STRUCTURAL LEARNING FOR LARGE SCALE IMAGE CLASSIFICATION

by

Yi Shen

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Computing and Information Systems

Charlotte

2013

Approved by:

____
Dr. Jianping Fan

____
Dr. Zbigniew W. Ras

____
Dr. Min Shin

____
Dr. Aidong Lu

____
Dr. Xingde Dai

ABSTRACT

YI SHEN. Structural learning for large scale image classification. (Under the direction of DR. JIANPING FAN)

To leverage large-scale collaboratively-tagged (loosely-tagged) images for training a large number of classifiers to support large-scale image classification, we need to develop new frameworks to deal with the following issues: (1) spam tags, i.e., tags are not relevant to the semantic of the images; (2) loose object tags, i.e., multiple object tags are loosely given at the image level without their locations in the images; (3) missing object tags, i.e. some object tags are missed due to incomplete tagging; (4) inter-related object classes, i.e., some object classes are visually correlated and their classifiers need to be trained jointly instead of independently; (5) large scale object classes, which requires to limit the computational time complexity for classifier training algorithms as well as the storage spaces for intermediate results.

To deal with these issues, we propose a structural learning framework which consists of the following key components: (1) cluster-based junk image filtering to address the issue of spam tags; (2) automatic tag-instance alignment to address the issue of loose object tags; (3) automatic missing object tag prediction; (4) object correlation network for inter-class visual correlation characterization to address the issue of missing tags; (5) large-scale structural learning with object correlation network for enhancing the discrimination power of object classifiers.

To obtain enough numbers of labeled training images, our proposed framework leverages the abundant web images and their social tags. To make those web images usable, tag cleansing has to be done to neutralize the noise from user tagging preferences, in particularly junk tags, loose tags and missing tags. Then a discriminative learning algorithm is developed to train a large number of inter-related classifiers for achieving large-scale image classification, e.g., learning a large number of classifiers for categorizing large-scale images into a large number of inter-related object classes and image concepts. A visual concept network is first constructed for organizing enumorus object classes and image concepts according to their inter-concept visual correlations. The visual concept network is further used to: (a) identify inter-related learning tasks for classifier training; (b) determine groups of visually-similar object classes and image concepts; and (c) estimate the learning complexity for classifier training. A large-scale discriminative learning algorithm is developed for supporting multi-class classifier training and achieving accurate inter-group discrimination and effective intra-group separation. Our discriminative learning algorithm can significantly enhance the discrimination power of the classifiers and dramatically reduce the computational cost for large-scale classifier training.

ACKNOWLEDGMENTS

There are a few people without whom this dissertation will not be written, and to whom I am greatly indebted.

Firstly, I would like to thank my Ph.D adviser, Prof. Jianping Fan, who has the attitude and the substance of a great scientist and teacher. He has not only generously and patiently provided me guidance and mentoring in my research journey, trained me with his professional knowledge and skills, but also influenced me with his passion and excitement in regard to research. Without his persistent help, nothing would be achieved in my doctoral study.

I would also appreciate all the other professors in my dissertation committee, who are Prof. Zbigniew Ras, Prof. Shin Min, Prof. Aidong Lu, Prof, Xingde Dai, for your helpful comments, advices and critiques on my dissertation proposal and the final dissertation, especially for your time and patience for attending my dissertation defense.

Many thanks to my friends, Ning Zhou, Chunlei Yang, Youjie Zhou and Zhiqiang Ma, who have been closely working with me during my Ph.D study. Without their kindly and generous helps, it would be tough to face the challenges alone.

I would also thank my former adviser, Prof. Xiangyang Xue of Fudan University, as well as Prof. Hong Lu, Prof. Yuejie Zhang, Prof. Yuefei Guo and Prof. Cheng Jin in the multimedia lab of Fudan University, for introducing the interesting topics of machine learning and training me with the fundamental knowledge and skills.

To Dr. Bo Geng of Peking University, Prof. Xiaofei He and Prof. Deng Cai of

Zhejiang University and all others whom I co-operate with in my doctoral study, I express my sincere appreciation for their helps.

Finally, Words can not be enough to express my appreciation to my parents, I dedicate this work to them for having raised me and being my greatest support for 27 years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

For many image understanding tasks, such as object recognition and scene categorization, machine learning techniques have been widely involved to learn the classifiers from large amounts of labeled training images. Due to the tremendous number of object classes existing in the real world and the diverse intra-class impurity for each object category, large scale image classification (i.e., training a large number of classifiers for recognizing or detecting large amounts of object classes and scene categories) has become important and essential problem for the computer vision and multimedia retrieval communities, where "large scale" means: (1) the number of object classes is sufficiently large to cover all the most popular real-world object classes and scenes (image concepts); (2) the number of labeled training images are adequante to describe the intra-category diversity of the object appearances effectively and sufficiently.

1.1 Large-scale Image Annotation Systems

Large-scale image classification has been raised as the key problem in many applications such as automatic image annotation, object recognition, tag recommendation, image query-by-example system, etc.

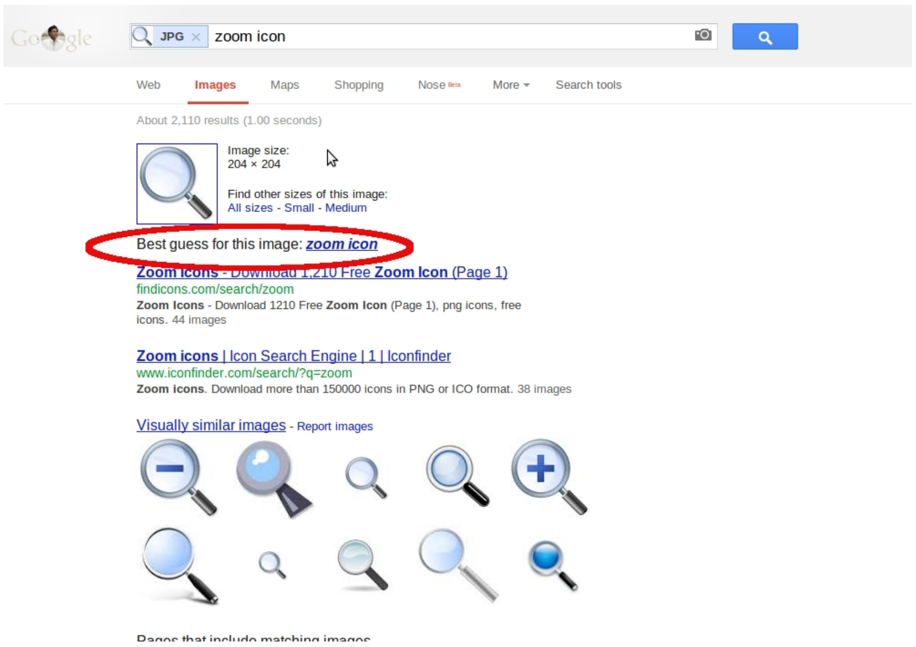
In the recent decade, with plentiful online social image systems (such as Flickr, Picasa, Google+, etc.) emerged as well as the very ubiquity of mobile devices such as smart phones and tablets, vast amounts of images have been uploaded on the web by users every day (according to the statistics of Flickr, in 2012, the number of photos uploaded to Flickr per day has reached 1.42 million). To organize the enormous number of images, it is attractive to leverage machine learning techniques for tag recommendation and automatic tag generation.

One of the earliest automatic content-based image annotation engine was ALIPR[1] made by Penn State University which supported real-time image auto-tagging with hundreds of predefined concepts in real-time. Recently, Google Inc has released an automatic image annotation system for android smart phones and iphones, Google Goggle. It leverages both the knowledge of Google Image Search as well as the mobility of smart phones, which make it possible to recognize text, landmarks, logos as well as common objects.

Since the number of images on the web has reached in the order of billions, image classification has also become a key component for content-based image search system. For large scale query-by-example image retrieval systems, text annotations are still exploited for indexing images in addition to low-level content feature indexes. For instance, Google Search-by-Image generates a best-fit text annotation for each input query image for accurately and fast retrieving images from its hybrid image index (which utilizes both texts and image contents as keys).

However, multiple issues have made it challenging to learn large scale image classifiers, which can be basically summarized into three major points:

1. It is hard to obtain large amounts of labeled training images which are qualified for classifier training. Professionally-labeled images are reliable but hiring



About 2,110 results (1.00 seconds)

Image size:
204 × 204

Find other sizes of this image:
All sizes - Small - Medium

Best guess for this image: *zoom icon*

Zoom Icons - Download 1,210 Free Zoom Icon (Page 1)
findicons.com/search/zoom
Zoom Icons - Download 1210 Free Zoom Icon (Page 1), png icons, free icons. 44 images

Zoom icons | Icon Search Engine | 1 | Iconfinder
www.iconfinder.com/search/?q=zoom
Zoom icons. Download more than 150000 icons in PNG or ICO format. 38 images

Visually similar images - Report images

Pages that include matching images

Figure 1: The Google Search-by-Image System automatically generates the machine label for each query image.

professionals to label large amounts of training samples is cost-sensitive. Even large-scale collaboratively-labeled social images are available online, socially-labeled images can not be directly utilized for classifier training due to low reliability of social tags and huge diversity of user tagging intentions.

2. The number of object categories is large which causes the learning complexity for classifier training becomes very high. In addition, object categories are visually inter-correlated, thus independent training of their classifiers is not appropriate.

3. The representations of the images should not only be able to capture the discriminative characteristics of the images, but also be efficient enough to be capable for those tasks when more than millions of images are involved.

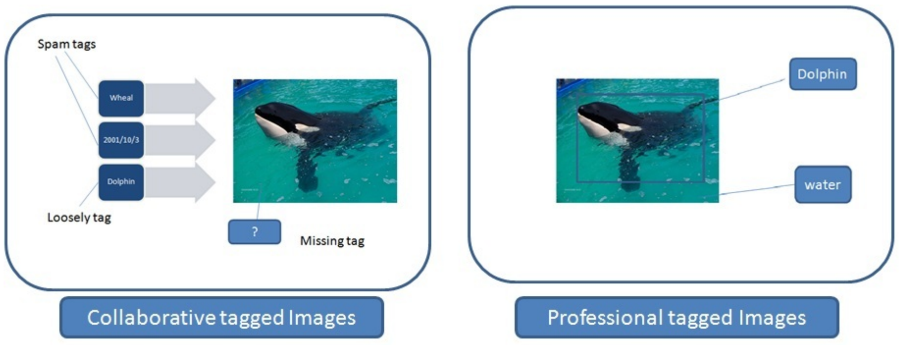


Figure 2: Weakly-tagged Image v.s. Professional-tagged Image

1.2 Leveraging Collaboratively-Tagged Social Images for Classifier Training

Collaborative image tagging system, such as Flickr [29], is now a popular way to obtain large-scale labeled images by relying on the collaborative effort of a large population of Internet users. In a collaborative image tagging system, people can tag the images according to their social or cultural backgrounds, personal expertise and perception. We call such the collaboratively-tagged social images as *loosely-tagged social images* because their social tags may not have exact correspondences with the underlying image semantics. Large-scale loosely-tagged social images can illustrate various visual properties of the object classes and their diverse appearances more sufficiently, thus it is very attractive to develop new algorithms that are able to leverage large-scale loosely-tagged social images for object classifier training.

It is not a trivial task to leverage large-scale loosely-tagged social images for object classifier training because of the following three critical issues: (a) *spam tags*, e.g., some tags are more related to popular query terms rather than the image semantics (spam tags are used to drive traffic to certain images for fun or profit), as long as some tags are improperly labeled by user's mistake; (b) *loose object tags*, e.g., multiple

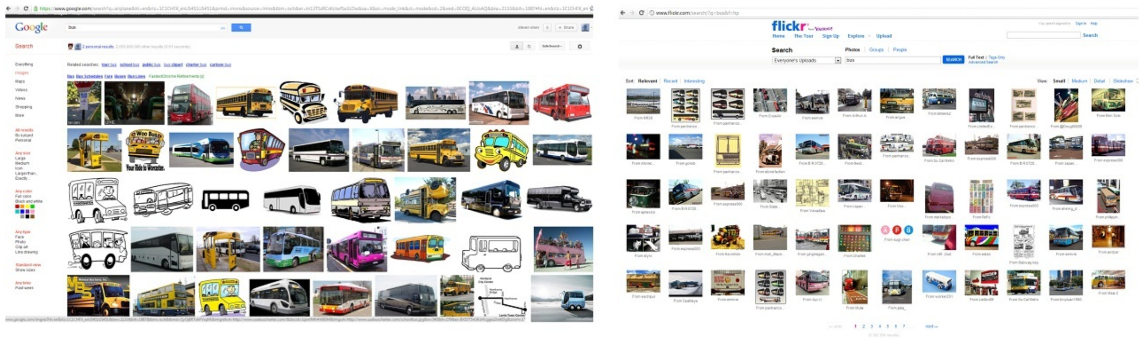


Figure 3: Images in collaborative image tagging system. Left: Google image search; Right: Flickr

object tags are loosely given at the image level without identifying the object locations in the images because it could be very time-consuming for providing object bounding boxes; (c) *missing object tags*, e.g., some object tags are missed (social tags for the loosely-tagged images are usually incomplete).

Without addressing the issue of *spam tags*, loosely-tagged images may contain large amounts of junk images and such the junk images may mislead the underlying machine learning tools for object classifier training. Without addressing the issue of *loose object tags*, extracting global visual features from whole images for object representation may dramatically decrease the discrimination power of the object classifiers. Without addressing the issue of missing object tags, loosely-tagged images with *missing object tags* may mislead the process for object classifier training because of the uncertainty of labels for the missing parts, e.g., we may treat the positive images as the negative images for object classifier training.

1.3 Constructing Large-Scale Multi-Class Classifiers

Classifier training is a critical issue for many computer vision tasks [12, 13, 14, 2], such as object detection and recognition, scene categorization, automatic image annotation. Many machine learning tools, such as multiple instance learning [38, 39, 40, 41], multi-label learning [42, 43, 44, 45, 46, 47, 48, 49], hierarchical learning [3, 4, 5, 6, 7, 9, 10, 56], have been developed for classifier training. Leveraging large-scale Web images for large-scale classifier training has recently become an attractive research topic. Compared with traditional approaches for image classification, "large-scale" reflects in both the input space and the output space: (a) the images that are available for classifier training at the input space are huge, which usually exceeds 10^5 [106, 107]; (b) the number of categories (i.e., object classes and image concepts) at the output space is also large, which usually includes hundreds or even thousands of categories.

One fundamental solution for large-scale image classification is to support *multi-class classifier training* [108, 114], and one popular approach for multi-class classifier training is to learn a set of pairwise binary Support Vector Machine (SVM) classifiers [111]. Because of the following problems, it is not a trivial task to simply extend such pairwise multi-class classifier training approach for dealing with the issue of large-scale image classification (e.g., learning a large number of classifiers for categorizing large-scale images into a large number of object classes and image concepts):

- (1) *Large Size of Output Space:* As the number of object classes and image concepts becomes large, traditional pairwise multi-class learning methods may fail due to huge computational cost;
- (2) *Inter-Concept Visual Correlations:* Some object classes and image concepts are visually-similar rather than independent, such inter-concept visual corre-

lations should be leveraged for large-scale classifier training to enhance their discrimination power and improve their efficiency and robustness;

- (3) *Imbalance of Training Images:* It is worth noting that the numbers (sizes) of available images for different categories (i.e., object classes and image concepts) could vary significantly, e.g., some popular categories may contain millions of images but some rare categories may contain just few images, which may bring serious imbalance issue for classifier training.

For many large-scale image sets, the number of categories (i.e., object classes and image concepts) has grown to the order of thousands or even more. For examples, there are totally 3,819 object classes and 908 scene categories (image concepts) in SUN image set [106] and the number of synsets in ImageNet[107] set has reached 21,841. Because of huge computational complexity [108], it is very hard if not impossible to simply extend traditional pairwise multi-class classifier training tools to learn a large number of classifiers for thousands of object classes and image concepts. Thus there is an urgent need to develop new machine learning frameworks which can deal with thousands of object classes and image concepts effectively, e.g., learning thousands of classifiers with reasonable computational cost.

In most existing approaches, each sample (image) belongs to only one category, e.g., basic assumption of exclusive categories. For large-scale image classification, the situation is much more complicated: (a) more than one label (category) could be associated with each image because one single image may contain multiple significant components (i.e., image regions for multiple object classes); (b) multiple labels (i.e., text terms for interpreting the relevant object classes and image concepts) could be dependent. Some existing work have leveraged the inter-label contexts to enhance multi-class classifier training [109, 110]. For example, ImageNet[107] image set uses a hierarchical tree (concept ontology) to organize a large number of object classes and image concepts according to their semantic correlations at the label space. Because both classifier training and automatic image classification are performed in the visual feature space rather than in the semantic label space, we should pay more attentions on inter-concept visual correlations rather than inter-label semantic correlations. To enhance the discrimination power of the classifiers, it is very attractive to train the classifiers for such visually-related object classes and image concepts jointly rather than independently.

It is also worthy to emphasize that the sizes of available training images are critical to the accuracy rates of the classifiers, e.g., integrating more training images for classifier training may usually result in higher accuracy rates on automatic image classification. Even for large-scale image sets such as SUN image set [106] and ImageNet[107] set, the numbers of available training images may vary significantly for different categories. For examples, the category "Wall" in SUN image set contains 20,213 images, while only one single image is available for hundreds of categories (i.e., object classes and image concepts) in the same image set. For those categories which contain small numbers of images, their available images may not be large enough to illustrate the inner-concept visual diversity sufficiently and learn reliable classifiers to discriminate themselves (i.e., those categories with insufficient images for classifier training) from other categories effectively. Thus lack of sufficient training images could still be a critical issue for some rare categories (i.e., less popular object classes and image concepts with small numbers of available images), which may further result in imbalance issue for classifier training.

Most existing approaches for multi-class image classification treat every category as equally important, while on the other hand, human beings may have different sensitiveness of the misclassification errors of the classifiers, e.g., human beings may be more sensitive to whether a classifier can distinguish the object class "trees" from the object class "cars", but they may be less sensitive to whether a classifier can separate the object class "oaks" from the object class "aspens".

To address these issues more effectively, a novel large-scale discriminative learning algorithm is developed in this paper to enable more effective multi-class classifier training, which is able to: (a) deal with the issue of large-scale image classification more effectively, e.g., handle not only large-scale training images at the input space but also thousands of categories (i.e., object classes and image concepts) in the output space; (b) leverage inter-concept visual correlations for large-scale classifier training which may result in the classifiers with high discrimination power; (c) define the importance of various image classification tasks for distributing more computation resources to more critical sub-tasks, e.g., less penalty can be assigned with the misclassification error for classifying the object class "oak" as the object class "aspen", more penalty can be assigned with the misclassification error for classifying the object class "oak" as the object class "car"; (d) learn from the training images that are available for other visually-related categories to deal with the imbalance issue and enhance the discrimination power of the classifiers, especially when the sizes of available training images for some categories are very small.

Our large-scale discriminative learning algorithm consists of following key components: (1) a visual concept network is first constructed for organizing a large number of object classes and image concepts according to their inter-concept visual correlations (it is worth noting that the visual feature space is the common space for classifier training and image classification); (2) a large number of object classes and image concepts (i.e., categories) are partitioned into a set of groups, where the object classes and image concepts in the same group share some common or similar visual properties and they are hard to be discriminated, on the other hand, the inter-group visual correlations are much weaker and it is much easy to distinguish the visually-dissimilar object classes and image concepts in different groups; (3) an importance factor is defined to measure the importance of various classification tasks, thus our multi-class image classification algorithm tends to put more efforts on inter-group classification (i.e., discriminating the visually-dissimilar object classes and image concepts from different groups) than inner-group classification (i.e., discriminating the visually-similar object classes and image concepts in the same group).

1.4 Structural Learning Framework

Motivated by the above issues, we propose our framework for large-scale image classification (shown in Figure 4) by addressing the previously mentioned issues, it consists of the following key components:

1. A clustering-based junk image filtering algorithm is developed to handle the issue of spam tags by automatically removing the images which are irrelevant

to the given tag.

2. To deal with loosely tagging issue, loosely-tagged images are partitioned into image instances (i.e. image regions) and an automatic instance-tag alignment algorithm is developed to assign multiple object tags to the most relevant image instances.

3. Object co-occurrence contexts are leveraged to predict the missing object tags for the uncertain image instances.

4. Inter-concept visual correlations are measured by inter-object visual similarity and a object correlation network is built to characterize the inter-concept visual correlations for large-scale object categories.

5. Structural Learning for leveraging the inter-object visual correlations to train large scale inter-related object classifiers jointly to enhance their discrimination power.

As the diagram in Figure 4 shows, our large scale image classification framework starts with the automatic junk image filtering (in Chapter 4), automatic instance-tag alignment (in Chapter 5.3), missing tag prediction (in Chapter 5.4). In Chapter 6, the GPU-accelerated clustering algorithm based on affinity propagation has been introduced. We have also developed a multi-task learning approach (in Chapter 7) to jointly train each object classifier for each object class with the classifiers for its first-order neighbours on the object correlation network.

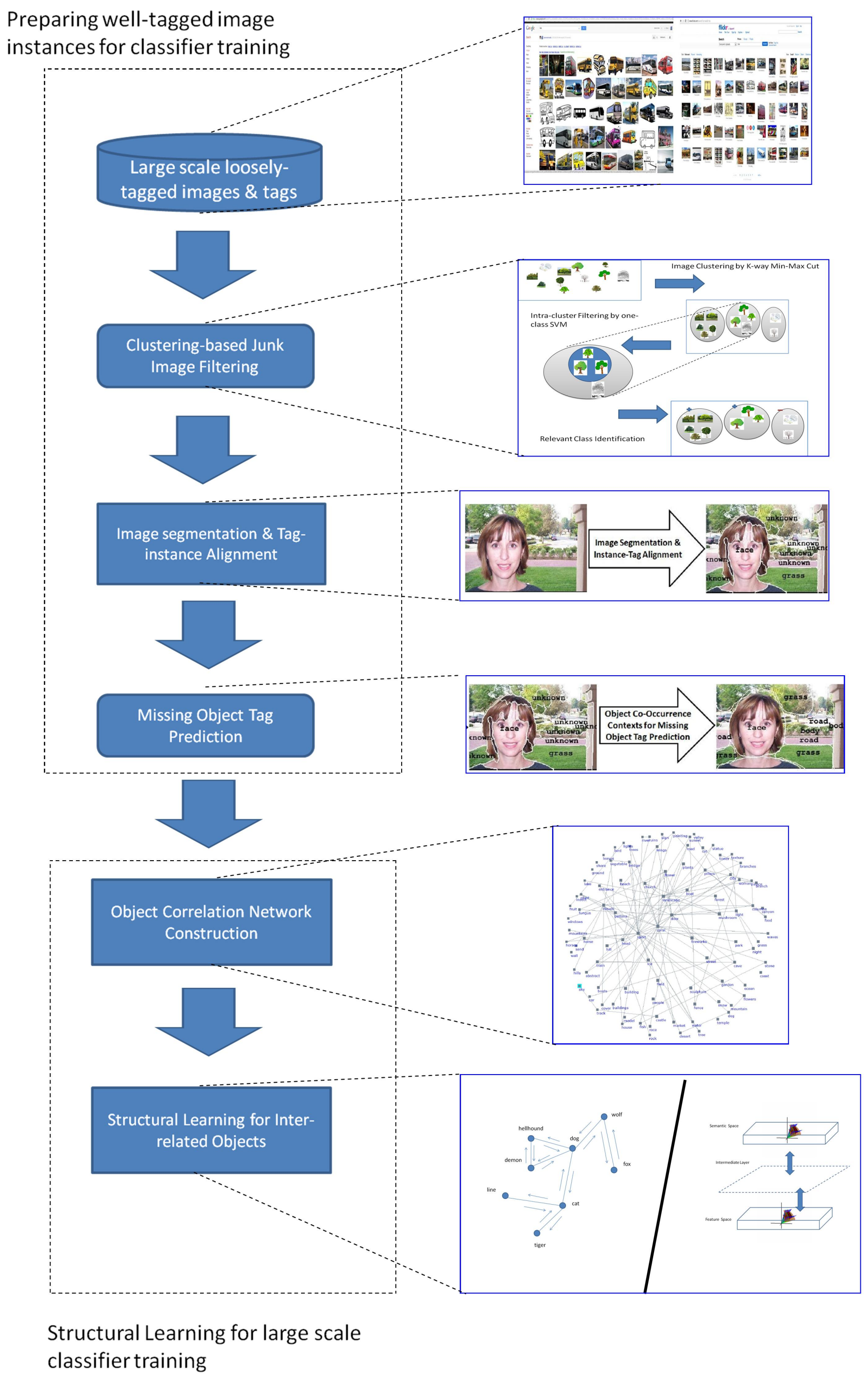

Preparing well-tagged image instances for classifier training
Large scale loosely-tagged images & tags
Clustering-based Junk Image Filtering
Image Clustering by K-way Min-Max Cut
Intra-cluster Filtering by one-class SVM
Relevant Class Identification
Image segmentation & Tag-instance Alignment
Image Segmentation & Instance-Tag Alignment
unknown
face
grass
Missing Object Tag Prediction
Object Co-Occurrence Contexts for Missing Object Tag Prediction
road
body
Object Correlation Network Construction
Structural Learning for Inter-related Objects
hellhound
wolf
dog
demon
fox
line
cat
tiger
Semantic Space
Intermediate Layer
Feature Space
Structural Learning for large scale classifier training

Figure 4: The proposed large scale image classification framework which consists of: junk image filtering, tag-instance alignment, missing tag prediction, object correlation network and structural classifier learning

Our structural multi-class classification method (in Chapter 8) is to address the following issues: (a) It is capable for large-scale classification tasks in which not only enormous image samples but also thousands of categories are involved; (b) The multi-class classifier is constructed by a series of binary discriminative classifiers which can be trained efficiently; (c) The inter-class visual correlations are leveraged to train the classifiers jointly, which not only helps distributed limited computation resources for sub-tasks, but also increases the discriminative power of the classifiers for visually-related categories; (d) categories with small size of samples can collaborate with their visually correlated categories and leverage their samples (images) to discriminate themselves with other non-correlated categories. Such process will increases the discriminative power of the whole classifier.

CHAPTER 2: RELATED WORK

The work of learning from large scale web images can be broadly divided to several sub-tasks: (1) preparing large amounts of web images with clean tags which are capable for machine learning tasks, where spam tags are identified and eliminated from the tag-list; (2) exploiting implicit tagging information from images and their loose object tags by identifying the exact relationship between tags and their true semantics in images, i.e., find and align the corresponding objects in the images with the object tags; (3) leveraging multiple types of relatedness information among the tags, including co-occurrence context, semantic dependency, feature space similarities, etc., for inter-related classifier training; (4) constructing a uniform classifier to handle the image classification problem with large scale inter-related classes.

2.1 Internet Images and their Applications

Some pioneering work have been done to leverage Internet images for computer vision tasks [15, 16, 17, 18, 19, 20, 21]. Fergus et al. [15] and Ben-Haim et al. [19] dealt with the precision problem by re-ranking the web images that are downloaded from an image search engine. On the other hand, Berg et al. [16] directly crawl large amounts of Internet images via a web search engine instead of an image search engine. Recently, Schroff et al. [17] have developed a new algorithm for harvesting image databases from the web by combining text, meta-data and visual information to achieve more precise image re-ranking. Cai et al. [20] have integrated the low-level visual features of the images, text terms of the associated text documents and linkage information of the relevant web pages for image clustering. All these existing techniques have made a hidden assumption, e.g., image semantics have an explicit correspondence with the associated text terms or nearby text terms [18, 21].

Some benchmark image sets have been generated for computer vision tasks. Catech 101 and Catech 256 have collected large amounts of images with clean background for 101 and 256 object classes [26, 27], where images are automatically downloaded from search engines (Google, etc) but manually tagged by professionals. LabelMe [25] is the first academic effort on collecting collaborative tags from a large number of users, but the diversity of LabelMe images and their tags are still limited. ESP game has been developed for acquiring large-scale labeled images through an online game [30]. On the other hand, ImageNet [28] has utilized commercial resources (Amazon Mechanic Turk) to hire social people (Internet users) to label its large scale dataset. By using large amounts of synsets of WordNet for image crawling, ImageNet focuses on collecting large amounts of Internet images and concept ontology is used for organizing large-scale image collections. These pioneering effects have generated some useful benchmark image sets with reasonable sizes and visual diversities. The key problems of these benchmark image sets can be broadly classified as two: the high labeling cost and the existence of missing object tags, e.g., object tags for these images are incomplete and some object classes in the images are not tagged.

2.2 Image Content Representation

To utilize machine learning techniques for computer vision tasks and character the visual properties of image data, image content representation, also known as "image features" or "feature extraction", has been a paramount topic in the area of computer vision, in which lots of research efforts has been made for recent decades. There are abundant types of image features which have been conducted, to characterize a variety of properties, such as color, texture, shape, etc, extracted from various sources.

In some earlier researches, visual feature vectors are extracted from the whole underlying image, using some statistical distribution to illustrate the visual outlook of the image in various aspects. Among hundreds of features for the color information, "*Color Histograms*" describes a global pixel distribution in some certain color space (most are in RGB and HSV); "*Color Moments*" embeds rough spatial information into color distribution and "*Dominant Colors*" describes the image with low-resolution color patterns [60, 61, 77]. Besides image pixel colors, textures and shape information in the image have also been characterized by multifarious visual features. One of the pioneering works on the texture description of an underlying area is the *"Tamura Features"* [63]. They defined the texture overview of a image by six high-level perceptual attributes from the psychology perspective: coarseness, contrast, directionality, linelinkness, regularity, roughness, which have widely used for the CBIR systems [64]. For texture discrimination, "*Gabor Feature*" [62], is one of the most frequent-used texture features, which capture orientated edge distribution by applying multiple Gabor filters in different scales and orientation on the image. "*Spatial Envelop*" [65], also known as the “GIST feature”, leverages a set of discriminant structural templates to illustrate the texture and shape contents in the image, which has been proved as a great success especially in large scale scene classification.

There are also a number of global features defined in the MPEG-7 standards [76, 77, 78]: (1) the *Texture Browsing Descriptor* summarizes the texture overview in a image by directionality, regularity and coarseness; (2) the *Homogeneous Texture Descriptor* embeds Gabor features for extracting homogeneous texture regions. (3) the *Edge histogram Descriptor* captures the spatial distribution of edges in the image.

Although the global features avoid a few hard issues such like object segmentation, they only capture the global visual overview of the images which have their success in some tasks like scene classification. For those tasks which required fine-grinded analysis of the images such object detection and recognition, the global features may provide insufficient information for object distinction.

For object detection and recognition, to find the discriminative characteristics of the objects in the images, the local features are often extracted instead of the global features. The most representative local feature is SIFT feature [66], which innovatively introduce the keypoint/descriptor scheme for object matching. Differed from global feature extraction, SIFT and its variants uses key-point detectors to locate the some interesting parts of the images and extract some descriptors from the neighbourhoods of the key-points. Later, a few variants of SIFT have been invented to improve both the effectiveness and efficiency of SIFT. To reduce the risk of over-fitting, PCA-SIFT [67] has combined SIFT descriptor extraction with principle component analysis. To speed up the extraction of SIFT features, SURF [68] provides an efficient approximation. ORB [70] replaces the SIFT key-point detector and descriptor extractor with a fast key-point detector, FAST [71] and some binary descriptors, BRIEF [72], to support real-time feature extraction. Taking the advantage of general purpose programming on GPUs , GPU-SIFT [69] is able to extract the SIFT features from an image in average milliseconds with a professional graphics adapter. Recently, Local Binary Patterns [73] have been found as a descent descriptor cooperating with HOG detector for object detection [74], especially for human detection [75].

To leverage the local features for classification tasks, the Bag of Words model provides a histogram representation for the image based on independent local features [79, 81], which quantize each local feature on an visual word from a pre-trained visual dictionary [80]. Since the histogram representation failed to retain spatial contexts among the local key-points, inherited from the idea of Pyramid Matching Kernel [82], Spatial Pyramid Matching Kernel [83] has been conducted for both efficient key-point matching as well as embedding spatial relationships into codeword histograms for dictionary-base features. Later, ScSPM [84] takes advantage of sparse coding and spatial pyramid matching kernel for image classification, and LLC [85] replaces the matching kernel with the linear binary codes for efficiency.

In the opposite of human engineered image features, the recent researches show that computers themselves are also capable for automatic feature learning. In [86] and [87], A. Ng *et al.*uses sparse coding to learn patch-based histograms directly from raw image pixels. Y. Bengio *et al.*[90] has proved that image representations can be learned with generic priors from raw images instead of domain specific knowledge. To justify, A. Krizhevsky *et al.*[91] have learned a strong image representation by a multi-layer deep learning network with abundant training images, which reaches the state-of-art performance in object recognition.

2.3 Image Segmentation and Multiple Instance Learning

To address the loosely-tagging issue, it is necessary to partition the images into regions, where each region is visually responsible to the semantic meaning of one certain tag. Normalized Cut [23] and JSEG[22] are two representative unsupervised methods to divide images into regions according to their visual contents. The first one performed a graph cut on the graph constructed from the relatedness between pixels, where the latter method utilized a region growth from a certain selected seed pixels.

A few previous work have been done on leveraging the loosely-labeled images for object classifier training by using multiple instance learning or semi-supervised learning. Multi-instance learning was first introduced by Dietterich to solve drug activity prediction problem [103], where each molecule may have different shapes but only one shape can bind a given protein. A typical multiple instance learning problem can be described as following: Given a training set $\{(B_1, y_1), (B_2, y_2), \ldots, (B_n, y_n)\}$, where each sample is a bag $B_i = \{x_{ij}\}$ contains several instances. For a positive bag, at least one instance (positive instance) is responsible to the label, while no positive instance is in negative bags. Current multiple instance learning approaches can be generally classified into three categories: (1) assigning the labels to all the instances to directly transform MIL problem to traditional supervised learning; (2) finding the witness instances in the positive bags which are responsible to the given tag; (3)

learning classifiers on bag level by exploiting the similarities between the bags which are summarized from the instance similarities.

