos as the most reliable method for recognizing a team.
Figure 5.9: Alternate models can be learned, such as shoulder patches or sock patterns.

able to detect learned logos that are distorted or partially occluded. In some cases, however, the system fails to detect learned logos that are relatively clean and undistorted. These might be due to contrast-reversal from a logo appearing against a different background, or perhaps missed detections of the underlying interest point detector.

While the team logos tend to be the most distinctive aspect of a player’s appearance, the system has learned alternate models for a variety of teams. Figure 5.9 displays two such examples, including a shoulder patch and sock pattern. The system might also learn an association between a team name and an appearance patch that is not part of a player at all.

Figure 5.10 displays instances of a particular type of false annotation. Variants of a white net pattern against a dark background are learned independently as reasonably high-confidence appearance models for several teams. This appears to be a result of overfitting. While this back-of-the-net appearance is quite common across the NHL image collection, certain variations of this appearance have a weak initial correspondence with particular teams. During iterative improvement, the system is able to tailor each team’s version of the net appearance model to fit only the specific instances of the net appearance in the training data that are annotated with the desired team name. In this way, a model can have few false positives in the training set, even though the described appearance is not meaningfully associated to the team name. We are currently considering approaches for reducing the occurrence of such overfitting while still allowing the improvement stage to encode meaningful differences in appearance.
Figure 5.10: Due to overfitting, our system learns bad models (a white net on a dark background) for several teams.

Figure 5.11: Words other than team names for which the system discovered strong models include the name of the Calgary Flames’ goalie, the name and location of the New York Islanders’ arena.
As mentioned previously, the system is not restricted to finding visual correspondences solely for team names. Figure 5.11 shows a few example detections of models learned for other words. Two of the models associate the distinctive pads of the Calgary Flames goalie Mikka Kiprusoff with the labels ‘mikka’ and ‘kiprusoff’. Most of the other high-confidence model–word associations link the appearance of a team logo with other words associated with the team. For instance, Figure 5.11(b) shows a learned association between the Islanders logo and words related to their home arena of “Nassau Coliseum in Uniondale, New York”. Other associations are with fragments of a team’s name that are not among the words we manually linked to identify a team. For instance, the system learned associations between the words \textit{wing} and \textit{los} and the logos of the Detroit Red Wings and the Los Angeles Kings. We associate the words \textit{angeles} and \textit{kings} with the same caption token, so that either word could be used to represent the team. The word \textit{los} was not one of the words we linked, but it still has a strong correspondence in our training set with the Los Angeles Kings logo. The fact that the components of a multi-word name can independently converge on similar visual descriptions means that it may be possible to automatically learn such compound names based on visual model similarity, instead of manually linking the components prior to learning.

Table 5.3 gives precision and recall values for all words (other than the team names) with at least 10 detections in the test images. These results may be somewhat understated, as the test image captions are not as consistent in mentioning these background words as they are with the team names. For instance, a detection for the word \textit{maple} of the Toronto Maple Leafs logo will count as a true positive if the caption of the test image is “Maple Leafs vs Lightning”, but count as a false positive if the caption is the semantically equivalent “Toronto vs Tampa Bay”.

\textbf{Annotation Performance on the LANDMARK Data Set}

The LANDMARK data set includes images of 27 famous buildings and locations with some associated tags downloaded from the Flickr website, and randomly divided into 2172 training and 1086 test image–caption pairs. Like the NHL logos, each landmark appears in a variety of
perspectives, scales and (to a lesser extent) orientations. Whereas a hockey logo is deformable and may appear in several different versions, the appearance of the landmarks can be strongly affected by varying lighting conditions, and different faces of the structure can present dramatically different appearances. Another difference is that the HOCKEY captions usually mention two teams, but each LANDMARK image is only associated with a single landmark.

Table 5.4 presents precision and recall information for the 26 of 27 landmarks for which high-confidence detections were made in the test set. Compared to the HOCKEY results (Table 5.2), recall is generally higher. This is probably because test image labels used as ground truth are somewhat more reliable in the LANDMARK set; there are fewer instances where a test caption mentions a landmark that is not present in the image. The only landmark of the set that was not learned to some degree was the Sydney Opera House, perhaps due to the smooth, relatively textureless exterior of the building. Our system recognized only one of the CN Tower images in the test set, probably because the more distinctive pattern of the observation level makes up such a small fraction of the overall tower.

Figure 5.12 shows 8 examples of the detected landmarks. In general, the learned models tend to cover highly textured areas of a landmark. In some cases, however, the profile of a
structure such as the towers of the Golden Gate Bridge or the Statue of Liberty is distinctive enough to form a strong model. Since many of the building details display different shadowing at different times of the day, multiple models are often learned to cover these different appearances. Due to the repetitive surface structure of many of the landmarks, an image often contains multiple detections of a single appearance model (e.g., Figure 5.12(c-e)). The system also learned associations between landmark names and some nearby objects, such as an adjacent office tower that the system associated with the Empire State building.
<table>
<thead>
<tr>
<th>Name</th>
<th>Precision</th>
<th>Recall</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rushmore</td>
<td>1.00</td>
<td>0.71</td>
<td>35</td>
</tr>
<tr>
<td>St. Basil’s Cathedral</td>
<td>1.00</td>
<td>0.69</td>
<td>35</td>
</tr>
<tr>
<td>Statue of Liberty</td>
<td>1.00</td>
<td>0.61</td>
<td>36</td>
</tr>
<tr>
<td>Great Sphinx</td>
<td>1.00</td>
<td>0.45</td>
<td>40</td>
</tr>
<tr>
<td>Notre Dame Cathedral</td>
<td>1.00</td>
<td>0.40</td>
<td>40</td>
</tr>
<tr>
<td>Stonehenge</td>
<td>1.00</td>
<td>0.36</td>
<td>42</td>
</tr>
<tr>
<td>St. Peter’s Basilica</td>
<td>1.00</td>
<td>0.15</td>
<td>41</td>
</tr>
<tr>
<td>Chichen Itza</td>
<td>1.00</td>
<td>0.05</td>
<td>37</td>
</tr>
<tr>
<td>CN Tower</td>
<td>1.00</td>
<td>0.03</td>
<td>34</td>
</tr>
<tr>
<td>Golden Gate Bridge</td>
<td>0.97</td>
<td>0.73</td>
<td>45</td>
</tr>
<tr>
<td>Christo Redentor</td>
<td>0.96</td>
<td>0.55</td>
<td>44</td>
</tr>
<tr>
<td>Eiffel Tower</td>
<td>0.95</td>
<td>0.61</td>
<td>33</td>
</tr>
<tr>
<td>Taj Mahal</td>
<td>0.89</td>
<td>0.52</td>
<td>33</td>
</tr>
<tr>
<td>Big Ben</td>
<td>0.88</td>
<td>0.68</td>
<td>44</td>
</tr>
<tr>
<td>Colosseum</td>
<td>0.87</td>
<td>0.33</td>
<td>39</td>
</tr>
<tr>
<td>Tower Bridge</td>
<td>0.82</td>
<td>0.79</td>
<td>47</td>
</tr>
<tr>
<td>White House</td>
<td>0.81</td>
<td>0.38</td>
<td>45</td>
</tr>
<tr>
<td>US Capitol</td>
<td>0.80</td>
<td>0.80</td>
<td>45</td>
</tr>
<tr>
<td>Reichstag</td>
<td>0.80</td>
<td>0.53</td>
<td>45</td>
</tr>
<tr>
<td>St. Paul’s Cathedral</td>
<td>0.75</td>
<td>0.69</td>
<td>48</td>
</tr>
<tr>
<td>Arc De Triomphe</td>
<td>0.71</td>
<td>0.57</td>
<td>42</td>
</tr>
<tr>
<td>Parthenon</td>
<td>0.71</td>
<td>0.29</td>
<td>35</td>
</tr>
<tr>
<td>Burj Al Arab</td>
<td>0.71</td>
<td>0.23</td>
<td>43</td>
</tr>
<tr>
<td>Leaning Tower</td>
<td>0.70</td>
<td>0.93</td>
<td>43</td>
</tr>
<tr>
<td>Empire State Building</td>
<td>0.62</td>
<td>0.55</td>
<td>38</td>
</tr>
<tr>
<td>Sagrada Familia</td>
<td>0.54</td>
<td>0.40</td>
<td>35</td>
</tr>
</tbody>
</table>

Table 5.4: Individual precision and recall values for the 26 of 27 landmarks detected with high confidence in the LANDMARK test set; $\Gamma = 10$ and $T_a = 0.95$. 
5.3.3 Summary of Results

The results presented in this section show that our system is able to find many meaningful correspondences between word labels and visual structures in three very different data sets. While the system can work equally well over a range of spatial parameter settings, the addition of spatial relationships to the appearance model can dramatically improve the strength and number of the learned correspondences. Our framework can deal with occlusion and the highly variable appearance of some objects by associating multiple visual models with a single word, but this depends on the presence of a sufficient number and a variety of training examples.

