Prajna: Towards Recognizing Whatever You Want From Images without Image Labeling

Xian-Sheng Hua, Jin Li
Microsoft Research, One Microsoft Way, Redmond, WA 98052, USA
{xshua; jinl}@microsoft.com

Abstract

With the advances in distributed computation, machine learning and deep neural networks, we enter into an era that it is possible to build a real world image recognition system. There are three essential components to build a real-world image recognition system: 1) creating representative features, 2) designing powerful learning approaches, and 3) identifying massive training data. While extensive researches have been done on the first two aspects, much less attention has been paid on the third. In this paper, we present an end-to-end Web knowledge discovery system, Prajna. Starting from an arbitrary set of entities as inputs, Prajna automatically crawls images from multiple sources, identifies images that have reliably labeled, trains models and build a recognition system that is capable of recognizing any new images of the entity set. Due to the high cost of manual data labeling, leveraging the massive yet noisy data on the Internet is a natural idea, but the practical engineering aspect is highly challenging. Prajna focuses on separating reliable training data from extensive noisy data, which is a key to the capability of extending an image recognition system to support arbitrary entities. In this paper, we will analyze the intrinsic characteristics of Internet image data, and find ways to mine accurate and informative information from those data to build a training set, which is then used to train image recognition models. Prajna is capable of automatically building an image recognition system for those entities as long as we can collect sufficient number of images of the entities on the Web.

Introduction

Building a system that can recognize *what*, *who*, and *where* from arbitrary images has motivated researchers in computer vision, multimedia and machine learning areas for decades. Significant progress has been made in recently years based on big multimedia data processing and deep neural networks techniques (Krizhevsky et al. 2012; Weston et al. 2013). However, how to generalize the system to enable recognizing a wider range of entities remains a challenge.

The three primary factors towards a real-world image recognition system are: 1) creating representative image features, 2) designing powerful learning methods, and 3) identifying massive training data. Representative image features are essential for describing and separating different entities. Learning methods, including modelling and learning, are critical for abstracting intrinsic patterns from the data. Training data generation is to ensure that the data contains sufficient representative variations to represent the different visual patterns. While considerable efforts have been put on the first two, much less has been devoted to the third one.

Currently, the usual approaches to generating training data are still based on manual labeling, though progress has been made to obtain the candidate training data from automated and large scale sources, such as search engines and/or social media portals. For example, click log from search engines has been shown as an efficient way to generate training data (Hua et al. 2013). However, the diversity and coverage are still concerns as most users typically only click attractive and representative images. Also, the entities of the clicked data may have limited overlap with the entities that a user is interested in. This paper will systematically study the problem of automatically generating training data from massive and noisy Internet data, and culminate with an end-to-end system that is capable of building automatic image recognition system for any given set of entities without image labeling. The primary challenges we have conquered include:

Separating relevant and informative data from noises: The search results from social media portals are in general very noisy. For example, images returned for query "Eiffel Tower" will contains not only photos of the landmark, but also large number of images that were taken on Eiffel Tower as well inside Eiffel Tower, and close-up photos of people. The search results returned by search engines typically has less noises in the top of the result list (say, on the first page), but it may contain all types of images such as clipart images, and more and more noises will appear as we approach the end of the list. How to filter out all those noises without manual effort is a big challenge.

Effectively leveraging the massive amount of data: It is easy to obtain massive amount of image data on the Internet as long as they are exposed to search engines or social media

portals. We may get more than 10,000 images for a certain entity from Flickr. However, how to effectively leverage this massive amount of data to improve recognition accuracy remains a challenge. In this paper, we argue that combining model based approach (i.e., we train recognition models for all the entities) and search based approach (i.e., build an index and use nearest neighbor search to recognize unknown image) is a more effective way to use this massive amount of data.

Building an efficient end-to-end pipeline: Our goal is to build a system that supports the recognition of any given set of entities. We need to investigate the efficiency in data selection, learning and recognition steps so that an image recognition system can be built in a reasonable amount of time (say, in a few hours). Most of the steps to this final goal are computational intensive. Effective parallelization strategies should be applied to resolve this challenge.