Dietterich proposed a class of methods which used axis-parallel rectangle to bound the common instances of all positive bags with least known negative instances (instances from negative bags) inside. Further Maron et al.[39] characterized the possibility of an instance to be positive by Diverse Density which measured the degree of a point to be close to the positive bags and far from the negative ones simultaneously in the feature space, as Equation 1:

$$\begin{aligned} DD(x) &= Pr(x = t|B_1, B_2, \ldots, B_n) \\ &\propto Pr(B_1, B_2, \ldots B_n|x = t) \\ &= \prod_{i=1}^{n} Pr(B_i|x = t) \\ &\propto Pr(x = t|B_i) \end{aligned}$$

$$Pr(x = t|B_i^+) = 1 - \prod_{i}^{n}(1 - Pr(x = t|x_{ij}))$$
$$Pr(x = t|B_i^-) = \prod i^n (1 - Pr(x = t|x_{ij}))$$
$$Pr(x = t|x_{ij}) = e^{-||x_{ij} - t||^2} \tag{1}$$

Since the formula for the diverse density is not convex and it is time consuming to find the global optima, later on, Zhang et al. developed EM-DD algorithm to use the centroids of the positive bags as the initials and utilize EM method to recursively find the maxima of the diverse density. However, both original DD and EM-DD algorithm didn't work well on high dimensional data due to the dimensional curse and the loss of generality.

Since the discriminative models such as support vector machine has been proved to gain competitive performance on high dimension data, MI-SVM and mi-SVM [104] are two attempts to perform multi-instance learning along with support vector machine. While mi-SVM directly embedded instance identification into SVM formula, MI-SVM assigned the instance which is farthest to the margin as the witness (which is responsible to the given tag) for each positive bag. However, so far there are no efficient optimization methods to solve either MI-SVM or mi-SVM other than iterative approximations. On the other hand, the assumption that the witnesses are far away from the margin may not be always held when MIL is applied to multi-class problem, especially when the classes are inter-related.

Another class of multiple instance learning algorithms tends to learn the classifiers at the bag level. Chen et al. [40] has developed DD-SVM to map and aggregate the instance features to a new space where the bases are DD local optimals. On the other hand, a marginalized MIL kernel has been designed by Kwok et al. [105] to measure the similarities between the bags.

Cour et al. [35] have developed a semi-supervised framework to learn the object (face) detectors from the loosely-labeled images (multiple faces associated with several names). Vijayanarasimhan et al. [32, 33] have developed a multi-label multiple instance learning approach to leverage the loosely-labeled images for object classifier training via active learning. Zhou et al. and Zha et al. have integrated multi-instance learning with multi-label learning for classifier training [44, 45]. Most of these existing techniques have made a hidden assumption that the object tags for each loosely-tagged image are complete, e.g., each image instance can be assigned into one of multiple object tags that are available at the image level. Recently, Umar and Ben [37] have developed a semi-supervised learning algorithm to leverage weakly-tagged images with missing tags for object classifier training. Our MIL algorithm shares a similar goal on dealing with the loose object tags and the missing object tags jointly, but our algorithm for multiple instance learning (MIL) with missing tag prediction focuses on an unsupervised approach rather than a semi-supervised approach.

2.4 Object Correlation and Structural Learning

The object co-occurrence contexts have been derived from large-scale image collections for supporting context-driven object detection and some pioneering work have been done recently. Qi *et al.* [43] and Tang *et al.* [42] have exploited the concept co-occurrences to enhance automatic image/video annotation, and some interesting statistical models have been developed to leverage such concept co-occurrence contexts for concept classifier training. Rabinovich *et al.* have done a pioneering work on integrating the object co-occurrence contexts to improve object detection [46]. Object co-occurrence context has provided more information in addition to visual information for learning image classifiers, but the co-occurrence context may be not consistent in different data sets due to the diversity of image sources.

Besides the object co-occurrence contexts, discovering class structure semantically in the label space has gained even more attention in the computer vision society. With different similarity (or distance) definitions, the structure of classes can be exploited in various ways: Google similarity distance[128] utilized the rankings in Google search engine to measure the relevance between keywords; ImageNet[107] inherits the semantic hierarchy of synsets in WordNet[129] to organize its image categories. Although the semantic context is an informative source to exploiting relatedness among the tags, due to the existence of semantic gap, there is a possibility that visually similar objects are not similar in the semantic space [], which has limited the usage of semantic context.

Multi-task learning is one potential solution to leverage the inter-concept similarity contexts for inter-related classifier training. One major problem for multi-task learning is that there is no good solution to determine the inter-related learning tasks automatically. Fan *et al.* have integrated the concept ontology and multi-task learning for hierarchical image classification [55, 56]. Based on the idea of Conditional Random Fields (directly modeling the posterior distribution as a Gibbs field) [51], Kumar *et al.* have proposed Discriminative Random Fields (DRF) to exploit the inter-patch correlations for object detection [47]. Yang *et al.* and Chen *et al.* have recently extended such the DRF technique for image/video concept detection [58, 59]. Structural learning has been developed for leveraging the inter-concept contexts for enhancing classifier training [51, 52, 53]. To utilize the structured information provided by either the tree-formed hierarchy or similarity-based graph, there have been multiple studies to embed the inter-class relationships for classifier training. Tsochantaridis et al. [132] proposed structural support vector machine which take the inter-class relationship into account in SVM optimization. Torralba et al. [134] developed JointBoost to utilize the inter-object correlations to improve the performance of object detection. Graphical models are also adapted to the computer vision tasks. Quattoni et al. [135] and Xiang et al. [136] embedded the semantic context graph by integrating discriminative conditional random field (CRF) for object recognition and image annotation.

2.5 Multi-Class Learning

Multi-class classification has been well studied in statistical machine learning. Many existing discriminative models, such as *Multi-class SVM*[111], are derived from their single-label versions, i.e. binary classification. In general, those approaches can be classified to three types based on the construction of binary classifiers: one-against-rest, one-against-one and hierarchical one.

As one of the earliest extensions from binary *support vector machine* [112] to multi-class problems, the simplest but so far still the most widely-applied is the 1-v-r (one against rest) SVM, which models the multi-class classification problem as a label ranking problem for each individual sample. It constructs K binary classifiers[113], where each binary classifier takes the instances from one specific class as the positive samples and treats the instances from other classes as the negative samples. Those K binary classifiers can be trained independently or by solving a joint optimization problem [114]as shown in Eqn. (2).

$$\begin{aligned} \min_{\{w_k\},\{\xi_i\}} \quad & \frac{1}{2}\sum_k |w_k|^2 + \sum_i \xi_i \\ \text{s.t.} \quad & \forall y \neq y_i : \langle w_{y_i}, x_i \rangle - \langle w_y, x_i \rangle \geq 1 - \xi_i \end{aligned} \tag{2}$$

The decision for a test sample is made by taking the maximal responsibility from K classifiers: $y = \max_k \langle w_k, x \rangle$. Although the training time of 1-v-r SVM grows linearly with the number of class K, some researches [115] have obtained that the generalization error of 1-v-r SVM is unbounded, which is a major weakness of 1-v-r SVM.

The second type of methods construct multi-class SVMs from pair-wise 1-v-1 (one against one) two-class sub-classifiers, i.e. each sub-classifier is a binary SVM which in charge of discriminating two categories from K categories. Thus, there would be $K(K-1)/2$ sub-classifiers in total. Several methods have been developed for combining those two-class sub-classifiers, varied by the ways of composition. Knerr's method[116] and LibSVM[117] use "AND" gates to combine the sub-classifiers, while Max-wins [118][119] approaches suggest a voting scheme over the outputs of binary classifiers. Later on, motivated by the tendency of the over-fitness by these two methods, Platt et al. introduced Decision Directed Acyclic Graph (DDAG)[115], in which $K(K-1)/2$ two-class classifiers are organized through a decision tree. Each binary classifier forms a node in DDAG, which eliminates one class from the candidate list. Besides the generalization performance, another major advantage of DDAG is that the number of binary classifier (which a testing sample needs to go through has been reduced to $K-1$, compared to $K(K-1)/2$ in previous 1-v-1 methods. In spite of some computational efficiency concerns 1-v-1 methods on the quadratic number of sub-classifiers, 1-v-1 SVMs are naturally adaptive to distributed computing environment[117], because of the small size of sub-classifier work sets and independence between sub-classifier. Thus, 1-v-1 scheme is still an attractive potential solution for multi-class problems.

In order to handle large scale data sets, in the recent decade, numerous researches attempt to build hierarchical structure upon categories to reduce both training cost and testing time. Some methods exploited recursive top-down hierarchies through employing standard clustering algorithm, such as K-means (Vural et al.[122], Liu et al. [123]), spectral clustering (Griffin et al.[124], Bengio et al.[125]) over the inter-class similarity matrix. On the other hand, Zhigang et al.[126] and Xia et al.[127] construct the class hierarchy by merging classes in a bottom-up way. Despite the top-down or bottom-up split of classes, there are two common problems which exist in those approaches: (1) Binary partition of a set of classes does not always hold good separability in term of binary classification [121]. (2) Neither training error or generalization error of high level binary classifiers are equally distributed to sub-classes Thus, Marzalek et al. [121] developed a method which splits a confusing class into two sub-classes if the decision can not be made in the current node. Recently, T. Gao et al.[120] embedded class partition and boundary estimation into the one uniform optimization problem by maximizing the number of classes involved in each binary classifier.

Recently, to handle the classification over millions of training images and thousands of categories, classical machine learning techniques are no longer capable for large-scale tasks. Bengio *et al.*[137] introduced an online linear classification techniques, WASBIE model, taking the advantage of both hinge loss and passive-aggressive model to train a multi-class classifier over 14,000 image categories iterativelly. With the invention of deep learning network and the great leap of computational power brought by general purpose graphics processing units, G. Hinton *et al.*has made a breakthough on large-scale image classification techniques by their deep convolution neural networks [91], which has reached the state-of-art performance.

CHAPTER 3: IMAGE FEATURE EXTRACTION AND IMAGE REPRESENTATION

To characterize the visual properties of the images more sufficiently, multiple feature descriptors have been extracted from the images. Generally, state-of-the-art image features can be classified into three types: (1) Global Features, such as color histogram, edge histogram descriptor, etc., which are used to characterize the global distribution of some certain visual properties (i.e. color, texture, etc.) over the whole image; (2) Local Features, which are used to capture the visual appearance of an object or an image region; (3) Bag of Words representation, which is used to model an image as a histogram vector of codewords from a visual dictionary.

3.1 Global Descriptors

Over last decades, multifarious global image descriptors have been invented to describe various characteristics of the images in different perspectives such as color, texture, shape etc. In the section, several types of global features are briefly reviewed: color histograms, Gabor descriptors, GIST features and Tamura features, which have been involved in our work.

As one of the most widely used image descriptors, *Color histograms* have been used to characterize the empirical color distribution in the images. To construct the color histograms from the images, the color space (usually RGB or HSV) is divided into a number of histogram bins (64 in our implementation). Each bin of the color histogram quantifies the frequency of its corresponding range of colors occurred in the image. The color histogram provides an overall layout of the colors in the image and it is invariant to rotation. However, an apparent drawback of the color histogram is that different views of the images may share the same distribution of pixel colors but varied in textures and shapes. Thus, instead of stand-alone representation, color histograms are usually utilized along with other types of features.

Gabor Descriptor [62] is a typical representation of gray scale image textures. A bank of Gabor filters $\{g_{mn}\}$ in multiple scales and orientations are firstly constructed, which results same number of filtered images by obtaining convolution with the original image and Gabor functions (shown in Equation 3).

$$W_{mn}(x,y) = \int I(x,y) g^*_{mn}(x-u, y-v) \cdot du \cdot dv \qquad (3)$$

where g^*_{mn} denote the complex conjugate of g_{mn}. The mean and standard deviation in each W_{mn} are picked to form the dimensions of the Gabor descriptor, respectively representing the intensity and diversity of the edges on the corresponding direction and scale.

Some earlier research [62] on texture features have shown that *Tamura Features* have reached a great performance on texture analysis on scene images. The Tamura Features have been defined as the six factors corresponding to human visual perception: coarseness, contrast, directionality, line-linkness, regularity and roughness, while three most important ones are measured to form Tamura feature vectors:

- Coarseness: represents the size of textures. Similar to Gabor feature, the value of coarseness is calculated by multi-scale operators. It aims to measure the size of largest texture exists in the image.

- Contrast: captures the difference in luminance, measured by the range of gray-scale and distribution of black and white.

- Directionality: an edge histogram that aims to differentiate between different oriented patterns and measure the distribution of directionality.

It is worth noting that one would utilize all aspects of global features simultaneously to obtain a uniform and relatively robust global image representation. Composite-kernel is one way to integrate several types of features by linear kernel combination, as which Equation (4) shows:

$$K(u, v) = \alpha \cdot K_c(u, v) + \beta \cdot K_g(u, v) + \gamma \cdot K_t(u, v) \tag{4}$$

where $K_c(\cdot, \cdot)$, $K_g(\cdot, \cdot)$, $K_t(\cdot, \cdot)$ denote kernel functions which measure color, Gabor and Tamura similarities respectively, and α, β, γ are weight parameters which can be either predefined or estimated from a set of training data.

It has been proved that the global features can successfully capture the visual overview of the image and successfully sustain scene classification in many applications. However, the major drawbacks of traditional global image features is that they emphasize the global distribution of image pixels, equally taking all the pixels into account. Thus, usually they do not achieve good performances in object recognition because backgrounds usually take respectable portions in an image.

3.2 Keypoint-based Local Features

The local features have achieved great success in the applications of object recognition and invariant viewpoint matching. In spite of numerous kinds of local features have been developed in recent years, the SIFT-like features have been demonstrated to achieve the best performance and been widely applied in object matching tasks [133].

Scale-invariant feature transform (SIFT) [66] is originally invented by D. Lowe for invariant viewpoint object matching. Briefly, the extraction of the SIFT features takes two steps: (1) key-point detection, in which local optima (both maxima and minima) of Difference of Gaussian (DoG) mapping at multiple scale of the gray-scale image are taken as the key-points; (2) descriptor extraction: for each key-point which has been extracted in the previous step, a 128-bin edge orientation histogram was extracted from the 4×4 neighbourhood of the key-point to capture the texture context near the key-point.

Besides Lowe's original implementation, multiple keypoint-based local features inherited from SIFT have been invented either to improve the generalization of SIFT descriptors or speed up the detection and descriptor extraction of key-points. PCA-SIFT [67] and GLOH [133] are two variants of SIFT where principal component analysis (PCA) is used to reduce the dimensionality and increase the generalization and distinctiveness.

Another well-known variant is SURF [68], which replaced DoG key-point extraction by fast Hessian matrix-based measurement along with a Haar wavelet distribution as the key-point descriptor. SURF has gained comparable performance as SIFT does, but it has achieved a significant reduction on the computational cost on both key-point detection and descriptor extraction.

Another successful way to improve the computational efficiency of making the SIFT features capable for real-time image feature extraction is GPU-SIFT [69], which utilized CUDA techniques to parallelize the descriptor extraction on the key-points over multiple GPU cores (there are hundreds of cores embedded in a typical graphic card for general purpose computing) to dramatically reduce the running time of original SIFT extraction to real-time, thanks to the increasing computational power of general purpose GPUs. Recently some researchers have also developed the GPU implementations for multiple previous-mentioned detectors and descriptors such as FAST, ORB, SURF, etc., which have been included in OpenCV, the most wide-used computer vision library.

3.3 Dense Sampled Local Descriptors

Despite the success of keypoint-based local features in object detection, key-point detectors have yet been proved to be as effective as global features for the tasks of general image classification such as object recognition or scene classification. Due to the fact that the local descriptors only capture the local characteristics of the pixels around the key-points without addressing the context from the whole image, In the particular circumstance of scene classification, simple matching of the local descriptors does not perform well as which has been achieved in object detection, for the following reasons: (1) Local descriptors rely on the detection of the key-points, the visual property of an image is described only by the key-points which has been determined by the local maxima and minima of the DoG, where the key-points are usually located near the edges or complex textures of the objects. However, in some cases of scene classification, there can be few key-points detected for the some region with simply texture, which may be responsible for some tag such as "water" or "sky", no matter how large the region is.

(2) It has been observed that the number of key-points in an image can be varied due to some issues irrelevant to the semantic of the image. For examples: (a) the number of key-points is sensitive to the size of the image. Usually in the same level of texture complexity, there are more key-points in a larger image than which in a smaller image. (b) the contrast of the image impacts to the number of key-points in the image. There are less key-points in a image with low contrast than a high contrast.

To neutralize the variance of key-points extraction, in some researches, key-point detectors are replaced by dense sampling for classification problems. The local descriptors are extracted from image regions which indicated by moving windows with different scales instead of neighbourhoods of key-points, covering all the pixels over the entire image. In opposite to doing a pair-wise matching for similarity comparison, the local descriptors extracted from moving windows are quantized into codewords from a pre-trained code book and similarity between two images are defined upon the histograms of codewords from each image. Further details will be discussed in Section 3.4 and 3.5.

3.4 Bag of Words Representations

Bag of Words (BoW), which is also known as Bag of Features (BoF), stands for a feature representation method which leverages a series of local features extracted from patches (which can be obtained by either a key-point detector or dense sampling) from a image in various locations and scales. Each single local feature is mapped into one or several codewords from a pre-trained visual dictionary (code book), where each codeword in the dictionary is taken as the representative of an area in the local feature space. Further, since the local features have been mapped to the codewords, the histogram of codewords is computed to represent the visual content of the image.

There are basically three advantages of bag of words representation: (a) The bag of words representation is developed as a feasible way to leverage multiple local features from different locations and scales in an image to generate a histogram-based feature vector with the same form as global features, which can fit into general image classification/retrieval pipelines. (b) For computing the visual similarity of two images, it only requires the comparison of two histograms in BoW representation instead of pairwise feature matching, which reduced the time complexity of similarity between two images from quadratic to linear to the number of patches. (c) Since the local features have been mapped into the codewords which are taken as representatives of the local feature space and the number of codewords is much less than the number of possible values of local descriptors, it can be considered that the BoW histogram feature space is a quantized subspace of the local feature histogram space, which improves the generalization power of further classifier training with Bow representation.

Normally learning a *Bag of Words* model consists of two sub-problems: (1) Learn a visual dictionary from a set of local features. (2) Map a local feature into one or several codewords from a trained visual dictionary.

3.4.1 Visual Dictionary Learning

There are abundant researches on visual dictionary learning. Current visual dictionary learning can be basically classified into two types according to the training samples: (1) *unsupervised dictionary learning*, where codewords are trained from unlabeled images; (2) *supervised dictionary learning*, where not only images but also their labels are available during the training processes. Although some researches suggest that the supervised dictionary learning approach tends to improve discriminative power than the unsupervised dictionary learning approach, because we need our dictionary-based features to be compatible for multiple large scale classification/retrieval tasks over thousands of categories, we tend to learn a universal dictionary for all the tasks from a relative smaller reference image set. Thereby, the dictionary we use for feature extraction has been trained by a unsupervised learning algorithm.

One simple but effective method of unsupervised dictionary learning is to cluster the local features into a limited number of representatives of the local feature universe. To get a visual dictionary viable for further feature extraction on large real image database, we have picked VOC image set which contains roughly 10k photos as the reference image set for codeword learning. For each image in the reference image set, we first went through the color SIFT local feature pipeline, which detected 300-1000

SIFT key-points in the image and extracted a 384-dimension color SIFT descriptor for each key-point. Thus, 1 million color SIFT descriptors extracted from the key-point neighbourhoods have been randomly picked to form a sample pool for clustering into 4096 groups.

One typical and widely-used class of clustering algorithms is k-means. However, although visual dictionary learning only needs to be done offline, the trivial k-means algorithm requires running time in order of $O(nmd)$, where n stands for the number of samples, m is the number of groups and d represents the dimensions of feature vectors. There have been multiple feasible approaches to reduce the running time for dictionary training into an acceptable period of time:

- distribute the distance computation of k-means algorithms over multiple processors.
- leverage CUDA framework to distribute computational cost over GPUs.
- use some approximated k-means algorithm such as *Hierarchical k-means* algorithm.

In our particular case, *Map-Reduce* [50], one of the most popular computing framework has been leveraged to distribute the computation cost of hierarchical k-means clustering over a 48-core Hadoop cluster. Within Map-Reduce framework, the k-means algorithm has been decomposed into a few mappers and reducers which can be automatically distributed by the Hadoop Map-Reduce scheduler: (1) each mapper computes the distance/similarity from a particular sample point to all cluster centroids. The nearest centroid is assigned to be the cluster center for the sample point and emitted, as Algorithm 1;

```
for $i \leftarrow 1 \dots k$ do
    $d_i \leftarrow ||v - c_i||_2$
end for
$i^* = \arg\min_i d_i$
emit (key $= c_{i^*}$, value $= v$)
```

Algorithm 1: K-Means-Mapper(v, $\{c_1, \dots, c_k\}$)

(2) All the samples, which shared the same particular cluster centroid, have been automatically fed to a single reducer by map-reducer shuffler and the coordinates of the new cluster centroid are recomputed by the current samples in the reducer, as Algorithm 2, where n is the total number of points assigned to cluster centroid c.

```
$c \leftarrow \frac{1}{n} \sum_{i=1}^{n} v_i$
emit $c$
```

Algorithm 2: K-Means-Reducer(c, $\{v_1, \dots, v_n\}$)

We iteratively and hierarchically run the map-reduce process until all the cluster centroids have been converged. The 384-dimensional coordinates of the centroid point in the leaf node corresponds to the representation of a codeword of the visual dictionary in the local feature space. With the visual dictionary, for an input image, we can quantized each local descriptor into codeword in the visual dictionary, and use a series of codewords to represent the visual content of the image instead of multiple local descriptors, which will be further discusses in Section 3.4.2

3.4.2 BoW Feature Code Extraction

To extract the dictionary-base feature (BoW Feature) for an input extraction, the image is first dense-sampled by moving windows in multiple scales. For each patch enclosed by the moving window, a 384-dimensional color SIFT descriptor is extracted to quantify the color and texture characteristics of the local patch area. The remaining problem of BoW Feature Extraction is to represent each 384-dimensional by the codewords in the pre-trained visual dictionary in Section 3.4.1.

For an input image, suppose there are totally r descriptors $\{s_1, \ldots, s_r\}$ extracted by dense-sampling and the codewords in the visual dictionary are denoted by k vectors $\{b_1, \ldots, b_k\}$ in the same local feature space. The intuitive method is *Vector Quantization*, which replaces each descriptor by the closest codeword in the visual dictionary:

$$\begin{aligned} \min_{\{a_{ij}\}} \quad & \sum_i ||s_i - a_{ij} \cdot b_j||^2 \\ \text{s.t.} \quad & a_{ij} \in \{0, 1\} \\ & \sum j a_{ij} = 1, \forall i \end{aligned} \tag{5}$$

However, hard vector quantization with a large dictionary may cause the curse of dimensions which leads to over-fit, sparse coding [86] has been advocated for mapping the local descriptors to the codewords in the dictionary. Rather than each raw descriptor has been assigned to a single representative codeword, sparse coding represents the descriptor by a linear combination of multiple codewords, with sparsity constraints to improve discriminative power. The coding coefficients $\{a^*_{ij}\}$ for the i-th color SIFT descriptor s_i is determined by a sparse coding optimization problem [86] in Equation (6).

$$\{a^*_{ij}\} = \arg\min_{\{a_{ij}\}} ||s_i - a_{ij} \cdot b_j||^2 + \lambda \cdot ||a_{ij}||_1 \tag{6}$$

To obtain the feature vector for the input image, the coding coefficients for its local descriptors needs to be aggregated into a vector of coefficients x which dimension equals to the size of the dictionary m, by either average pooling strategy,

$$\forall j \in \{1 \ldots m\} : x_j = \frac{1}{n} \sum_i a_{ij}^* \tag{7}$$

or max pooling strategy,

$$\forall j \in \{1 \ldots m\} : x_j = \max_i |a_{ij}^*|_1 \tag{8}$$

where each coefficient of the dictionary-based feature vector, x_j, representing the strongest occurrence of the j-th codeword in the input image.

The χ kernel function, which usually appears for comparing histograms, is advocated to measure the visual similarity context $\kappa(x, y)$ between two dictionary-based feature vectors x and y:

$$\kappa(x, y) = e^{-\chi^2(x,y)/\sigma} = \prod_{j=1}^{m} e^{-\chi_j^2(x_j, y_j/\sigma_j} \tag{9}$$

where $\sigma = [\sigma_1, \cdots, \sigma_m]$ is the set of the mean values of the χ^2 distances. The χ^2 distance $\chi_j^2(x_j, y_j)$ between x_j and y_j is defined as:

$$\chi_j^2(x_j, y_j) = \frac{1}{2} \cdot \frac{|x_j - y_j|^2}{x_j + y_j} \tag{10}$$

where x_j and y_j are the j-th coefficient of two codeword histograms.

The major advantage of BoW presentation in object recognition and scene categorization is that BoW not only captures the local characteristics around the key-points but also provides a global distribution of the the local features. In addition, the utilization of codewords can further enhance the generalization of BoW representation, which improves the robustness and distinctiveness in object/scene classification tasks.

Despite the proficiency of Bag of Words representation in classification/retrieval cases, in many applications such as image retrieval or online image classification, feature extraction for an input image needs to be done within one second in real-time. On the other hand, both vector quantization and sparse coding requires fast k-nearest neighbour retrieval in high dimensional feature space, yet brute-force ranking to locate k-nearest neighbour goes through every codeword in the visual dictionary and computes the pairwise distance/similarity between the query descriptor and each codeword, which demands $O(md)$ running time for a single descriptor and $O(rmd)$ for an image, where r is denoted for the number of patches in the image.

To support real-time Bag of Words coding extraction, k-nearest neighbours are enforced to be retrieved within sub-linear time complexity. Therefore, a high dimensional kd-tree index has built on top of the visual dictionary to support approximated nearest neighbour search, as Algorithm 3 shows.

```
$i \leftarrow \text{random} \mod d$
$v_m \leftarrow$ the median of $\{v_1(i) \ldots, v_n(i)\}$.
node.idx $\leftarrow i$
node.value $\leftarrow v_m$
node.left $\leftarrow$ kd-tree$(\{v_j | v_j(i) <= m\}, depth + 1)$
node.right $\leftarrow$ kd-tree$(\{v_j | v_j(i) <= m\}, depth + 1)$
```

Algorithm 3: kd-tree$(\{v_1, \ldots, v_n\}, depth)$

With the codeword index available, the time complexity of approximated k-nearest neighbour search has been reduced to $O(d \log n)$ by a heuristic searching algorithm (Algorithm 4).

```
$i \leftarrow$ node.idx
$v_m \leftarrow$ node.value
if $s > ||v_k^* - v_q||_1$ then
    break
end if
if $||v_m - v_q||_1 < ||v_k^* - v_q||_1$ then
    replace $v_k^*$ with $v_m$ and re-sort $\{v^*\}$
end if
if $v_m(i) \leq v_q(i)$ then
    kd-search($v_q$, $s + v_q(i) - v_m(i)$, node.left)
    kd-search($v_q$, $s$, node.right)
else
    kd-search($v_q$, $s$, node.left)
    kd-search($v_q$, $s + v_m(i) - v_q(i)$, node.right)
end if
```

Algorithm 4: kd-search(v_q, s, node)

Table 1: Speed boost with dictionary index

	Brute Force Search	kd-tree Index
300 color SIFT descriptors	25 secs	< 1 sec
1000 color SIFT descriptors	83 secs	< 1 sec
300 raw SIFT descriptors	40 secs	< 1 sec
1000 raw SIFT descriptors	107 secs	3 secs

Table 1 shows that compared to the brute-force search, k-d tree index has successfully reduced the running time to meet the requirement of real-time feature extraction.

3.5 Spatial Pyramid Matching Kernel

In the Bag of Words model, similarity between two images can be efficiently calculated by comparing two dictionary-based histograms rather than pairwise matching. Nevertheless, though the histogram-based global features as well as BoW representation still enjoy their success in large-scale whole-image classification, as a matter of fact, spatial contexts among local image patterns have been completely lost when the visual contents of the images are only characterized by histograms: Once a set of local descriptors of the input image are encoded into a dictionary-based feature, no spatial correlations between those local keypoints/patches have been preserved.

Spatial Pyramid Matching Kernel [83], derived from *Pyramid Matching Kernel* [82], has been invented to embed spatial locality information of local patches into histogram kernel computation. Pyramid Matching Kernel was originated to approximately aggregate of pairwise correspondence kernels between two sets of descriptors by dividing the local feature space by a series of cascaded grids with multiple levels. At each level l, the whole feature space is evenly divided into 2^{dl} high dimensional cells. While the descriptors fall into the same cell of level l are counted in the same histogram bin, the min kernel function of two image X and Y at level l is defined as Equation 11,

$$I(H_X^l, H_Y^l) = \sum_{i=1}^{2^{dl}} \min(H_X^l(i), H_Y^l(i)), \tag{11}$$

where $H_X^l(i)$ and $H_Y^l(i)$ respectively stands for the value of the i-th histogram bin of image X and Y at level l.

With the min kernel function at each cascaded level, the Pyramid Matching Kernel $\kappa(X, Y)$ of two images X, Y has been aggregated with exponentially-decayed weights, in Equation 12,

$$\begin{aligned}\kappa(X, Y) &= I^L + \sum_{l=0}^{L-1} \frac{1}{2^{L-l}}(I^l - I^{l+1}) \\ &= \frac{1}{2^L} I^0 + \sum_{l=1}^{L} \frac{I^l}{2^{L-l+1}},\end{aligned} \tag{12}$$

where I^l was the abbreviation of $I(H_X^l, H_Y^l)$.

As the Pyramid Matching Kernel plays as the role of an alternative of dictionary-based histogram to speed up local descriptor matching, the Spatial Pyramid Matching Kernel has been motivated by taking the advantage of spatial relationship among local patches/keypoints, though it has inherited the exact same form from the Pyramid Matching Kernel in calculation. Instead of separating high level feature space, the Spatial Pyramid Matching Kernel divides the image by multiple 2-D cascaded grid in multiple level, and kernel function I^l at level l is redefined by,

$$I^l = \sum_{i=1}^{2^l} I(H_X, H_Y | g_l(i)), \tag{13}$$

where H_X and H_Y respectively stands for the dictionary-based features for the images X and Y, $I(H_X, H_Y | g_l(i))$ denotes the intersection of H_X and H_Y within the i-th grid of level l.

Similar to the Pyramid Matching Kernel, the Spatial Pyramid Matching Kernel also sum up kernel function from all the levels with decayed weights, as Equation 12, for the matching in a smaller grid gain more weight than one in a larger grid. Therefore the global spatial contexts have been enforced by applying the spatial pyramid matching, but with a slight increase of computation.

CHAPTER 4: OBJECT TOPIC EXTRACTION AND JUNK IMAGE FILTERING

Since there are huge amount of images available on the Internet, collaborative image tagging system (such as Flickr) and Google Image search has become popular resources to obtain large scale images for many computer vision tasks. It is very attractive to leverage those online images for automatic image annotation applications. Although abundant social images are available on the Internet, it is not a trivial task to obtain the images which are associated appropriately with desired object tags. As we mentioned previously, many online images are not helpful for object classifier training because of the following cases: (1) one image is improperly associated to its corresponding tag, i.e. the visual content of the image does not match the semantic meaning of the tag (error tags or spam tags); (2) the image tag itself is not interesting in term of object classification, i.e. little relevance is observed between the semantics of those tag (e.g. date tags, action tags, etc.) and the most popular real-world object classes. Thus, our desired large scale image sets should be associated with high-quality tags and satisfy: (a) There are sufficient images associated to each tag, which provides enough information to illustrate the principal visual properties of the relevant images for the given tag; (b) The image-tag association is reliable: for each image, the associated tags can correctly reflect the semantics of its visual content (although loosely tagging is accepted and one image may be associated to multiple tags). The goal of the first step of our image classification system is to obtain such a clean image with reliable tag-image associations.

There are various ways to download massive amounts of images from the Internet. The classic way is to design a web crawler, which starts from a list of seed web sites, recursively goes though the links on the web pages and downloads the images and their corresponding texts from the current web pages. However, nowadays Internet search services (including search engines like Google and built-in search services such as Flickr keyword search) are highly developed and open to users. Even the quality of returned images are unsatisfied due to the existence of large number of junk images (which are irrelevant to the semantic to the keyword that is used as query term), there is a simplified way to obtain large scale images just by utilizing the results returned by search engines. Therefore, the task for image collection has been basically reduced to two problems: (1) determine a set of interesting tags and use them as the keywords (query terms) to search engines; (2) filter junk images (which are irrelevant to the given tags) from the returned images.

4.1 Object Topic Extraction

In a collaborative image tagging space, each loosely-tagged image is associated with the image holder's tags of the underlying image content and other users' tags or comments. It is worth noting that entity extraction can be done more effectively in a collaborative image tagging space. In this paper, we first focus on extracting the social tags which are strongly related to the most popular real-world object classes and their actions. The social tags, which are related to image capture time and place, are also very attractive, but they are beyond the scope of our work. Thus the image tags

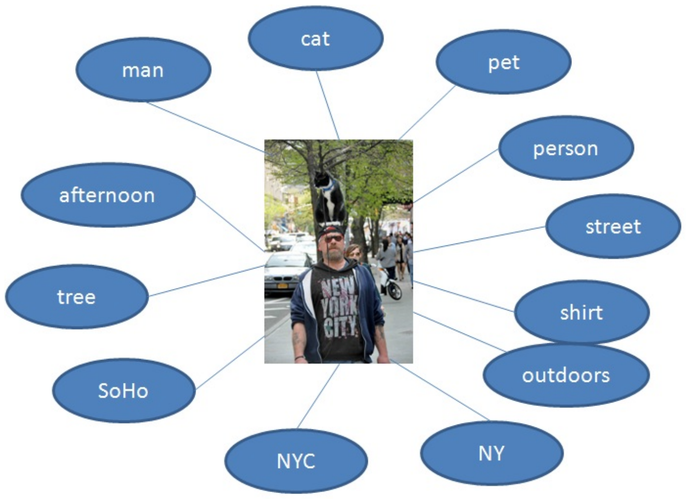


Figure 5: Sample Image with Collaborative Tags: not all tags are useful for object classification (wrong tags or uninteresting tags) and some objects are missed from tags

are first partitioned into two categories: noun phrases *versus* verb phrases. The noun phrases are further partitioned into two categories automatically: *object-relevant tags* (i.e., object tags) and *object-irrelevant tags*. The verb phrases are further partitioned into two categories automatically: *action-relevant tags* (i.e., object action tags) and *action-irrelevant tags*.

The occurrence frequency for each object-relevant tag and each action-relevant tag is counted automatically by using the number of relevant loosely-tagged images. The misspelling tags may have low frequencies (i.e., different people may make different typing mistakes), thus it is easy for us to correct such the misspelling tags and their images are added into the relevant tags automatically. Two tags, which are used for tagging the same image, are considered to co-occur once without considering their orders. A co-occurrence matrix is obtained by counting the frequencies of such pairwise tag co-occurrences.