5.4 Conclusions

We have proposed an unsupervised method that uses language both to discover salient objects and to build distinctive appearance models for them, from cluttered images paired with noisy captions. The algorithm simultaneously learns appropriate names for these object models from the captions. Note that we do not incorporate any prior knowledge into our system about which objects or words are of interest. The system has to learn these from the pairings of the images and captions in the training data. We have devised a novel appearance model that captures the common structure among instances of an object by using pairs of points together with their spatial relationships as the basic distinctive portions of an object. We have also introduced a novel detection method that can be reliably used to find and annotate new instances of the learned object models in previously unseen (and uncaptioned) test images. Given the abundance of existing images paired with text descriptions, such methods can be very useful for the automatic annotation of new or uncaptioned images, and hence can help in the organization of image databases, as well as in content-based image search.

At this juncture, there are two complimentary directions for future work. First, we would like to use insights from computational linguistics to move beyond individual words to groups of words (e.g., word combinations such as compound nouns and collocations, or semantically-
related words) that correspond to distinct visual patterns. Second, though local features provide a strong basis for detecting unique objects, they are less ideal for detecting object categories or general settings. However, the grouping problem persists for most types of visual features and most forms of annotation. We expect that many of the mechanisms for addressing the grouping problem for local features will also apply to other feature classes, such as contours.

5.5 Parameter Settings

This section elaborates on how we set the values of various parameters required by our model and methods, including: parameters of our image and object representation, and those of our detection confidence score from Section 5.1, and parameters of the model–word correspondence confidence score (used in learning) from Section 5.2.

For our image representation presented in Section 5.1.1, we use a cluster set of size 4000 to generate the quantized descriptors \( c_m \) associated with each image point. We set the neighborhood size, \(|n_m|\), to 50 as experiments on the CVPR data set indicated this was an appropriate tradeoff between having distinctiveness and locality in a neighborhood. (Neighborhoods in the range of 40 – 70 points produced roughly equivalent results.) Recall from Section 5.1.2 that in an appearance model, each vertex is associated with a vector of neighboring cluster centers, \( c_i \). We set \(|c_i| = 20\) to minimize the chance of missing a matching feature due to quantization noise, at the expense of slowing down the model detection function.

We set the parameters of the word–appearance correspondence measure, \( \text{Corr}(w, G) \), according to the following assumptions: Reflecting high confidence in the captions of training images, we set \( P(r_{wi} = 1|s_i = 1) = 0.95 \), since we expect to observe an object’s name in a caption whenever the object is present in the corresponding image. Similarly, we expect that few images will be labeled \( w \) when the object is not present. However, even setting a low background generation rate \( P(r_{wi} = 1|s_i = 0) = 0.5 \) could lead to a relatively large number of false \( w \) labels if few image–caption pairs contain the object \( P(s_i = 1) \) is low). We
Chapter 5. Probabilistic Appearance Models

expect the false labels to be a small fraction of the true labels. Since the expected rate of true word labels is $P(r_{wi} = 1|s_i = 1)P(s_i = 1)$ and the expected rate of false word labels is $P(r_{wi} = 1|s_i = 0)P(s_i = 0)$, we therefore set:

$$P(r_{wi} = 1|s_i = 0) = 0.05 \cdot \frac{P(r_{wi} = 1|s_i = 1)P(s_i = 1)}{P(s_i = 0)}.$$  \hfill (5.17)

Similarly, during model learning, we wish to assert that a high-confidence model should have a low ratio of false-positives to true-positives. Therefore we set:

$$P(q_{Gi} = 1|s_i = 0) = 0.05 \cdot \frac{P(q_{Gi} = 1|s_i = 1)P(s_i = 1)}{P(s_i = 0)}.$$  \hfill (5.18)

However, when evaluating correspondences between words and neighborhoods to find good seed models, we set this target number of false-positives as a fraction of the true positives much higher at 1. We do so because the starting seed models are expected to be less distinctive, hence even promising seeds may have a large number of false positives. Performing experiments on a held-out validation set of 500 HOCKEY training images, we found the precise value of $P(H_0)/P(H_C)$ to have little effect on the overall performance; we thus set $P(H_0)/P(H_C) = 100,000$.

At the initialization stage, up to 20 neighborhood clusters $\mathcal{N}$ are selected to generate seed models, which are further modified in up to 200 stages of iterative improvement.

If the detection threshold $T_d$ is set too low, the detection algorithm can report a large number of low-probability detections, which can impose a computation burden on later stages of processing. However, the threshold should not be set so high that detections that might lead to model improvements are ignored. We err on the side of caution and set $T_d = 0.001$. The parameters of the background likelihood (used in measuring detection confidence) are set according to sample statistics of a collection of 10,000 commercial stock photographs. Based on this wide sample of visual content, the distance variance $\sigma_{xB}^2$ is set to 200, and the scale variance $\sigma_{\lambda B}^2$ is set to 5. Each of the model spatial relationship variances is set to a fraction $1/\Gamma$ of the corresponding background variance, i.e.:

$$\sigma_x^2 = \frac{\sigma_{xB}^2}{\Gamma}, \sigma_{\lambda}^2 = \frac{\sigma_{\lambda B}^2}{\Gamma}, \text{ and } \sigma_{\phi}^2 = \frac{\pi^2}{3\Gamma}.$$
where $\Gamma$ is the spatial tolerance parameter that determines the expected degree of spatial variability among observed object models. In most experiments, we set $\Gamma = 10$, a value we found to result in reasonably good performance on the HOCKEY validation set. After estimating the mixture of Gaussian distribution based on the stock photo collection, we observed that average variance among the background feature clusters was 1.31, so we set the model feature variance to be slightly higher at $\sigma_f^2 = 1.5$. Based on empirical findings on the held-out subset of the HOCKEY training data, we set $P(H_B)/P(H_G) = 100,000$, though we found a broad range of values to yield similar results.
Chapter 6

Hierarchical Appearance Models

Computer vision tasks from image retrieval to object class recognition are based on discovering similarities between images. For all but the simplest tasks, meaningful similarity does not exist at the level of basic pixels, and so system designers create image representations that abstract away irrelevant information. One popular strategy for creating more useful representations is to learn a hierarchy of parts in which parts at one level represent meaningful configurations of subparts at the next level down. Thus salient patterns of pixels are represented by local features, and recurring configurations of features can, in turn, be grouped into higher-level parts, and so on, until ideally the parts represent the objects that compose the scene. The hierarchical representations are inspired by and intended to reflect the compositional appearance of natural objects and artifacts. For instance, each level of the Leaning Tower of Pisa appears as a ring of arches while the tower as a whole is composed of a (nearly) vertical stack of levels.