Related Work

Large-scale image recognition: Image recognition has been studied for decades. In the early years, researchers were more focused on small scale dataset (in terms of both the numbers of categories and images). In recently years, with the availability of ImageNet dataset, more and more research efforts are devoted to recognizing 1K or more categories. Krizhevsky et al. used deep neural network to model 1K ImageNet categories, where GPUs are utilized to speed up the training process. Le et al. used a distributed cluster to learn features from unlabeled data based on deep network and then use the features to recognize 20K ImageNet categories. Weston et al. presented a scalable approach in (Weston et al. 2011), which learns a low-dimensional joint embedding space for both images and labels. All the above works focus on features, models or learning process. In contrast, this paper focuses on the data aspect, which have much fewer published works. There are a few approaches that used search engine data or noisy data from social media portals as training source. They typically rely on the learning algorithms to depress the noises (Tang et a. 2009), with only mild or no effort to filtering the data. Such approaches are not suited for handling high noises in the automatically crawled large-scale image dataset.

Image search re-ranking: The main idea of image search re-ranking is to leverage the intrinsic properties of initial search results (typically from an existing search engine) to refine the order of the list (Hsu et al. 2007; Jing et al. 2008; Tian et al. 2008). The basic assumption is that most of images in the search results are relevant and large image groups (in terms of visual similarity) with relatively high initial relevant scores have higher chance to be relevant. The intra-entity propagation in this paper shares similar idea as image re-ranking. However, our data include large number of images from social sources, which have much higher diversity and noises. We need more sophisticated methods to filter out the noises.

Concept Learning from Search Engine Results: Two approaches to learn visual concepts from search results of commercial search engines were reported in (Chen et al. 2013 and Divvala et al. 2014). However, both efforts used the top results from search engines, and assumed that majority of images in the search results are relevant to the entity.

Main contributions

The primary contributions of this paper are threefold:

(1) We have developed two image selection methods, a clustering based method and propagation based method, to mine relevant and informative images and labels from noisy Internet data.

(2) We have combined classification and search based recognition, and are able to more effectively utilize the large-scale training data to achieve a better recognition accuracy.

(3) We have built a scalable end-to-end distributed computing system that can automatically build a customized image recognition system for an arbitrary set of entities in a short turnaround time with minimal human intervention.

System Overview

We name the system "Prajna", which is a synonym of wisdom and have the meaning of "the insight in the true nature of reality". The system architecture is illustrated in Figure 1. There is a backend sub-system (recognizer building) and a frontend sub-system (image query serving). For the backend system, the input is a list of entity names that users would like to build a recognition model for. For example, it can be a list of animals, plants, attractions of a country, all landmarks in the world, etc. Then, image list generation module uses entity names as queries to get ranked image lists from search engines (such as Bing and Google) and social media portals (such as Flickr and PhotoBucket). The crawled image data (containing the entity names, image URLs, page URLs and metadata from the search engines or image portals), are then passed to an additional crawler to get the image files from the Internet. The crawled images are stored into a raw image database. Additionally, they are sent to the image feature extractors to extract classification and indexing features, followed by a module to clean the noises in the image set based on image features. Thereafter, the filtered dataset is sent to the training module and indexing module to build classification models and inverted index based on classification features and indexing features, respectively. The final outputs of the backend system are (1) a set of classification models representing each individual entity trained from the filtered image set; (2) an inverted index of the filtered images.

The Prajna frontend works as an image recognition system. The user submits an image, either as raw bytes captured from a mobile device/camera, or as a URI link from Internet, to the system. Prajna extracts classification features and indexing features from the image, and sends them to the classifier to find top matched categories, and to the inverted index to search for top matched images. The corresponding entities of the top matched images will be aggregated to generate a ranked list, and are then combined with the ranked list from the classifiers to generate the final ranked entity list with confidence scores. The top five entities are shown to users. Optionally, Prajna also shows the corresponding Wikipedia page or information from other knowledge base of the top entities. Alternatively, Prajna can also show image and web search results using the top entity names as queries. These information help users verify whether the recognized entities are correct.