The object-relevant tags and the action-relevant tags are further partitioned into two categories according to their interestingness scores: *interesting tags* and *uninteresting tags*. In this research, multiple information sources have been exploited for determining the interesting tags more accurately. For a given tag C, its interestingness score $\omega(C)$ depends on: (1) *its occurrence frequency* $t(C)$ (e.g., higher occurrence frequency corresponds to higher interestingness score); and (2) *its co-occurrence frequency* $\vartheta(C)$ with any other tag in the vocabulary (e.g., higher co-occurrence frequency corresponds to higher interestingness score). The occurrence frequency $t(C)$ for a given tag C is equal to the number of loosely-tagged images that are tagged by the given tag C. The co-occurrence frequency $\vartheta(C)$ for the given tag C is equal to the number of loosely-tagged images that are tagged jointly by the given tag C and any other tag in the vocabulary.

The *interestingness score* $\omega(C)$ for a given tag C is defined as:

$$\omega(C) = \xi \cdot \log(t(C) + \sqrt{t^2(C) + 1}) + (1 - \xi) \cdot \log(\vartheta(C) + \sqrt{\vartheta^2(C) + 1}) \qquad (14)$$

where ξ is the weighting factor.

All the interesting tags, which have large values of $\Omega(\cdot)$ (i.e., top 5000 tags in our current experiments), are treated as *image topics*. In this work, we focus on only the image topics that are used to interpret the most popular real-world object classes and object actions (i.e., object tags). It is worth noting that one single loosely-tagged image may be assigned into multiple object classes when the relevant object tags are used for tagging the loosely-tagged image jointly. Collecting large-scale training images for the most popular real-world object classes and learning their object classifiers more accurately are crucial for many computer vision tasks.

4.2 Junk Image Filtering

Since current search engines index images by either the keywords which are extracted from surrounding texts of images or noisy user tags, there may not have exact correspondence between the tags and their linked images. Junk images, which show little semantic relevance to a given tag, may exist. To utilize the online images for computer vision tasks, a junk image filtering algorithm needs to be developed to eliminate those irrelevant images from the image list of each tag.

For different users, their motivations for spam tagging are significantly different, thus the junk images for spam tagging should contain different visual content and have different visual properties with the relevant images. Thus, junk images, which come from different users with different motivations, should be distributed more diversely in appearances than the relevant images do. Since the intrinsic appearances among relevant images might be diverse as well, for each tag, a three-step junk image filtering approach has been developed to eliminate the junk images and identify those relevant images:

1. K-way Min-Max Cut Clustering algorithm to partition the whole set of images (which are associated with a given tag) into K groups according to their visual similarity.

2. One-class SVM algorithm to eliminate the outliers which have significant visual inconsistency with other images in the same group.

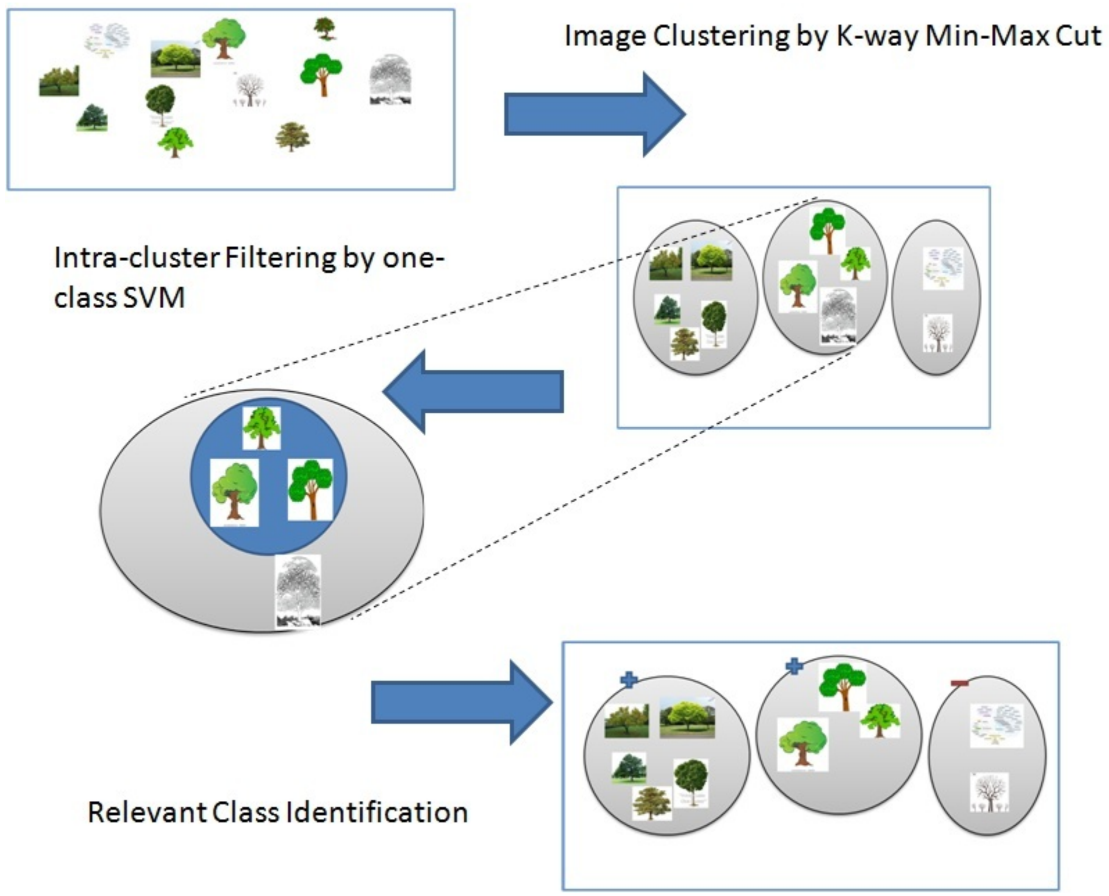


Image Clustering by K-way Min-Max Cut
Intra-cluster Filtering by one-class SVM
Relevant Class Identification

Figure 6: Overview of Three-step Junk Image Filtering Approach

3. Random walk algorithm to identify those groups which are most relevant to the given topic (i.e, the given tag).

For a given image concept (tag) C with n associating images, a image graph is constructed according to their visual similarity, where the nodes are the images and the edges are weighted by their visual similarity $\kappa(\cdot,\cdot)$. All the images associated with the tag C can be partitioned into K groups automatically by minimizing the following objective function for K-way Min-Max Cut:

$$\min_{G} J(C,K,G) = \sum_{i=1}^{K} \frac{s(G_i, G \backslash G_i)}{s(G_i, G_i)} \tag{15}$$

where $G = \{G_i | i = 1 \dots K\}$ is used to represent K image clusters (groups), $G \backslash G_i$ denotes the set of image clusters without G_i. The cumulative intra-cluster similarity and inter-cluster similarity are defined respectively as:

$$s(G_i, G_i) = \frac{1}{|G_i|^2} \sum_{u \in G_i} \sum_{v \in G_i} \kappa(u, v),$$

$$s(G_i, G \backslash G_i) = \frac{1}{|G_i| \cdot (n - |G_i|)} \sum_{u \in G_i} \sum_{v \notin G_i} \kappa(u, v). \tag{16}$$

To solve the optimization problem Equation (15) in matrix form, we define $X = [X_1, \dots, X_i, \dots, X_K]$ where X_i is the binary indicator vector in which each element is to identify whether the corresponding image has been assigned to cluster i, i.e.:

$$X_i(u) = \begin{cases} 1 & u \in G_i \\ 0 & u \notin G_i \end{cases}$$

and W is defined as a $n \times n$ symmetric matrix with entries $W_{uv} = \kappa(u, v)$, D is defined as a diagonal matrix which is calculated from W: $D_{uu} = \sum_{i=1}^{n} W_{uv}$.

The optimal partition of the image set associated with tag C can be obtained by solving:

$$\begin{aligned} \min_X \quad J(X) &= \sum_{i=1}^{K} \frac{X_i^{\mathrm{tr}}(D - W)X_i}{X_i^{\mathrm{tr}} W X_i} \\ &= \sum_i \frac{X_i^{\mathrm{tr}} D X_i}{X_i^{\mathrm{tr}} W X_i} - K \end{aligned} \tag{17}$$

Further let $\tilde{W} = D^{-1/2} W D^{-1/2}$ and $\tilde{X}_i = D^{1/2} X_i / ||D^{1/2} X_i||$, the above objective function can be reduced to a standard spectral graph cut problem:

$$\min_{\tilde{X}} \sum_{i=1}^{K} \frac{1}{\tilde{X}_i^{\mathrm{tr}} W \tilde{X}_i} \tag{18}$$

Therefore, the optimal solution for Equation (15) can be obtained by solving the eigen-problem:

$$\tilde{W} \cdot \tilde{X}_i = \lambda_i \cdot \tilde{X}_i, i \in [1 \dots K] \tag{19}$$

By the K-way Min-Max cut, the whole set of images tagged by concept C have been partitioned K groups. Due to the high diversity of junk images, there are two possible situations: (1) Some junk images have been distributed to some clusters which the majority image of the cluster are relevant to the given concept; (2) Some junk images have formed some junk clusters in which most images are irrelevant to the concept. To eliminate the junk images in the first case, one-class support vector machine is utilized to generate a core set from each group individually, where as minority, junk images are guaranteed to sit outside the core set.

One-class SVM [130] is a unsupervised method, which find the smallest possible hyper-sphere to enclose all samples $\{x_i\}$ with an error-tolerated boundary. The algorithm makes a trade off between the size of hyper-sphere and the error penalty by solving the optimization problem in Equation (20):

$$\begin{aligned} \min \quad & R^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \forall x_i : ||x_i - z||^2 \leq R^2 + \xi_i, \xi_i \geq 0 \end{aligned} \tag{20}$$

where R and z indicate the radius and the center of the hyper-sphere, xi_i is the error penalty and C is the regulation parameter. The core of the category is formed by those x_i which tightly enclosed by the hyper-sphere, which satisfies $\xi_i = 0$ correspondingly. The core set of samples $\{x_i\}$ is formed according to distances (similarities) to the center z: $\{x_i \mid ||x_i - z||^2 \leq R^2\}$. Further, image samples outside the core are discarded from each group due to the higher possibility to be the junk images.

Since all the image groups have been cleansed by one-class SVM, the remaining problem is to identify the relevance between the groups and the given tag C. In order to leverage the inter-cluster visual correlations for achieving more effective relevance measurement, a random walk process is performed for automatic relevance score refinement. For the given object tag C, an inter-cluster correlation network (i.e., K image clusters and their inter-cluster visual correlations) can be automatically determined upon the image clusters filtered by one-class SVM. We use $\rho_l(G_i)$ to denote the relevance score for the ith image cluster G_i at the lth iteration. The relevance scores for all these K image clusters at the lth iteration will form a column vector $\overrightarrow{\rho(G_i)} \equiv [\rho_l(G_i)]_{K\times 1}$. We further define Φ as an $K \times K$ transition matrix, its element ϕ_{G_i,G_j} is used to define the probability of the transition from the image cluster G_i to its inter-related image cluster G_j. ϕ_{G_i,G_j} is defined as:

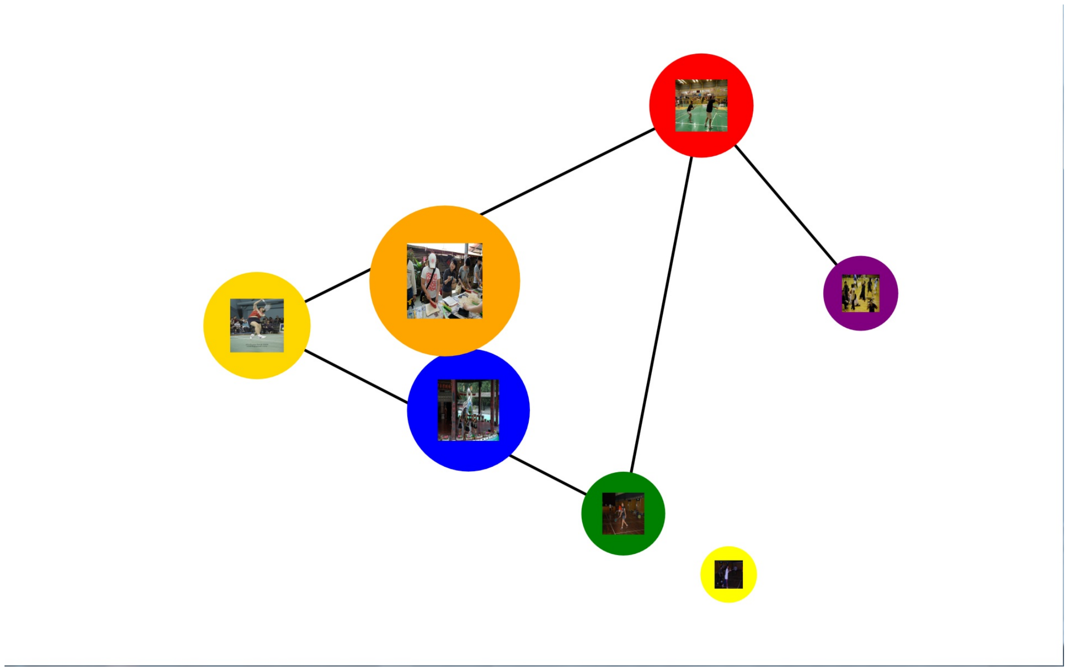
$$\phi_{G_i,G_j} = \frac{c(G_i, G_j)}{\sum_{G_h \in C} c(G_i, G_h)} \tag{21}$$

where $c(G_i, G_j)$ is the inter-cluster visual similarity context between two image clusters G_i and G_j as defined as following:

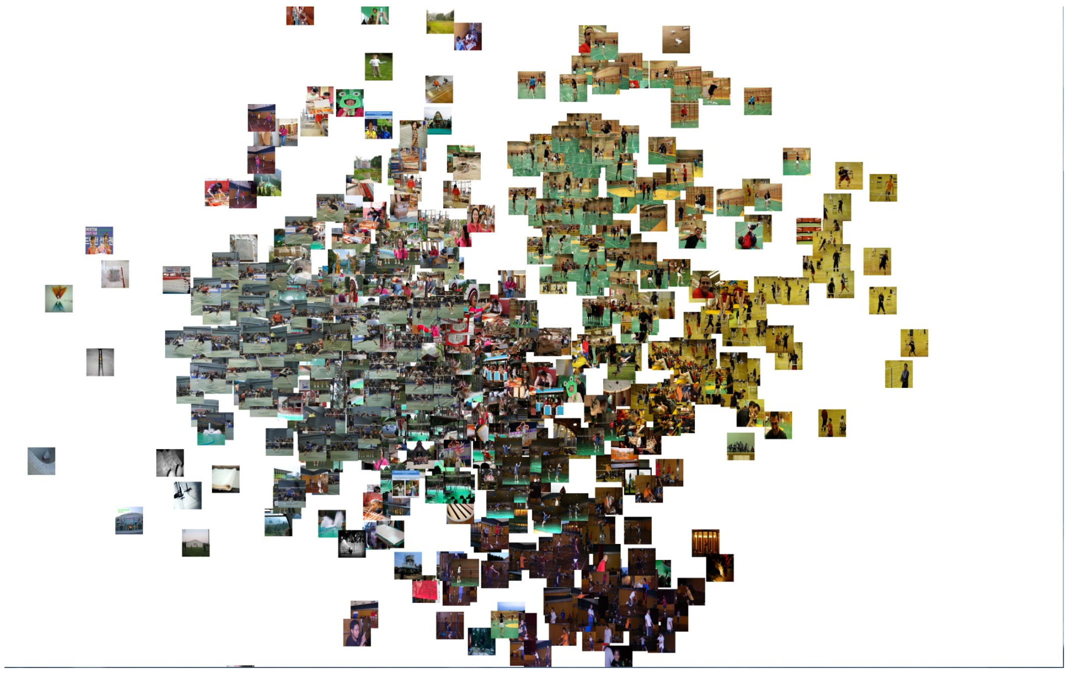
$$c(G_i, G_j) = \frac{2s(G_i, G_j)}{s(G_i, G_i) + s(G_j, G_j)} \tag{22}$$

The random walk process is then formulated as:

$$\rho_l(G_i) = \theta \sum_{j \in \Omega_j} \rho_{l-1}(G_j)\phi_{G_i,G_j} + (1-\theta)\rho(C, G_i) \tag{23}$$



(a)



(b)

Figure 7: Image clustering results: (a) inter-cluster correlation network; (b) filtered junk images.

where Ω_j is the first-order nearest neighbors of the image cluster G_j on the cluster correlation network, $\rho(C, G_i) = |x_i \mid r_C(x_i) \leq R$ and $x_i \in G_i|$ is the initial relevance score for the image cluster G_i which is derived from ranking function $r_C(x_i)$ of the search engine and θ is a weight parameter.

This random walk process will promote the image clusters that have many connections on the inter-cluster correlation network, e.g., the image clusters that have close visual properties (i.e., stronger visual similarity contexts) with other image clusters. On the other hand, this random walk process will also weaken the isolated image clusters on the inter-cluster correlation network, e.g., the image clusters that have weak visual correlations with other image clusters. This random walk process is terminated when the relevance scores converge.

By performing random walk over the inter-cluster correlation network, our relevance score refinement algorithm can re-rank the relevance between the image clusters and the given object tag C more precisely. Thus the top-k image clusters, which have higher relevance scores with the given object tag C, are selected as the most relevant image clusters for the given object tag C. Through integrating the cluster correlation network and random walk for relevance re-ranking, our spam tag detection algorithm can filter out the junk images effectively.

Although it can be observed from results that our junk image filtering algorithm is performing with a low recall rate, it is worth noting that since images return by search engine are usually enormous, there always be enough image available in spite of strictness of relevance measurement. Figure 9 shows the results of junk image filtering upon returned images from Flickr and Figure 8 measures the effectiveness

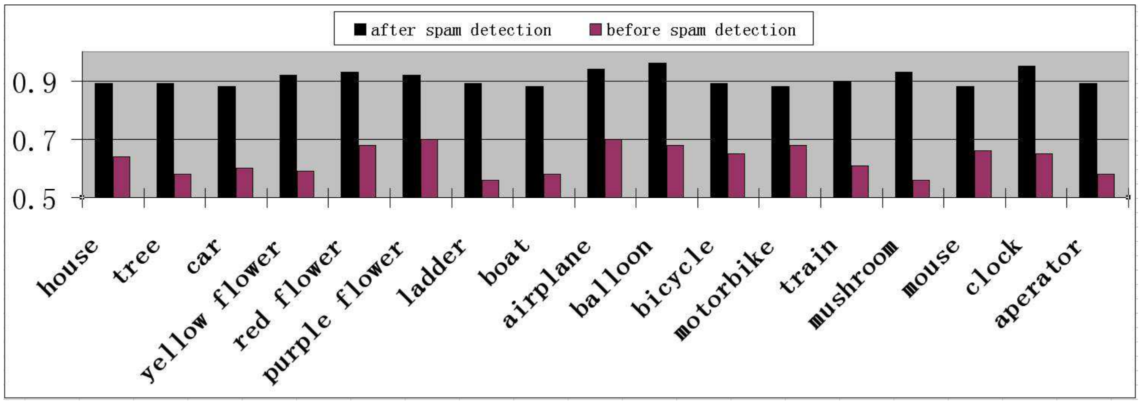
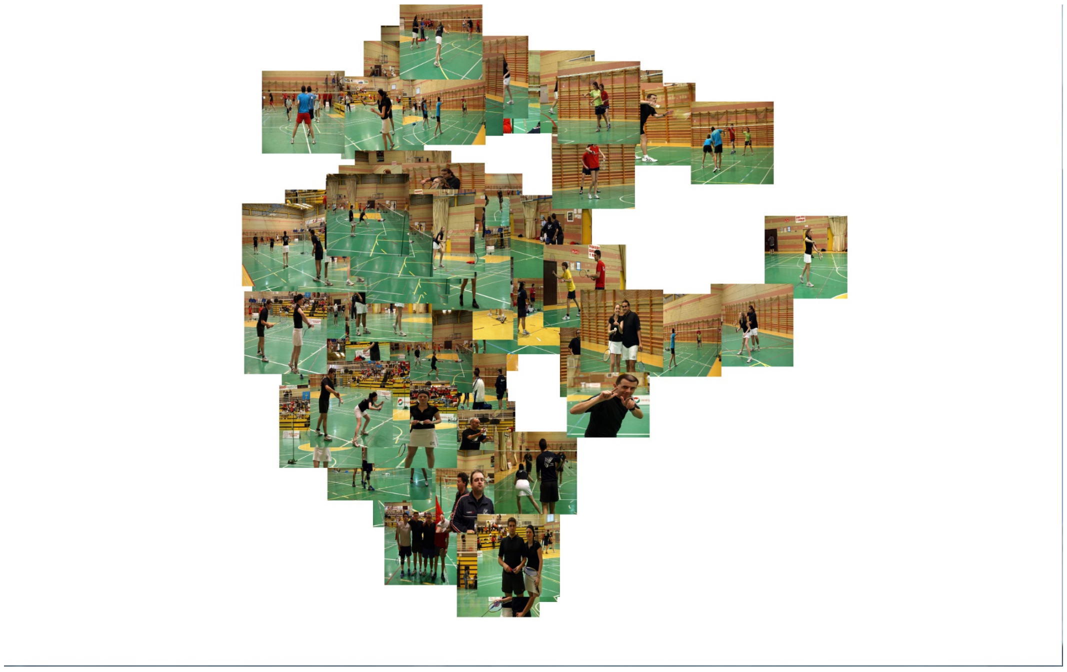


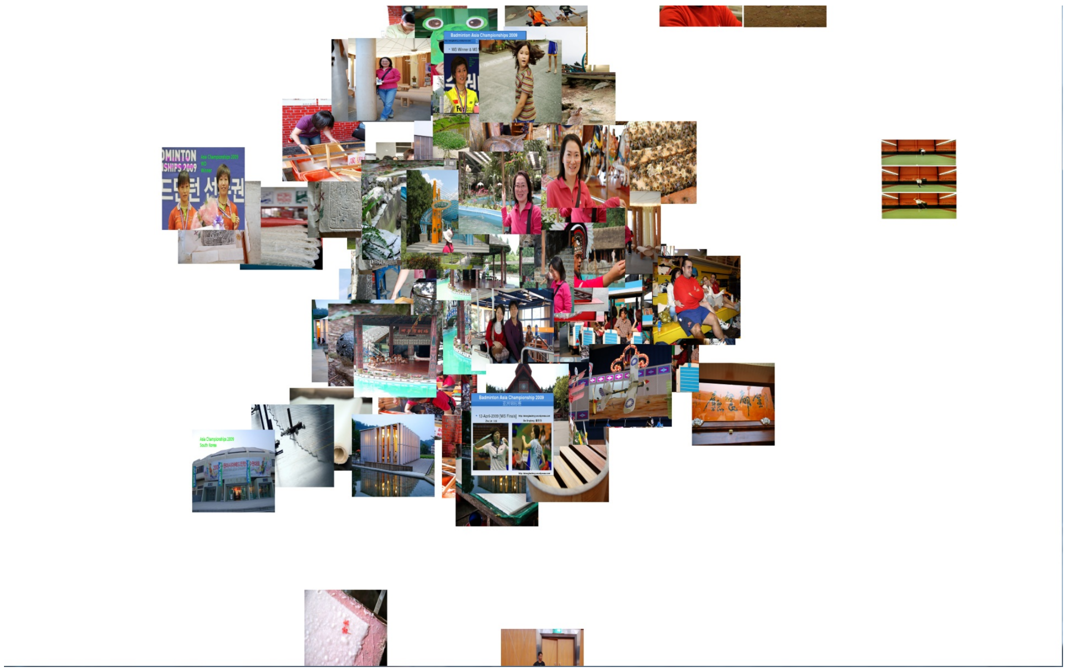
Figure 8: The comparison on the accuracy rates after and before performing junk image filtering

.

of our algorithm in the classification task. By filtering out the junk images, we can automatically create large-scale training images with more reliable labels for object classifier training.



(a)



(b)

Figure 9: Junk image filtering results on Flickr. (a): relevant cluster. (b): junk cluster

CHAPTER 5: MULTIPLE INSTANCE LEARNING WITH MISSING OBJECT TAGS

Multiple object tags are usually associated with the loosely-tagged images, but some object tags may be missed because of time limitation or lack of patience for tagging the object classes completely. Thus it is not a trivial task to leverage large-scale loosely-tagged images with missing tags for object classifier training: (a) each object may possess only a small part of the loosely-tagged images and the information for object locations and sizes are missed, extracting the global visual features from whole images may not be representative and discriminative for object representation and incorporating such global visual features for object classifier training may result in low discrimination power; (b) because of the appearances of missing object tags, positive images (which contain the relevant object classes but they are not tagged) may be treated as negative bags, and the appearances of positive instances in the negative bags will seriously mislead the underlying machine learning tools for object classifier training.

To address both the issues of loose object tags and missing object tags more effectively, a novel algorithm for multiple instance learning (MIL) with missing tag prediction is developed and its flowchart is illustrated in Figure 10. To achieve automatic instance-tag assignment and missing tag prediction, each loosely-tagged image is first partitioned into multiple image regions by integrating the segmentation results

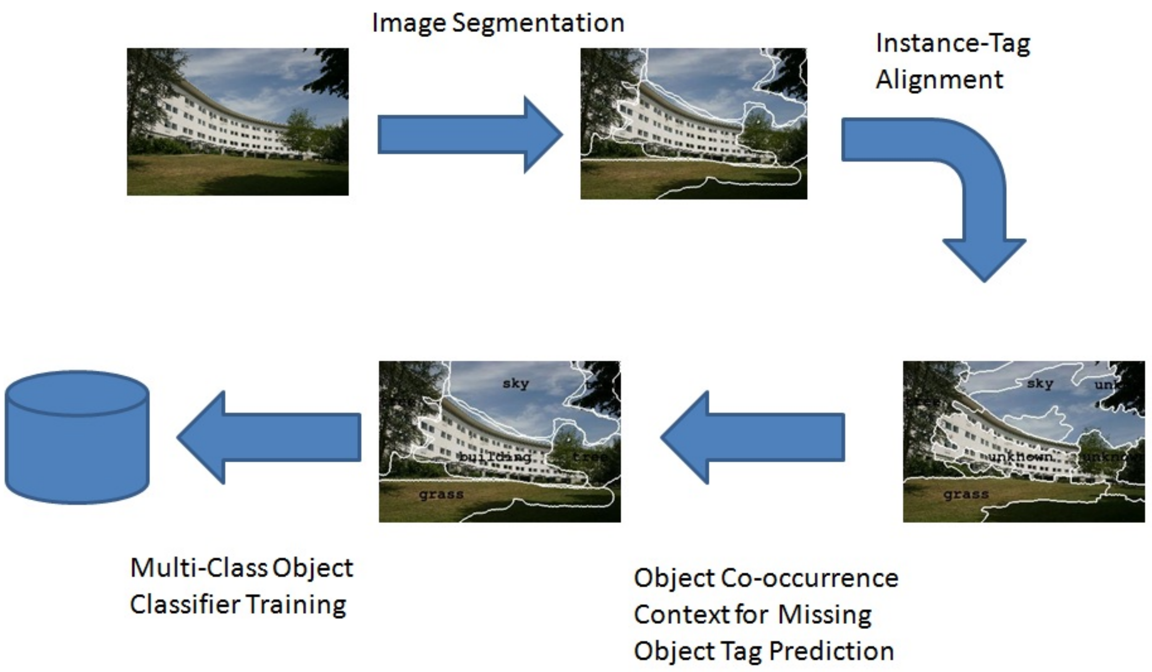


Image Segmentation
Instance-Tag Alignment
sky
building
tree
grass
unknown
Multi-Class Object Classifier Training
Object Co-occurrence Context for Missing Object Tag Prediction

Figure 10: The flowchart for our multiple instance learning algorithm with missing tag prediction.

of JSEG and NCut [22, 23, 24]. Each image region (i.e., image instance) is considered as a semantic atom and only one object tag can be assigned to such semantic atom, e.g., instance semantics is unique.

5.1 Bag of Instances

For a given object tag C_l, two bags of loosely-tagged images are labeled for object classifier training: positive bags (positive images) Ω_l *versus* negative bags (negative images) $\bar{\Omega}_l$. For the given object tag C_l, its positive bag (positive image) contains at least one positive instance, e.g., at least one image region is relevant with the given object class (object tag). On the other hand, its negative bag (negative image) may also contain positive instances because of the existence of missing object tags, e.g., some loosely-tagged images, which contain the object class C_l but the object tag C_l

is missed, may be treated as negative images for C_l. Thus one important assumption for traditional MIL scenario, e.g., all the instances in the negative bags are negative, is not true in our MIL scenario. Therefore, it is very attractive to develop new MIL algorithms that are able to address both the issues of loose object tags and missing object tags jointly. In this research, we have developed a novel MIL algorithm by assigning multiple object tags into the most relevant image instances, determining uncertain image instances whose object tags are not available on the tag list (missing object tags), and predicting the missing object tags for the uncertain image instances automatically.

The existence of missing object tags may fail most existing MIL algorithms because most of them share a common assumption: positive instances from positive bags should be close each other and be far away from the instances in negative bags, e.g., positive bags contain at least one positive instance and negative bags should strictly contain no positive instance. Such assumption is true and reasonable in traditional MIL scenarios because they assume that the bag-level labels are complete (missing object labels are not permitted).

Because of the appearances of missing object tags, it is worthy to re-consider the common assumption for multiple instance learning. When some object tags are missed, the common assumption for positive bags is still reliable, but the common assumption for negative bags is not always true because the negative bags may also contain positive instances e.g., positive images may be treated as the negative bags when the corresponding object tags are missed.

5.2 Distributed Affinity Propagation for Instance Clustering

Working with large scale image set, the size of image instances has been grown even enormous which is far out of the computational capability of classical clustering algorithms (that require to store huge size of pairwise similarity matrix for large amounts of image instances) on a single computer. Therefore it is necessary to take the advantage of distributed computing environments to enable clustering on large scale data set.

Affinity Propagation (AP) [131] is a clustering algorithm that takes the pairwise similarities as the input, and it works by sending two types of messages, *Responsibility* $R(r_i, r_j)$ and *Availability* $A(r_k, r_i)$, between the data items (loosely-tagged images). By iteratively computing Responsibilities and Availabilities, the algorithm assigns a most likely cluster center to each instance.

- Responsibility $R(r_i, r_j)$ updates:

$$t_1(r_i) = \arg\max_{r_j}\{A(r_j, r_i)+\kappa(r_i, r_j)\}; \quad t_2(r_i) = \arg\max_{r_j : r_j \neq t_1(r_i)}\{A(r_j, r_i)+\kappa(r_i, r_j)\} \tag{24}$$

$$R(r_i, r_k) = \kappa(r_i, r_k) - [A(t_l(r_i), r_i)) + \kappa(r_i, t_l(r_i))] \tag{25}$$

where $l = 2$ when $t_1(r_i) = r_k$, otherwise $l = 1$, $\kappa(r_i, r_j)$ is the visual similarity context between the web images r_i and r_j.

- Availability updates:

$$q(r_k) = \sum_{r_j} \max\{0, R(r_j, r_k)\}; \qquad A(r_k, r_k) = q(r_k) - R(r_k, r_k) \tag{26}$$

$$A(r_k, r_i) = \min\{0, q(r_k) - \max\{0, R(r_i, r_k)\}\} \tag{27}$$

Though the classic Affinity Propagation [131] can be seamlessly integrated with MapReduce framework, with the growing number of instances, the computation cost for pairwise responsibility/availability has even exceeded the computational capability of a typical Hadoop cluster. Hence, instead of parallelize the classic Affinity Propagation, an approximated Affinity Propagation algorithm has developed, taking advantage of the acceleration from the latest general purpose GPUs, which will be further introduced in Chapter 6. By applying such GPU-accelerated clustering algorithm, visually similar image instances are grouped into the instance clusters according to their visual similarity contexts $\kappa(\cdot, \cdot)$.

5.3 Automatic Instance-tag Alignment

For a given object tag C_l, two bags of loosely-tagged images are labeled for object classifier training: positive bags (positive images) Ω_l *versus* negative bags (negative images) $\bar{\Omega}_l$. For a given object tag, all the image instances in its positive bags and negative bags are grouped into a set of clusters by using our distributed clustering algorithm as described in Section 4. The image instances in the same cluster will share similar visual properties and their semantics can be described effectively by using the same object tag, e.g., consistency between the visual similarity and the semantic similarity. It is worth noting that the relevant image instances for the same object tags may have huge inner-class visual diversity and they may be partitioned into multiple instance clusters but their semantics can be described effectively by the same object tag. Because of the consistency between the visual similarity and the semantic similarity, such distributed instance clustering process can significantly reduce the uncertainty on the relatedness between the semantics of the image instances and the given object tag C_l.

For the given object tag C_l, an agreement value $P(C_l|r_i)$ is calculated for each instance r_i to characterize its relatedness with the given object tag C_l. To address the issues of loose object tags and missing object tags jointly, we make the following assumptions: (a) *Tag-Instance Consistency*: In our MIL scenario, if two image instances r_i and r_j (they may be positive or negative with the given object tag C_l) are similar on their visual properties, they should have similar agreement value to be assigned into the same object tag y_p, e.g., $P(y_p|r_i) = P(y_p|r_j)$, where $P(y_p|r_i)$ and $P(y_p|r_j)$ are the agreement values for r_i and r_j to be assigned with the object tag y_p. For the given object tag C_l, we focus on assigning the given object tag C_l into its positive instances with maximum agreement values $P(C_l|r_i)$, but we will completely ignore the object tags for C_l's negative instances. (b) *Instance-Cluster Tag Consistency*: The semantics for all the image instances in the same clusters can be interpreted effectively and efficiently by using a small set of available object tags, e.g., instance-cluster tag consistency (consistency between the visual similarity and the semantic similarity).