With this strategy in mind, we build upon the approach described in Chapter 5 (originally presented in [47]) to produce a system with more accurate image annotation and improved object localization. Given images of cluttered scenes, each associated with potentially noisy captions, our previous system can discover configurations of local features that strongly correspond to particular caption words. Our system improves the overall distribution of these local configurations to optimize the overall correspondence with the word. While individual learned
Figure 6.1: Object model detection and learning progresses in stages. Gradient patterns in the original image (a) are grouped into local features (b). Configurations of local features with strong word correspondence are captured as part models (c). Finally, we represent meaningful configurations of part models as multipart models (d).

parts are often sufficient to indicate the presence of particular exemplar objects, they have limited spatial extent and it is difficult to know whether a collection of part detections in a particular image are from multiple objects or multiple parts of a single object. Our system learns meaningful configurations of parts wherever possible, allowing us to reduce false annotations due to weak part detections and provide a better indication of the extent of detected objects. Figure 6.1 illustrates how low-level features are assembled in stages to form a multipart model (MPM) for the Leaning Tower. MPMs are more robust to occlusion, articulation and changes in perspective than a flat configuration of features.

6.1 Images, Parts and Multipart Models

Our system learns multipart appearance models (MPMs) by detecting recurring configurations of lower-level ‘parts’ that together appear to have a strong correspondence with a particular caption word. Though our overall approach could be appropriate for a variety of part features, in this chapter our parts are local configurations of interest points as in Chapters 4 and 5.

As in chapter 5, an image is represented as a set of local interest points, \( I = \{ p_m | m = 1 \ldots |I| \} \). These points are detected using Lowe’s SIFT method [58], which defines each
A part appearance model describes the distinctive appearance of an object part as a graph \( G = (V, E) \). Each vertex \( v_i \in V \) is composed of a continuous feature vector \( f_i \) and each edge \( e_{ij} \in E \) encodes the expected spatial relationship between two vertices, \( v_i \) and \( v_j \). Each model detection is assigned a confidence score, \( \text{Conf}_{\text{detect}}(O, G) \in [0, 1] \), based on how well an observed set of points \( O \) fits a part model \( G \).

Multipart models are very similar in structure to the local appearance models. As shown in Figure 6.2, a multipart model is a graph \( H = (U, D) \) where vertices \( u_j, u_k \in U \) are part appearance models and each edge \( d_{jk} \in D \) encodes the spatial relationships between them, using the same relationships as in the part model: \( d_{jk} = (\Delta x_{jk}, \Delta \lambda_{jk}, \Delta \phi_{jk}, \Delta \phi_{kj}) \).

6.2 Discovering Parts

Multipart models are composed of the same type of individual appearance models that were discovered in chapter 5. However, models trained to maximize stand-alone detection performance are generally not ideal as parts of a larger appearance model. Singleton appearance
models need to act as high-precision detectors while MPM parts can be individually more ambiguous and rely on the MPM layer to weed out false-positive detections by imposing co-occurrence and spatial constraints. Therefore, when learning MPM parts, we can accept some loss of precision in exchange for better recall and better spatial coverage of the object of interest. We implement this shift toward weaker parts with better coverage by replacing the part initialization process in Chapter 5 with our own improved process and by limiting the size of learned part models to eight vertices.

6.2.1 Model Initialization through Image Pair Sampling

We replace the clustering-based model initialization method of chapter 5 with an approach that makes earlier use of language information. The previous method summarizes the visual information within each neighborhood of an image set as a quantized bag-of-features descriptor called a neighborhood pattern and then uses clustering to group similar neighborhood patterns. Next, the system checks for promising correspondences between the occurrence patterns of each neighborhood cluster and each word. Finally, clusters with the best correspondences for each word are used to extract initial two-vertex appearance models.

This clustering approach has several drawbacks. The neighborhood patterns are noisy due to features quantization and detector errors. Therefore a low similarity threshold is needed to reliably group similar appearances. However, this allows unrelated neighborhoods to join the cluster. Especially on large image sets, this can add substantial noise to the cluster occurrence pattern, obscuring its true word correspondences. Therefore recurring visual structure corresponding to rarer object views is often overlooked.

Our initialization method avoids feature quantization and uses word labels early-on in the process. Instead of using a neighborhood pattern, we compare visual features directly. Instead of clustering visual structure across the entire training set, we look for instances of shared appearance between pairs of images with the same word label. For a given word \( w \), the system randomly samples pairs of images \( I_A \) and \( I_B \) from those with captions containing \( w \) and
identifies neighborhoods in the two images that share visual structure.

We identify shared neighborhoods in three steps. First, the system looks for uniquely-matching features that are potential anchors for shared neighborhoods. Following [58], we identify matching features that are significantly closer to each other than to either feature’s second-best match, \( i.e. \), features \( f_m \in I_A \) and \( f_n \in I_B \) that satisfy equations 6.1 and 6.2:

\[
|f_m - f_n|^2 \leq \psi_u |f_m - f_k|^2, \forall f_k \in \{I_B - f_n\} \tag{6.1}
\]

\[
|f_m - f_n|^2 \leq \psi_u |f_l - f_n|^2, \forall f_l \in \{I_A - f_m\} \tag{6.2}
\]

where \( \psi_u < 1 \) controls degree of uniqueness of anchor matches. For each pair of uniquely-matching features, the system checks for supporting matches in the surrounding neighborhood. These supporting matches aren’t required to be unique, so the corresponding uniqueness quantifier \( \psi_s > 1 \). For each supporting match pair \( f_i \in I_A \) and \( f_j \in I_B \), the system then verifies that the spatial relationships between the unique feature and the supporting feature in the two images \( (r_{mi} \text{ and } r_{nj}) \) are consistent. A shared neighborhood has a pair of unique matches and at least two spatially consistent supporting matches.

Given this evidence of shared visual structure, we construct a set of two-vertex part models, each with one vertex based on the unique match and the other on a strong supporting match. These two-vertex models represent shared visual structure between two images labeled with word \( w \). To check whether the models correspond with \( w \), the system detects each model \( G \) across the training image set and compares its occurrence pattern with that of \( w \). Below, we explain how we sample image pairs and filter the resulting initial part models to maximize overall coverage of the object.

### 6.2.2 Part Coverage Objective

In chapter 5, the system develops the \( n \) neighborhood clusters with the best correspondence with \( w \) into full appearance models. This approach concentrates parts on the most common
views of an object, neglecting less common views and appearances associated with \( w \). Our method instead selects initial part models so that, as a group, they have good coverage of \( w \) throughout the training set.

The ideal set of models would have multiple, non-overlapping detections in every training set image annotated with word \( w \) and no detections elsewhere. We evaluate how well a given distribution of model detections approaches this ideal using a correspondence measure \( F \) between a vector \( r_w \) indicating images with \( w \) in the caption and a weighted vector \( Q_w \) indicating images containing detections for multiple parts. The part initialization process greedily grows and modifies a collection of non-overlapping two-vertex part models to maximize \( F(r_w, Q_w) \). At each iteration, we draw a pair of images from the sample distribution \( s_w \) and use them to generate potential part models. The algorithm then calculates, for each potential model, the effects on the correspondence score \( F \) of adding the model to the current part set, of replacing each of the models in the current set and of rejecting the model. The algorithm implements the option which leads to the greatest improvement in correspondence. The process stops once no new models have been accepted in the last \( N_{pairs} \) image-pair samples.

Given a set of \( k \) training images, \( Q_w = \{ Q_{wi} | i = 1, \ldots, k \} \) is a detection weight vector representing the distribution of appearance model detections throughout the training set. If \( n_i \) is the number of independent model detections in image \( i \), \( Q_{wi} = 1 - \nu^{n_i}, \nu < 1 \). The weight vector approaches 1 when there are multiple detections per image, but with each successive detection has a smaller effect on \( Q_{wi} \). Therefore there is little potential reward (or penalty) for introducing new part detections in images that already have several different parts. The detection weight vector \( Q_w \) also influences the sample distribution: \( s_w \sim 1 - Q_w \). This focuses the search for new models on images that do not already contain several model detections.