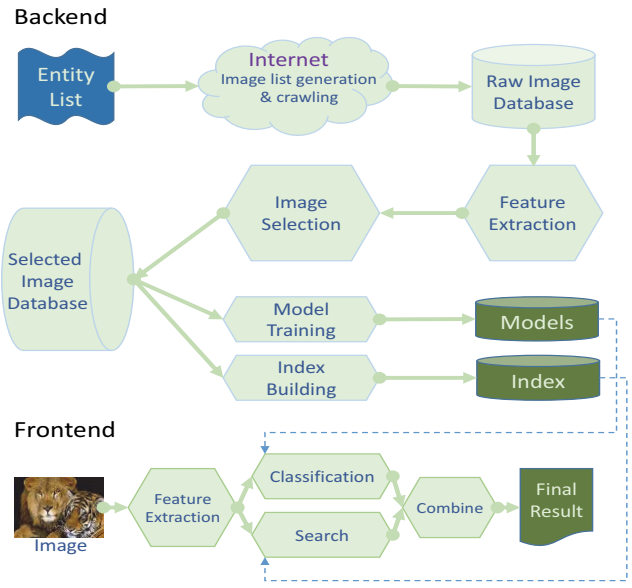

Figure 1. "Prajna" system overview.

Prajna backend and frontend work together as a pipeline for automatically building an image recognition system on the fly to recognize whatever entities that users care, as long as sufficient image data can be discovered from the Internet.

Image Selection

After images are crawled, Prajna extract two sets of features for the images in the raw image database. The first is a 2048-dimensional feature. For this, a process similar to (Krizhevsky et al. 2012) is used to train a deep neural network based on the 1K ImageNet training dataset. The only difference is that we reduce the number of nodes of the fully-connected layers from 4096 to 2048 for better generalization capacity. We remove the last layer of the net, so that the trained deep neural network takes a raw image as input, and outputs a 2048-dimentional feature vector, which is used for image label filtering and model training. The second feature is visual words for indexing. In particular, up to 300 hundreds T2S2 visual words are extracted per image (Winder et al. 2007) and the vocabulary size is 1 million.

Clustering Based Image Selection

Among the crawled image dataset, it is difficult for computers to determine whether an individual image is relevant to an entity (otherwise, we would have solved the image recognition problem). On the other hand, making group membership decision and decide whether a group of images that are sharing certain similar visual patterns is a relevant group is much easier. Therefore, we cluster the images for each entity based on their visual similarities using K-Means, and then try to separate relevant clusters from irrelevant clusters by analyzing the characteristics of the clusters.

Selecting an appropriate *K* in K-Means is always a non-trivial problem. We tried to use other approaches like Affinity Propagation but K-Means generates the best result. And we also found K actually does not affect the cluster relevance estimation (in image level) too much as long as we choose a K that tends to over segment the image set. The rational is that our goal is to decide whether an image cluster is relevant, instead of to generate perfect clustering results. In our paper, we choose a *K* so that the average number of images in a cluster is equal or close to 50.

We then use three types of features to make group membership decision, and select relevant image clusters.

Intra-Cluster Features

Intuitively, bigger clusters with visually consistent images have higher chance to be relevant to the entity. Figure 2 (a) shows two exemplary clusters for entity "Smith Tower" (a landmark entity in "Seattle" entity set, which is used in our experiments). The left cluster, which is a relevant one, has 126 images with a relatively small average image distance, while the irrelevant cluster on the right has 45 images and a much larger average distance.

Figure 2. Image cluster examples for entity "Smith Tower".

To better characterize the data distribution of an image cluster, we calculate pair-wise image distances for all image pairs in the cluster, from which we derive a set of features to

describe the image distribution in the cluster. If the cluster has *n* images, then we will have $n \times (n-1)/2$ distances. We derived the following features to discover relevant clusters:

- *Size of the cluster (number of images)*
- *Percentage of the cluster (number of images in the cluster divided by number of images of this entity)*
- *Minimum, maximum, average and standard deviation of the pair-wise distances*.
- *Normalized image distance distribution*

Most of these features are self-explanatory. Let all the distances be denoted by

$$D = \left\{ d_i, 0 \le i < \frac{1}{2} n \times (n-1) \right\} \quad (1)$$

We normalize the distances to a number in [0, 1]:

$$d_i' = \frac{d_i - \min\{d_i\}}{\max\{d_i\} - \min\{d_i\}}, \quad (2)$$

We then divide [0, 1] to *K* bins (for example, 10 bins, which is [0, 0.1), [0.1. 0.2),⋯, [0.9, 1]), and count the number of normalized distances in each bin to form a histogram of the normalized image distances.