Based on these assumptions, all the image instances in the same cluster G_i contribute cumulatively on cluster agreement value $f(y_p|G_i)$ and it is defined as:

$$f(y_p|G_i) \propto \frac{1}{|G_i|} \sum_{r_i \in G_i} P(y_p|r_i) \tag{28}$$

where $|G_i|$ is the total number of image instances in the same instance cluster G_i. If

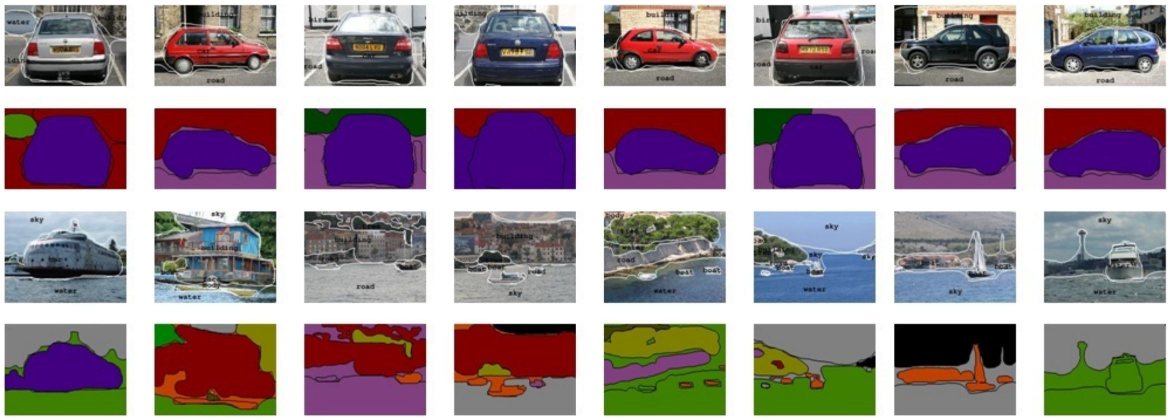


Figure 11: Automatic instance-tag alignment and uncertain instance identification

the instance cluster G_i is detected as the positive cluster for the given object class (object tag) C_l, the object tag for the instance cluster G_i is set as $y_p = C_l$.

For the given object tag C_l, the cumulative visual similarity context $\bar{\delta}(r_i, \Phi_l)$ between the image instance r_i and all its positive instances in its most positive clusters Φ_l is defined as:

$$\bar{\delta}(r_i, \Phi_l) = \frac{1}{|\Phi_l|} \sum_{v \in \Phi_l} \kappa(r_i, v) \tag{29}$$

where $\kappa(r_i, v)$ is the visual similarity context between two image instance r_i and v. The cumulative visual similarity context $\bar{\delta}(r_i, \Psi_l)$ between the image instance r_i and all the negative instances in C_l's most negative clusters Ψ_l is defined as:

$$\bar{\delta}(r_i, \Psi_l) = \frac{1}{|\Psi_l|} \sum_{u \in \Psi_l} \kappa(r_i, u) \tag{30}$$

where $\kappa(r_i, u)$ is the visual similarity context between two image instance r_i and u.

To assess the agreement between the image instance r_i and the given object tag C_l (i.e., whether r_i is positive with C_l), it is very important to make sure that r_i is far from C_l's most negative clusters Ψ_l and close to C_l's most positive clusters Φ_l. Thus the agreement value $P(C_l|r_i)$ between the image instance r_i and the given object tag

C_l is defined as:

$$P(C_l|r_i) \propto \left\{ \frac{1}{|\Phi_l|} \sum_{G_j \in \Phi_l} \bar{\delta}(r_i, \Phi_l) f(C_l|G_j) - \frac{1}{|\Psi_l|} \sum_{G_k \in \Psi_l} \bar{\delta}(r_i, \Psi_l) f(C_l|G_k) \right\} \tag{31}$$

where Φ_l and Ψ_l are the most positive clusters and the most negative clusters for the given object tag C_l, $|\Phi_l|$ and $|\Psi_l|$ are the total numbers of positive instances and negative instances in Φ_l and Ψ_l.

If the agreement value $P(C_l|r_i)$ for the image instance r_i is positive, $P(C_l|r_i) > 0$, r_i is treated as the positive instance for the given object tag C_l and C_l is used to tag the positive instance r_i. If the agreement value $P(C_l|r_i)$ for the image instance r_i is negative, $P(C_l|r_i) < 0$, it is treated as the negative instance for the given object tag C_l and we will completely ignore the object tag for the negative instance r_i. If the agreement value $P(C_l|r_i)$ for the image instance r_i is close to zero, $P(C_l|r_i) \approx 0$, it is treated as *uncertain image instance* and we will further leverage the object co-occurrence contexts to predict the missing object tag for the uncertain image instance r_i. Some experimental results on automatic instance-tag alignment and uncertain instance identification are given in Figure 13.

5.4 Missing Tag Prediction

After all the available object tags are assigned into their most relevant image instances automatically, we can obtain large amounts of labeled image instances (most positive instances) for all these available object tags and we can also count their pairwise co-occurrences in large-scale loosely-tagged images. An object co-occurrence network is constructed to represent such the object co-occurrence contexts explicitly

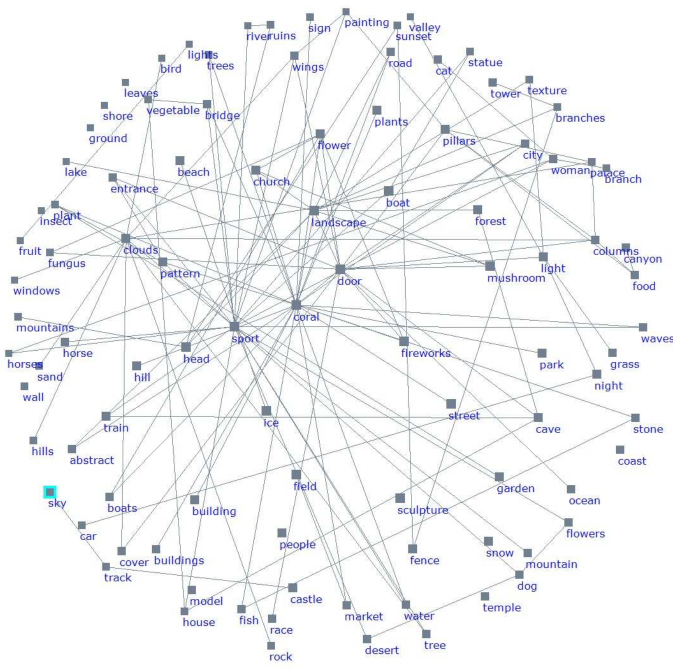


Figure 12: Our object co-occurrence network for missing tag prediction.

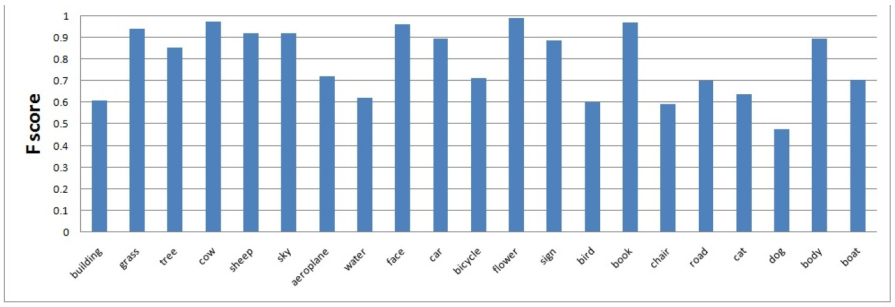


Figure 13: The F scores for instance identification algorithm on MSRC image set (using ground truth segmentation)

and provide a good environment to leverage such the object co-occurrence contexts for missing tag prediction.

For two given object tags C_i and C_j, their co-occurrence context $\omega(C_i, C_j)$ is defined as:

$$\omega(C_i, C_j) = -P(C_i, C_j) \log \frac{P(C_i, C_j)}{P(C_i) + P(C_j)} \tag{32}$$

where $P(C_i, C_j)$ is the co-occurrence probability for two object tags C_i and C_j in large-scale loosely-tagged images, and $P(C_i)$, $P(C_j)$ are the individual occurrence probabilities for C_i and C_j in large-scale loosely-tagged images.

The object tags (object classes), which have large values of the co-occurrence contexts, are connected to form an object co-occurrence network. The object co-occurrence network for our image collections is shown in Figure 12, where each object tag is linked with multiple co-occurrence object tags with larger values of $\omega(\cdot, \cdot)$. Our object co-occurrence network can provide a good environment for predicting the object classes (object tags) that are frequently used for tagging the same images.

For the given object class C_l, we completely ignore the object tags for its negative instances in the negative bags and the positive bags. Thus we will not predict the missing object tags for the uncertain image instances in C_l's positive bags because such uncertain image instances are irrelevant with the given object tag C_l (negative instances for C_l). As a result, the issue of predicting missing object tags for the uncertain image instances is simplified as an easier problem: assessing the relatedness between the uncertain image instances (in the negative bags) and the given object tag C_l, e.g., assessing whether the uncertain image instances in C_l's negative bags are

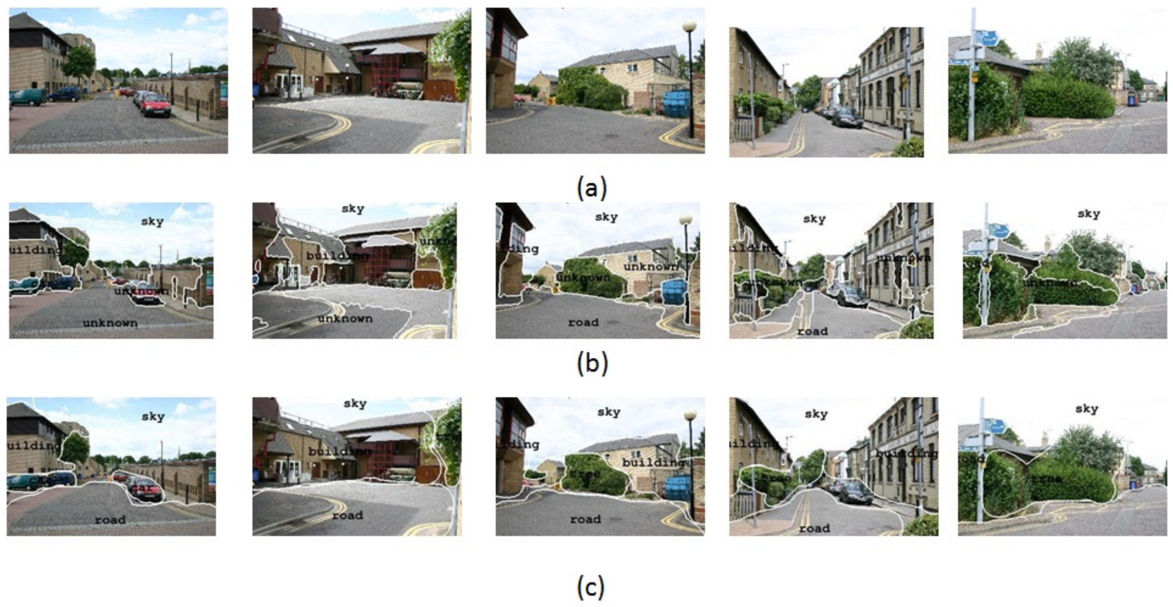


Figure 14: Missing tag prediction: (a) original images; (b) instance-tag alignment and uncertain instance identification; (c) missing tag prediction.

positive with the given object tag C_l.

For the given object tag C_l, there is a small set of object classes, Θ_l, which co-occur most frequently with C_l in large-scale loosely-tagged images and they have large values of $\omega(C_l, C_i)$, $C_i \in \Theta_l$. Uncertain image instances in the negative bags of C_l may belong to four categories: (a) negative instances for C_l and C_l's most frequently co-occurring object tags in Θ_l are also not used to tag C_l's negative images; (b) negative instances for C_l and some C_l's most frequently co-occurring object tags in Θ_l are used to tag C_l's negative images; (c) positive instances for C_l and some C_l's most frequently co-occurring object tags in Θ_l are used to tag C_l's negative images; (d) positive instances for C_l and all C_l's most frequently co-occurring object tags in Θ_l are not used to tag C_l's negative images.

For the first case, the uncertain image instance r_u is negative with C_l and it should be close to C_l's most negative clusters Ψ_l, thus its agreement value $\lambda_1(C_l, r_u)$ with the given object tag C_l is then refined as:

$$\lambda_1(C_l, r_u) = P(C_l|r_u) + \frac{1}{|\Psi_l|} \sum_{G_k \in \Psi_l} \bar{\delta}(r_u, \Psi_l) f(C_l|G_k) \tag{33}$$

where $P(C_l|r_u)$ is the agreement value for the uncertain image instance r_u to be assigned into the given object tag C_l by using Equation (31).

For the second case, the uncertain image instance r_u is negative with C_l and it should be close to all these most negative clusters for C_l and its most frequently co-occurring object tags in Θ_l, thus its agreement value $\lambda_2(C_l, r_u)$ with the given object tag C_l is then refined as:

$$\begin{aligned} \lambda_2(C_l, r_u) &= P(C_l|r_u) + \frac{1}{|\Psi_l|} \sum_{G_k \in \Psi_l} \bar{\delta}(r_u, \Psi_l) f(C_l|G_k) \\ &+ \frac{1}{|\Theta_l|} \sum_{C_i \in \Theta_l} \frac{\Delta_{iu}}{|\Psi_i|} \omega(C_l, C_i) \sum_{G_k \in \Psi_i} \bar{\delta}(r_u, \Psi_i) f(C_i|G_k) \end{aligned} \tag{34}$$

where Ψ_i is the most negative clusters for the object tag C_i, $C_i \in \Theta_l$, $|\Theta_l|$ is the total number of object classes in Θ_l, $\Delta_{iu} = 0$ if $C_i \in \Theta_l$ is used to tag the corresponding negative image that contains r_u, otherwise, $\Delta_{iu} = 1$.

For the third case, the uncertain image instance r_u is positive with C_l and it should be close to all these most positive clusters for C_l and its most frequently co-occurring object tags in Θ_l, thus its agreement value $\lambda_3(C_l, r_u)$ with the given object tag C_l is then refined as:

$$\begin{aligned} \lambda_3(C_l, r_u) &= P(C_l|r_u) + \frac{1}{|\Phi_l|} \sum_{G_k \in \Phi_l} \bar{\delta}(r_u, \Phi_l) f(C_l|G_k) \\ &+ \frac{1}{|\Theta_l|} \sum_{C_j \in \Theta_l} \frac{\Delta_{ju}}{|\Phi_j|} \omega(C_l, C_j) \sum_{G_k \in \Phi_j} \bar{\delta}(r_u, \Phi_j) f(C_j|G_k) \end{aligned} \tag{35}$$

where Φ_j is the most positive clusters for the object tag C_j, $C_j \in \Theta_l$, $\Delta_{ju} = 0$ if $C_j \in \Theta_l$ is used to tag the corresponding negative image that contains r_u, otherwise, $\Delta_{ju} = 1$.

For the fourth case, the uncertain image instance r_u is positive with C_l and it should be close to C_l's most positive clusters, thus its agreement value $\lambda_4(C_l, r_u)$ with the given object tag C_l is then refined as:

$$\lambda_4(C_l, r_u) = P(C_l|r_u) + \frac{1}{|\Phi_l|} \sum_{G_k \in \Phi_l} \bar{\delta}(r_u, \Phi_l) f(C_l|G_k) \tag{36}$$

The maximum agreement value $\lambda_{max}(C_l, r_u)$ between the uncertain image instance r_u and the given object tag C_l is then determined by:

$$\lambda_{max}(C_l, r_u) = \arg\max \{\lambda_i(C_l, r_u)|i = 1, 2, 3, 4\} \tag{37}$$

If $\lambda_{max}(C_l, r_u)$ is equal to $\lambda_3(C_l, r_u)$ or $\lambda_{max}(C_l, r_u)$ is equal to $\lambda_4(C_l, r_u)$, e.g., $\lambda_{max}(C_l, r_u) = \lambda_3(C_l, r_u)$ or $\lambda_{max}(C_l, r_u) = \lambda_4(C_l, r_u)$, the uncertain image instance r_u is treated as positive instance for C_l and the given object tag C_l is assigned with r_u. Otherwise, the uncertain image instance r_u is treated as negative instance for C_l. Some experimental results on missing tag prediction are given in Figure 14 .

CHAPTER 6: GPU-ACCELERATED AFFINITY PROPAGATION FOR LARGE SCALE DATA SETS

Given a set of data points and their pair-wise similarities, Affinity Propagation [131] is an unsupervised clustering algorithm, which finds a subset of exemplars by passing messages between data points. By performing affinity propagation, each data point is associated with an exemplar, which results in data points associated to same exemplars are grouped into clusters.

Compared to classic clustering algorithms such as K-means, K-medoids, graph cut, etc, affinity propagation takes two advantages on large scale data set: (1) Since the structure for large scale data set is usually unknown, the number of clusters usually need to be given as an input parameter for most classic clustering algorithms. In comparison, affinity propagation automatically determines the number of clustering from the similarity matrix and the preferences of data points which are provided by users; (2) In some applications such as text or image clustering, correspondences among sample points are provided by kernel matrices with unknown high dimensional coordinates for samples. As long as affinity propagation takes only the similarity matrix and the preferences of data points as the input, it can be directly applied to work with user-defined similarities without making any changes.

However, like all similarity-based methods, affinity propagation has encountered the problem of limited size of storage devices. For Instance, it is almost impossible to obtain a dense similarity matrix when the number of data points reaches as large as 1 million. Moreover, since the messages passed between the data points are also in the order of $O(n^2)$ (where n stands for the number of data points) without truncation, both the time complexity and the space complexity of original affinity propagation are beyond the computational capability of a single computer when it is applied on some large scale data sets. One way to solve the problem is to utilize modern parallel/distributed computing frameworks, such as Map-reduce, GPGPU, etc, to collaborate thousands or more computing units, while another way is to reduce the computational cost by sparsifying the similarity matrix.

In this chapter, a GPU-based solution of approximated affinity propagation will be introduced, by taking the advantage of both matrix sparsification and parallel computing. Firstly, we have developed a top-down hierarchical affinity propagation algorithm, which partitions the large-scale clustering problem into thousands of small problems which normal processors can handle. Second, thanks to CUDA technology, we spread those small problems over hundreds of GPU cores. By achieving parallel computing over GPUs, it has dramatically improved the efficiency of affinity propagation and significantly reduced the running time. Combined approximated hierarchical algorithm with CUDA technology, it is capable to group large-scale high dimensional data points with a small cluster of computer or even with a single computer equipped with a latest CUDA graphic card.

Table 2: Notations in Chapter 6

n	number of points
m	number of edges in the graph
$s(i,j)$	similarity between point x_i and point x_j
$r(i,k)$	responsibility transferred from x_i to x_k
$a(i,k)$	availability from exemplar x_k to x_i
$e(i)$	if data point i is nominated as an exemplar
$c(k)$	the preference of data point x_k to be an exemplar

6.1 Classic Affinity Propagation Algorithm

Affinity propagation algorithm transfers two types of messages between data points: (a) Responsibility $r(i,k)$, which is passed from data point i to candidate exemplar point k, indicates the preference of data point i to select the point k as its exemplar; (b) Availability $a(i,k)$, which is a feedback from the exemplar point k to the data point i, shows what degree does the point k fit to be a cluster center for the data point i. Responsibilities are calculated as Equation 38,

$$r(i,k) = s(i,k) - \max_{k' \neq k}\{a(i,k') + s(i,k')\} \tag{38}$$

and Equation 39 and 40 shows the update of Availabilities, for $i \neq k$,

$$a(i,k) = \min\left\{0, c(k) + r(k,k) + \sum_{i' \notin \{i,k\}} \max\{0, s(i',k)\}\right\}, \tag{39}$$

and self-availabilities indicate preferences,

$$a(k,k) = c(k) + \sum_{i' \neq k} \max\{0, r(i',k)\}. \tag{40}$$

.

The algorithm iteratively updates responsibilities and availabilities until conver-

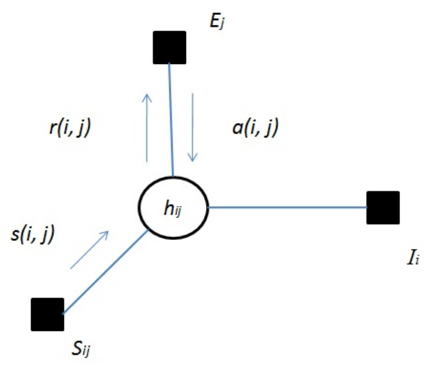


Figure 15: Single point in Binary Factor Graph

gence. The exemplar associated to the data point x_i is determined by Equation 41:

$$f(i) = \arg\max_{k}\{r(i,k) + a(i,k)\} \tag{41}$$

Theoretically, affinity propagation is derived from max-product algorithm (max-sum in log domain) in a binary factor graph as shown in Fig. 15 [93], where $\{h_{ij}\}$ are hidden binary variables which indicate whether the point x_i has chosen the point x_j as its exemplar. The goal is to maximize the object function in Equation 42, which models a joint energy combined with the similarities (weights on edges) and preferences (weights on nodes).

$$\sum_{ij} S_{ij}(h_{ij}) + \sum_{j} C_j(e_j) \tag{42}$$

, subjects to following constraints:

$$C_j(e_j) = c(j)e(j), S_{ij}(h_{ij}) = s(i,j)h_{ij} \tag{43}$$

$$\sum_{j} h_{ij} = 1 \tag{44}$$

$$e_j = h_{jj} = \max_{i} h_{ij} \tag{45}$$

Equation 44 constrains that each point only be associated with one exemplar and Equation 45 ensures the basic fact that if the point x_j is selected as the exemplar of some other points, it must be the exemplar of itself. As discussed in [131, 93, 94], the optimization problem in Equation 42 has been proved to be an NP-hard problem and the passing message algorithm in [131] is an approximated MAP solution, which has been working well in most cases.

The classic affinity propagation algorithm takes full similarity matrix as the input and passes the messages between the points. Therefore without any optimization to reduce the cost, the time complexity is $O(tn^2)$, where t is the number of iterations. It also requires $O(n^2)$ space to store the pair-wise similarity matrix, availabilities and responsibilities. Apparently $O(n^2)$ space complexity is not acceptable for large scale data points. However, if affinity propagation is performed on a pre-constructed sparse graph, since the messages are only passed though the edges, the running time complexity will be reduced to $O(tm)$ and only $O(m)$ space is needed.

6.2 Previous Optimization over Affinity Propagation

There are some pioneering works which attempted to reduce the cost of affinity propagation by approximations. Y. Jia *et al.*. proposed a two-step method, known as FSAP [95]. Instead of passing the message on the whole fully-connected graph, FSAP does a pre-processing step which prunes the graph by performing a k-nearest-neighbor search on each data points. Later, another graph pruning method[96] was developed by Y. Fujiwara and T. Kitahara. Differed from FSAP, Fujiwara's method prunes the graph by estimating upper and lower bound of availability and responsibility on each pair of data points, which tends to yield the exact same clustering results as which the classic algorithm [131] produces. However, both these two methods rely on the calculation of the complete similarity matrix, which limits their performance on large scale data. FSAP requires the pair-wise similarities to perform k-NN search unless the structure of data points is pre-computed (like with a k-d tree); Fujiwara's method goes through all the pairs of data points to estimate lower and upper bounds of messages, where also complete similarity matrix is also computed. Alternatively, besides those approximated solution, W. Wang *et al.*have moved affinity propagation algorithm under map-reduce framework, which takes the advantage of cloud computing instead of running with large scale data on a single computer [97].

In the recent decades, thanks to the development of general purpose computing on GPUs, a remarkable number of machine learning methods has been refined to be compatible for GPU acceleration, e.g., GPU-accelerated K-means was developed by J. Hart and J. Hall in [98]. In [99], V. Garcia *et al.*speeded up K-nearest neighbour search by using Nvidia's CUDA technique [100]. More generally, M. Hussein and W. Abd-Almageed have developed a band approximation which uses GPU to accelerate all kernel methods on sparse Gram matrices [101]. As an application, the performance of affinity propagation algorithm can also be improved by [101] in case the similarity matrix is sparse.

We also would like to mention hierarchical affinity propagation [94] , which is developed by I. Givoni *et al.*. Although neither is motivated by efficiency issue nor it improved performance in practice, our proposed method actually comes from the idea of hierarchical affinity propagation.

6.3 Approximated Affinity Propagation and CUDA Implementation

Our proposed algorithm works on a bottom-up framework as shown in Fig. 16. Differed from directly bottom-up aggregation in hierarchical affinity propagation [94], which is shown in Fig. 17, it is not possible to propagate the responsibilities and availabilities over huge amounts of data points on lower layers. Instead, we conduct a top-down work set selection scheme to ensure the number of points on each layer is small enough for a single processor to handle. At the certain layer l, if the number of data points in current work set exceeds a pre-defined limit T_n (which is related to the computation power of a processor unit and its storage size), the current work set will be partitioned into p (which is determined by T_n) overlapped sub-worksets and the algorithm proceeds to layer $l+1$ to find exemplars for each sub-workset separately. Only the exemplars from all p sub-worksets take part in the clustering in layer l, which means to be an exemplar in layer l, that point must be an exemplar in layer $l+1$. The algorithm is described as in Algorithm 5.

6.3.1 Workset Selection

As we stated in Algorithm 5, work set partition (or sub-workset selection) is performed on each layer except the bottom one. The ideal partition method meets following conditions: (a) the partition method is simple enough that it does not take a lot of resources; (b) the structure of data points is approximately preserved that the close points tend to be close after partition and no points disappear from all sub-worksets. (c) it computes as less pair-wise similarities as possible, because $O(n^2)$ time/space complexity is too expensive in most large scale cases. (d) The lost mes-

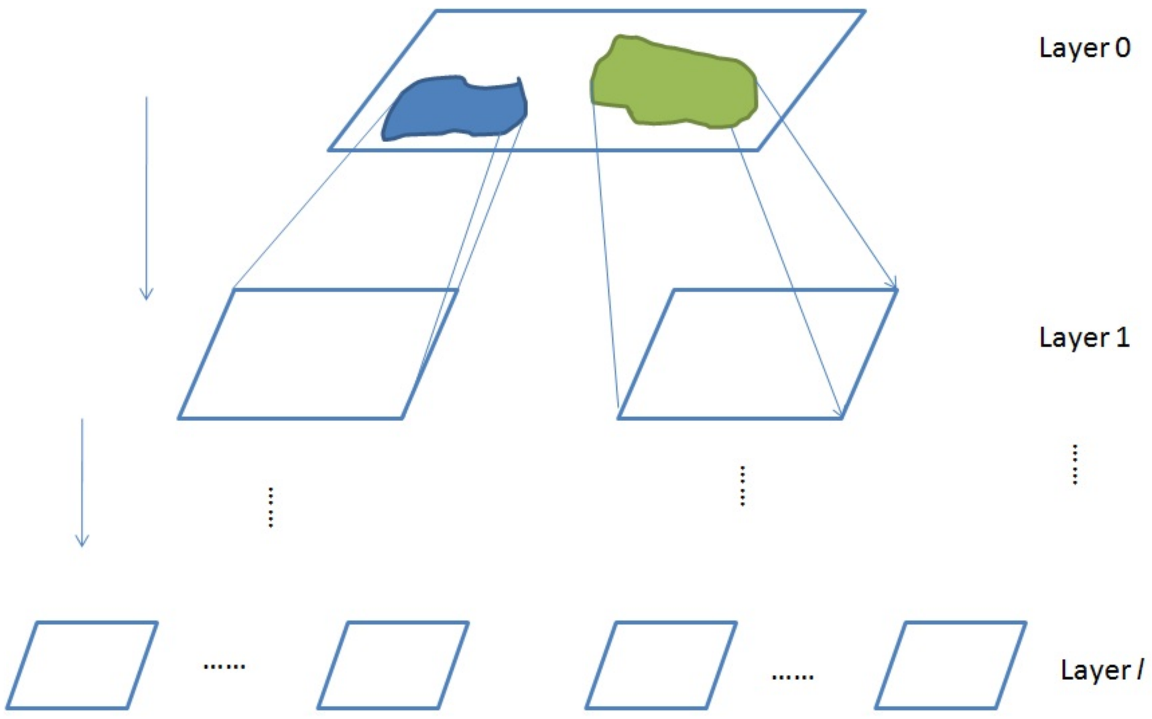


Figure 16: Proposed Affinity Propagation

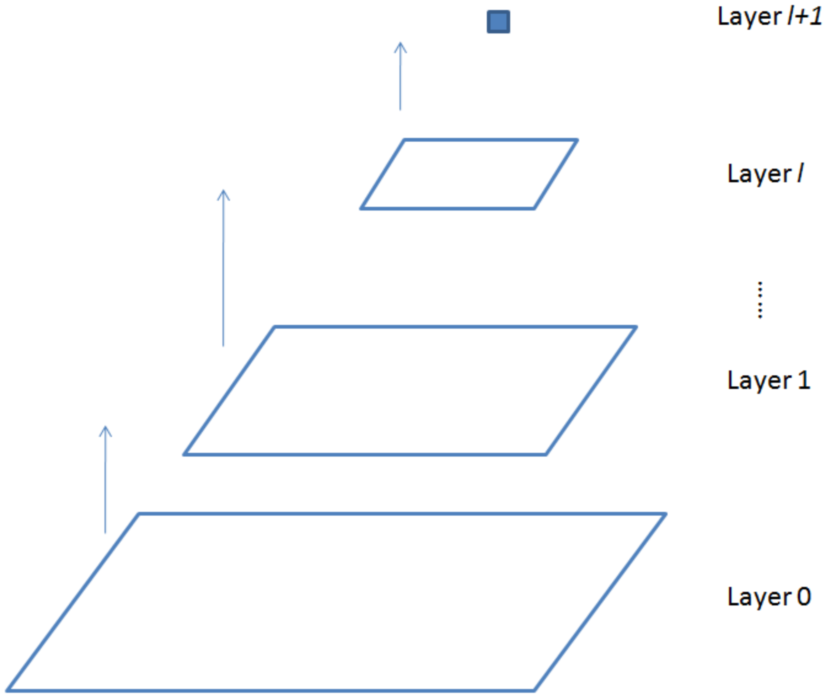


Figure 17: Hierarchical Affinity Propagation

```
APCluster ($l$, $n$, $\{x_i\}$)
if $n \leq T_n$ then
  Calculate the similarity matrix $S$ over $\{x_i\}$
  Perform normal affinity propagation based on $S$
  return $\{f(x_i)\}$
else
  Partition $\{x_i\}$ into $p$ sub-workset: $\{(n_k, \{x_i^k\})\}$
  for each sub-workset $\{(n_k, \{x_i^k\})\}$ do
    APCluster($l + 1$, $n_k$, $\{x_i^k\}$)
  end for
  $Y_{l+1} = \{y_i^k | y_i^k \text{is an exemplar of sub-workset } k\}$
  Calculate similarity matrix $S$ over $Y_{l+1}$
  Perform normal affinity propagation based on $S$ and get a exemplar set $Y_l$
  Calculate similarity matrix $S'$ between $\{x_i\}$ and $Y_l$
  Iteratively update availabilities and responsibilities between $\{x_i\}$ and $Y_l$
  Update $f(x_i)$ by Eq. 41, where $f(x_i) \in Y_l$
  return $\{f(x_i)\}$
end if
```

Algorithm 5: the Proposed Approximated Affinity Propagation Algorithm

sages due to partition are less important, e.g., the messages, which are transmitted between data points that far away from each other, tend to be unnecessary.

It seems to be a way to partition work set by the results from k-nearest-neighbor search. However, in case that the similarities can be arbitrary and no additional graph information is provided, it actually takes $O(n^2)$ time to find k-nearest neighbors for all the data points. We have conducted a much simpler but more efficient way for the partition, as shown in Fig. 18. It starts from a random point x_i. With its k-nearest neighbours, these $k + 1$ points form a sub-workset. To ensure every data point has been picked by some sub-workset, a frequency list $\{d(i)\}$ is maintained, where $d(i)$ records how many times point x_i has been chosen to form sub-workset in the current work set. We randomly pick a point with least $d(i)$ as the start point to generate a sub-workset, until any data point belongs to at least one sub-workset and the number

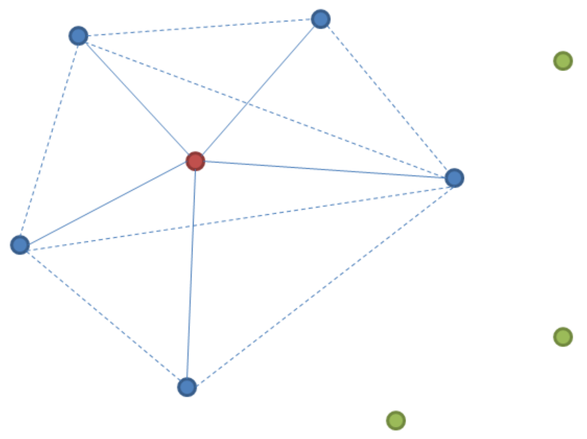


Figure 18: Starting point x_i and its k-nearest neighbours form a sub-workset.

of sub-worksets reaches p. The total complexity of this selection method is $O(pn)$.

6.3.2 Message between Layers

Since the exemplars are the representatives of low-layer data points in a upper layer, it is reasonable and necessary to make sure that candidate exemplars in upper layers are assigned with different preferences. According to [94], inter-layer messages have been introduced for addressing such differences, as shown in Fig. 19. The message $\tau^{l-1}(k)$ from lower layer to high layer, affected responsibility of exemplar k at layer $l-1$ and the message $\phi^{l+1}(k)$ is a preference feedback of exemplar k from layer l to layer $l+1$, which are calculated as Eq. 46 and Eq. 47:

$$\tau^{l-1}(k) = c^l(k) + r^l(k,k) + \sum_{k' \neq k} \max\{0, r^l(k',k)\} \tag{46}$$

$$\phi^{l+1}(k) = \max_{k'} \left\{a^l(k,k') + s(k,k')\right\} \tag{47}$$

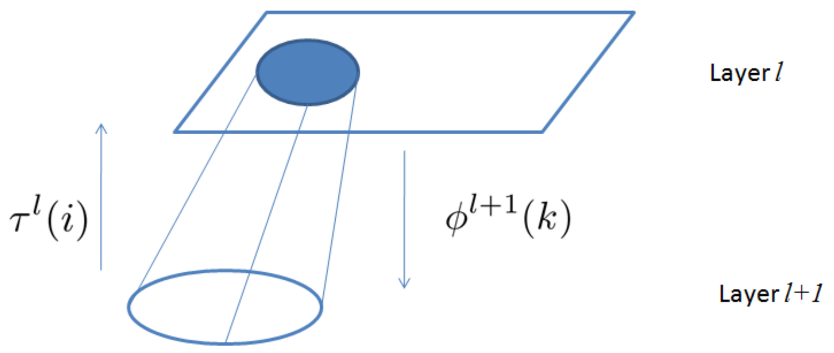


Figure 19: Messages passed between adjacent layers.

Correspondingly, responsibility updating formula in Eq. 38 is changed to Eq. 48:

$$r^l(i,k) = s^l(i,k) + \min\left\{\tau_i^l, -\max_{k'\neq k}(a^l(k,k') + s^l(k,k'))\right\} \tag{48}$$

and preference $c^l(k)$ is updated by $\hat{c}^l(k) = c^l(k) + \phi^l(k)$. There is a detail proof of layer-wise message transferring in [94], which is omit in this paper.