As the overall objective function \( F(r_w, Q_w) \), we use F-measure. Later, during individual model improvement, in which precision is more important, we use correspondence confidence, \( \text{Conf}_{corr}(G, w) \) from [47] as our objective.

Besides optimizing the explicit objective function, the initialization system also avoids re-
dundant models with many overlapping detections. Two models are considered to be redundant when their detections overlap nearly as often as they occur separately. When a new two-vertex model is considered, if selected it must replace any models that it makes redundant. Algorithm 3 summarizes the steps of part initialization.

**Algorithm 3** Chooses initial part models to maximize overall coverage \( F(r_w, Q_w) \)

**FindInitialParts**\( (r_w) \)

1. Start with \( n_i = 0, \forall i \), therefore \( Q_w \) and \( s_w \) are uniform.
2. Draw \( I_A \) and \( I_B \) (without replacement) from \( s_w \).
3. Find shared neighborhoods and construct potential part models \( G \).
4. Filter \( G \) based on \( F_{0.25} \) correspondence with \( w \).
5. For each remaining \( G \):
   - Calculate overlap of \( G \) with current part models.
   - If \( G \) overlaps with existing models, calculate \( Q_w^* \) for \( G \) replacing overlapping models.
   - Else calculate \( Q_w^* \) for addition and for replacement of each individual model.
   - Accept best change \( Q_w^* \) if \( F_1(r_w, Q_w^*) > F_1(r_w, Q_w) \).
   - Update \( Q_w \).
6. Update \( s_w \) and go to step 2.
7. If \( N_{pairs} \) samples with no change in model set accepted, return.

### 6.3 Building Multipart Models

After learning distinctive part models, but before assembling them into multipart models, we perform several stages of processing. Algorithm 4 summarizes both the preprocessing steps and the MPM initialization and assembly process, with reference to the subsections below that explain the steps of the algorithm.
Algorithm 4 Uses parts associated with word $w$ to assemble multipart models.

\textbf{ConstructMPMs($w$)}

1. For each part $G$ associated with $w$, find the set $\mathcal{O}_G$ of observations of $G$ in training images.

2. Identify and remove redundant parts (section 6.3.1).

3. For each $G$, set the spatial coordinates of each observation $O_G \in \mathcal{O}_G$ (section 6.3.2):
   
   - Choose representative vertex $v_c$ to act as center of $G$.
   - For each $v_i \in v_G$, find average relationship, $\bar{r}_{ic}$, between co-occurrences of $(v_i, v_c) \in \mathcal{O}_G$.
   - For each $O_G \in \mathcal{O}_G$, and each observed vertex $p_i \in O_G$ calculate expected position of $x_c$ based on $(\bar{r}_{ic}, x_i)$. Part spatial coordinate $x_G$ is the average expected center $\bar{x}_c$.

4. Sort parts by $\text{Conf}_{\text{corr}}(G, w)$.

5. For each $G$:
   
   - Skip expansion if most $O_G \in \mathcal{O}_G$ are already incorporated into existing MPMs (section 6.3.3).
   - Iteratively expand $G$ into an MPM $H$ using same method as part models (section 6.3.4):
     
     - Expand MPM $H$ to $H^*$ by adding new part or spatial relationship.
     - Detect $H^*$ across the training image set (section 6.3.5).
     - If new MPM–word correspondence, $\text{Conf}_{\text{corr}}(H^*, w) > \text{Conf}_{\text{corr}}(H, w)$, $H \Leftarrow H^*$.

   - If at least $N_{MPM}$ multipart models have been created, return.
6.3.1 Detecting Duplicate Parts

Our initialization method avoids excessive overlap of initial part models. However, during model refinement, two distinct part models can converge to cover the same portion of an object’s appearance. Near-duplicate parts must be pruned or they could complicate the search for multipart models since they could be interpreted as a pair of strongly co-occurring, independent parts.

Rather than detect near-duplicates by searching for partial isomorphisms between part models, we look for groups of parts that tend to be detected in the same images at overlapping locations. If a vertex $v_{Ai}$ in model $G_A$ maps to the same image point as vertex $v_{Bj}$ in model $G_B$ in more than half of detections, then we draw an equivalence between $v_{Ai}$ and $v_{Bj}$. If more than half of the vertices in either part are equivalent, we remove the part with the weakest word–model correspondence confidence $\text{Conf}_{\text{corr}}(G, w)$.

6.3.2 Locating Part Detections

The parts described in [47] encode spatial relationships among local interest points; we construct multipart models by discovering spatial relationships between such detected parts. However, while a local interest point detector provides that point’s scale, orientation and location, the part detector does not. We therefore set the spatial coordinates for each part detection based on the underlying image points in a way that is robust to occlusion and errors in feature detection.

For each part we select a central vertex and for each detection we estimate the center’s spatial coordinates. The center vertex need not be observed in every detection, since each observed vertex contributes to a weighted estimate of the center’s coordinates. Figure 6.3 illustrates this approach. We use the estimated location and orientation of the center and multiply the estimated scale of the center vertex by a part-specific factor so that the detected part scale reflects the normal spread of the part’s vertices.
Figure 6.3: The spatial coordinates of a part detection are tied to a central vertex $c$. We estimate $c$’s coordinates based on observed vertices, even if $c$ itself is not observed.

### 6.3.3 Choosing Initial Multipart Models

Our system uses the most promising individual part models as seeds for constructing multipart models. Parts that have good correspondence with a word are likely to co-occur with other parts in stable patterns from which large MPMs with good spatial coverage can be constructed. However, if only the strongest part models are expanded, the resulting MPMs may be too clustered around only the most popular views of the object. This would neglect views with weaker individual parts where MPMs can make the biggest difference in precision.

Therefore initial model selection proceeds as follows. Part models are evaluated in the order of their correspondence with a word $w$. A model is expanded if at least half of its ‘good’ detections (in images labeled with $w$) have not been incorporated into any of the already-expanded MPMs. Selective expansion continues until the list of part models is exhausted or $N_{MPM}$ distinct multipart models have been trained for a given word.

### 6.3.4 Refinement and Expansion of Multipart Models

In order to expand the multipart models, we take an approach very similar to the method described in chapter 5, in that we use the correspondence strength $Conf_{corr}(H, w)$ between a multipart model $H$ and word $w$ to guide the expansion of these two-vertex graphs into larger multipart models. The correspondence score reflects the amount of evidence, available in a set of training images, that a word and a part model are generated from a common underlying source object, as opposed to appearing independently.

Each iteration of the expansion algorithm begins by detecting all instances of the current
multipart model in the training set (section 6.3.5) and identifying additional parts that tend to
co-occur with a particular spatial relationship relative to the multipart model. We expand the
multipart model by adding new vertices (part models) and edges (spatial relationships) one at
time from among the candidate parts. An expansion of the multipart model $H$ is accepted if
it improves $\text{Conf}_{\text{corr}}(H, w)$ (starting a new iteration), and rejected otherwise. The expansion
process continues until potential additions to $H$ have been exhausted.

### 6.3.5 Detecting Multipart Models

As in part model detection, multipart detection must be robust to changes in viewpoint, occlu-
sion or lighting that can cause individual part detections to be somewhat out of place or missing
entirely. We use a simple generative model illustrated in Figure 6.4 to explain the pattern of
part detections both in images that contain a particular multipart model and those that do not.