Inter-Cluster Features

Intra-cluster features are not sufficient to separate relevant clusters out of irrelevant ones in some situations. Figure 3 shows one reasonable-size (i.e., the left one, with 20 images) and one large-size (the right one, with 89 images) clusters with visually consistent images for "Smith Tower". Unfortunately, none of them corresponds to "Smith Tower", and cannot be removed by using intra-cluster features only.

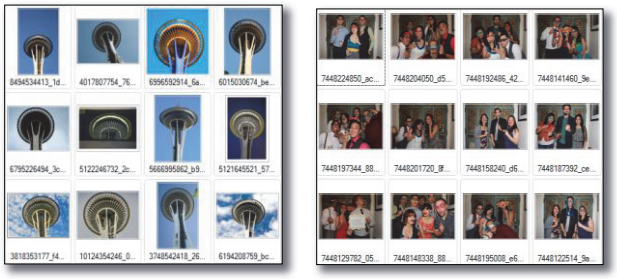

Figure 3. Tight but wrong clusters (entity "Smith Tower").

We observe that the left cluster in Figure 3 is actually a good set of images for another entity "Space Needle", and image clusters similar to the right one may appear in the clusters of other entities as well due to it mainly consists of photos of a group of persons in a room, a common scene in social photos. This enlightens us to use inter-cluster features to filter out this type of clusters. We calculate the distances of each cluster to all the clusters of other entities. When calculating distance of two clusters, the cluster centroids are used to represent the corresponding clusters. For example, suppose C_i is a particular cluster of entity E_i, then the distances between E_0 and all clusters of other entities E_1 to E_{M-1} are the distances between C_0 to $C_1 \ldots C_{M-1}$. Let there be L distances, the following features can be used as inter-cluster features: *minimal distance*, *maximal distance*, *average distance*, *standard deviation*, and *normalized image distance distribution* (similar to how we generate the intra-cluster distance distribution). Intuitively, based on this set of features, the two clusters of "Smith Tower" in Figure 3 will be removed as they are too close to some clusters of other entities.

Intra-Entity Feature

Within a particular entity, how close an image cluster to the rest of the clusters reflects how "close" this cluster to the primary "subjects" (in terms of visual appearances) of this entity. In general, the closer it is to the primary theme, the higher probability that it is a relevant cluster. Based on this intuition and similar to the above two features, we use *minimum*, *maximum*, *average*, *standard deviation* and the *distribution of cluster pair-wise distances* to describe the intra-entity property of this cluster.

Cluster Relevance Estimation

After deriving the intra-cluster, inter-cluster and intra-entity features, we train a group membership classifier that determines how relevant a cluster is to the entity. To train this classifier, we label a set of clusters with a relevance score to a certain entity. This labeling work only needs to be done once for all entity sets, and the trained classifier can be reused onto new entity set Prajna encounters. The classifier can be, for example, SVM (support vector machine) based (which is also the one we used in this paper), Decision Tree based, etc. The classifier is then applied to estimate the relevance of a cluster to its corresponding entity for other entity sets.

Propagation Based Image Selection

The purpose of propagation is to leverage the relationships among the entities and the corresponding images to further eliminate noises. Propagation is applied on both individual entity level (intra-entity level) and the entire entity set level (inter-entity level). Intra-entity level propagation shares similar ideas with image search re-ranking (Hsu et al. 2007; Jing et al. 2008), which typically is done by a random walk process to propagate and refine the relevance scores of the images to the query. After intra-entity propagation, only images with high relevance scores are kept.

Unlike intra-entity propagation which propagates image relevance to the targeted entity, inter-entity propagation propagates entity-relevance pairs among all the images in this entity set. The vertexes of the graph are all the images for the entity set, the values are the corresponding entities and the relevant scores of the image corresponds to the entity (which are the outputs from the previous step).

After inter-entity based propagation, images will be associated with the top one entity name in the entity name list. While most correlations will not change, the changed ones indicate the corresponding image is actually more relevant to another entity.

Combined Recognition

The proposed system combines model based recognition and search based recognition into one recognition engine.

A variety of learning models can be applied to build recognition models. The deep neural network model gives the best recognition results on ImageNet dataset. Fisher vector as feature plus online SVM also got good results in non-neural-network based approaches. In this paper, to make it easier to compare and reimplementation, we adopt online SVM approach though the dimension of the features we are using is much lower than a typical Fisher Vector. We trains M models for the M entities via online SVM, and the algorithm is briefly described as below. More details about the SGD based online SVM learning can be found in (Crammer et al. 2001 and 2006).