6.3.3 GPU-accelerated Implementation

Nowadays, Graphics Processing Units (GPUs) are used to render fancy graphics on monitors. Thanks to the recent development on general purpose computing on GPU (GPGPU), not only a remarkable number of gaming graphic cards are capable to perform scientific computing, but also some graphic card (such as Tesla series) are specifically designed for super computing. As a main stream model of GPGPU, CUDA and its tools provide an easy way for researchers to make their programs adaptive to GPUs.

For scientific programming, CUDA provides a high-level programming interface

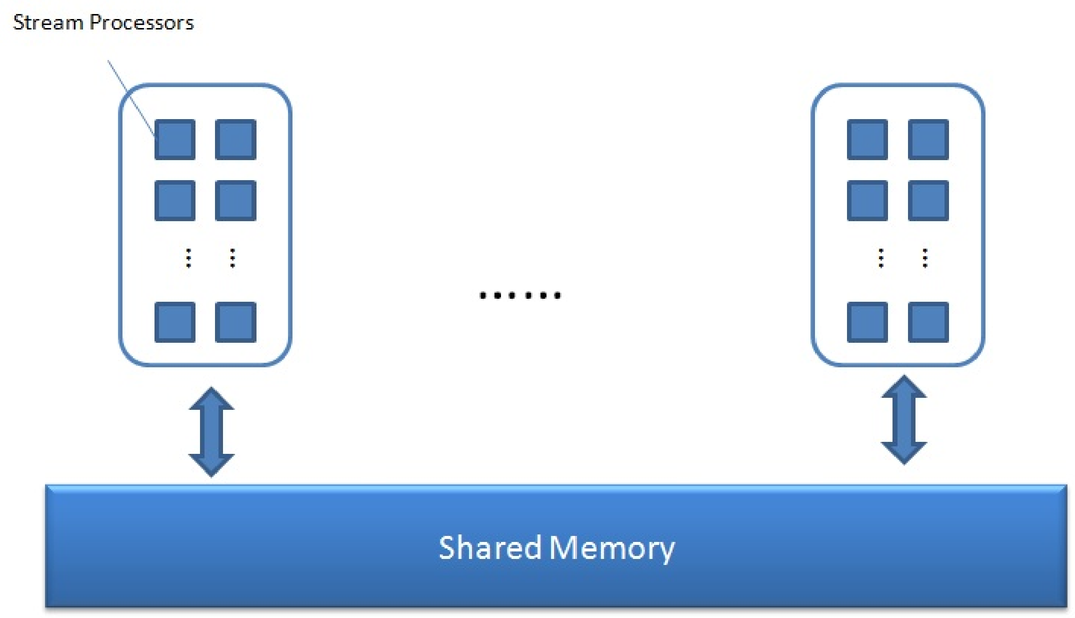


Figure 20: CUDA Architecture

that programmers are able to take advantage of general purpose computing by GPUs easily. Users are required to specify a series of code segments which are called "kernels". Kernels are independent to each other that each kernel will be allocated with a streaming processor and its corresponding storage automatically by CUDA, as shown in Fig. 20. In our implementation, each kernel handles the affinity propagation of a single work set. It is important to mention that the storage on a graphic card is limited, so CUDA is not a good candidate solution to directly handle those problems which can not fit into the size of main memory, e.g., the huge similarity matrix of affinity propagation. Therefore, it is necessary to prune the matrix computation before trying to speed up with CUDA, which we have done in previous sections.

In our framework, as shown in Fig. 21, a dispatcher, which is running on CPU, controls if a work set is ready to be running or is good for output. When a new work

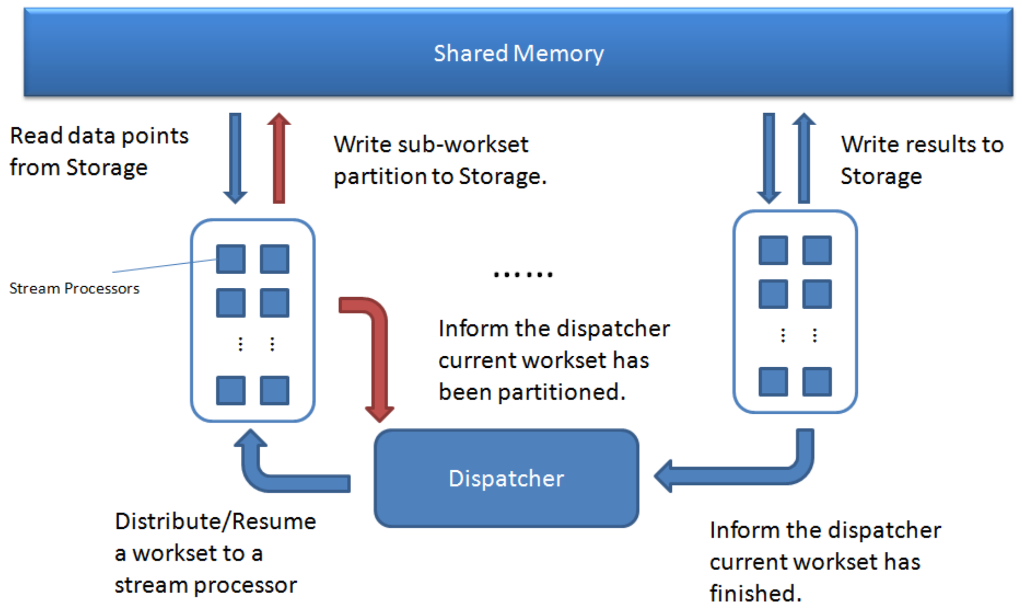

Shared Memory
Read data points from Storage
Write sub-workset partition to Storage.
Write results to Storage
Stream Processors
......
Inform the dispatcher current workset has been partitioned.
Dispatcher
Distribute/Resume a workset to a stream processor
Inform the dispatcher current workset has finished.

Figure 21: CUDA implementation of proposed affinity propagation algorithm

set is generated, dispatcher first allocate the work set a spare streaming processor, if it is available. As in Algorithm 5, if the number of points in the work set exceeds the limit, the streaming processor generates a series of new sub-worksets and return them to the dispatcher. In that case, the dispatcher suspends the original work set and spreads new sub-worksets to streaming processors. When all sub-worksets have finished and returned their outputs, the dispatcher disturbs the original work set again, as well as those outputs collected from sub-worksets. This process continues until the last work set, which is the topmost layer with all data points, returns its results.

6.3.4 Efficiency

For each work set which contains n_k data points, if $n_k < T_n$, normal affinity propagation are simply applied on the work set, which requires at most $O(T_n^2)$ time and

$O(T_n^2)$ space. For those work sets with $n_k > T_n$, we partition the work set into small sub-worksets by doing k-nearest neighbor search p times. Therefore, the time cost of partition is $O(pn_k)$ and the space used is $O(n_k)$. In practice, since each steaming process is able to handle brute force AP cluster over thousands of data points, a two-layer hierarchy is enough to cluster up to 10^6 data points. Therefore, as we set $n_k = \sqrt{n}$ and $p = O(\sqrt{n})$, the total time complexity is $O(n^{1.5})$ and the space needed is $O(n)$, which is theoretically lower than which in [95] and [96].

With the help of CUDA framework, the work sets are spread over hundreds of processor units (GPU cores). Ideally, the total running time will be reduced to $O(n^{1.5}/n_c)$, where n_c stands for the number of processor units. However, since data transferring between stream-processors and shared memory is time-consuming, the real running time is approximated to $O(n^{1.5}/n_c + n_c \cdot n)$.

6.4 Experiments

We implemented our proposed approximated affinity propagation algorithm both on CUDA framework and normal CPU. The experiment platform is a laptop with a Nvidia Geforce GT 555M display card, which contains 144 GPU cores and 3GB display memory, installed with Intel Core i7-2630QM 2.0Hz CPU (8 cores), 8GB main memory, running 64-bit Windows 7.

In order to demonstrate both robustness and efficiency of our proposed algorithm, our experiments consists of two parts: in the first part, clustering accuracy is focused. we have tested both the proposed method and reference methods on some relatively small data sets, which normal CPU-based methods are able to handle, to compare our results with both ground truths and outputs from reference methods; in the second part, we have moved onto large scale data sets. Since the sizes of such data sets reach or exceed the limit of traditional CPU-based methods, we mainly focused on demonstrating speed boost of our proposed method. Besides the original affinity propagation algorithm, two reference methods, FSAP [95] and Fujiwara's method [96], are involved in our evaluation to compare with.

6.4.1 Clustering Accuracy

Because Fujiwara's method always produces the same results as which the original method does, it is meaningless to take Fujiwara's method into accuracy evaluation. Thus, in this section, it's obviously a comparison between two approximations, our proposed method and FSAP. We begin to conduct our experiment by constructing and testing on a synthetic 2D data set. The data set contains 20K 2-D points, generated by the mixture of 15 random 2-D Gaussian functions [102]. The similarity between two data points is defined as the negative of their Euclidean distance. To quantitatively evaluate the performance, we run the original affinity propagation algorithm [131] on the synthetic data set to obtain benchmark results which further compared with the results output by other methods with the same preference setting. The degree of preserving original outputs is measured by exactness as Eq. 49 .

$$Exactness = \frac{\sum_{ij}(I(f(i), f(j)) \cdot I(f_0(i), f_0(j)) + 1)}{2 \cdot n^2} \tag{49}$$

where $f_0(i)$ is the ground truth outputs by the original affinity propagation, $f(i)$ is the output from current candidate method, $I(\cdot, \cdot)$ is an identical function which is

Table 3: Exactness comparison on the synthetic 2D dataset with benchmark output contains 150 exemplars.

	Number of Exemplars	Exactness
Proposed Algorithm	147	0.844
FSAP	145	0.885

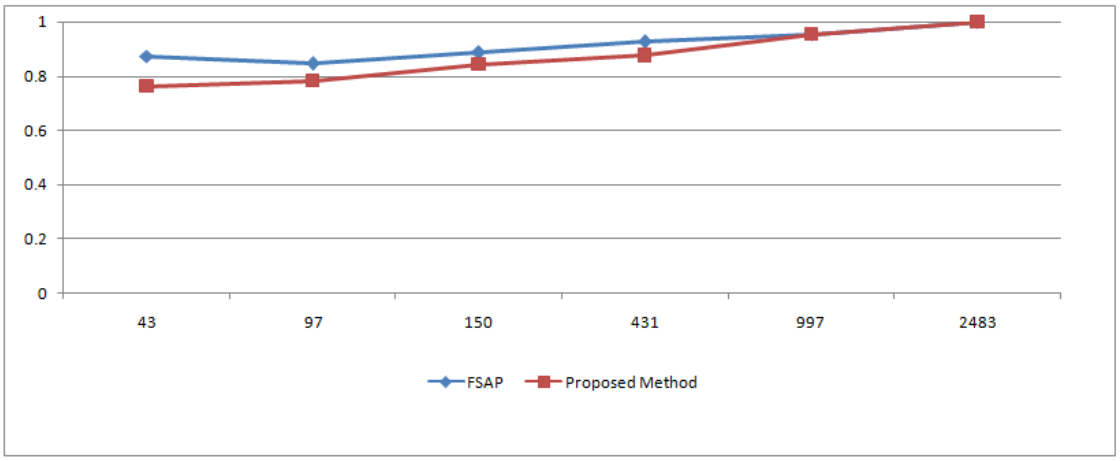


Figure 22: Exactness vs. number of Exemplars. x-axis indicates the number of exemplars found by the original algorithm. y-axis shows the exactness calculated by Eq.49.

defined by:

$$I(f(i), f(j)) = \begin{cases} 1, & f(i) = f(j) \\ 0, & f(i) \neq f(j) \end{cases} \tag{50}$$

Table 3 shows the exactness of our proposed on the synthetic 2D data set, with the number of exemplars is around 150. Although the number of exemplars found by our proposed algorithm is closer to the benchmark than FSAP's results, the proposed method seems not tend to preserve the same output by the original algorithm. This is principally because the proposed method only calculates part of the similarities, rather than FSAP goes through the complete similarity matrix. We have also tuned preferences of input data points which changed the number of exemplars, to evaluate the exactness with different number of exemplars, in Fig. 22. Besides exactness, error within clusters is also a reasonable and convincing measurement. Since affinity propagation is a similarity-based algorithm, alternatively, we have calculated average in-cluster similarity, as Eq. eqn:ap:similarity. Higher average in-cluster similarity tend to indicate better clustering.

$$\bar{S} = \frac{\sum_{f(i)=f(j)} s(i,j)}{\sum I(f(i) = f(j))} \tag{51}$$

As shown in Fig. 23, the proposed algorithm has not only achieved better approximation than FSAP, it also produces as good results as original algorithm does under the measurement of average in-cluster similarity.

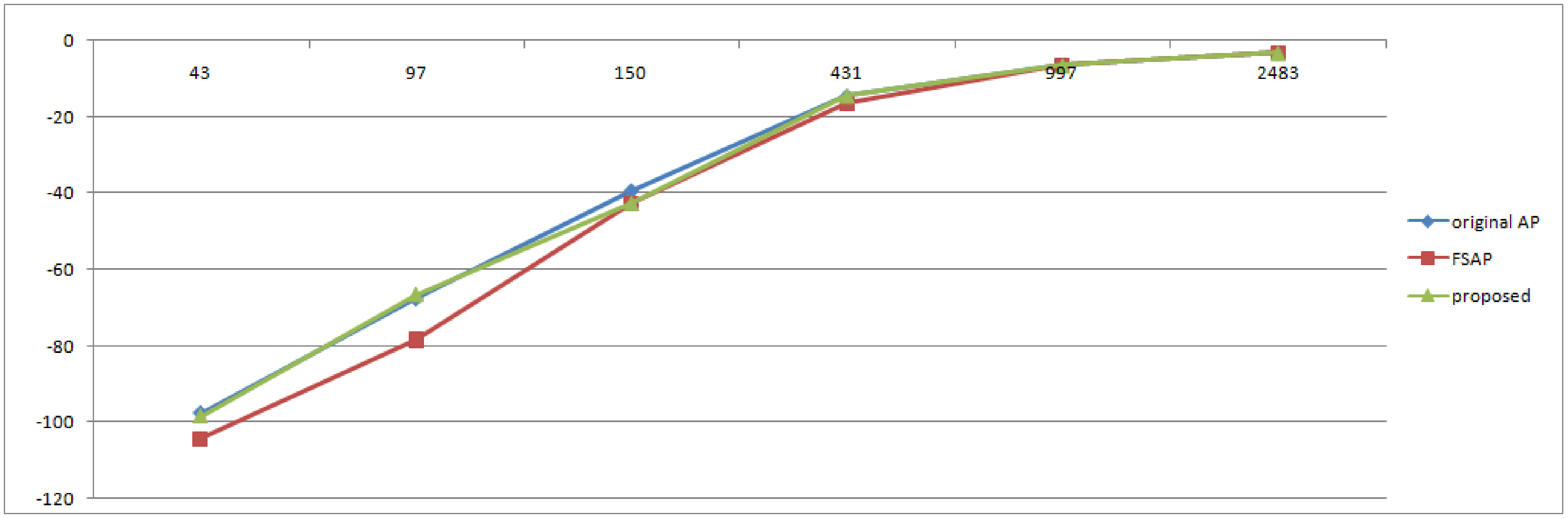


Figure 23: Average in-cluster similarity v.s number of exemplars: x-axis represents the number of exemplars found by the original algorithm; y-axis stands for the average in-cluster similarity by Eq. 51.

6.5 Clustering Efficiency

As we mentioned, our proposed work is an approximation which significantly reduces the running time and the space requirement of affinity propagation. To support the theoretically analysis in Section 6.3.4, we have examined our algorithm, as well as some preference methods, on several data sets.

To begin with, running time evaluation is performed on the synthetic 2D data set. Fig. 24 shows the running time over different sizes of data sets, the GPU-accelerated algorithm significantly have taken significantly less running time than other approximation of affinity propagation.

To demonstrate our method can be used to perform large scale data clustering, we also evaluated our method on a selected data set from ImageNet [107] which consists of 1,044,396 images. Fig. 25 shows our algorithm is capable to perform AP clustering on large scale data set. Sample image clustering results on ImageNet are provided in supplementary materials.

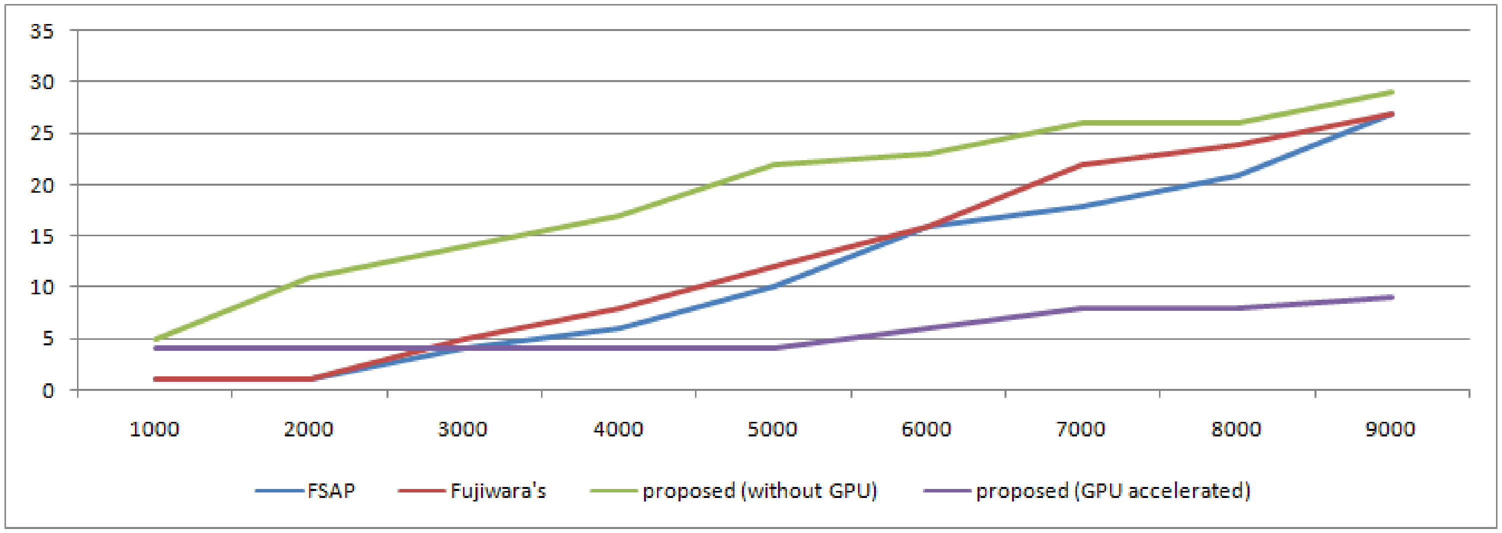


Figure 24: Running time on synthetic 2D dataset: x-axis represents number of data points; y-axis indicates time in seconds

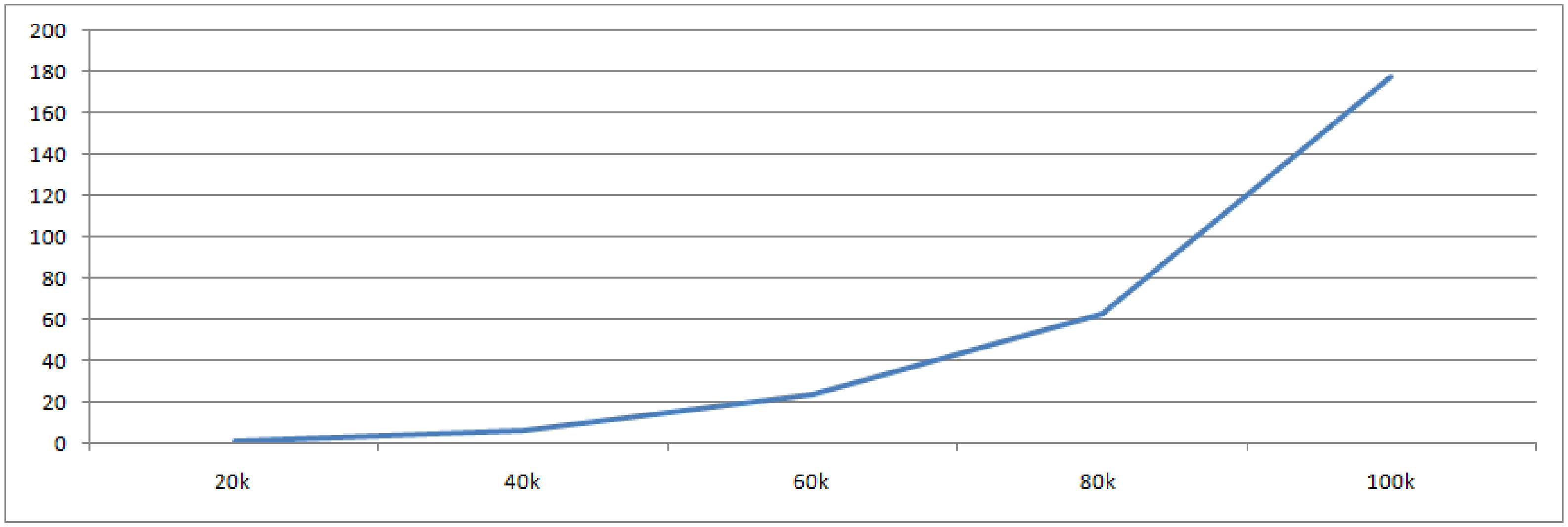


Figure 25: Running time on ImageNet dataset: x-axis represents number of data points; y-axis indicates time in minutes

Table 4: Conclusive comparison between proposed algorithm and reference methods

	Running Time	Space Complexity	Running Time on sparse graph	Preserve Exactness	GPU-accelerated
original AP	$O(n^2)$	$O(n^2)$	$O(m)$	N/A	no
FSAP	$O(n^2)$	$O(m)$	$O(m)$	no	no
Fujiwara's	$O(n^2)$	$O(m)$	$O(n^2)$	yes	no
Proposed	$O(n^{1.5})$	$O(n)$	$O(n^{1.5})$	no	yes

6.6 Conclusion

In this chapter, we have proposed a novel affinity propagation approximation, which (a) found an approximation which make affinity propagation capable to large scale data clustering; (b) significantly reduced the running time of affinity propagation. It has combined the idea of graph pruning and hierarchical affinity propagation, also taken the advantage of general purpose computing. To summarize, we provide a conclusive comparison between our proposed work and other methods, as shown in Table. 4

CHAPTER 7: NETWORK-ORIENTED MULTI-TASK LEARNING FOR OBJECT CLASSIFIER TRAINING

To address the inter-relatedness among object categories in classifier training, generally two fundamental problems are conducted: (1) how to characterize and measure the inter-relatedness among object categories; (2) with object inter-relatedness available, how to utilize those information to achieve better classification performance.

Since the relationships among object categories are complicated, we constructed a object correlation network which simplified object inter-relatedness by pair-wise object correlations without losing the inter-class structure. With the pair-wise object correlations and the object structure, a multi-task learning framework is designed to jointly train the classifiers for the inter-correlated object categories.

7.1 Object Correlation Network

In order to determine the inter-related learning tasks and support discriminative learning, a visual concept network is constructed in this paper for organizing a large number of object classes and image concepts according to their inter-concept visual correlations in the visual feature space. The visual concept network consists of two key components: (a) *categories* (i.e., object classes and image concepts); and (b) *inter-concept cumulative visual similarity contexts* between their relevant image instances.

Our algorithm for visual concept network construction consists of three key steps: (a) For each category, one-class SVM algorithm [130] is used to determine a small set of most representative image instances for characterizing its principal visual properties effectively and sufficiently; (b) For two given categories, their pairwise inter-concept visual correlation is computed by simulating the visual similarity contexts among their relevant image instances in the visual feature space; (c) The categories (i.e., object classes and image concepts), which have large values of the inter-concept visual correlations, are linked together to form the visual concept network.

7.1.1 Extracting Most Representative Images

The first step for visual concept network construction is to determine a set of the most popular real-world object classes and image concepts. In this paper, more than $1,000$ most popular real-world object classes and image concepts are extracted from ImageNet image set [107], and their most representative images are extracted from ImageNet images for characterizing their principal visual properties effectively.

To reduce the computational cost as well as the influence from noise images, a one-class SVM algorithm is used to automatically determine the most representative images for each category. One-class SVM algorithm [130] is a unsupervised method, which can find the smallest possible hyper-sphere to enclose all the most representative images with an error-tolerated boundary. The one-class SVM algorithm makes a trade off between the size of hyper-sphere and the error penalty by solving the optimization problem in Equation (52):

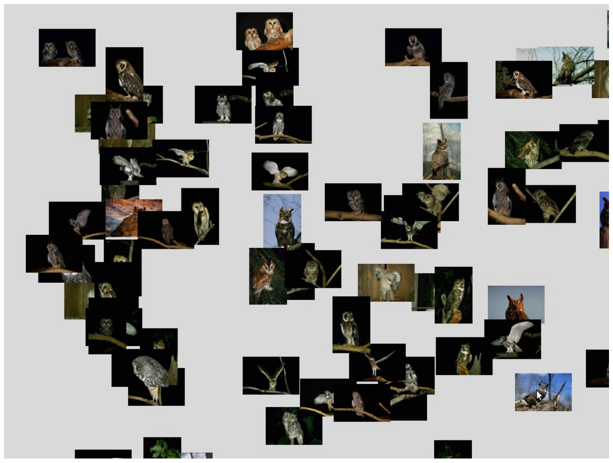
$$\begin{aligned} \min \quad & R^2 + C\sum_i \xi_i \\ \text{s.t.} \quad & \forall x_i : ||x_i - z||^2 \leq R^2 + \xi_i, \xi_i \geq 0 \end{aligned} \tag{52}$$

where R and z indicate the radius and the center of the hyper-sphere, xi_i is the error penalty and C is the regulation parameter. The cores of the given category (i.e., the most representative images for the given object class or image concept) are formed by those image instances x_i which are tightly enclosed by the hyper-sphere and satisfy $xi_i = 0$ correspondingly.

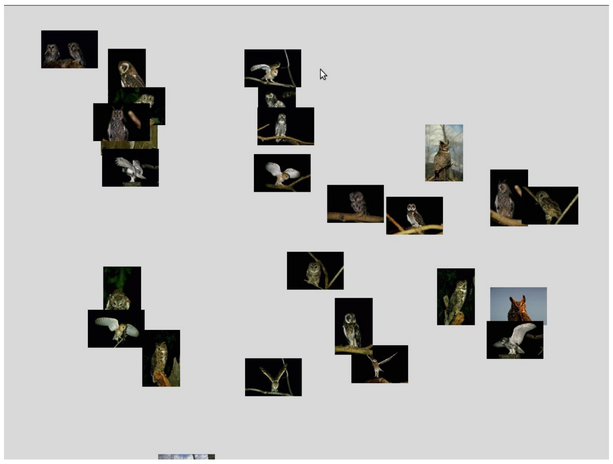
One example is shown in Fig. (26) to illustrate the results for most representative image extraction for the category "owl", one can observe that our algorithm can extract the most representative images effectively from large amounts of ImageNet images by filtering out less representative images. For a given category, there are two purposes to extract such the cores (i.e., the most representative images) instead of directly using entire set of ImageNet images to characterize its principal visual properties: (a) The most representative images (i.e., cores) can represent the principal visual properties for the given category effectively and sufficiently and the outliers (i.e., less representative images whose visual properties are significantly different from the principal visual properties for the given category) have been eliminated automatically; (b) For a given category, the number of most representative images remaining in the core is much less than the original number of ImageNet images, thus the computational cost for calculating the inter-concept visual correlations can be reduced dramatically.

7.1.2 Inter-Concept Visual Correlation

An object correlation network is constructed to characterize the inter-object visual correlations explicitly and provide a good environment to determine the inter-



(a)



(b)

Figure 26: Core-image Extraction: (a) the original images for the category "owl"; (b) the extracted core images for the category "owl".

related learning tasks directly in the feature space. Our object correlation network consists of two key components: *object classes* and their *inter-object correlations*. For two given categories (i.e., object classes and image concepts) C_u and C_v, their inter-concept visual correlation $\gamma(C_u, C_v)$ is characterized by simulating the pairwise similarity contexts among their most representative images. The average-max aggregation is enforced to compute the visual similarity context between two image sets for two given categories C_u and C_v:

$$\begin{aligned} S_u(C_u, C_v) &= \frac{1}{N_u} \sum_{i=1}^{N_u} \max_{j=1 \ldots N_v} K(u_i, v_j) \\ S_v(C_u, C_v) &= \frac{1}{N_v} \sum_{j=1}^{N_v} \max_{i=1 \ldots N_u} K(u_i, v_j) \\ \gamma(C_u, C_v) &= \frac{1}{2}(S_u(C_u, C_v) + S_v(C_u, C_v)) \end{aligned} \tag{53}$$

where u_i is the ith representative image for the category C_u, v_j is the j-th representative image for the category C_v, N_u and N_v are the sizes of the most representative images for the categories C_u and C_v, and $K(u_i, v_j)$ is the kernel function to calculate the visual similarity between two most representative images u_i and v_j from two categories C_u and C_v.

Because the number of object classes and image concepts could be very large and the size of their most representative images is even enormous, it is time-consuming to obtain the pairwise inter-concept visual correlations. An alternative way for calculating the inter-concept visual correlations is to apply min-max pulling to determine the most representative vectors for each category. Let L be the length of the ith feature vector and $u_i(l)$ denotes the value of l-th dimension of the ith feature vector u_i, the min-max pulling approach constructs a new feature vector s_u for each category u by taking the strongest occurrence of each dimension in its core, as shown in Equation (54):

$$\forall l = 1 \ldots L : s_u(l) = \max_{i=1 \ldots N_u} |u_i(l)| \tag{54}$$

The inter-concept visual similarity between the categories u and v is measured by the similarity between their most representative vectors: $\gamma^*(C_u, C_v) = K(s_u, s_v)$.

Table 5: Inter-object correlations.

object pair	ϕ	object pair	ϕ	object pair	ϕ	object pair	ϕ
grass-building	0.04	horse-cow	0.31	cat-dog	0.85	car-bicycle	0.53
bird-aeroplane	0.54	car-aeroplane	0.55	boat-grass	0.02	dog-cow	0.51
sign-building	0.63	sheep-horse	0.66	body-dog	0.83	body-cat	0.79
road-building	0.72	road-sign	0.59	boat-aeroplane	0.73	mountain-road	0.62
cat-grass	0.03	tree-cat	0.72	book-grass	0.06	sky-book	0.02
boat-sky	0.06	aeroplane-mountain	0.43	mountain-water	0.75	body-sky	0.06

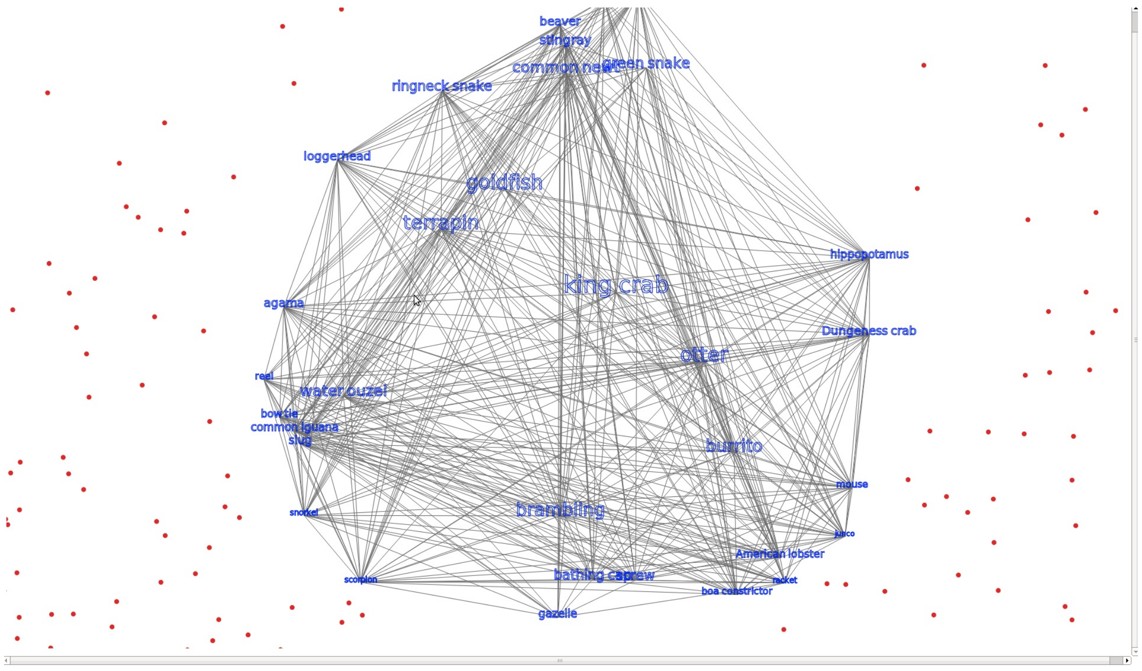


Figure 27: The visual concept network for our experimental image set.

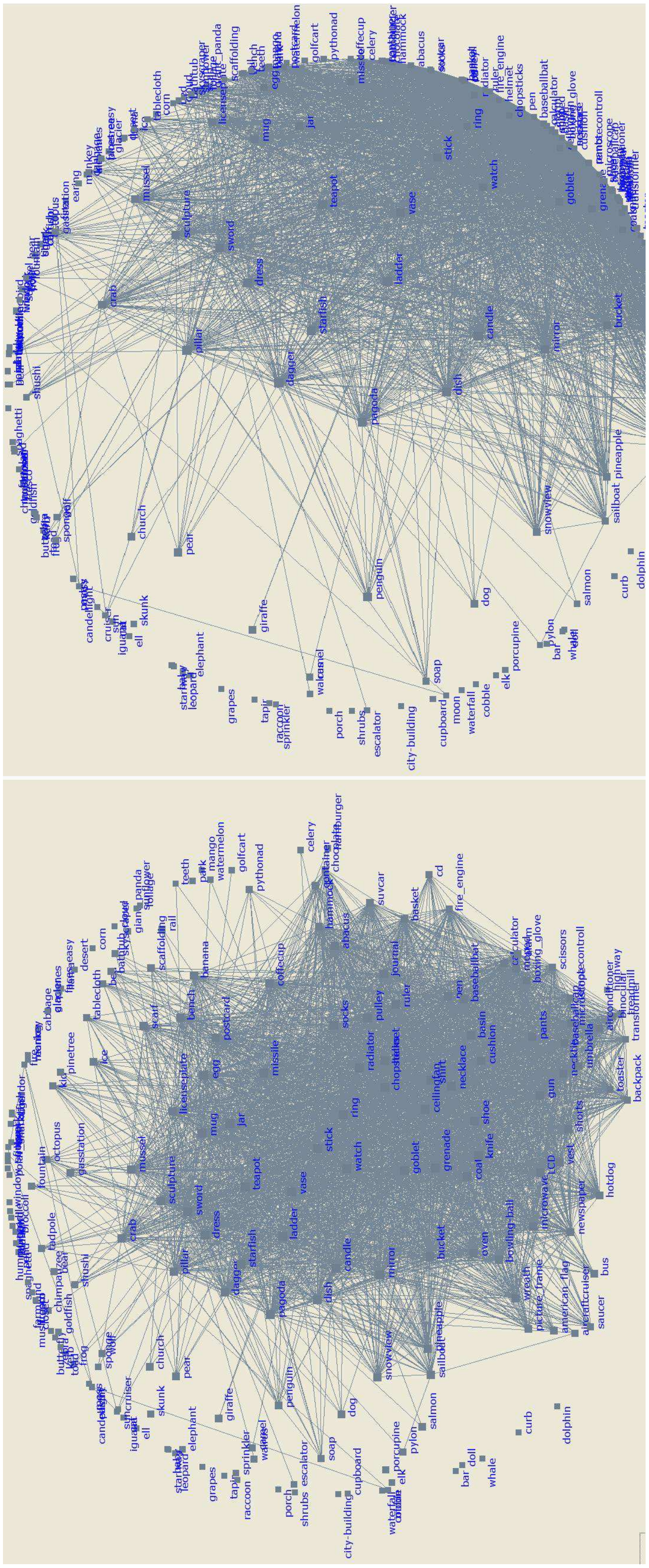


Figure 28: Different views of our object correlation network.