Each image $i$ has an independent probability $P(h_i = 1)$ of containing the multipart model
$H$. Given $h_i$, the presence of each model part is determined independently ($P(u_{ij} = 1|h_i)$).
The foreground probability of a model part being present is relatively high ($P(u_{ij} = 1|h_i =
1) = 0.95$), while the background probability, $P(u_{ij} = 1|h_i = 0)$, is equal to its normalized
frequency across the training image set. If a part is present, it tends to have a higher observed
detection confidence, $o_{ij}$ ($p(o_{ij}|u_{ij} = 1) = 2o_{ij}, p(o_{ij}|u_{ij} = 0) = 2(1 - o_{ij})$). If the multipart
model is present ($h_i = 1$) and contains an edge $r_{jk}$, and the parts $u_{ij}$ and $u_{ik}$ are present, then
the observed spatial relationship $s_{ijk}$ between the two parts has a relatively narrow distribution
centered at the edge parameters. Otherwise, all spatial relationships follow a broad background
distribution.

In any given image, there may be many possible assignments between multipart model
vertices and observed part detections. We choose assignments in a greedy fashion in order to
maximize $P(h_i = 1|o_i, s_i)$. First we choose the best-fit assignment of two linked vertices, then
one by one we choose the vertex assignment that makes the largest improvement in $P(h_i =
1|o_i, s_i)$ and is consistent with existing assignments.
The prior probability $P(h_i = 1)$ depends on the complexity of the MPM, with more complex multipart models having a lower prior probability. Specifically:

$$P(h_i = 1) = \alpha^{|U|} \cdot \beta^{|D|},$$

(6.3)

where $\alpha, \beta < 1$ and $|U|$ and $|D|$ are, respectively, the number of vertices and edges in $H$. This factor prevents large, complex models from being detected when only a tiny fraction of their vertices are present. The constants were selected based on detection experiments on synthetic part distributions.

## 6.4 Results

Once we have discovered a set of individual part models and learned multipart models from configurations of the parts, we can use these learned structures to annotate new images. We begin by detecting all part models in the image (even those that are relatively weakly detected or have relatively low individual correspondence confidence). Based on these part observations, we then evaluate detection confidence for all learned MPMs. As in chapter 5, our annotation confidence for both parts and multipart models is the product of detection confidence, $\text{Conf}_{\text{detect}}(i, H)$, and correspondence confidence $\text{Conf}_{\text{corr}}(H, w)$. Overall annotation confidence is the maximum annotation confidence over word $w$’s detected models in image $i$. 
For ease of comparison, we ran our system on three image sets described in [47]. In all three cases, the changes to part initialization combined with the addition of MPM models improve the precision and recall of annotation on new images compared to the system in chapter 5. The degree of improvement seems to depend on the scale and degree of articulation of named objects.

In experimentation on the small TOYS image set, we find that the particular values of our system parameters do not have a significant effect on our results. The same parameter values chosen based on the TOYS set results are carried over to the two larger and more significant sets without modification. We set uniqueness factors $\psi_u = 0.9$ and $\psi_s = 1.2$. $N_{pairs} = 50$ allows a large number of failed pair samples before ending initial model search. $\nu = 0.75$ allows $Q_{wi}$ to build gradually. We set the maximum number of MPMs per word, $N_{MPM} = 25$, more than the number of distinct views available for individual objects in these image collections. Finally, we choose MPM detection parameters $\alpha = 0.25$ and $\beta = 0.33$ based on experiments on synthetic data.

### 6.4.1 Experiments on the TOYS Data Set

This section examines the annotation performance of our new system on the TOYS data set first presented in [45] (see chapter 3). Figure 6.5 displays some example images from the test set along with the highest-confidence MPM for each object and the associated annotations produced by our system. Each MPM part is displayed as a yellow, five-sided figure indicating canonical position, orientation and scale. The underlying interest points for each part are drawn in red and the edges connecting MPM parts are drawn in blue. The detections illustrate how MPMs are able to integrate many local patches of distinctive appearance into a single structure. However, the MPM coverage is uneven, with some areas of the objects covered with a large number of overlapping parts while coverage in other areas is relatively sparse.

While MPMs do integrate local detections, Figure 6.6 indicates that they have only a minor effect on overall precision and recall in the TOYS set. The new initialization improves recall by
Figure 6.5: Sample detections of objects in the TOYS test set. In these examples, all named objects were correctly detected.

Figure 6.6: A comparison of precision–recall curves over the TOYS test set, for five systems: MPMs plus individual parts, individual parts alone, PAMI [47], ICCV [46], and CVPR [45]. CVPR used a local bag-of-features model that was initialized with singleton features. ICCV added spatial relationships and neighborhood initialization. PAMI added detection and annotation confidence scores and builds models with more spatial relationships. The new initialization method improves recall somewhat, while the addition of MPMs corrects a slight drop in precision.
<table>
<thead>
<tr>
<th>Name</th>
<th>Precision</th>
<th>Recall</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Franklin</td>
<td>1.00</td>
<td>0.88</td>
<td>33</td>
</tr>
<tr>
<td>Rocket</td>
<td>1.00</td>
<td>0.80</td>
<td>44</td>
</tr>
<tr>
<td>Drum</td>
<td>1.00</td>
<td>0.69</td>
<td>32</td>
</tr>
<tr>
<td>Bus</td>
<td>1.00</td>
<td>0.51</td>
<td>57</td>
</tr>
<tr>
<td>Bongos</td>
<td>1.00</td>
<td>0.50</td>
<td>36</td>
</tr>
<tr>
<td>Bug</td>
<td>1.00</td>
<td>0.49</td>
<td>51</td>
</tr>
<tr>
<td>Dino</td>
<td>1.00</td>
<td>0.38</td>
<td>42</td>
</tr>
<tr>
<td>Ernie</td>
<td>1.00</td>
<td>0.28</td>
<td>39</td>
</tr>
<tr>
<td>Cash</td>
<td>0.97</td>
<td>0.78</td>
<td>46</td>
</tr>
<tr>
<td>Horse</td>
<td>0.96</td>
<td>0.86</td>
<td>28</td>
</tr>
</tbody>
</table>

Table 6.1: Performance results on the TOYS test set; precision = 99%, recall = 60%.

a small amount while losing some precision at lower recall levels. Adding the multipart layer corrects the precision while retaining most of the gains in recall. We attribute the relatively small impact of the changes to the already good performance of the existing system. Most of the remaining missed object detections are more difficult cases with a high degree of occlusion. Also, some of the objects (Bug Bus and Dino) are small enough that individual part models can cover most of the area of distinctive appearance.

Table 6.1 shows the per-object precision and recall values of our current system, including MPMs. In this and subsequent tables, the Frequency column shows the number of captions within the test set that contain at least one instance of the corresponding word. All of the precision and recall values we report are based on word occurrence in the captions of the test set; if the system does not detect an object for a word that appears in the caption, that instance is counted as a false positive, even if the named object does not actually appear in the image.

Overall, our system achieves about a 3% improvement in recall on the TOYS set over our previous approach. As in past evaluations, the two books (Franklin and Rocket) with large, detailed planar surfaces were easiest to detect. The two most difficult objects (Dino and Ernie) are notable for their curved surfaces and lack of distinctive fine-scale texture. That said, most of the recall improvement was due to a roughly three-fold increase in recall for the Dino object.
6.4.2 Experiments on the HOCKEY Data Set

The HOCKEY set includes 2526 images of National Hockey League (NHL) players and games, with associated captions, downloaded from a variety of sports websites. It contains examples of all 30 NHL teams and is divided into 2026 training and 500 test image–caption pairs. About two-thirds of the captions are full sentence descriptions, whereas the remainder simply name the two teams involved in the game.