To build image index, we extract up to 300 T2S2 visual words per image. We then build an inverted index that maps each visual word to a list of images. The algorithm for using the inverted index to do recognition is as follows:

Algorithm 1: Search based image recognition

Input: An image *I*, and an inverted index *Index*

1: Extract visual words from *I*, denoted as $\{\boldsymbol{w_i}\}$.

2: For each $\boldsymbol{w_i}$, find the list of images in the *Index*, denoted by: $\boldsymbol{L_i} = \{(\boldsymbol{I_{ij}}, \boldsymbol{h_{ij}})\}$, where $\boldsymbol{h_{ij}}$ is the visual words count in image $\boldsymbol{I_{ij}}$.

3: Merge $\{\boldsymbol{L_i}\}$ by summing up the visual word hit counts of the same images, and order the images by the hit counts in descending order:

$$\textbf{sum}\{\boldsymbol{h_{ij}}\} \text{ group by } \boldsymbol{I_{ij}} \textbf{ from } \{\boldsymbol{L_i}\} \textbf{ order by } \boldsymbol{h_{ij}}. \quad (3)$$

4: Get the entity names for the top images in the list and output it as the recognition result, and the confidence score is estimated by the number of hits divided by the number of visual words.

Combined Recognition

Most image recognition works use model based approaches and very few researchers reported search based approaches (Wang et al. 2007). The advantage of model based approach is that the learned models are "abstract" representations of the entities and are powerful to handle object models with reasonable variations. On the other hand, the disadvantage is that some outstanding "prototypes" will not be captured by the abstracted models if the amount of training data for these prototypes is not sufficient. These "prototypes" actually can be well handled by search based approaches if near duplicates and/or partial duplicates can be found in the training set. Search based approaches also scale better as the index operation is less expensive than model recognition.

In the proposed image recognition system, we attempt to combine both approaches, aiming at leveraging the large amount of automatically selected training data in a more effective way. There are two difficulties in combining these two approaches.

First, search based approach requires a training set with high accuracy and large coverage at the same time. However, these two requirements are contradict to each other. Second, it is unclear on how to combine the outputs from model based classification and inverted index search. The predicted scores for the candidate labels in the classification results are often not in accordant with the probabilities whether the corresponding predictions are correct. Theoretic research in combing the two results are still under developing. In Prajna, we propose the following empirical approach to estimate a "relative confidence score" for model based classification and calibrate it with the weighted confidence score from search based recognition.

The basic idea is to estimate how "significant" the classification score is by comparing the score with all other scores from the model vectors of other entities. A naive way to get a relative confidence score of a prediction is to compare the score of the top one result and that of the second result. Larger difference indicates higher relative confidence score and vice versus. However, multiple entity names may be related to one particular image. For example, for a picture of a lion and tiger, the scores of lion and tiger may be both high and close to each other. To solve this, we check more than one consecutive pairs and also take the "similarity" of the entity pairs into account. Suppose the prediction scores of top n entities are $t_1, t_2, \ldots, t_n$, then the relative confidence score of this prediction is defined as:

$$C^{(r)} = \max\{t_i\} + \alpha \max\{t_i - t_{i+1}\} - \beta \sum_{i \neq j} \frac{D(E_i, E_j)}{|t_i - t_j| + \gamma}. \quad (4)$$

On the right side of equation (3), the first term aims to keep the information in the original prediction score, the second term promote predictions with outstanding scores, and the third term penalizes predictions that give similar scores to "distinct" entities, where the distance between two entities is estimated by the average pair-wise cluster distances. (α, β, γ) are empirical numbers. By using the relative confidence score, the combine recognition scheme becomes:

Algorithm 2: Combined recognition

1: Prediction by model based classification.

2: Get relative confidence score $\boldsymbol{C^{(r)}}$ for the prediction. If $\boldsymbol{C^{(r)}}$ is bigger than a threshold, then output result. Otherwise, go to 3.

3: Get an entity list by searching over the index. Calculate a weighted confidence scores. If the top score is higher than $\boldsymbol{\eta C^{(r)}}$, output the search based recognition result; otherwise, output model based recognition results. $\boldsymbol{\eta}$ is an empirical parameter.

System Level Considerations

In this section, we will discuss system-level considerations, particularly on how to scale out the system and make the system extensible.