The co-occurrence correlation $\rho(C_u, C_v)$ between two object classes C_u and C_v is defined as:

$$\rho(C_u, C_v) = -P(C_u, C_v) log \frac{P(C_u, C_v)}{P(C_u) + P(C_v)} \tag{55}$$

where $P(C_u, C_v)$ is the co-occurrence probability for two object classes C_u and C_v in our image collections, $P(C_u)$ and $P(C_v)$ are the occurrence probabilities for C_u and C_v.

For two given object classes C_u and C_v, their visual similarity context $\gamma(C_u, C_v)$ and their co-occurrence correlation $\rho(C_u, C_v)$ are first normalized into the same scale. The inter-object correlation $\phi(C_u, C_v)$ between two object classes C_u and C_v is finally defined as:

$$\phi(C_u, C_v) = \eta \cdot \bar{\gamma}(C_u, C_v) + (1 - \eta) \cdot \bar{\rho}(C_u, C_v) \tag{56}$$

where η is the weighting factor, $\bar{\gamma}(C_u, C_v)$ and $\bar{\rho}(C_u, C_v)$ are the normalized visual similarity context and co-occurrence correlation. The weighting factor is set as $\eta = 0.7$ in our current implementation because the visual similarity contexts are more important than the co-occurrence correlations for inter-object correlation characterization.

The visual concept network for 1,000 most popular real-world object classes and image concepts from ImageNet is shown in Fig. (27) and Fig. (28), where each object class or image concept is linked with multiple visually-similar object classes and image concepts with larger values of $S(\cdot, \cdot)$. It is worth noting that different object classes and image concepts can have different numbers of visually-similar object classes image concepts on our visual concept network. Our hyperbolic visualization algorithm is performed to layout the visual concept network according to the strengths of the inter-concept visual correlations $S(\cdot, \cdot)$, where the inter-concept visual correlations are represented as the weighted undirected edges and the length of such weighted undirected edges are inversely proportional to the strengths of their inter-concept visual correlations $S(\cdot, \cdot)$. Thus the geometric closeness between the object classes and image concepts is directly related to the strengths of their inter-concept visual correlations, so that such graphical representation of the visual concept network can reveal a great deal about how these object classes and image concepts are visually correlated.

Our visual concept network can provide a good environment for: (1) characterizing the inter-concept correlations directly in the visual feature space and such inter-concept visual correlations can also indicate the potential correlations among the classifiers for the visually-similar categories; (2) identifying seed categories and estimating the learning complexity for classifier training, e.g., it is much harder to distinguish two visually-similar categories than discriminating two visually-dissimilar categories; (3) categorizing a large number of object classes and image concepts on the visual concept network into a set of groups, where the visually-similar object classes and image concepts are clustered in the same group and their classifiers are strongly inter-related and should be trained jointly rather than independently. For the visually-similar object classes and image concepts in the same group, learning their inter-related classifiers jointly can bring powerful inference scheme to enhance their discrimination power significantly.

Since the object classes are inter-related and such inter-object correlations can be represented explicitly by the object correlation network and can be represented pre-

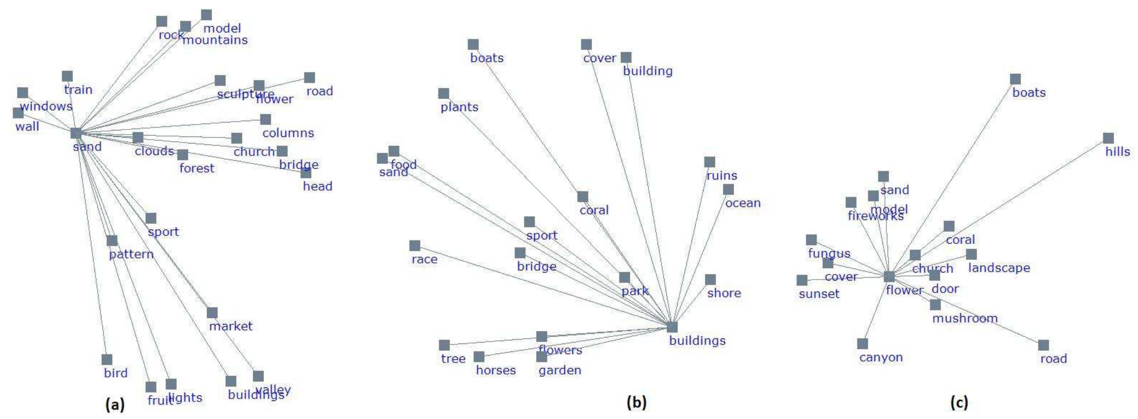


Figure 29: The inter-related object classes which may easily confuse the machine learning tools.

cisely by the strengths of their inter-object correlations $\phi(\cdot,\cdot)$. When a large number of object classes come into view, directly modeling of such the inter-object correlations over the whole object correlation network becomes computationally intractable. In this paper, a multi-task structural SVM scheme is developed by incorporating the first-order nearest neighbors (i.e., clique for each object class on the object correlation network), multi-task learning and structural SVM to leverage the inter-object correlations to achieve more accurate training of a large number of inter-related object classifiers.

For a given object class, its first-order nearest neighbors on the object correlation network are strongly correlated and their training instances may share similar visual properties. Some examples for such inter-related object classes are illustrated in Figure 29 and Figure 30. Isolating these inter-related object classes and training their classifiers independently are not appropriate. In order to leverage the inter-object correlations for training the inter-related object classifiers jointly, it is very important to develop new algorithms for integrating multi-task learning with structural SVM to

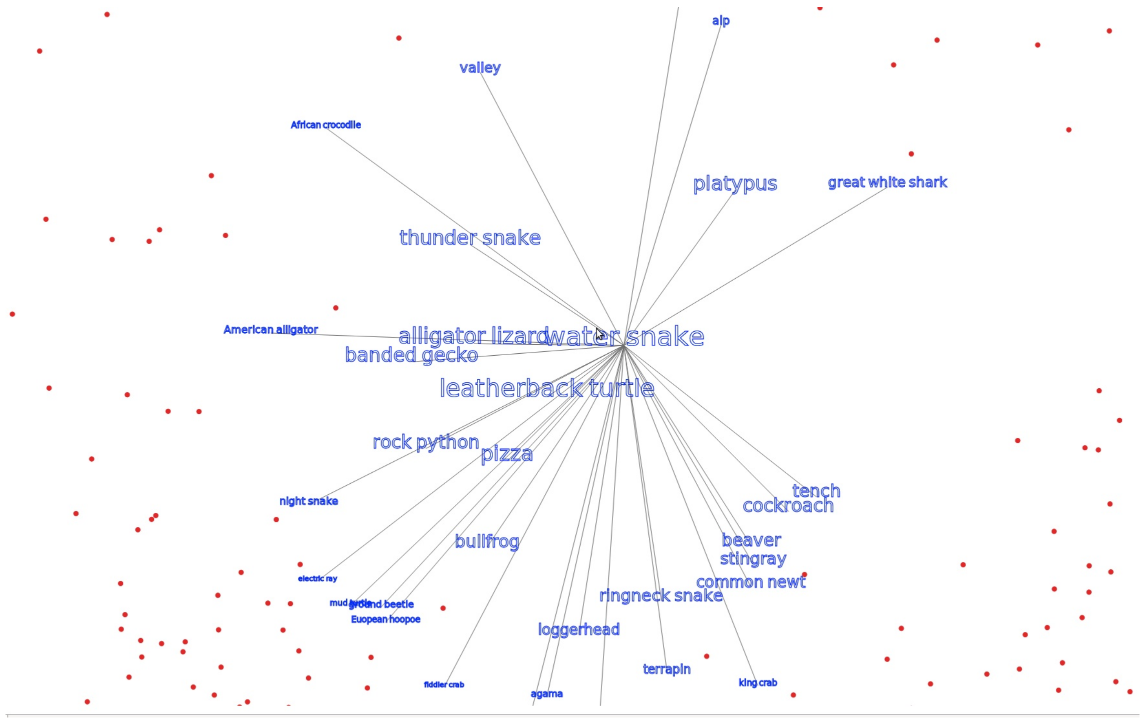


Figure 30: Some examples for the inter-related object classes and image concepts.

model the inter-task relatedness more precisely. The idea behind multi-task learning is that if multiple inter-related learning tasks share a common prediction component, such common prediction component can be estimated more accurately by considering these inter-related learning tasks together. When multiple object tags are jointly used for image tagging, the corresponding object classes should be inter-related rather than independent and thus learning the classifiers for these inter-related object classes should be considered jointly. Our object correlation network can provide a good environment for identifying such the inter-related learning tasks directly in the feature space. The idea behind structural SVM is to exploit the inter-label correlations in the label space for supporting structural prediction.

Table 6: Examples of category groups, where the categories with similar visual properties are clustered into the same group.

Group 1:	hammerhead, axolotl, tree frog, green snake, dugong, mouse, pudding, trifle, French loaf, pretzel, cheeseburger, hotdog, nacho, zucchini, carbonara, dough, pizza, potpie, burrito, eggnog
Group 2:	ostrich, barbell, bassoon, bell cote, bolo tie, bow, canteen, hook, megalith, nailfile, oboe, padlock, pinion, plunger, shovel, swing, thimble, torch, tuning fork
Group 3:	magpie, hornbill, toucan, admiral, giant panda, rock beauty, jake-o'-lantern, mortarboard, puck, ski mask, chocolate sauce
Group 4:	kite, European swift, headset
Group 5:	vine snake, nematode, soup bowl, consomme, espresso
Group 6:	harvestman, black and gold garden spider, black widow, tiger beetle, long-horned beetle, monarch, manhole cover
Group 7:	bee eater, adjustable wrench, bib, cap opener, circular saw, corkscrew, face powder, garrison cap, hacksaw, hammer, letter opener, plane, radio telescope, stethoscope, stupa, switchblade, toilet seat, wing tip, wire cutter, toilet tissue
Group 8:	black swan, flat-coated retriever, Tibetan mastiff, Newfoundland, American black bear, sloth bear, gorilla, chimpanzee, gibbon, siamang, guenon, colobus, capuchin, howler monkey, lesser panda, bearskin, car wheel, bell pepper
Group 9:	echidna, sea anemone, bittern, limpkin, sea urchin, zebra, lionfish, spider web, cardoon
Group 10:	abaya, academic gown, metronome, obelisk, suit
Group 11:	baseball, basketball, croquet ball, ping-pong ball, rugby ball, soccer ball, volleyball, rapeseed, gravel
Group 12:	beer bottle, beer glass, broom, church, cocktail shaker, hourglass, mosquito net, perfume, pop bottle, punching bag, wine bottle, cider vinegar

7.1.3 Group Generation via Affinity Propagation

When the visual concept network is available, it can provide a good environment to determine the groups of visually-similar object classes and image concepts. In this paper, Affinity Propagation algorithm [131] is performed over the visual concept network (i.e., inter-concept visual similarity matrix) to determine the groups of visually-similar object classes and image concepts.

We conclude this section with a brief comparison between our visual concept network and some concept hierarchies. Compared with semantic [107] or co-occurrence networks [11], our visual concept network is constructed directly in the visual feature space. It is worth noting that the visual feature space is the common space for classifier training and automatic image classification. Some tree-structured hierarchies are also constructed in the visual feature space [122, 123, 126, 121] and some of them also employ graph-based partition over the visual similarity matrix, but our visual concept network treats all these object classes and image concepts to be equally important rather than organizing them in a hierarchical way. There is no evidence that the "super-classifiers" for the high-level image concepts in a concept hierarchy (which are modeled by using hierarchical support vector machine) can have good separability or bounded generalization errors, thus it is not a good idea to make a restriction on the object classes and image concepts as done in the hierarchical SVM algorithm. On the other hand, our visual concept network may bring more powerful inference scheme for classifier training, and it is further used to: (1) determine inter-related learning tasks and identify the seed categories for classifier training; (2) detect the groups of visually-similar object classes and image concepts, and (3) estimate their learning complexity for classifier training.

7.2 Multi-task Support Vector Machine

A multi-task structural SVM scheme is developed by incorporating the object correlation network, multi-task learning and structural SVM to enhance the discrimination power of a large number of inter-related object classifiers: (a) The object correlation network is used to identify the inter-related learning tasks directly in the feature space, e.g., training multiple inter-related object classifiers jointly; (b) The inter-task relatedness is characterized explicitly by using the strengths of the inter-object correlations $\phi(\cdot,\cdot)$ and a *common prediction component* is used to model the inter-task relatedness and shared among these inter-related object classifiers; (c) The structural SVM is integrated with multi-task learning to model the inter-task relatedness more precisely and estimate the common prediction component more accurately. By seamlessly integrating multi-task learning with structural SVM, our multi-task structural SVM algorithm is able to exploit the inter-object correlations explicitly in the input space (i.e., the feature space for classifier training and testing), thus it can provide a new approach for inter-related classifier training and address the issue of multiple tags more effectively.

In our multi-task structural SVM scheme, a *common regularization term* W_0 of the SVM classifier is used to model the inter-task relatedness among multiple SVM classifiers for the inter-related object classes. For one given object class C_j, its classifier is defined as:

$$f_{C_j}(x) = \sum_{C_t \in \mathcal{T}} \gamma_t (W_0 + V_t)^{tr} \Phi_t(x) \tag{57}$$

where W_0 is the common regularization term shared among the classifiers for multiple inter-related object classes centered by C_j as shown in Figure 29, V_t is the *individual regularization term* for the classifier between the given object class C_j and its neighbor C_t, γ_t is the weight how class C_t contributes the classification of C_j.

The common regularization term W_0 is used to model the inter-task relatedness and shared among the classifiers for multiple inter-related object classes. W_0 can be estimated more reliably by minimizing their joint objective function J for T inter-related learning tasks.

$$J = \frac{1}{2}(\|W_0\|^2 + \sum_{t=1}^{T} \lambda_t \|V_t\|^2) + c_0 \sum_{t=1}^{T} \sum_{i=1}^{n_j} \xi_{ti} + \sum_{t=1}^{T} c_t \sum_{i=1}^{n_t} \eta_{ti} \tag{58}$$

where $\xi_{ti} \geq 0$, $\eta_{ti} \geq 0$, n_j and n_t are the total number of training instances for the object classes C_j and C_t.

7.3 Optimizing the joint SVM problem

To solve the joint optimization problem, we use the Lagrangian Principle. We add a dual set of variables, one for each constraint and get the Lagrangian L of the optimization problem:

$$\begin{aligned} L = J \; & - \; \sum_{t=1}^{T} \sum_{i=1}^{n_j} \beta_{ti} \left(\langle W_0 + V_t, \Phi_t(x_{ji}) \rangle + \xi_{ti} - 1 \right) + \sum_{t=1}^{T} \sum_{i=1}^{n_t} \overline{\beta_{ti}} \left(\langle W_0 + V_t, \Phi_t(x_{ti}) \rangle - \eta_{ti} + 1 \right) \\ & - \; \sum_{t=1}^{T} \sum_{i=1}^{n_j} \sigma_{ti} \xi_{ti} - \sum_{t=1}^{T} \sum_{i=1}^{n_t} \overline{\sigma_{ti}} \eta_{ti} \end{aligned}$$

We now seek a saddle point of the Lagrangian L, e.g., the partial difference of L satisfied:

$$\begin{aligned} \frac{\partial L}{\partial W_0} &= W_0 - \sum_{t=1}^{T} \sum_{i=1}^{n_j} \beta_{ti} \Phi_t(x_{ji}) + \sum_{t=1}^{T} \sum_{i=1}^{n_t} \overline{\beta_{ti}} \Phi_t(x_{ti}) \\ \frac{\partial L}{\partial V_t} &= \lambda_t V_t - \sum_{i=1}^{n_j} \beta_{ti} \Phi_t(x_{ji}) + \sum_{i=1}^{n_t} \overline{\beta_{ti}} \Phi_t(x_{ti}) \\ \frac{\partial L}{\partial \xi_{ti}} &= c_0 - \beta_{ti} - \sigma_{ti}, \quad \frac{\partial L}{\partial \eta_{ti}} = c_t - \overline{\beta_{ti}} - \bar{\sigma}_{ti} \end{aligned}$$

and we can get:

$$\begin{aligned}
W_0 &= \sum_{t=1}^{T}\sum_{i=1}^{n_j}\beta_{ti}\Phi_t(x_{ji}) - \sum_{t=1}^{T}\sum_{i=1}^{n_t}\overline{\beta_{ti}}\Phi_t(x_{ti}) \\
V_t &= \frac{1}{\lambda_t}\left(\sum_{i=1}^{n_j}\beta_{ti}\Phi_t(x_{ji}) - \sum_{i=1}^{n_t}\overline{\beta_{ti}}\Phi_t(x_{ti})\right) \\
c_0 &= \beta_{ti} + \sigma_{ti}, \quad c_t = \overline{\beta_{ti}} + \overline{\sigma_{ti}}
\end{aligned}$$

The dual form of the problem is then simplified as:

$$L = \sum_{t=1}^{T}\sum_{i=1}^{n_j}\beta_{ti} + \sum_{t=1}^{T}\sum_{i=1}^{n_t}\overline{\beta_{ti}} - \frac{1}{2}\left(\|W_0\|^2 + \sum_{t=1}^{T}\lambda_t\|V_t\|^2\right)$$

Given the training image instances for T inter-related object classes on the object correlation network (i.e.,the object classes in the same group), the margin maximization process for object classifier training is then transformed into a quadratic problem:

$$\begin{aligned}
\max_{\beta_{ti},\bar{\beta}_{ti}} L = \sum_{t=1}^{T}\sum_{i=1}^{n_j}\beta_{ti} + \sum_{t=1}^{T}\sum_{i=1}^{n_t}\overline{\beta_{ti}} - \frac{1}{2}[\sum_{t,s=1}^{T}\sum_{i=1}^{n_j}\sum_{l=1}^{n_l}\beta_{ti}\beta_{sl}K_{ts}(x_{ji},x_{jl}) \\
-\sum_{t,s=1}^{T}\sum_{i=1}^{n_j}\sum_{l=1}^{n_s}\beta_{ti}\overline{\beta_{sl}}K_{ts}(x_{ji},x_{sl}) - \sum_{t,s=1}^{T}\sum_{i=1}^{n_t}\sum_{k=1}^{n_j}\overline{\beta_{ti}}\beta_{sk}K_{ts}(x_{ti},x_{kj}) \\
+\sum_{t,s=1}^{T}\sum_{i=1}^{n_t}\sum_{k=1}^{n_s}\overline{\beta_{ti}\beta_{sk}}K_{ts}(x_{ti},x_{sk}) + \sum_{t=1}^{T}\frac{1}{\lambda_t}(\sum_{i,l=1}^{n_j}\beta_{ti}\beta_{tl}K_t(x_{ji},x_{jl}) \\
-2\sum_{i=1}^{n_j}\sum_{l=1}^{n_t}\beta_{ti}\overline{\beta_{tl}}K_t(x_{ji},x_{jl}) + \sum_{i,l=1}^{n_t}\overline{\beta_{ti}\beta_{tl}}K_t(x_{ti},x_{tl}))]
\end{aligned}$$

subject to: $\quad \forall_i: \quad \forall_t: \quad \beta_{ti} \geq 0, \quad \overline{\beta_{ti}} \geq 0$

To deal with the structural prediction problem, it is very attractive to construct a joint kernel function that is better suited to joint-space support estimation. In this research, a tensor products is incorporated to define the joint kernel $\kappa((x_i,y_i),(x_j,y_j))$

as:

$$\kappa((x_i, y_i), (x_j, y_j)) = \kappa(x_i, x_j)\kappa_s(y_i, y_j) \tag{30}$$

where $\kappa(x_i, x_j)$ is the visual kernel for the visual features and $\kappa_s(y_i, y_j)$ is the semantic kernel to characterize the semantic similarity context between two object classes and their labels y_i and y_j (i.e., inter-object correlation on the label space).

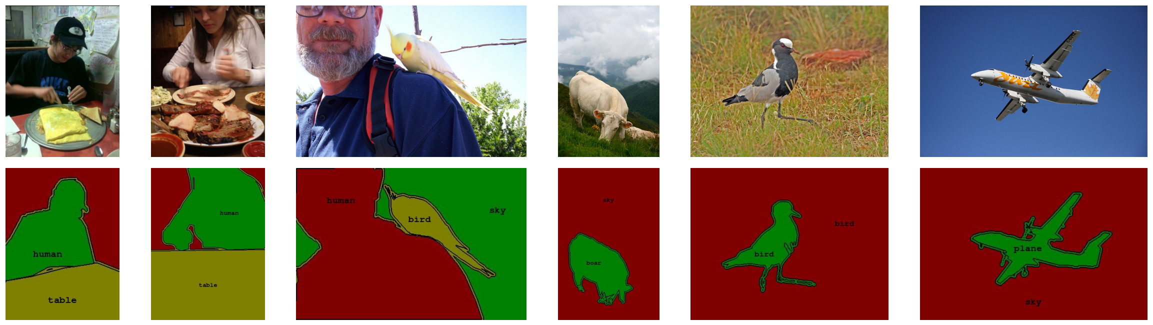
By learning from a joint training image set $\Omega = \{(x_{it}, y_{it})|i = 1, \cdots, n; t = 1, \cdots, T\}$ for T inter-related object classes on the object correlation network, the classifier for the given object class C_j can be determined as:

$$f_{C_j}(x) = \sum_{h,t=1}^{T} \gamma_t \kappa(t, h) \left(\sum_{i=1}^{n_j} \beta_{hi} \kappa(x_{ji}, x) - \sum_{i=1}^{n_h} \overline{\beta_{hi}} \kappa(x_{hi}, x) \right)$$
$$+ \sum_{t=1}^{T} \frac{\gamma_t}{\lambda_t} \left(\sum_{i=1}^{n_j} \beta_{ti} \kappa(x_{ji}, x) - \sum_{i=1}^{n_t} \overline{\beta_{ti}} \kappa(x_{ti}, x) \right) \tag{59}$$

One can observe that our classifiers for the inter-related object classes consist of two components: (a) individual prediction component; and (b) common prediction component.

By learning two different sets of the weights β and $\bar{\beta}$ for the training instances simultaneously, our multi-task structural SVM algorithm can automatically establish two independent decision boundaries for both the common prediction component (shared among the inter-related discriminant functions) and the individual prediction component of the discriminant function for each particular object class. The training instances, which are used to construct the common prediction component for multiple inter-related object classifiers (i.e., support vectors with large values of $\bar{\beta}$), are less important for the individual prediction components for these inter-related object

Table 7: Object detection results by our multi-task structural SVM algorithm.



classifiers (i.e., with smaller values or even zero values of weights β).

The weights $\bar{\beta}$ are fixed for all these T inter-related discriminant functions to characterize their common prediction component. By integrating the training instances for multiple inter-related object classes to learn a common prediction component and separating it from their individual prediction components, our multi-task structural SVM algorithm can significantly enhance the discrimination power and the generalization ability of the inter-related object classifiers.

The inter-related object classifiers for a set of object classes are trained in our current implementation. After the object classifiers are trained, they are further used to identify the objects from the test images and recommend the suitable tags for automatic image annotation. Some experimental results on object detection (tag recommendation) are given in Table 7 and further classification results accompanied with multiple learning are shown in Figure 31 and Figure 32. One can observe that our multi-task structural SVM scheme can directly learn the object detectors from the loosely-tagged images and precisely localize the objects in the images.

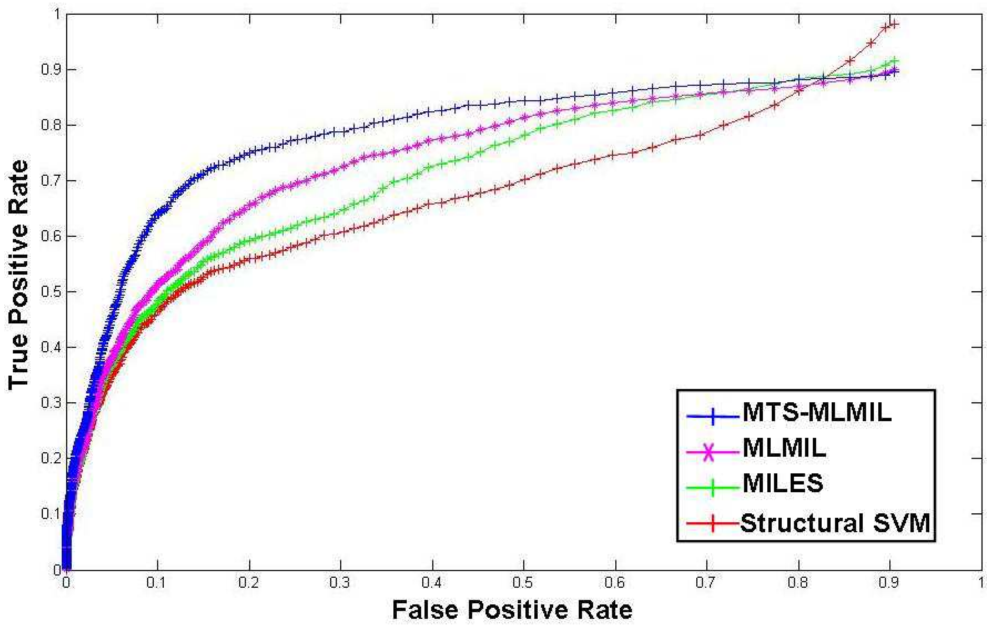


Figure 31: ROC curves for performance comparison: our multi-task structural SVM (MTS-MLMIL) algorithm *versus* other most relevant algorithms.

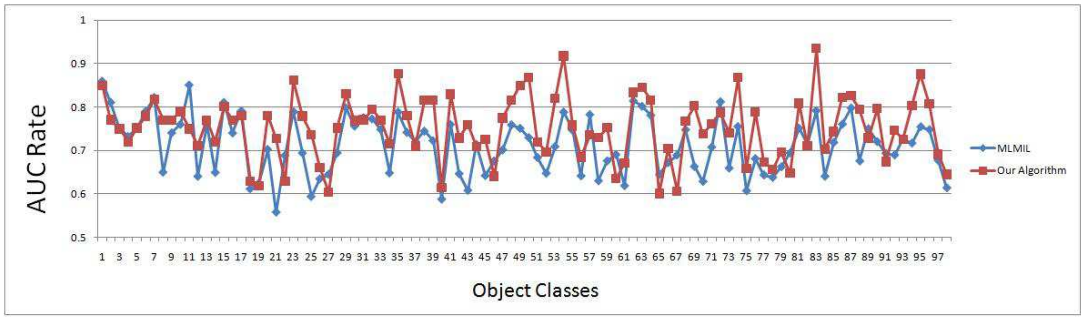


Figure 32: AUC (area under ROC curve) rate for performance comparison: our new MIL algorithm *vs* MLMIL algorithm.

CHAPTER 8: STRUCTURAL LEARNING WITH CONCEPT NETWORK

To support large scale image classification, one of the fundamental problems is to design a multi-class classification framework: (1) The proposed multi-class classifier is constructed by a series of binary discriminative classifiers (SVM) which possess good generalization in computer vision tasks. Obviously, to intuitively learn $O(m^2)$ (m denotes the number of the object categories) pair-wise binary SVM is not acceptable due to the computational complexity when m has become very large. Thus, only a relatively small number of selected classifiers can be learned, which handle specific sub-tasks but are sufficient to the discriminant logic of the whole multi-class classification problem; (2) In the perspective of a single object category, its classifier should be jointly learned with the classifiers for its highly visually-correlated categories, not only to improved the discriminative power of the classifier but also to leverage samples from inter-correlated classes for those classes with relatively fewer samples.

It is also worth noted that as an effective way to boost the computational capability for large scale problems, distributed computing has been playing an important role in large scale computing. Specifically in our classifier training task, there are two aspects to parallelize our multi-class classifier training: (1) the training process of binary SVM which takes large amounts of training samples can be parallelized intuitively by invoking existing parallel-SVM toolkits; (2) the construction of the multi-class meta-classifier from binary SVMs needs to be adaptive to parallelization, thus the training process of binary SVMs can respectively distributed to different computing units. Our parallelization concerns are mainly focus on the latter term. However, as the common sense of parallel computing, the difficulty of parallelization comes from poor separability of the problem. In another word, less inter-dependency between subtasks leads better parallelization performance, which somehow conflicts with our joint classifier design. Thus, a potential trade-off may have to be made to balance the computational cost and the accuracy of the correlated classifiers.

With the object correlation network in Chapter 7.1, the inter-class correlations are explicitly measured by the pair-wise object similarities. Therefore, the remaining problem is to construct an acceptable number of binary classifiers to complete the discriminative logic in term of multi-class classification, which also explicitly or implicitly take the inter-object correlations into account.

8.1 Classifiers on Edges

As it has been observed that the object correlation network which has been built in Chapter 7.1 is a sparse connected graph, i.e., all the nodes are connected by at least one route with each other. One way to construct the multi-class classifier is to learn a pair of binary classifiers $f(x|c_i, c_j)$ and $f(x|c_j, c_i)$ for each pair of object nodes c_i, c_j which are linked by an edge in the object correlation network, where each $f(x|c_i, c_j)$ is a binary SVM which functions to:

$$f(x|c_i, c_j) = \begin{cases} 1 & P(x \in c_i) > P(x \in c_j) \\ -1 & P(x \in c_j) > P(x \in c_i) \end{cases} \tag{60}$$

Derived from Multi-task Learning which has been described in Chapter 7, $f(x|c_i, c_j)$

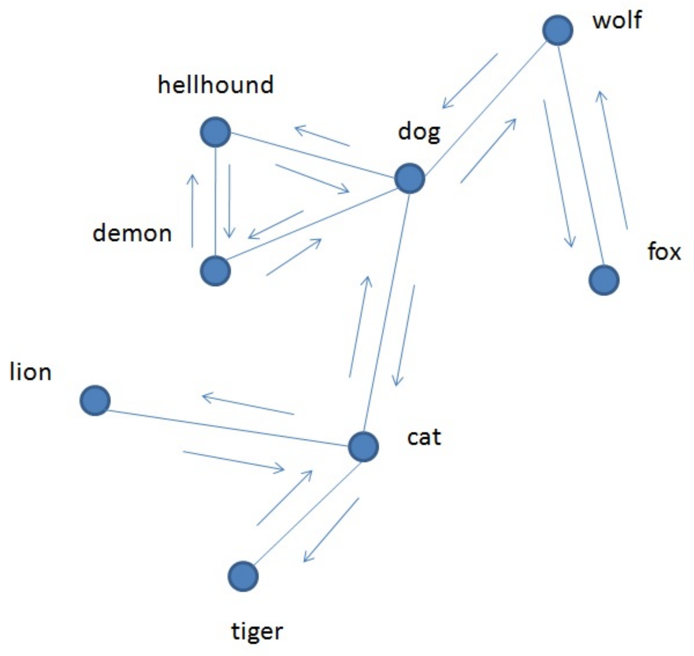


Figure 33: Classifier on Edges: two binary classifiers are trained for each pair of connected object nodes

can be modeled as following:

$$f(x|c_i, c_j) = W_{c_i} + \lambda_{c_i c_j} V_{c_i c_j} \tag{61}$$

where W_{c_i} is the common part of all classifiers which are relative to c_i and $V_{c_i c_j}$ is the individual part which discriminates the samples of c_i from c_j. The classifier in Equation (61) can be learning by solving the optimization problem in Equation (58) in Chapter 7.

For the decision process of the potential framework, a random walk algorithm is performed to find out the object node which can best describe the incoming sample x, which can be formulated as following:

$$P_t(c_i|x) = (1-\theta_1-\theta_2)\cdot P_{t-1}(c_i|x)+\theta_1\cdot \sum_{c_j\in\Omega(c_i)} f(x|c_i, c_j)\cdot P_{t-1}(c_j|x)-\theta_2\cdot \sum_{c_j\in\Omega(c_i)} f(x|c_j, c_i)\cdot P_{t-1}(c_j|x) \tag{62}$$

where $\Omega(c_i)$ denotes the first-order neighbours of c_i and θ_1, θ_2 are weight parameters and initially $P_t(c_i|x) = 1/m$.

In the above proposed framework, only m joint optimization problems need to be solved in the training process which are efficient enough for large scale joint classifier training. However, since the existence of bridges in the object correlation graph, the proposed method has the same disadvantage which many tree-structured hierarchical methods (where all links are bridges) share: the performances of bridge classifiers plays more important roles in the random walk process than the performances of non-bridge classifiers. Nevertheless, no evidence shows those bridge classifiers can achieve are guaranteed to return correct results. Since in the object correlation network, highly correlated object concepts are linked, which indicates the generalization power of classifiers for discriminating linked concepts is usually low. On the other hand, with more bridges in the object network, it requires less time for both the training process and the testing process. Thus, a potential trade-off between the computational complexity and the robustness of the proposed framework need to be made in order to achieve reliable performance. However, the good news is that it is intuitive to parallelize this approach since the classifier training process at t-th iteration only depends on the classifiers at $(t-1)$-th iteration, which may be another potential perspective to bound the training time.