Figure 6.7 shows sample multipart model detections on test-set images and the associated team names. Compared to MPMs in the TOY and LANDMARK sets, most MPMs in the HOCKEY set are relatively simple. They typically consist of 2 to 4 parts clustered around the team’s chest logo. Since the chest logos are already reasonably well covered by individual part models, there is little reward for developing extensive MPMs. In principle, MPMs could tie together parts that describe other sections of the uniform (socks, pants, shoulder insignia) like those shown in Figure 6.7(e), but this type of MPM (seen in Figure 6.7(f)) is quite rare. There may be too
Figure 6.8: A comparison of precision–recall curves over the HOCKEY test set, for three systems: multipart and single-part models combined, single-part models alone and the system described in chapter 5. The new initialization method substantially improves overall recall. However, the addition of MPMs has little effect. The is probably because the distinctive portions of a player’s appearance are of limited size and they do not tend to co-occur in repeating patterns.

much articulation and (more importantly), too few instances of co-occurrence of these parts in the training set to support such MPMs.

Figure 6.8 indicates that our new approach for initializing part models leads to about a 12% improvement in recall. Considering the barriers to achieving high recall on the HOCKEY set (discussed in chapter 5), this represents a substantial gain. Our initialization system is better able to identify regions of distinctive appearance than the approach in [47]. For instance, one of the best-recognized NHL teams using our method was completely undetected in [47]. On the other hand, the addition of MPMs does not improve annotation performance at all. This is probably due to the relatively small size of distinctive regions in the HOCKEY images combined with a degree of articulation and occlusion that make larger models unreliable.

Table 6.2 shows the annotation performance of our system with respect to individual team names. The system has high-confidence detections for 27 of the 30 teams, 4 more than before. At 95% precision, overall recall was 26%, 12% higher than the previous method. With the new
initialization approach, the Washington Capitals became one of the better-recognized teams where the previous system had not detected them in the test set at all. This may be due to the new system’s focus on images sharing particular caption words.

### 6.4.3 Experiments on the LANDMARK Data Set

The LANDMARK data set includes images of 27 famous buildings and locations with some associated tags downloaded from the Flickr website, and randomly divided into 2172 training and 1086 test image-caption pairs. Like the NHL logos, each landmark appears in a variety of perspectives and scales. Compared to the hockey logos, the landmarks usually cover more of the image and have more textured regions in a more stable configuration. On the other hand, the appearance of the landmarks can vary greatly with viewpoint and lighting and many of the landmarks feature interior as well as exterior views.

Figure 6.9 provides some sample detections of multipart models in the LANDMARK test set. The MPMs can integrate widely-separated part detections, thereby improving detection confidence and localization. However, many of the models still display a high degree of part overlap, especially on objects such as the Arc de Triomphe with a dense underlying array of distinctive features. In addition, MPM coverage of the object, while better than individual parts, is not as extensive as it could be. For instance, the system detects many more parts on the western face of Notre Dame than are incorporated into the displayed MPM. In the future, we may wish to modify the MPM training routine to explicitly reward spatial coverage improvements. Finally, MPMs often seem to have one or two key parts with a large number of long-range edges. This edge structure may unnecessarily hamper robustness to occlusion.

Regardless of their limitations, Figure 6.10 indicates that MPMs can significantly improve annotation precision. The new initialization system improves overall recall by about 10%, and the addition of MPMs lifts the precision of the curve towards the 100% boundary.

Table 6.3 breaks the results down by landmark. The structures on which our system achieved the poorest results were St. Peter’s Basilica, Chichen Itza and the Sydney Opera
### Table 6.2: Individual precision and recall values for 27 of the team names (of 30) detected with high confidence in the HOCKEY test set; precision = 95%, recall = 26%.

<table>
<thead>
<tr>
<th>Name</th>
<th>Precision</th>
<th>Recall</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tampa Bay Lightning</td>
<td>1.00</td>
<td>0.61</td>
<td>49</td>
</tr>
<tr>
<td>Pittsburgh Penguins</td>
<td>1.00</td>
<td>0.45</td>
<td>29</td>
</tr>
<tr>
<td>Minnesota Wild</td>
<td>1.00</td>
<td>0.37</td>
<td>35</td>
</tr>
<tr>
<td>Washington Capitals</td>
<td>1.00</td>
<td>0.35</td>
<td>17</td>
</tr>
<tr>
<td>Los Angeles Kings</td>
<td>1.00</td>
<td>0.31</td>
<td>36</td>
</tr>
<tr>
<td>Dallas Stars</td>
<td>1.00</td>
<td>0.29</td>
<td>42</td>
</tr>
<tr>
<td>Detroit Red Wings</td>
<td>1.00</td>
<td>0.26</td>
<td>42</td>
</tr>
<tr>
<td>San Jose Sharks</td>
<td>1.00</td>
<td>0.26</td>
<td>23</td>
</tr>
<tr>
<td>Buffalo Sabres</td>
<td>1.00</td>
<td>0.25</td>
<td>32</td>
</tr>
<tr>
<td>Calgary Flames</td>
<td>1.00</td>
<td>0.23</td>
<td>26</td>
</tr>
<tr>
<td>Columbus Blue Jackets</td>
<td>1.00</td>
<td>0.18</td>
<td>11</td>
</tr>
<tr>
<td>Philadelphia Flyers</td>
<td>1.00</td>
<td>0.17</td>
<td>46</td>
</tr>
<tr>
<td>Carolina Hurricane</td>
<td>1.00</td>
<td>0.17</td>
<td>30</td>
</tr>
<tr>
<td>New York Rangers</td>
<td>1.00</td>
<td>0.14</td>
<td>42</td>
</tr>
<tr>
<td>Montreal Canadiens</td>
<td>1.00</td>
<td>0.13</td>
<td>23</td>
</tr>
<tr>
<td>Colorado Avalanche</td>
<td>1.00</td>
<td>0.09</td>
<td>23</td>
</tr>
<tr>
<td>Anaheim Ducks</td>
<td>1.00</td>
<td>0.08</td>
<td>27</td>
</tr>
<tr>
<td>Vancouver Canucks</td>
<td>1.00</td>
<td>0.05</td>
<td>40</td>
</tr>
<tr>
<td>New York Islanders</td>
<td>0.96</td>
<td>0.45</td>
<td>60</td>
</tr>
<tr>
<td>Toronto Maple Leafs</td>
<td>0.92</td>
<td>0.33</td>
<td>73</td>
</tr>
<tr>
<td>New Jersey Devils</td>
<td>0.89</td>
<td>0.29</td>
<td>59</td>
</tr>
<tr>
<td>Florida Panthers</td>
<td>0.88</td>
<td>0.28</td>
<td>25</td>
</tr>
<tr>
<td>Ottawa Senators</td>
<td>0.88</td>
<td>0.12</td>
<td>58</td>
</tr>
<tr>
<td>Chicago Blackhawks</td>
<td>0.86</td>
<td>0.34</td>
<td>35</td>
</tr>
<tr>
<td>Nashville Predators</td>
<td>0.83</td>
<td>0.25</td>
<td>20</td>
</tr>
<tr>
<td>Atlanta Thrashers</td>
<td>0.80</td>
<td>0.23</td>
<td>35</td>
</tr>
<tr>
<td>Boston Bruins</td>
<td>0.75</td>
<td>0.18</td>
<td>17</td>
</tr>
</tbody>
</table>
Figure 6.9: Sample detections of objects in the LANDMARKS test set.
Figure 6.10: A comparison of precision–recall curves over the LANDMARKS test set, for three systems: current, current without spatial relationships, and PAMI (chapter 5). The new initialization method substantially improves overall recall. In this case, the addition of MPMs improves the precision of the new detections. Distinctive portions of landmarks sometimes have stable relationships between one another.

House. The first two of these suffer from a multiplicity of viewpoints, with training and test sets dominated by a variety of interior viewpoints and zoomed images of different parts of the structure. The Sydney Opera House’s expressionist design has relatively little texture and is therefore harder to recognize using local appearance features.