Distributed Computing

A large number of image can be collected from search engines and social media portals. For example, for an entity set with 1000 entities, if we collect 6000 images per entity, there

will be 6 million images in total. If we are dealing with multiple entity sets, the computation cost can be huge. It is usually impractical for the entire computation be completed in one machine in a reasonable period of time. To solve this, we used a map-reduce based distributed system with thousands of nodes to complete most of the tasks. Different parallelization methods are applied for different steps according to the natural and complexity of the processes, as shown in Figure 4. For example, feature extraction and within entity clustering can be easily executed using direct parallelization approach; inter-cluster feature extraction and inter-entity propagation need to be run in divide and merge manner; Online SVM training can be parallelized using divide and aggregate strategy.

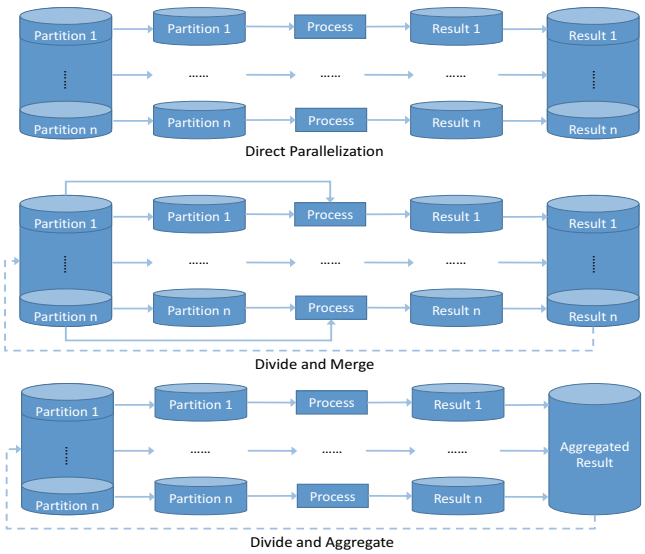

Figure 4. Three parallelization strategies.

Extending Entity Set on the Fly

If some entities that are related to the current entity set are not supported by the current image recognition system, it is easy to add the new entities into the existing system, because that all the steps, including getting image list, downloading, feature extraction, clustering, cluster feature extraction, cluster classification, inverted indexing, and SVM model training, can be done incrementally. Therefore, new entities can be added to the system at low computation and time cost.

Handling Multiple Entity Sets

Though we mainly use "Seattle", an outdoor scenery entity set as example, the system described can work on arbitrary entity set selected by the users. We have also considered the problem to make Prajna work for more than one entity sets at the same time in one system. The primary issue is how to determine the right entity sets (one or more) to do the combined recognition when there are multiple models and indexes available in the system.

There are a few potential approaches that can be adopted here. One approach is to use a multi-layer classification, in which a top layer classifier determines which entity subsets are to be further applied. An alternative approach is to use inverted indexes to determine which entity set should be used. That is, we will search for the given image from the indexes of all entity sets, and the entity subsets that have the most number of matched images or visual words will be selected. A third approach is to run classifications on all entity sets, and compare the relative confidence scores to pick one or more entity sets. We can also combine one or more of the above approaches to fuse the recognition result.

Experiments

The Prajna system has a large number of components. Due to space limit, we will focus on evaluating how image selection and combined recognition improve the overall recognition accuracy.

Evaluation Datasets

Four entity sets are used for evaluation: ImageNet animal subset, Disneyland attractions, Seattle attractions and Aquarium sea creatures. For ImageNet subset, we select all the animal related entities from the ImageNet 1K dataset 0, which has 163 synsets. The reason to pick ImageNet dataset is that it has a manually labeled training set, validation set and test set, so we can compare and evaluate the accuracy of the system built upon unlabeled data. For Seattle entity set, we manually labeled 5000 images obtained from Google image search as the test dataset. Due to the high cost of manual labeling, for the other two entity sets, clicked images from Clickture-Full dataset (Hua 2013) is used as the test data.

For ImageNet dataset, there are 1.3M images in the training set, 50K images in the evaluation set, as well as 150K image in test set for the 1000 synsets. After selecting all the images that are associated with the 163 animal related synsets, we get 211.9K images for training, 8.15K for validation, and 24.45K for testing. For the click data from Clickture, we use the corresponding animal names to retrieve the clicked images in Clickture.