8.2 Multi-class Classifier Hierarchy

As mentioned previously, the performance of the classifier-on-edge approaches highly depends on those bridge classifiers on the object network. Thus instead of directly

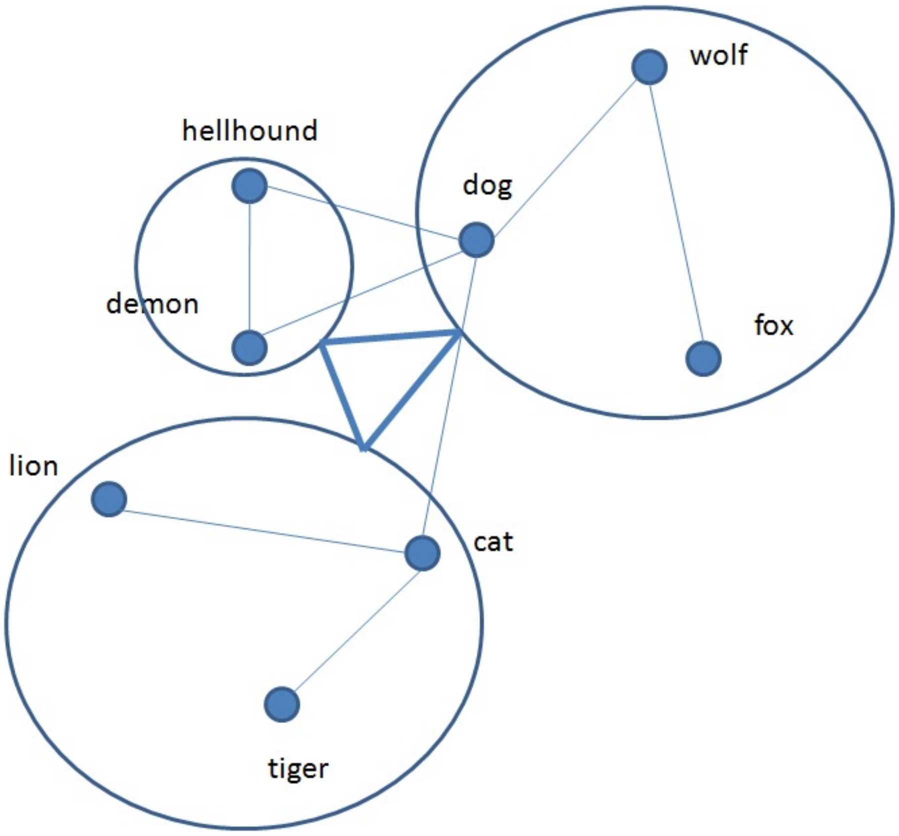


Figure 34: Group-based Approach: It is constructed by two kinds of classifiers: inter-group meta-classifiers and intra-group classifiers

working on the object correlation network, an alternative approach is to only partition the object correlation network into a certain number of sub-graphs. Objects in the same sub-graph form an object group.

In practice, a clustering method, Affinity Propagation [131] is utilized to partition the object similarity matrix (which is calculated in Chapter 7.1) into K groups. Ideally, the inter-group class correlations are lower than the intra-group ones. Thus the proposed approach attempts to reduce the multi-class classification into a two-layer problem. As Figure 34 shows, the meta-classifier $f_{G_iG_j}$ is trained to discriminate between two groups of classes and the intra-group classifier $f_{c_ic_j}$ is trained jointly by multi-task learning which has been described in Chapter 7 if and only if c_i and c_j are in the same group.

The decision process of the group-based approach is intuitive: for an incoming testing sample x, the meta-classifiers $\{f_{G_iG_j}\}$ firstly decide which group x should goes in and the intra-group classifiers of the corresponding group further make the final decision which the object category that x belongs to. The total number of binary classifiers need to be trained is in the order of $O(K^2 + m)$. In the best case, if the graph is evenly partition into K groups, where $K \approx \sqrt{m}$, the number of binary classifiers would be reduced to the order of $O(m)$.

Compared to the classifier-on-edge method which has been described previously, meta-classifiers between relatively-low correlated groups seems to be more reliable than the classifiers of randomly occurred bridges in the object correlation graph. However, it worth noting that as a two-layer hierarchical method, the proposed methods share the common problem with other hierarchical approaches: Neither training error or generalization error of high level binary classifiers are equally distributed to sub-classes (as shown in Figure 35). Moreover, T. Gao et al.[120] addressed the issue that the measurement of separability in class partition and which in boundary estimation are not consistent in those two-layer hierarchical methods.

8.3 Discriminative Learning for Large-Scale Classifier Training

In this section, we introduce the details for designing our multi-class classification method: (a) a large set of L sub-classifiers $f_l(x) \rightarrow \{-1, +1\}$ are first trained and each sub-classifier focuses on distinguishing two given categories; (b) the large set of L sub-classifiers are further used to generate the multi-class classifier $F(x)$.

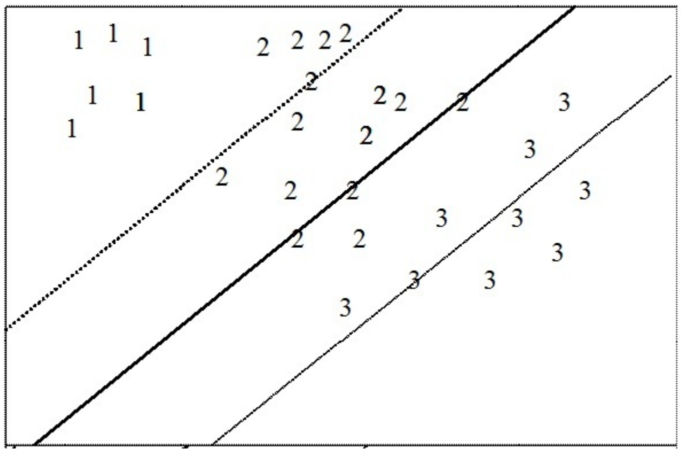


Figure 35: The figure shows a classifier to separate the 1st and 2nd from 3rd. Errors are unequally distributed among class 1 and class 2 with hierarchical methods

Given a test sample (image instance) x, the output of our multi-class classifier is aggregated from the binary outputs of all these L binary sub-classifiers by using a voting scheme, which is shown as Equation (63):

$$F(x) = \sum_{l=1}^{L} I_l \cdot f_l(x) \tag{63}$$

where I_l denotes the coding vector of the l-th sub-classifier. The coding vector $I_l \in \{-1, 0, +1\}^m$ is a m-dimension column vector (m is the number of the relevant categories) and it is used to indicate the function of the l-th sub-classifier f_l when the sub-classifier f_l is used to separate two given categories. If $I_l(c_i) = +1$, the category c_i is in the positive set for training the sub-classifier f_l, while $I_l(c_i) = -1$ indicates that the category c_i is in the negative set for training the sub-classifier f_l. The category c_i is ignored for training the sub-classifier f_l if $I_l(c_i) = 0$. The output of multi-class classifier $F(x) \rightarrow R^m$ is also an m-dimension column vector, where the value of each dimension is used to indicate the voting to the corresponding category from all these L binary sub-classifiers. The final decision is made by a max-voting strategy [119]:

$$y = \arg\max_{c} F(x, c) \tag{64}$$

where $F(x, c)$ is the value of c-th dimension of $F(x)$.

The remaining issues are to determine the code vector I_l for each binary sub-classifier $f_l(x)$ and train the sub-classifier f_l for the relevant categories which are involved in the coding vector I_l. Rather than performing a two-step method which first sets I_l and then learns the sub-classifier f_l according to the given I_l, our approach is to learn the pair $\{I_l,\ f_l\}$ simultaneously, where a pair of seed categories $\{c_l^+,\ c_l^-\}$ is automatically identified from our visual concept network (e.g., category clustering over our visual concept network for group generation).

8.3.1 Learning sub-classifiers

Given a pair of seed categories $\{c^+,\ c^-\}$, the goal of our sub-classifier learning algorithm is to train a binary sub-classifier f as well as its coding vector I. The sub-classifier f should be able to separate the positive category c^+ from the negative category c^- (i.e., $I(c^+) = +1$ and $I(c^-) = -1$ are satisfied). Unlike [120] maximizing the number of categories which participate in the binary classification task (i.e., the number of non-zero elements in I), our learning algorithm focuses on only those categories in the positive group G^+ and the negative group G^- ($c^+ \in G^+$ and $c^- \in G^-$). For a given binary classification task, the positive group G^+ and the negative group G^- are identified automatically by performing AP clustering over our visual concept network for group generation (see Chapter 7.1). Our algorithm tends to learn a binary SVM classifier to minimize the training error while maximizing the number of categories in $G^+ \cup G^-$. Given a set of training instances $\{x_i, y_i\}$, the sub-classifier $f(x) = w^T \cdot x + b$ is learned by solving the optimization problem in Equation (65) in case c^+ and c^- are not in the same group (i.e. $c^+ \neq G^-$):

$$\begin{aligned} \min_{w,b,I,\{\xi_i\}} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{N}|I(y_i)|\xi_i - A\sum_{i=1}^{N}|I(y_i)| \\ \text{s.t.} \quad & \forall c \in G^+ : I(c) \in \{0, +1\}, \\ & \forall c \in G^- : I(c) \in \{0, -1\}, \\ & \forall c \notin G^+ \cup G^- : I(c) = 0 \\ & I(c^+) = +1, I(c^-) = -1, \\ & \forall i : I(y_i)(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0 \end{aligned} \tag{65}$$

where N is the total number of training instances (only the instances with $y_i \in G^+ \cup G^-$ are actually involved in the training process for the sub-classifier f_l), C and A are the regularization parameters which also bound the average slack (hinge loss) of each category [120]:

$$\forall c, I(c) \neq 0 : \frac{1}{N_c}\sum_{i:y_i=c}\xi_i \leq \frac{A}{C} \tag{66}$$

Because it is a NP-Hardness issue for optimizing the problem in Equation (65), an iterative approximation [120] is introduced to solve the problem in practice. The optimization variables are divided into 2 sets: (1) the coding vector: I; (2) SVM decision plane and boundaries: $\{w, b, \{\xi_i\}\}$. We alternatively optimize over two sets of variables, until the condition in Equation (66) is violated.

When the coding vector I is fixed, the optimization problem in Equation (65) is reduced to a standard binary SVM problem. We can use any kind of SVM classifier training algorithms, such as SMO [117], to optimize over $\{w, b, \{\xi_i\}\}$. When the SVM decision plane $\{w, b\}$ is fixed, I can be refined by adding a currently unconcerned category c^* ($I(c^*) = 0$) with the least average slack under current configuration $\{w, b\}$:

$$\begin{aligned} L(c|w,b) &= \frac{1}{N_c} \sum_{i:y_i=c} \max\{0, 1 - (2 \cdot 1[c \in G^+] - 1)(w^T x_i + b)\} \\ c^* &= \underset{c:c \in G^+ \cup G^- \wedge I(c)=0}{\operatorname{argmax}} L(c|w,b) \end{aligned} \tag{67}$$

where $1[\cdot]$ is the indicator function. The optimization process stops when the condition in Equation (66) is not met by c^*, and thus we get the optimal I and $\{w, b\}$.

To reduce the total number of binary sub-classifiers, it is worth considering that our multi-class classifier may also be used to classify other categories which are not either in the positive group G^+ or in the negative G^- in case all the instances for these categories $c \notin G^+ \cup G^-$ happen to lie on the same side of the hyper-plane of the sub-classifier $f(x) = w^T x + b$. Thus, for all the categories c not in the seed groups, a post evaluation is done to get the best average slack of the sub-classifier $f(x)$ for the category c:

$$\begin{aligned} L_c^(t) &= \frac{1}{N_c} \sum_{y_i=c} \max\{0, 1 - t \cdot (w^T x_i + b)\} \\ L_c^* &= \min_{t \in \{-1,1\}} L_c(t) \\ t^* &= \underset{t \in \{-1,1\}}{\operatorname{argmin}} L_c(t) \end{aligned} \tag{68}$$

We add the category c with the code t^* into the code vector I if the best average slack (hinge loss) L_c^* is below the threshold $\frac{A}{C}$, and it indicates that the category c meets the condition in Equation (66). Unlike those categories in the seed groups, the categories, which are added recently and are not included in the seed groups, are only taken as the bonuses. Such bonus categories do not affect the effectiveness for training the sub-classifier $f(x)$.

It is also possible that two seed categories c^+ and c^- are in the same group G, thus training such intra-group sub-classifier is similar to the process for training the inter-group sub-classifier. The only difference is that the sub-classifier has to decide a sign for the codes of the categories in G other than c^+ and c^-:

$$\begin{aligned} \min_{w,b,I,\{\xi_i\}} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{N} |I(y_i)|\xi_i - A\sum_{i=1}^{N} |I(y_i)| \\ \text{s.t.} \quad & \forall c \in G : I(c) \in \{-1, 0, +1\}, \\ & \forall c \notin G : I(c) = 0, \\ & I(c^+) = +1, I(c^-) = -1, \\ & \forall i : I(y_i)(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0. \end{aligned} \tag{69}$$

The post evaluation process is meaningless for the intra-group sub-classifier because the visually-similar categories (i.e., object classes and image concepts) in the same group have higher inter-concept visual correlations than the visually-dissimilar categories in different groups have. The complete algorithm for training a sub-classifier $\{f(x), I\}$ is described as Algorithm 6.

8.3.2 Selection of Seed Categories

One single sub-classifier in Chapter 8.3.1, which focuses on discriminating two given categories, is only one component of our multi-class classifier. When all these sub-classifiers are available, the remaining problem is how to utilize them to generate the

```
Initial: $I(c^+) \Leftarrow +1$, $I(c^-) \Leftarrow -1$, $\forall c \notin \{c^+, c^-\} : I(c) \Leftarrow 0$
repeat
   Train $\{w, b\}$ by standard binary SVM algorithm with fixed $I$
   if $G^+ \neq G^-$ then
      $\forall c \in G^+ \wedge I(c) = 0 : t(c) \Leftarrow +1$
      $\forall c \in G^- \wedge I(c) = 0 : t(c) \Leftarrow -1$
   else
      $\forall c \in G \wedge I(c) = 0 : t(c) \Leftarrow \underset{y \in \{-1,1\}}{\operatorname{argmin}} \frac{1}{N_c} \sum_{i:y_i=c} \max\{0, 1 - y \cdot (w^T x_i + b)\}$
   end if
   $\forall c \in G^+ \cup G^- \wedge I(c) = 0 : L(c) \Leftarrow \frac{1}{N_c} \sum_{i:y_i=c} \max\{0, 1 - t(c) \cdot (w^T x_i + b)\}$
   $c^* \Leftarrow \underset{\forall c \in G^+ \cup G^- \wedge I(c)=0}{\operatorname{argmin}} L(c)$
   if $L(c^*) < \frac{A}{C}$ then
      $I(c^*) \Leftarrow t(c^*)$
   end if
until $L(c^*) \geq \frac{A}{C}$ or $\forall c \in G^+ \cup G^- : I(c) \neq 0$
if $G^+ \neq G^-$ then
   for $\forall c \notin G^+ \cup G^-$ do
      $\forall t \in \{-1, 1\} : L(t) \Leftarrow \frac{1}{N_c} \sum_{i:y_i=c} \max\{0, 1 - t \cdot (w^T x_i + b)\}$
      $t^* \Leftarrow \underset{\forall t \in \{-1,1\}}{\operatorname{argmin}} L(t)$
      if $L(t^*) < \frac{A}{C}$ then
         $I(c) \Leftarrow t^*$
      end if
   end for
end if
return $\{f(x) \Leftarrow \operatorname{sign}(w^T x + b), I\}$
```

Algorithm 6: The algorithm for training a sub-classifier $\{f(x), I\}$.

multi-class classifier by covering all the aspects for multi-class image classification, as defined in Equation (63). The functionality of a sub-classifier f_l can be characterized by an $m \times m$ indicator matrix B_l which is derived from its coding vector I_l:

$$\mathrm{B}_l(u,v) = 1[I_l \cdot I_l^T(u,v) < 0], \forall u, \forall v. \tag{70}$$

The binary value at the position (u,v) in B_l indicates whether the sub-classifier $f + l$ can separate the category c_u from the category c_v.

To train a reliable multi-class classifier, we need to train a minimum number of sub-classifiers which can handle the multi-class image classification task for all the pairs of the relevant categories completely. Since the training process of a sub-classifier f_l is initialed by a pair of seed categories $\{c_l^+, c_l^-\}$, our remaining problem is to select a set of such seed pairs with smallest size which their sub-classifiers can completely handle the multi-class image classification task. Thus the seed selection problem can be modeled as the following matrix decomposition problem:

$$\begin{array}{ll} \min\limits_{d_{ij}} & \sum\limits_{i,j \in \{1 \ldots m\}} d_{ij} \\ \text{s.t.} & \forall u, v \in \{1 \ldots m\} : \sum\limits_{i,j=1 \ldots m} d_{ij} \cdot \mathrm{B}_{ij}(u,v) > 0 \\ & d_{ij} \in \{0,1\}, \forall i, j \in 1 \ldots m \end{array}$$

where B_{ij} is the indicator matrix of the sub-classifier f_{ij} and it is initialed by the seed pair $\{c_i, c_j\}$ and d_{ij} is the variable to indicate whether f_{ij} is selected. Unfortunately, it is not possible to optimize on Equation (71) because the indicator matrix B_{ij} is not available until the sub-classifier f_{ij} is trained. Alternatively, a greedy algorithm is developed by sequentially adding the sub-classifiers until all the pairs of the relevant categories are separated.

Our discriminative learning algorithm prefers to choose the category pair $\{c_u, c_v\}$ with small values of the inter-concept visual similarity contexts $S(\cdot, \cdot)$ as the seed categories. The reason is that discriminating two visually-dissimilar categories from different groups (with less inter-concept visual similarity $S(\cdot, \cdot)$) is a more common task and the corresponding sub-classifier has more possibility to be shared with other tasks, on the other hand, discriminating two visually-similar categories in the same group (with big inter-concept visual similarity $S(\cdot, \cdot)$) is a more strict task and the corresponding sub-classifier has less possibility to be shared with other tasks. For example, we can use the sub-classifier for tiger vs. tree to discriminate lion from tree and the sub-classifier may still get good performance, on the other hand, it could be very difficult to use the sub-classifier for tiger vs. lion to discriminate the visually-dissimilar pairs tiger and tree. Another reason why the visually-dissimilar category pairs get more priority is that the visually-dissimilar categories are easy to be separated and users may have low error tolerance for such easy tasks. For some highly visually-similar categories, it is very important to train an individual sub-classifier to discriminate them specifically. Our multi-class classifier training algorithm is described in Algorithm 7.

Our multi-class classifier training algorithm has the following advantages: (1) Our multi-class classifier tends to minimize the number of sub-classifiers for achieving multi-class classification task, which is very attractive for large-scale image classification. (2) All the sub-classifiers are trained independently without incorporating an inter-concept (inter-classifier) hierarchy. Thus the error of one sub-classifier does

```
Initial: $\mathrm{B} \Leftarrow \mathrm{E}_{m\times m}$, $l \Leftarrow 0$
repeat
  $l \Leftarrow l+1$
  $u^*, v^* \Leftarrow \underset{u,v\in\mathcal{Y}}{\operatorname{argmax}}(1-S(u,v))\cdot 1[B(u,v)=0]$
  $\{f_l, I_l\} \Leftarrow$ Invoke Algorithm 6 to train the sub-classifier initialized by $u$, $v$
  $B_l \Leftarrow 1_{m\times m}[I_l \cdot I_l^T < 0]$
  $\mathrm{B} \Leftarrow \mathrm{B} + B_l$
until $(|\mathrm{B}|_1 = m^2) \vee (l > \text{limit})$
$L = l$
$F(x) \Leftarrow \sum_{l=1}^{L} f_l(x)\cdot I_l$
return $F(x)$
```

Algorithm 7: Our proposed multi-class classification algorithm.

not affect others, which may guarantee the robustness of our multi-class classifier. (3) The inter-concept visual correlations are leveraged to identify the seed groups for classifier training. The pairwise sub-classifiers for the visually-similar categories are trained jointly, which can reduce the computational cost but also increase the discrimination power of the sub-classifiers.

8.4 Algorithm Evaluation and Experimental Results

Our experimental studies are carried on two image sets: VOC 2007 and the ImageNet 1K[107]. VOC 2007 is a benchmark for image classification which contains around 24,000 annotated object images for 20 categories. While the number of categories VOC 2007 is relatively small and it provides a good environment with reliable annotations for sanity check of our algorithm, ImageNet 1k image set contains 1.2 million images for 1,000 object classes which are crawled from Internet. Even not all those 1,000 object classes have atomic semantic meaning (i.e., to be on the leaf nodes of a concept ontology), their semantics are mutually exclusive. In addition, the annotations of the object bounding boxes are provided over a subset of images:

345,685 object images with at least 100 images for each category are annotated in the training set, while the testing set contains 110,627 object images. We have compared our proposed method with other representative methods on both of these image sets, such experiments can be used to compare the performance differences for these algorithms and provide good evidence of the correctness of our proposed algorithm.

As the inter-concept similarity matrix $S(u, v)$ is calculated according to the techniques in Chapter 7.1, we can visualize the inter-concept similarity matrix on a 2-D plane by using Multi-Dimensional Scaling (MDS) in Equation 71.

$$\min_{\{x_1 \ldots x_m\}} \sum_{u,v \in \mathcal{Y}} (1 - ||x_u - x_v|| - S(u, v))^2 \tag{71}$$

Even human eyes cannot find an explicit structure from Fig. (36), it still reflects that some categories are close enough to form the groups of visually-similar object classes. After all the object classes are partitioned into a set of groups, the scatter plot in Fig. (37) clearly shows that the pixels near the main diagram diagonal are lighter than others, which indicates there is a significant gap between the values of the intra-group visual similarity and the values of the inter-group visual similarity.

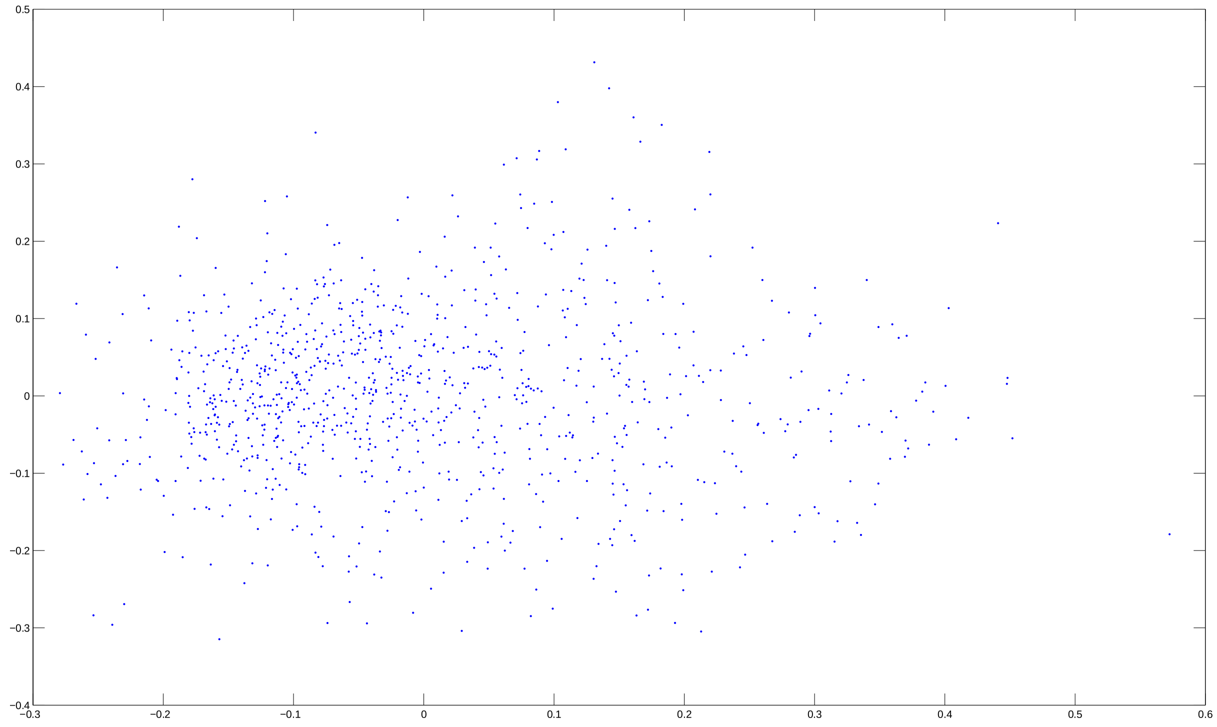


Figure 36: 2-D Plot of all 1,000 categories by performing multi-dimensional scaling over the inter-concept visual similarity matrix.

Table (8) shows some examples of the groups of visually-similar object classes. As we can see, the visually-similar object classes in the same group have good semantic consistency, which demonstrates the good evidence of the correctness and effectiveness of our algorithms for feature extraction and visual concept network construction.

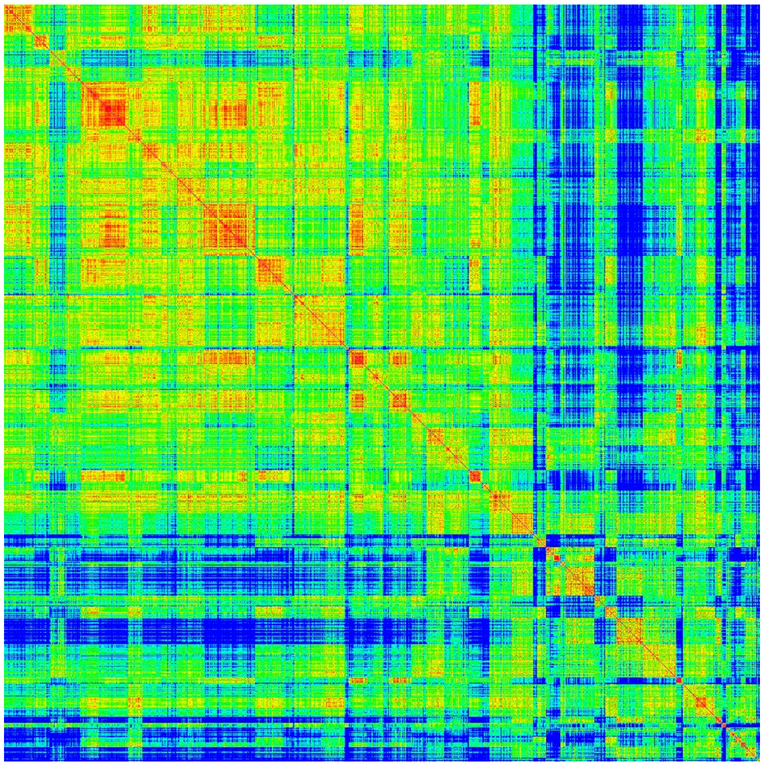
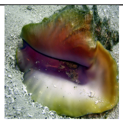
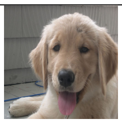
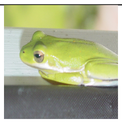
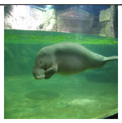
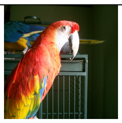
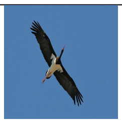
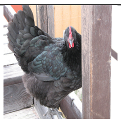
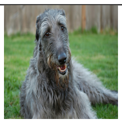


Figure 37: Scatter plot of the inter-concept visual similarity matrix, axes are sorted according to groups.

Table 8: Examples of category groups extracted from ImageNet 1K image set: those categories in the same group share some common or similar visual properties.

						...
Group #0	tench	great white shark	electric ray	stingray	common newt	...
						...
Group #1	goldfish	conch	cowrie	Pekinese	golden retriever	...
						...
Group #2	tiger shark	airship	bathtub	bullet train	collar	...
						...
Group #3	hammerhead	axolotl	tree frog	green snake	dugong	...
						...
Group #4	cock	frilled lizard	macaw	tusker	black stork	...
						...
Group #5	hen	great grey owl	koala	Scottish deerhound	Australian terrier	...

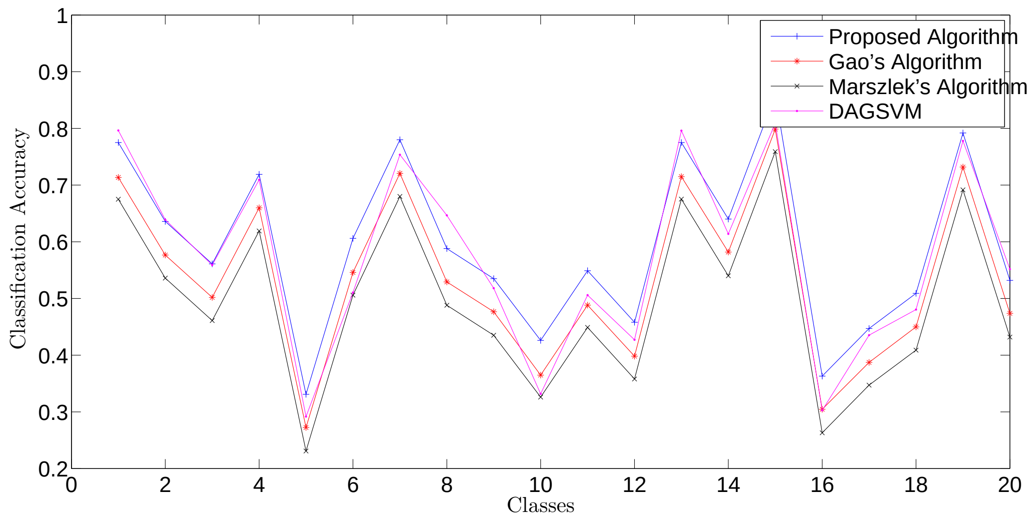


Figure 38: Classification accuracy on VOC 2007 image set for our algorithm, Marszlek's algorithm, Gao's algorithm and DAGSVM.

Our classification performance evaluation starts with a popular benchmark for image classification: VOC 2007 image set. Three representative and well-known approaches for multi-class classification are selected for algorithm comparison: (1) DAGSVM [115]; (2) Marszalek's algorithm [121]; (3) Gao's algorithm [120]. Fig. (38) shows the classification accuracy comparison between our algorithm and three representative algorithms. Our algorithm has reached significant better performance than Gao's and Marszlek's algorithm in term of classification accuracy. Since the number of object classes in VOC 2007 are too small, the inter-concept visual correlations are not significant and thus our algorithm did not achieve a remarkable win against DAGSVM.

Though the VOC 2007 image set is not large enough to show the viability of our algorithm on large scale image classification, the evaluation on VOC 2007 has still provide the evidence of the correctness of our multi-class classification algorithm. To demonstrate the advantage of our algorithm on large scale image set, ImageNet

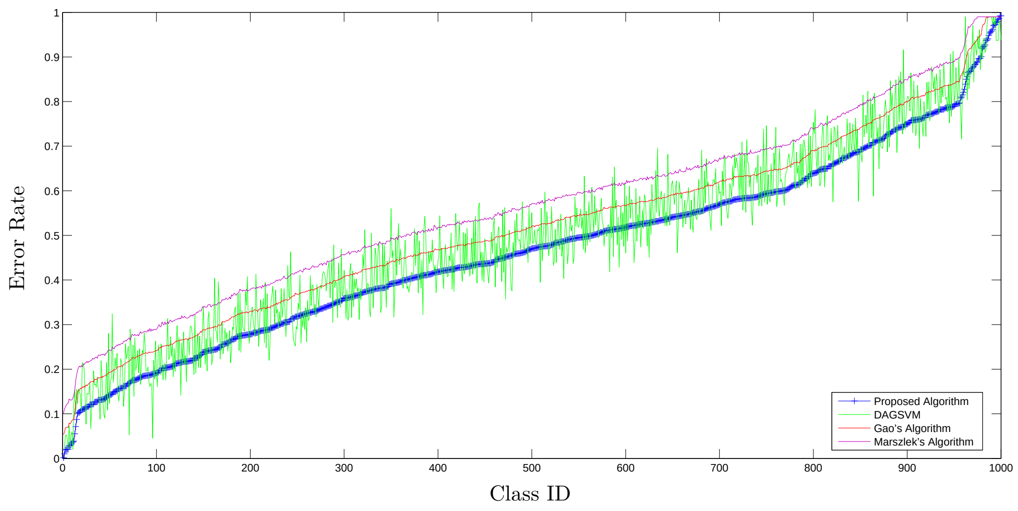


Figure 39: Category-specified performance comparison among our algorithm, DAGSVM, Marszlek's algorithm and Gao's algorithm, in *flat error*: class IDs on x-axis have been sorted by the classification error of our algorithm.

1k can be treated as the an excellent benchmark set. Because ImageNet 1K[107] set is organized according to a concept hierarchy, two types of classification errors are considered in this paper to calculate the classification accuracy for algorithm evaluation: flat error fe and hierarchical error he, as in Equation (72).

$$\begin{aligned} fe &= \frac{1}{n}\sum_{i=1}^{n} 1[f(x_i) \neq y_i] \\ he &= \frac{1}{n}\sum_{i=1}^{n} d(f(x_i), y_i)/(2 \cdot h) \end{aligned} \tag{72}$$

where $d(f(x_i), y_i)$ is the length of short path between the predicted object class $f(x_i)$ (which is predicted by the object classifier $f(x_i)$) and its real object class y_i for the given object image x_i in the concept hierarchy and h is the maximum height of the hierarchy tree.

The performance differences are shown in Fig. (39) for flat error and Fig. (40) in an error histogram manner. As we can see, our proposed method is slightly better

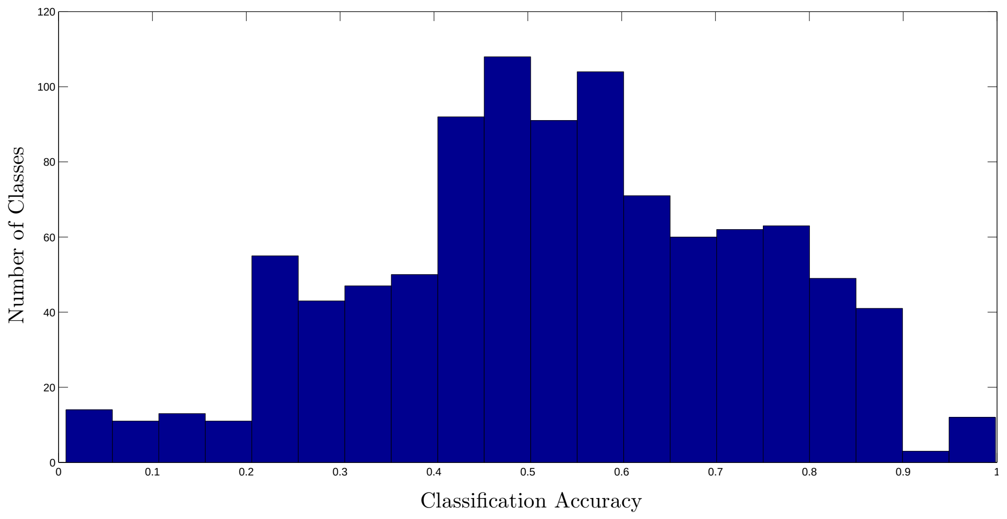


Figure 40: Flat-error distribution of our algorithm over categories: for most categories, the flat error rate of our algorithm is around 50%.

than other three methods on flat error, while DAGSVM's performance is just about the same as ours. If we take the semantic distance between the object classes into account (i.e., hierarchical errors), our approach has gained a significant improvement over all other three methods. Furthermore, the inter-concept visual similarity is used in our visual concept network rather than the inter-concept semantic relationship, thus it is not fair to employ the semantic inter-concept distance measures to penalize the degree of misclassification in our visual concept network. Thus a local hierarchical error lhe is defined, which inherits from he:

$$lhe = \frac{1}{n}\sum_{i=1}^{n}\left(1 - S(f(x_i), y_i)\right). \tag{73}$$

Table (9) clearly shows that our proposed method can reach a significant increase on the performance in term of hierarchical error. Both he and lhe have proved that our theoretical analysis in Section 8.3 is correct.