6.5 Conclusions

Our initialization method and multipart models are designed to work together to improve annotation accuracy and object localization over the approach in chapter 5. Our initialization mechanism boosts recall and part coverage by detecting potential parts that would have been overlooked by the previous system, providing for a better distribution of parts over the image set and including more individually ambiguous parts. The MPM layer boosts precision and localization by integrating parts that may be individually ambiguous into models that can cover an entire view of an object.
Table 6.3: Individual precision and recall values for the 30 structures in the LANDMARK test set; precision = 98%, recall = 51%.
Together, our new methods significantly improve annotation accuracy over previous results on the experimental data sets, with the amount of improvement strongly dependent on the image set. Our improvements to part initialization and training have significantly increased recall, though sometimes at the expense of precision. For objects with recurring patterns of distinctive parts, the MPM layer can filter out bad detections, resulting in a substantially improved precision–recall curve.

The annotation performance improvement due to MPMs depends strongly on the image-caption collection. MPMs are most useful for combining evidence from multiple part models that co-occur in repeatable spatial configurations but that may be individually weak or ambiguous. We speculate that MPMs do not improve results on the TOYS and HOCKEY sets because either the objects (or a sufficiently distinctive subregion) can be described effectively by a single part, or parts do not recur often enough in stable configurations (e.g. hockey chest, shoulder and sock patterns). Landmarks, being larger and often with more ambiguous local structure, benefit more from MPMs. We expect that the MPM layer will become more important as individual parts become more ambiguous (such as parts for describing object classes).

Our initialization mechanism and the development of multipart models also improves object localization. Parts have less spatial overlap than before, they cover portions of the object that are less individually distinctive and they are better-distributed across object views. MPMs tie together recurring patterns of parts, allowing us to distinguish between the presence of multiple parts and multiple objects. Future work could further improve localization by ensuring that MPMs use all available parts to maximize spatial coverage and are themselves well-distributed across object views.
Chapter 7

Conclusions and Discussion

7.1 Review of contributions

Previous methods for learning language–vision correspondences use caption information to learn a mapping between words and predetermined visual elements. In our method, we use word–appearance correspondence to build our visual elements, forming structured appearance models by grouping local interest features. In addition, we can apply the same approach to the learned appearance models, using word–appearance correspondence to build extensive multipart appearance models. These hierarchical models are invariant to translation, scale and orientation and robust to occlusion, articulation and small changes in perspective.

Our novel detection method can reliably detect both structured and hierarchical appearance models in new scenes, allowing us to annotate previously uncaptioned images. The same detection method plays a critical role during model learning by locating instances of incomplete models throughout the training image set. Our novel word–appearance correspondence measure compares the distribution of these detections with the target word distribution to help decide whether a proposed change to the appearance model should be accepted. Through this process, we automatically expand and revise our structured and hierarchical appearance models in order to improve correspondence with caption words. With this approach, our system can
learn meaningful word–appearance pairs from complex, cluttered, real-world scenes without strong supervisory information such as bounding boxes or reliable, single-word captions.

### 7.1.1 Novel Mechanisms in other Contexts

It is important to ask how the novel mechanisms introduced in this work (such as structured and hierarchical appearance models, iterative improvement and initialization mechanisms as well as our correspondence measure) would be used in different problems. First, while they would require some changes, much of the framework would apply directly to the problem of learning object categories from loosely-labeled data, given more appropriate low-level features. Second, the correspondence measure and the iterative improvement cycle focused on optimizing this measure could be used to help learn from loosely-labeled data for a wide variety of possible appearance models. Finally, our structured and hierarchical appearance models are robust to changes in occlusion, orientation and scale and could be applied in a variety of recognition contexts.

### 7.1.2 Robustness to Visual Ambiguity

The visual ambiguities our system encounters in the training data can be roughly divided into two categories: uncertainly in the object’s position (due to cluttered backgrounds, multiple objects and multiple labels per image and lack of bounding boxes) and variation in the object’s appearance.

Lack of knowledge of the object’s position, occlusion, scale and orientation in the training data presents a difficult initialization problem. While our system is largely successful in overcoming the initialization problem, we rely on the fact that somewhat effective appearance models can be derived from grouping only a few highly-distinctive local features. More ambiguous low-level features that are better-suited to recognizing object categories could require a much more extensive and expensive search through potential configurations to find useful
seed appearance models. In such a situation, it would be useful to leverage a few clear, uncluttered images of the named object with unambiguous labels to help obtain initial models that could be refined based on the more abundant, less structured data.

While our system is reasonably robust to variations in object appearance due to scale, rotation, occlusion, deformation and small changes in viewpoint, our appearance models are in other respects somewhat brittle. Changes in lighting on a 3d surface, large changes in viewpoint or within-class appearance variation can all cause a recognition failure. Our main response to this variation is to automatically learn multiple appearance models per object, such as variations in team logos for hockey players, or different front, back, interior, day or night views for landmarks. In many cases, this is unavoidable, but it does balkanize the space of object appearances, and may leave individual appearances without enough examples to form a word–appearance association. In situations where the object appearance is more continuously variable (such as object categories), it is more appropriate to choose the underlying features so that they can span these variations.

7.1.3 Robustness to Unreliable Captions

Similar to visual ambiguity, we can classify the challenge of noisy, unreliable captions into two categories. One the one hand, we are able to deal effectively with the problem of extra, noisy unwarranted labels or multiple distinct appearances assigned the same label. However, our system is not currently designed to deal well with missing labels or alternate labels used to describe the same appearance.

Since we know that our learned appearance models rarely represent the whole range of appearance of an object, we allow multiple models to be associated with a given word. Therefore we expect that any given appearance will only be detected in a subset of the images with the associated label. Therefore, spurious label instances (such as an image labeled 'Maple Leafs' where no Leafs player is visible) do not undermined our correspondence confidence. Similarly, our system can handle noisy, unrelated labels because these are unlikely to generate reliable
word–appearance pairs.

However, in our current design, the presence of missing or synonymous labels has a more serious effect. Given that a particular appearance model will be detected in only a subset of labeled images, adding the possibility that only a small subset of the object’s appearances would be labeled makes it much harder to distinguish a meaningful co-occurrence from chance. The problem of synonyms can be ameliorated by considering groups and relationships among words. If there are large numbers of simply missing labels, however, datasets would probably have to be significantly larger to establish high-confidence correspondences.

7.2 Discussion

In the following sections, we describe some of the larger potential tasks for future work. As in any system, there is an array of worthy but smaller-scale improvements which we omit for the sake of (relative) brevity.

7.2.1 Acknowledging Connections Between Models

It is an inconvenient truth that treating visual models as independent does not, in fact, make them so. The extent to which we acknowledge this has been an important and under-examined aspect of the system since the first implementation. The translation model described in chapter 3 could use correspondence between an appearance model and a word to ‘explain away’ a weaker correspondence with a second word. However in some cases, such as a multiword name, this is not a desirable effect. In any event, the translation model was too slow to use in the inner loop of an iterative improvement scheme. In chapters 4 and 5, we developed part models independently, which simplifies parallelization. However, as discussed in chapter 6, attending to possible connections between models avoids the cost of training redundant models and can significantly improve annotation results.

One area where considering possible interactions among models could yield additional
benefits is MPM localization. We have considered using the relatively-distinctive part models learned in chapter 6 as anchors for a more systematic attempt to find the boundaries of learned objects. An MPM founded on individually distinctive parts could search in the vicinity of its vertices for additional, more ambiguous parts in order to improve localization.

Perhaps the most significant step toward modeling part dependencies is to represent object appearance using a shared hierarchy of parts. This approach might allow us to represent a large number of appearance models with relatively few low-level parts. Our initial investigation into building a shared part hierarchy for local interest features could not identify a sufficient number of shared part configurations. In retrospect, this is not surprising since we were looking for parts that were relatively ubiquitous among named objects but not shared with the background. There are likely to be few such parts unless named objects as a group have an appearance distinct from the unnamed objects composing the background. It is probably more appropriate for the first level in such a hierarchy of shared parts to consist of suspiciously common configurations of features without regard for word correspondence. It might also be helpful to reduce the dimensionality of the local features or in some other way reduce their specificity so that a small configuration of features can span small differences in appearance between objects of interest.