Up to 6150 images per entity are collected, among which (up to) 6000 are from Flickr and 150 are from Bing image search. For ImageNet animal subsets, each individual animal name of a synset is used as one query and then the search results of all the queries for the same synset are merged.

Performance Comparison

In the first experiment, we compare different image selection approaches on the ImageNet animal subset. We will also compare our results with the ones based on ImageNet labeled training data and clicked data in Clickture. As there is no other published research for dealing with highly-noisy la-

belled web image data, we do not compare our approaches with existing approaches in the literature.

Evaluation on ImageNet animal subset

Table 1 shows the experimental settings in terms of the sources of the training data, image selection approaches, as well as the recognition engines. All evaluations are run on the ImageNet test set.

There are indeed quite a few parameters need to set empirically, which is inevitable for a complex real-world end-to-end system. Thanks to the high computation power of the computer cluster, a parameter sweeping procedure is done to find the best parameters on the ImageNet validation dataset based on experimental setting 10 in Table 1. The below table shows the parameters as well as the sweeping intervals. To speed up parameter sweeping, clustering and propagation results are stored as intermediate data and then different parameters are applied in image selection, learning and combined recognition steps. The best parameters obtained on the validate set is 0.15, 0.8, 0.9, 20, 0.01, 3, 7, and 0.9.

Table 1. Experiment settings.

ID	Training Set/Image Selection	Recog. Engine
Ex01	ImageNet training set (animal subset)	Model + Index
Ex02	Clickture-Full	Model + Index
Ex03	Flickr/Bing unfiltered	Model + Index
Ex04	Flickr/Bing + cluster selection	Model + Index
Ex05	Flickr/Bing + propagation	Model + Index
Ex06	Flickr/Bing + cluster selection/propagation	Model
Ex07	Flickr/Bing + cluster selection/propagation	Index
Ex08	Flickr only + cluster selection/ propagation	Model + Index
Ex09	Bing only + cluster selection/ propagation	Model + Index
Ex10	Flickr/Bing + Cluster selection/ propagation	Model + Index

Table 2. Parameter sweeping.

ID	Parameter	Range	Interval
Th1	Cluster based selection classifier	[-1.0, 1.0]	0.05
Th2	Intra-entity propagation selection	[0.1, 1.0]	0.1
Th3	Inter-entity propagation selection	[0.1, 1.0]	0.1
Th4	Online SVM partition number	[20, 50]	5
Th5	Learning rate	[0.001, 0.1]	10 times
Th6	Rounds of learning on each node	[1, 10]	1
Th7	Rounds of model averaging	[1, 20]	1
Th8	Relative confidence threshold	[0.5, 4]	0.1

Table 3 shows the recognition accuracy on the ImageNet test set (animal subset) and the actual size of training data used. From this table, it can be observed that: (1) Ex10, the final proposed approach, gives the second highest recognition accuracy. It is slightly lower than the accuracy based on the ImageNet manually labeled data. The result shows leveraging large amount of noisy data on the Internet with a selection step is a promising solution to automatically generate training data, thus potentially we can expand image recognition to any given set of entities as long as we can collect sufficient number of images from the Internet (and the entities are somewhat model-able by the visual features). From Ex10, even the cleaned data set is not as accurate as the manually labeled one, but it has a larger amount of data, which provides useful additional variation to build an effective classifier. The gap between Ex01 and Ex10 may also indicates that there is still room to improve in the image selection step. (2) Ex03 has much lower accuracy than Ex10, which shows image selection is essential to ensure the quality of the training data. (3) Ex02 is worse than Ex01, Ex06, Ex08 and Ex10, which proves the volume and variations of clicked data are not sufficient to capture the visual variations of the entities. (4) Ex10 is better than Ex04 and Ex05, which shows combining clustering based selection and propagation based selection boosts the accuracy. The difference between Ex04 and 05 shows clustering based selection is more effective for noise removal. An explanation is that propagation based approach is less effective when the noise level is high. (5) Ex10 is better than Ex06 and Ex07, which means combining model-based recognition and search-based recognition improves the recognition performance. (6) Ex10 is better than Ex08 and Ex09, which shows both Flickr data and Bing data contributes to the performance and Flickr contributes more due to more images from Flickr crawling.

Table 3. Experiment results.