We have also discovered how the number of sub-classifiers affects the classification

Table 9: Overall classification performance comparison in both flat error and local hierarchical error among four algorithms: our algorithm, DAGSVM, Marszlek's algorithm and Gao's algorithm.

	Flat Error	Local Hierarchical Error
The proposed algorithm	0.4675	0.1837
DAGSVM	0.4956	0.3386
Marszlek's algorithm	0.5660	0.3743
Gao's algorithm	0.5171	0.3605

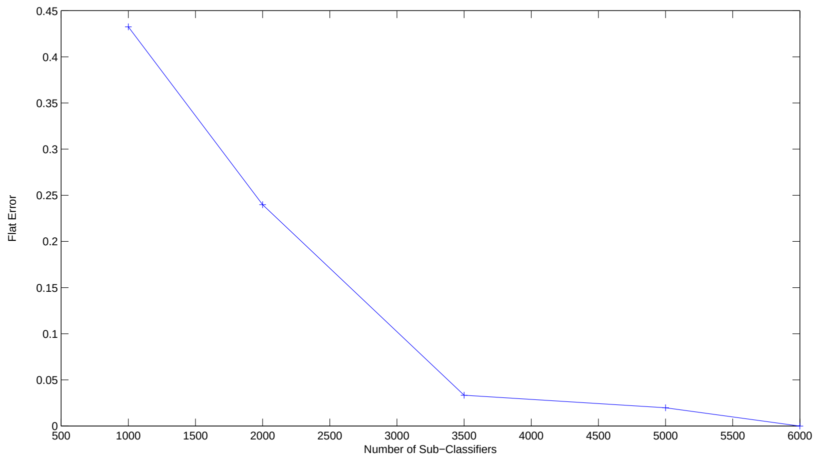


Figure 41: Number of sub-classifiers versus Performance.

performance. As our algorithm finally stopped at 6,000 sub-classifiers (i.e., a pre-set limitation), we made a few stop points at 1000, 2000, 3500, 5000 sub-classifiers and evaluate these existing sub-classifiers. Fig. (41) presents the difference between those incomplete sub-classifiers and the final classifier on their error rates. There is actually very slight difference between 3,500 sub-classifiers and 6,000 ones, which has proved that our assumption is correct (i.e., local intra-group sub-classifiers contribute less than global inter-group sub-classifiers).

8.5 Conclusion

A novel discriminative learning algorithm is developed in this chapter for achieving large-scale image classification (e.g., learning a large number of classifiers to categorize large-scale images into a large number of object classes and image concepts). A visual concept network is constructed for: (a) determining the groups of visually-similar object classes and image concepts and identifying the seed categories; (b) detecting the inter-related learning tasks; and (c) estimating the learning complexity for classifier training, e.g., it is much hard to discriminate the visually-similar object classes and image concepts in the same group than discriminating the visually-dissimilar object classes and image concepts in different groups. Our large-scale discriminative learning algorithm can dramatically reduce the computational cost for large-scale classifier training and significantly enhance their discrimination power. Our experiments on large-scale image set (ImageNet [107]) have obtained very competitive results.

CHAPTER 9: CONCLUSIONS

In this dissertation, a structural learning framework for large-scale image classification has been proposed to leverage large amounts of loosely tagged images for training a large number of inter-related classifiers. To ensemble web images for computer vision tasks, we have concluded the following three issues: (1) junk tags; (2) loosely tags; (3) missing tags. Some accomplished key components of the proposed framework have been introduced to respectively address different issues: (a) a junk image filtering algorithm has been developed to handle spam tagging issue; (b) a cluster-based tag-instance alignment approach has been conducted to assign image level tags into regions; (c) missing object tags are discovered by a missing tag prediction method; (d) object correlation network is built to characterize inter-concept correlations; (e) for large scale problems, a multi-task structured learning algorithm has been enforced, which constructed the multi-class classifier by joint-trained binary sub-classifier.

9.1 Junk Image Filtering for Image Cleansing

To leverage online images to enrich the training images for supervised learning, we have developed a junk image filtering algorithm to cleanse the spam user tags from large-scale collaboratively-tagged social images, which takes three major steps:

1. The images tagged by one particular tag have been gathered and separated into

multiple groups by an efficient clustering algorithm.

2. For each group, the one-class SVM algorithm has been enforced to discard the outlier images, which ensures the remaining images in the group are visually consistent.

3. The correlation between groups has been analyzed to determine the positive groups with high precision but low recall. The images from the negative groups have been removed.

It has been proved and observed that the remaining images must be visually similar to the majority of the society of images labeled by the particular tag, where the improper label-image relationships have been removed.

9.2 Multi-Instance Learning with Missing Tags

Our multi-instance learning algorithm consists of two parts, tag instance alignment and missing object prediction, have dealt with two issues respectively: loosely tag and missing tag.

For the tag instance alignment, we further segmented each image into a number of regions and grouped the millions of regions into homogeneous instance clusters by our efficient clustering algorithm. Our clustering algorithm takes advantage of both GPU acceleration from CUDA techniques and high performances inherited from the affinity propagation algorithm. The dominant label of each group is extracted as the representative tag of the homogeneous instance cluster, taking account of both in-group election and inter-group correlation, and finally be transferred to instance tags.

For missing tag prediction, we have concluded the sources of missing tags into four cases, by the coherence of the unlabeled instance with the labeled homogeneous instance clusters. For each instance which tag is missing, its nearby labeled instance clustered determines whether and which it should be labeled.

By our tag instance alignment algorithm and missing object prediction scheme, the following two goals have been accomplished: (1) The labels at the image level have been transferred to the instance tags which are compatible to most traditional machine learning techniques; (2) The labeling noises by user's tagging preferences have been highly diminished due to the cleansing of improper tags.

9.3 Network-oriented Multi-task Structured Learning Algorithm

We have also proposed a network-based structured learning approach for large scale image classification, taking advantage of the joint learning for correlated classes and the network-guided multi-class classifier.

The most important part of our approach is the object correlation network. The correlation between the object categories have been analyzed by only the visual coherence of the representative images from each image category, but also the co-occurrence of object tags. With the object correlation network, the structure of large scale object categories have visually and quantitatively exposed, which guided the further classifier training process.

From the pair-wise relationships of the objects on the the object correlation network, the classifiers for highly-correlated categories have been trained jointly by our proposed multi-task supervised learning algorithm derived from support vector machine. Since the generalization power of the discriminative models are task-driven and determined by the correlation between two sets of samples, object classifiers among the visually-correlated categories are strongly inter-related and should be trained jointly rather than independently.

To deal with large scale image classification with thousands of categories, a multi-class classification scheme is proposed, which ensembles a group of binary classifiers with a structured hierarchy guided by the object correlation network. Despite the influence from the object correlation network, the structure hierarchy has been determined but the discriminant power of actual binary sub-classifiers, which guarantees the correctness and efficiency of our multi-class classifier to be capable for large scale problems with thousands of categories.

9.4 Future Work

From the perspective of applications, our proposed algorithm can be potentially utilized in the follow tasks:

1. The social media website can take advantage of our algorithm to generate machine tags for user-uploaded photos and return the recommended machine tags to users to saving the cost of user tagging.

2. For present image search engines, our algorithm can provides a machine label for the query images from users as well as indexes in the database, which tends to produce more accurate results and reduce the response time.

However, the principal issue which prevents our system from the capability of real-time image classification is the efficiency. Despite the limitation of our accessible computational power, the multi-step design of our framework naturally increases the response time of our system. In our framework, the time-consuming parts for input query prediction are image segmentation, feature extraction and multi-classifier response computation.

In modern computing theory, it is important to reduce the dependency among time-consuming components for taking advantage of parallel computing. One of potential solution to reduce the response time of our system is to embed a coarse-to-fine image feature scheme into our framework, in which coarse but efficient features are used in the high-level classifiers and accurate but slow features are taken by the fine level classifiers as the opposite. Thereby the extraction of finer features and high-level classifier responses can be computed simultaneously.

Besides, another issue of our system is that image segmentation has cost too much computation. As a matter of fact, for natural photos uploaded by users, attention maps seem to be a better candidate to locate objects than general purpose unsupervised image segmentation, for the following reasons: (1) it is not necessary to classify objects or scenes which users are not interested in; (2) the interesting objects in a real photo are far less than its segmentation regions; (3) general purpose image segmentation on real photos is not as reliable as it is on benchmark image sets. (4) the cost of attention map generation is far less than which image segmentation costs. For the above four reasons, it looks attractive to embed user attention detection as a viable replacement for image segmentation.

However, thanks to the recent breakthrough on deep learning network, both the accuracy and efficiency of large-scale image classification can be potentially improved.

It is also worth noting that in our proposed framework, the computation of the components such as image segmentation, multiple feature extraction are coupled, e.g., some lower-level computation such as pixel gradient calculation may have been done multiple times in feature extraction as well as in image segmentation. In contrast, if we integrate deep learning network into our framework, attention map detection, feature extraction, inter-category visual correlation and classifier outputs can be unified in the same network. Since multi-layer deep learning network sensors semantics from bottom levels to top levels and receives decayed supervision from top levels to bottom levels, we can potentially design our framework as following: (1) The bottom level is the sensor level, which observes pixel intensities with multi-resolution bounding boxes and clusters similar boxes into groups via self-organized maps. (2) The second level calculates the attention map from the outputs of bottom levels; (3) The third level is the soft-max level, which functions similar to codeword feature extraction to quantize outputs from lower level into codewords; (4) for the upper levels but not top level, soft-max level design for classical deep learning network are inherited to unsupervised group similar contents from low-level edges to high-level objects; (5) In contrast to the classification level in classical deep learning network, in our potential design, the top level is replaced by our structured learning framework; Thus, our framework can potentially co-operate with the deep learning networks, which unifies the image representation and discovery of inter-category visual correlations, which tends to get multi-level image representation compatible to both fast inter-category distinction and fine-grinded classification.

REFERENCES

[1] J. Li, J.Z. Wang, "Real-Time Computerized Annotation", ACM Multimedia 2006.

[2] M.S. Lew, N. Sebe, C. Djeraba, R. Jain, "Content-based multimedia information retrieval: State of the art and challenges", *ACM Trans. on Multimedia Computing, Communications, and Applications* (TOMCCAP), vol. 2, no.1, pp.1-19, 2006.

[3] J. Luo, A.E. Savakis, A. Singhal, "A Bayesian network-based framework for semantic image understanding", *Pattern Recognition*, vol. 38, no.6, pp.919-934, 2005.

[4] J. Huang, S.R. Kumar and R. Zabih, "An automatic hierarchical image classification scheme", ACM Multimedia, Bristol, UK, 1998.

[5] N. Vasconcelos, "Image indexing with mixture hierarchies", IEEE CVPR, 2001.

[6] J. Li and J. Z. Wang, "Automatic linguistic indexing of pictures by a statistical modeling approach.", *IEEE Trans. on PAMI*, vol. 25, no. 9, pp. 1075-1088, 2003.

[7] J. Fan, Y. Gao, H. Luo, R. Jain, "Mining multi-level image semantics via hierarchical classification", *IEEE Trans. on Multimedia*, vol. 10, no.1, 2008.

[8] J. Fan, Y. Gao, H. Luo, "Integrating concept ontology and multitask learning to achieve more effective classifier training for multilevel image annotation", *IEEE Trans. Image Processing*, vol.17, pp.407-426, 2008.

[9] L. Fei-Fei, P. Perona, "A Bayesian hierarchical model for learning natural scene categories", IEEE CVPR, 2005.

[10] K. Barnard and D. Forsyth, "Learning the semantics of words and pictures", IEEE ICCV, pp.408-415, 2001.

[11] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora and S. Belongie, "Object in Contexts", *International Conference on Computer Vision*, 2007

[12] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta and R. Jain, "Content-based image retrieval at the end of the early years", *IEEE Trans. on PAMI*, 2000.

[13] Y. Rui, T. S. Huang, and S.-F. Chang, "Image retrieval: Current techniques, promising directions and open issues", *Journal of Visual Communication and Image Representation*, Vol. 10, pp.39-62, 1999.

[14] R. Datta, D. Joshi, J. Li, J. Wang, “Image retrieval: Ideas, influences, and trends of the new age”, *ACM Comput. Surv.*, vol. 40, no.2, 2008.

[15] R. Fergus, L. Fei-Fei, P. Perona, A. Zisserman, “Learning object categories from Google’s image search”, CVPR, 2006.

[16] T. Berg, A. Berg, J. Edwards, M. Mair, R. White, Y. Yeh, E. Learned-Miller, D. Forsyth, “Names and faces in the news”, CVPR, 2004.

[17] F. Schroff, A. Criminisi, A. Zisserman, “Harvesting image databases from the web”, ICCV, 2007.

[18] A. Quattoni, M. Collins, T. Darrell, “Learning visual representations using images with captions”, CVPR 2007.

[19] N. Ben-Haim, B. Babenko, S. Belongie, “Improving image search via content based clustering”, CVPR SLAM, 2006.

[20] D. Cai, X. He, Z. Li, W.-Y. Ma, J.-R. Wen, “Hierarchical clustering of WWW image search results using visual, textual, and link information”, ACM Multimedia, 2004.

[21] J. Fan, Y. Shen, N. Zhou, Y. Gao, “Harvesting large-scale weakly-tagged image databases from the Web”, IEEE CVPR, 2010.

[22] Y. Deng, B.S. Manjunath, “Color image segmentation”, IEEE CVPR, 1999.

[23] J Shi, J Malik, “Normalized cuts and image segmentation”, *IEEE Trans. on PAMI*, 2000.

[24] B. Russell, A. Efros, J. Sivic, W. Freeman, A. Zisserman, “Using multiple segmentations to discover objects and their extent in image collections”, IEEE CVPR, 2006.

[25] B. Russell, A. Torralba, K.P. Murphy, W.T. Freeman, “LabelMe: a database and web-based tool for image annotation”, *Intl. Journal of Computer Vision*, vol.77, no.1, 2008.

[26] G. Griffin, A. Holub, P. Perona, The Caltech-256, Caltech Technical Report.

[27] L. Fei-Fei, R. Fergus, P. Perona, “Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories”, CVPR, 2004.

[28] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database”, CVPR, 2009.

[29] Flickr, www.flickr.com

[30] L. von Ahn, L. Dabbish, "Labeling images with a computer game", ACM CHI, 2004.

[31] B.J. Frey, D. Dueck, "Clustering by Passing Messages Between Data Points", *Science*, 2007

[32] S. Vijayanarasimhan, K. Grauman, "Keywords to visual categories: Multiple-instance learning for weakly supervised object categorization", CVPR 2008.

[33] S. Vijayanarasimhan, K. Grauman, "What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations", CVPR 2009.

[34] C. Galleguillos, B. Babenko, A. Rabinovich, S.J. Belongie, "Weakly supervised object localization with stable segmentations", ECCV , pp.193-207, 2008.

[35] T. Cour, B. Sapp, C. Jordan, B. Taskar, "Learning from ambiguously labeled images", CVPR, 2009.

[36] C.R. Rosenberg, M. Hebert, "Training object detection models with weakly labeled data", BMVC 2002.

[37] U. Syed, B. Taskar, "Semi-supervised learning with adversarially missing label information", NIPS, 2010.

[38] Q. Zhang, W. Yu, S. A. Goldman, J. E. Fritts, "Content-based image retrieval using multiple-instance learning", ICML, 2002.

[39] O. Maron, A. L. Ratan, "Multiple-instance learning for natural scene classification", ICML, 1998.

[40] Y. Chen, J. Bi, J. Z. Wang, "MILES: multiple instance learning via embedded instance selection", *IEEE Trans. PAMI*, vol.28, no.12, pp.1931-1947, 2006.

[41] P. Viola, J.C. Platt, C. Zhang, "Multiple instance boosting for object detection", ICML, 2006.

[42] J. Tang, X. Hua, M. Wang, Z. Gu, G. Qi, X. Wu, "Correlative linear neighborhood propagation for video annotation", *IEEE Trans. on SMC*, vol. 39, no.2, pp.409-416, 2009.

[43] G.-J. Qi, X.-S. Hua, Y. Rui, J. Tang, T. Mei, H.-J. Zhang, "Correlative multi-label video annotation", ACM Multimedia, pp.17-26, 2007.

[44] Z. Zha, X.-S. Hua, T. Mei, J. Wang, G.-J. Qi, Z. Wang, "Joint multi-label multi-instance learning for image classification", CVPR, 2008.

[45] Z.-H. Zhu, M.-L. Zhang, "Multi-instance multi-label learning with application to scene classification", NIPS, 2006.

[46] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, S. Belongie, "Objects in context", ICCV, 2007.

[47] S. Kumar, M. Hebert, "Discriminative random fields", *Intl. Journal of Computer Vision*, 2006.

[48] C. Galleguillos, A. Rabinovich, S. Belongie, "Object categorization using co-occurrence, location and appearance", CVPR, 2008.

[49] W. Jiang, S.-F. Chang, A. Loui, "Context-based concept fusion with boosted conditional random fields", IEEE ICASSP, 2007.

[50] J. Dean, S. Ghemawat, "MapReduce: Simplified data processing on large clusters", OSDI'04, 2004.

[51] J. Lafferty, A. McCallum, F. Pereira, "Conditional random field: Probabilistic models for segmenting and labeling sequence data", Proc. ICML, 2001.

[52] T. Joachims, T. Finley, C. Yu, "Cutting-plane training of structural SVMs", *Machine Learning*, vol. 77, no.1, pp.27-59, 2009.

[53] M. Blaschko, C. Lampert, "Learning to localize objects with structured output regression", ECCV, LNCS 5302, pp.2-15, 2008.

[54] A. Torralba, K. P. Murphy, W. T. Freeman, "Sharing features: efficient boosting procedures for multi-class object detection", IEEE CVPR, 2004.

[55] J. Fan, Y. Gao, H. Luo, R. Jain, "Mining multilevel image semantics via hierarchical classification", *IEEE Trans. on Multimedia*, vol. 10, no.2, 2008.

[56] J. Fan, Y. Gao, H. Luo, ""Integrating concept ontology and multi-task learning to achieve more effective classifier training for multi-level image annotation", *IEEE Trans. on Image Processing*, vol. 17, no.3, pp.407-426, 2008.

[57] T. Evgeniou, C.A. Micchelli, M. Pontil, "Learning multiple tasks with kernel methods", *Journal of Machine Learning Research*, vol.6, pp.615-637, 2005.

[58] J. Yang, Y. Liu, E. X. Ping, A.G. Hauptmann, "Harmonium models for semantic video representation and classification", SIAM Conf. on Data Mining, 2007.

[59] M.-Y. Chen, A.G. Hauptmann, "Discriminative fields for modeling semantic concepts in video", RIAO Large-Scale Semantic Access to Content, May 30-June 1, 2007.

[60] Novak, C.L.; Shafer, S.A. "Anatomy of a color histogram". *Computer Vision and Pattern Recognition*, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on 1518 June 1992 Page(s):599 - 605

[61] YuH.,LiM., ZhangH. andFengJ.“Colortexturemomentforcontentbasedimageretrieval”. *Proc. IEEE Intl Conf. onImageProcessing.* September,2002

[62] P. Howarth and S. Ruger, “Evaluation of Texture Features for Content-Based Image Retrieval” . *Proceedings of the International Conference on Image and Video Retrieval*, 2004.

[63] H. Tamura, S. Mori, and T. Yamawaki. “Texture features corresponding to visual perception”. *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-8, no. 6, 1978, 460 - 473.

[64] V. Castelli and L. D. Bergman (Eds.). “Image Databases: Search and Retrieval of Digital Imagery”. Wiley: New York, 2002.

[65] A. Oliva, A. Torralba. “Modeling the shape of the scene: a holistic representation of the spatial envelope”. *International Journal of Computer Vision*, Vol. 42(3): 145-175, 2001.

[66] D. G. Lowe, “Distinctive Image Features from Scale-Invariant Keypoints”. *International Journal of Computer Vision*, 2004.

[67] Y. Ke and R. Sukthankar, “PCA-SIFT: A More Distinctive Representation for Local Image Descriptors”, *Computer Vision and Pattern Recognition (CVPR)*, 2004.

[68] H. Bay, T. Tuytelaars and L.V. Gool, “SURF: Speeded Up Robust Features”, *Proceedings of the ninth European Conference on Computer Vision (ECCV)*, 2006.

[69] S.N. Sinha, J-M. Frahm, M. Pollefeys and Y. Genc, “GPU-Based Video Feature Tracking and Matching”, *EDGE 2006, workshop on Edge Computing Using New Commodity Architectures*, 2006

[70] E. Rublee, V. Rabaud, K. Konolige, G. Bradski. “ORB: an efficient alternative to SIFT or SURF”. *Computer Vision* (ICCV), 2011 IEEE International Conference on (pp. 2564-2571). IEEE.

[71] E. Rosten and T. Drummond. “Machine learning for highspeed corner detection”. *European Conference on Computer Vision*, volume 1, 2006.

[72] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. *Brief: Binary robust independent elementary features. European Conference on Computer Vision*, 2010.

[73] T. Ojala, M. Pietikainen, and D. Harwood, “A Comparative Study of Texture Measures with Classification Based on Feature Distributions”, *Pattern Recognition*, vol. 29, pp. 51-59

[74] P. Viola, M.J. Jones. “Rapid Object Detection using a Boosted Cascade of Simple Features”. *IEEE CVPR*, 2001.

[75] X. Wang, T.X. Han, S. Yan, "An HOG-LBP Human Detector with Partial Occlusion Handling," *IEEE International Conference on Computer Vision*, Kyoto, 2009

[76] B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan, and A. Yamada. "Color and texture descriptors". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, 2001, 703 - 715

[77] T. Sikora. "The MPEG-7 visual standard for content description - an overview". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, 2001, 696 - 702

[78] J. M. Martinez, Ed. "MPEG-7 Overview". ISO/IEC JTC1/SC29/WG11 No. 6828 (2004). On-line. http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm

[79] L. Fei-Fei, R. Fergus, and A. Torralba. "Recognizing and Learning Object Categories", *CVPR* 2007

[80] M. Aharon, M. Elad, and A. Bruckstein. "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation". *Signal Processing, IEEE Transactions on*, 54(11):4311 4322, 2006

[81] Q. Tian, S. Zhang, W. Zhou, R. Ji, B. Ni, N. Sebe, "Building descriptive and discriminative visual codebook for large-scale image applications", *Multimedia Tools and Applications*, vol.51, no.2, pp.441-477, 2011.

[82] K. Grauman and T. Darrell, "The Pyramid Matching Kernel: Discriminative Classification with Sets of Image Features", *IEEE International Conference on Computer Vision (ICCV)*, 2006

[83] Lazebnik, Svetlana, Cordelia Schmid, and Jean Ponce. "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories." *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on.* Vol. 2. IEEE, 2006.

[84] J. Yang, K. Yu, Y. Gong, T.S. Huang. "Linear spatial pyramid matching using sparse coding for image classification". *CVPR*, 2009.

[85] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong. "Locality-constrained Linear Coding for Image Classification". *CVPR*, 2010.

[86] H. Lee, A. Battle., R. Raina, A. Y. Ng. "Efficient sparse coding algorithm", *Annual Conference on Neural Information Processing Systems (NIPS)*, 2006.

[87] R. Raina, A. Battle, H. Lee, B. Packer, A.Y. Ng. "Self-taught learning: transfer learning from unlabeled data". *the 24th International Conference on Machine learning*, 2007.

[88] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, Y. Ma, “Robust face recognition via sparse representation”, *IEEE Trans. PAMI*, 2009.

[89] X. Yuan, S. Yan, “Visual classification with multi-task joint sparse representation”, IEEE CVPR, pp.3493-3500, 2010.

[90] Y. Bengio, A. Courville, P. Vincent, “ Representation Learning: A Review and New Perspectives”, *Arvix*, 2012.

[91] A. Krizhevsky, I. Sutskever, G.E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks”, *NIPS*, 2012.

[92] J. Fan, D. Keim, Y. Gao, H. Luo, Z. Li, “JustClick: Personalized image recommendation via exploratory search from large-scale Flickr images”, *IEEE Trans. on CSVT*, vol. 19, no.2, pp.273-288, 2009.

[93] I.E. Givoni and B.J. Frey, “A Binary Variable Model for Affinity Propagation”, *Neural Computation*, Vol 21 (2009).

[94] I.E Givoni, C. Chung and B.J. Frey, “Hierarchical Affinity Propagation”, *UAI* 2011

[95] Y. Jia, J. Wang, C. Zhang and X.S. Hua, “Finding Image Exemplars Using Fast Sparse Affinity Propagation”, *ACM Multimedia* 2008

[96] Y. Fujiwara, G.Irie and T. Kitahara, “Fast Algorithm for Affinity Propagation”, *Internation Joint Conference on Artificial Intelligence* 2011

[97] W. Wang, H. Zhang, F. Wu and Y. Zhuang, “Large-Scale of E-Learning Resources Clustering with Parallel Affinity Propagation”, *ICHL* 2008

[98] J. C. Hart and J. D. Hall, “GPU Acceleration of Iterative Clustering”, Manuscript accompanying poster at GP^2: The ACM Workshop on General Purpose Computing on Graphics Processors, and SIGGRAPH 2004 poster

[99] V. Garcia, E Debreuve, F Nielsen and M. Barland, “K-Nearest Neighbor Search: Fast GPU Based Implementations and Applications to High-Dimensional Feature Matching”, *IEEE Internationcal Conference on Image Processing* 2010

[100] CUDA, “http://www.nvidia.com/object/cuda_home_new.html”

[101] M.Hussein and W. Abd-Almageed, “Efficient Band Approximation of Gram Matrices for Large Scale Kernel Methods on GPUs”, *ACM/IEEE Conference on High Performance Computing, SC* 2009

[102] Y. Pei and O. Zaiane, “A Synthetic Data Generator for CLustering and Outlier Analysis”

[103] T.G. Dietterich, R.H. Lathrop and T. Lozano-Perez, "Solving the multiple instance problem with axis-parallel rectangles", *Artificial Intelligence* 1997, 89 (12): 3171

[104] S. Andrews, I. Tsochantaridis and T. Hofmann, " Support Vector Machines for Multiple-Instance Learning", *NIPS 2002*

[105] J.T. Kwok and P-M Cheung, "Marginalized Multi-Instance Kernels", *IJCAI 2007*

[106] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. "SUN Database: Large-scale Scene Recognition from Abbey to Zoo", *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[107] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and Fei-Fei Li, "ImageNet: A Large-Scale Hierarchical Image Database.", *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2009.

[108] J. Deng, A. C. Berg, K. Li and Fei-Fei Li, "What does classifying more than 10,000 image categories tell us?", *European Conference on Computer Vision(ECCV)*, 2010.

[109] N. Ghamrawi, A. McCallum, "Collective Multi-Label Classification", *Conference on Information and Knowledge Management (CIKM 2012)*

[110] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, S. Belongie, "Object in Context", *International Conference on Computer Vision (ICCV)*, 2007

[111] C. Hsu, C. Lin, "A Comparison of Methods for Multi-class Support Vector Machines", *IEEE Transactions on Neural Networks*, 13(2):415-425, 2002a

[112] C. J.C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery.*

[113] V. Vapnik. "Statistical Learning Theory", 1998

[114] K. Crammer and Y. Singer, "On the Algorithmic Implementation of Multi-class Kernel-based Vector Machines", *Journal of Machine Learning Research*, 2(2001), 265-292

[115] J. C. Platt, N. Cristianini, J. Talor, "Large Margin DAGs for Multiclass Classification", *Advances in Neural Information Processing Systems*, 547-553, 2000

[116] S. Knerr, L. Personnaz, and G. Dreyfus, "Single-layer learning revisited: a stepwise procedure for building and training a neural network", In *J. Fogelman, editor, Neu- rocomputing: Algorithms, Architectures and Applications*, Springer-Verlag, 1990.

[117] C. Chang and C. Lin, "LIBSVM : a library for support vector machines", *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011

[118] T. G. Dietterich and G. Bakiri, “Solving multiclass learning problems via error-correcting output codes”, *Journal of Articial Intelligence Research*, 2 (1995), 263286.

[119] E. Allwein, R. E. Schapire, Y. Singer, “Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers”, *Journal of Machine Learning Research* 1 (2000) 113-141

[120] T. Gao and D. Koller, “Discriminative Learning of Relaxed Hierarchy for Large-scale Visual Recognition”, *International Conference on Computer Vision (ICCV)*, 2011

[121] M. Marszalek and C. Schmid, “Constructing Category Hierarchies for Visual Recognition”, *European Conference on Computer Vision (ECCV)*, 2008

[122] V. Vural and J. G. Dy, “A hierarchical method for multi-class support vector machines”, *International Conference on Machine Learning (ICML)*, 2004.

[123] S. Liu, H. Yi, L. Chia, and R. Deepu, “Adaptive hierarchical multi-class svm classier for texture-based image classication”, *IEEE International Conference on Multimedia and Expo (ICME)*, 2005.

[124] G. Grifn and P. Perona, “Learning and using taxonomies for fast visual categorization”, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[125] S. Bengio, J. Weston, and D. Grangier, “Label embedding trees for large multiclass task”, *Annual Conference on Neural Information Processing Systems (NIPS)*, 2010.

[126] Z. Liu, W. Shi, Q. Qin, X. Li and D. Xie, ”Hierarchical Support Vector Machines”, *Geoscience and Remote Sensing Symposium (IGARSS)*, 2005.

[127] S. Xia, J. Li, L. Xia, and C. Ju, “Tree-structured support vector machines for multi-class classication”, *International Symposium on Neural Networks (ISNN)*, 2007

[128] R. Cilibrasi and P Vitanyi, “The Google Similarity Distance”, *IEEE Trans. on Knowledge and Data Engineering*, 19:3(2007), 370383.

[129] George A. Miller, “WordNet: A Lexical Database for English”, *Communications of the ACM*, Vol. 38, No. 11(1995): 39-41.

[130] B. Scholkopf, J. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. “Estimating the support of a high-dimensional distribution”. *Neural Computation*, 13, 2001, 1443-1471.

[131] B. J. Frey, D. Dueck, “Clustering by Passing Messages Between Data Points”, *Science*, 2007

[132] I. Tsochantaridis, T. Hofmann, T. Joachims and Y. Altun. "Support Vector Machine Learning for Interdependent and Structured Output Spaces". *International Conference on Machine Learning (ICML)*, 2004.

[133] K. Mikolajczyk, B. Leibe and B. Schiele, "Local Features for Object Class Recognition", *International Conference on Computer Vision (ICCV)*, 2005

[134] A. Torralba, K. P. Murphy and W. T. Freeman, "Sharing features: efficient boosting procedures for multi-class object detection", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[135] A. Quattoni, M. Collins and T. Darrell. "Conditional Random Fields for Object Recognition". *Annual Conference on Neural Information Processing Systems (NIPS)*, 2004.

[136] Y. Xiang, X. Zhou, Z. Liu, T. S. Chua, C. W. Ngo. "Semantic Context Modeling with Maximal Margin Conditional Random Fields for Automatic Image Annotation". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[137] J. Weston, S. Bengio, N. Usunier, "WSABIE: scaling up to large vocabulary image annotation", *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence*, 2011.

PUBLICATIONS

[1] Y. Shen, J. Fan, "Discriminative Learning for Image Classification", Submitted to *Pattern Recognition.*

[2] Y. Shen, J. Fan, "Multi-Label Multi-Instance Learning with Missing Object Tags", *Multimedia Systems.*

[3] N. Zhou, Y. Shen, J. Peng, J. Fan, "Learning Inter-related Visual Dictionary for Object Recognition", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12).*

[4] J. Fan, Y. Shen, N. Zhou and Y. Gao,"Harvesting Large-Scale Weakly-Tagged Image Databases from the Web", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10).*

[5] Y. Shen, J. Peng, X. Feng and J. Fan, "Multiple Instance Learning with Missing Object Tags", *ICIMCS 2011: 9-12* (best paper candidate)

[6] N. Zhou, Y. Shen, J. Peng, X. Feng, J. Fan, "Leveraging auxiliary text terms for automatic image annotation", *WWW 2011: 175-176*

[7] Y. Shen and J. Fan, "Leveraging Loosely-Tagged Images and Inter-Object Correlations for Tag Recommendation", *ACM Multimedia 2010.*

[8] F. Xie, Y. Shen, X. He, "K-way min-max cut for image clustering and junk images filtering from Google images", *ACM Multimedia 2010: 803-806.*

[9] Y. Shen, C. Yang, Y. Gao and J. Fan, "Harvesting weakly tagged images for computer vision tasks", in *Proc. SPIE*, Vol. 7450, 75400Z(2010).

[10] J. Fan, Y. Shen, C. Yang and N. Zhou, "Structured Max-Margin Learning for Inter-Related Classifier Training and Multi-Label Image Annotation", *IEEE Trans. Image Processing, 2010.*

[11] Y. Zhang, C. Yang, Y. Shen, Y. Gao, J. Fan, H. Luo, "A Cross-Modal Approach to Cleansing Weakly Tagged Images", *IEEE MultiMedia 17(4): 18-25 (2010).*

[12] Y. Shen, J. Fan, "Multi-task multi-label multiple instance learning", *Journal of Zhejiang University - Science C 11(11): 860-871 (2010).*

[13] J. Fan, H. Luo, Y. Shen, C. Yang, "Integrating Visual and Semantic Contexts for Topic Network Generation and Word Sense Disambiguation", *ACM Conf. on Image and Video Retrieval (CIVR'09).*

[14] J. Weng, Y. Shen, M. Chi, X. Xue, "Temporal context as cortical spatial codes", *IJCNN 2009: 3348-3355.*