7.2.2 Improving Scalability

We have demonstrated that our system can learn word–appearance associations for a dozen objects from training sets with anywhere from about a hundred to a few thousand annotated images. We suspect that many potential word–appearance pairs from less-common views of named objects are overlooked due to lack of evidence in the training data. Of course, much larger annotated image sets exist for which a scarcity of examples is much less likely to be a problem. How would our system have to change to handle millions of training images? Leaving aside issues such as data structures and programming languages (to say nothing of hardware budgets), we focus on barriers to parallelization and scaling in the image annotation and training stages.
Probably the most important obstacle to scalability in the annotation process is the large increase in the number of named objects and appearance models that can be expected to accompany a large increase in the training set size. Currently, annotation involves searching for every learned appearance model in every image. Clearly, this approach does not scale well for a large model set. Our current detection function is not quite a brute force search. First, we look for feature-level correspondences between image points and model vertices, then expand to viable pair-level correspondences and finally whole graphs. This is faster than a typical sliding-window detector (though see [84]), but we should be able to do better by using an index-and-verify approach.

Our current feature set and model structure are good fits for an indexing approach to detection. Using very similar features, this approach has been quite successful for retrieving images of matching exemplars (e.g. [68]). This technique would be less effective, however, when using individually less-distinctive features (for instance, when recognizing object classes). Grouping ambiguous features can produce stronger elements for indexing, but making correct grouping decisions at indexing time can be very difficult ([39]). In this case, a hierarchy of shared parts would help alleviate the problem by making explicit the specific local feature configurations that are useful for indexing into the object model set.

Model training does not have the same performance constraints as image annotation, but here the challenge of scalability is more complex. While our current detection function is highly parallelizable, it is invoked on the entire training set for each training iteration of each appearance model. This huge number of detector calls make up the main computational cost of model training. As when annotating images, a switch to an indexing-based detection function is possible. However, indexing achieves efficiency by amortizing computation time across many potential models; for a single model, our current detection function is significantly faster. To effectively use indexing in detection, we would have to search for a large collection of half-trained models at once. This would introduce a large number of interactions and make parallelization harder. Some type of incremental detection that amortizes detection cost over
a sequence of closely-related models may be a better fit for the training stage. In addition, a hierarchical model approach with shared parts might improve scalability by, at low levels, limiting the number of shared parts to be learned and, at higher levels, reducing the number of relevant parts per image.

### 7.2.3 Moving Beyond Local Interest Points

There are many ways to characterize the look of an object, and characteristics that are distinctive for one type of object are in others merely noise. Feature types that strongly distinguish exemplars are less helpful, or even distracting, when attempting to identify object classes or relatively structureless materials. To learn all available correspondences between the language and visual domains, an ideal system would probably need to use a variety of types of features and learn types that are relevant for each object. Such tight integration is quite challenging because types of features differ not only in the sorts of things they can characterize, but also in their own fundamental character.

For instance, consider how our approach might be adapted to learn to recognize not just exemplar objects, but classes of objects with some degree of variation in surface appearance. Local interest features based on exemplar surface details can be so distinctive that even a two-vertex appearance model often has a strong enough correspondence with a particular object to serve as a starting point for model refinement. For object classes, features such as contours (e.g. [1, 7, 36, 81]) that don’t rely on such details are far more individually ambiguous. This means that far more grouping must be accomplished ‘in the dark’, without the guidance of sufficiently-strong correspondence with any caption word. In addition, surface structure that was useful in the exemplar case only buries the important contours of object classes in irrelevant detail. Therefore a system designed to learn word correspondences with object classes would probably have to conduct a broader and deeper search for initial object models, relying far more on low-level grouping hints such as collinear contour fragments.
7.2.4 Making Better Use of Language

In our work, we seek connections between words and appearance beyond one-to-one correspondences between a word and a visual feature. We seek correspondences that require grouping to be fully realized. We do not assume that a single visual feature is sufficiently informative to correspond to a word. On the other hand, we do assume that an individual word is enough by itself to correspond to a particular appearance. To deal with the ambiguities and inconsistencies of natural-language captions, we must relax this assumption. Even when dealing with exemplars such as NHL team uniforms, correspondence is improved by manually forming word associations such as “Boston” OR “Bruins”. We would like to automatically discover these meaningful word associations rather than construct them by hand.

The process could be facilitated by a preprocessing stage that attempts to measure the semantic similarity of caption words by analyzing the patterns of word occurrence in the caption text (analogous to our neighborhood cluster initialization). A variety of distributional similarity measures have been proposed for use in natural language processing applications (surveyed and compared in [54, 60]). Given a word such as ‘car’, it would be relatively straightforward to modify our system to allow related words such as ‘automobile’ or ‘hatchback’ to make a weighted contribution to the word–appearance correspondence measure.

The results of a distributional similarity measure can be improved upon by considering correspondences with recurring visual structures—i.e., the visual associations with words can contribute to our assessment of how similar the words are. In essence, we can discover meaningful representations in both the language and vision domains by trying to maximize the correspondence between the language and vision components of the training set. Changes to the language representation could be restricted to reinforcing semantic similarity scores between words or might involve forming explicit structures such as “car OR automobile”.

One of the obstacles to using captions to help label objects is that words often have multiple senses. Think of the difference in the meaning of ‘bank’ in “bank of the Mississippi” versus “Bank of America”. This word sense ambiguity might impede the search for language–
vision correspondences because a particular visual model is probably only appropriate for a particular word sense. On the other hand, the sense of a word is often much less ambiguous when considered in context (e.g. [16]). Associating visual models with some kind of ‘word neighborhood’ that can take into account such context may help address the word sense ambiguity problem. Learning visual correspondences based on particular senses of a word may also facilitate automatic word-sense disambiguation through nearby pictures (as in [6]).

A potential drawback to allowing complex structures on both the language and vision sides is increased opportunity to hallucinate correspondences. There are more opportunities for a seeming correspondence to occur in the training data by chance simply because the space of complex models is so much larger than the space of simpler models (or single elements). In order to combat this tendency, it may be necessary to adjust measured correspondence based on the complexity of the models. In effect, the improvement in correspondence must be enough to justify an increase in the model’s degrees of freedom.

In addition to forming correspondences with more complex noun-equivalents, we could conceivably learn visual correspondences for verbs (as in [59]) or word phrases that describe a relationship between elements in a scene. For instance, Gupta and Davis [40] how exploiting these comparative language elements can resolve ambiguities in noun reference as well as provide a much more complete understanding of the structure of a scene. Many of these elaborations, however, are most practical and useful if, as above, we can expand the range of things which the system can learn to recognize.

7.2.5 Building a Firmer Foundation for Grouping

Many decisions must be made in the course of developing a complex system. Our method for discovering language–vision correspondences is no exception. The process of constructing the system resembles the way the system constructs an appearance model; designed bit-by-bit, with decisions reached through tug-of-war between trench-level exigencies and birds-eye correspondence with the intuitive vision of language-driven grouping. Grouping is a messy
business, and a certain number of expedient shortcuts and empirical thresholds are unavoidable. Still, the system would benefit tremendously if we found a more mathematical approach to frame the problem.

A clever machine learning algorithm paired with a lot of data can go a long way toward eliminating parameters set by hand. However, the most popular machine learning algorithms are most readily applied when feature vectors have a fixed size and structure. Of course, the problem of efficiently learning configurations and structures has not been neglected in the machine learning literature (see, for instance [82]), but we are not yet familiar enough with this area to say what methods could be adapted to our problem. Whether it is to be found in machine learning or some other field, in a sense the search for the right formalism is another correspondence problem. Is there some special viewpoint, some way of configuring the aspects of our task that allows us to import meaning from another domain and bring some powerful procedure to bear? Or does the road to final victory run through the trenches?
Bibliography