ID	Train. Image	Sel. Ratio	Top 1 Acc.	Top 5 Acc.
Ex01	211.9K	-	0.52581	0.80822
Ex02	69.7K	-	0.28290	0.54233
Ex03	1311.1K	100%	0.37146	0.60123
Ex04	920.3K	70.2%	0.47894	0.75431
Ex05	1101.8K	84.0%	0.42540	0.68417
Ex06	808.9K	61.7%	0.46030	0.73106
Ex07	808.9K	61.7%	0.20112	0.46978
Ex08	750.3K	64.6%	0.48133	0.77686
Ex09	98.5K	66.7%	0.32986	0.57636
Ex10	808.9K	61.7%	0.51526	0.79648

Evaluation on Other Datasets

In the second experiment, we evaluate the accuracy of the proposed framework on the other three entity sets: Seattle attractions on a manually labeled dataset and click dataset from Clickture, Disneyland attractions and Aquarium sea creatures on Clickture dataset. The focus is on evaluating the image selection approaches as there are no sufficient manually labeled training datasets for these three entity sets,

Table 3 shows the evaluation results on these three entity sets under setting Ex10 (with image selection) and Ex03 (no image selection). It is observed that image selection step pays a significant role in improving the final recognition accuracy. In addition, we can see that with the increase of the size of the entity set, the recognition accuracy decreases significantly. One reason that Disneyland entity set has low recognition rate is that many entities have limited number of images collected from the Internet (especially from Flickr), such as “Café Rix” (a restaurant in Disneyland park, 12 image) and “BouTiKi” (a store, 48 images).

In the above experiments, the manually labeled dataset and click dataset are both derived from search engines, which is not a perfect evaluation setting for Prajna as most of the images we are using to train the system are from Flickr, which have a different image-label characteristics

(e.g., containing many personal photos). Search results and click log data also contain image types that do not usually present in Flickr, e.g., clipart images. That explains why we get better accuracy on animal set compared with these three entity sets. In addition, if we train DNN recognition models directly from the filtered training data, we should get a higher recognition accuracy.

Table 3. Results on Seattle, Disneyland & Aquarium.

Entity Set (#)	Selection	Test Set	Top 1 Acc.	Top 5 Acc.
Seattle (106)	Yes	Manual 5K	0.47813	0.73902
	No		0.32342	0.54239
	Yes	Clickture 2.4K	0.40517	0.65297
	No		0.29368	0.51980
Disneyland (623)	Yes	Clickture 112.8K	0.25834	0.50322
	No		0.10136	0.23169
Aquarium (239)	Yes	Clickture 46.9K	0.37253	0.61458
	No		0.26878	0.51970

Conclusion

In this paper, we proposed an end-to-end system "Prajna" to mine knowledge from the Internet and build image recognition systems with minimal human involvement. Potentially the system can recognize any given set of entities as long as we can collect sufficient large number of (noisy) image data from the Internet. The basic idea is to automatically select relevant and informative training data from the noisy data by leveraging the intrinsic characteristics of the data distribution as well as the correlations among images and labels. We apply well known distributed computing strategies so that it is feasible to build a recognition engine for a given set of entities in a reasonable period of time.

Prajna uses the entire Internet as a knowledge base (though noisy) to mine structured knowledge of image data. We believe that this is a promising direction to build real-world image recognition systems as well as to power recognition related applications (such as photo tagging). Prajna represents our first attempt to build image recognition system from web data, and there are still rooms to improve for many Prajna modules. For example, we can potentially adapt selection parameter according to the distribution of the original noisy dataset; make use of the textual metadata of the images; use more sophisticated approaches to combine model based approach and search based approach; balance computation cost on each node to further speed up the system building process; learn entity set specific features for recognition; automate approaches to form an entity set for a given topic; estimate the accuracy of the recognition system without labeled data, etc.. We hope this work will inspire more researchers and groups to use web data to mine structured knowledge.

References

Ahn, L. von; Dabbish, L. 2004. Labeling images with a computer game. SIGCHI.

Chua, T.-S.; et al. 2009. NUS-WIDE: A Real-World Web Image Database from National University of Singapore. ACM CIVR.

Chen, X.; Shrivastava, A.; Gupta, A. 2013. NEIL: Extracting Visual Knowledge from Web Data. ICCV.

Crammer, K.; Singer, Y. 2001. On the Algorithmic Implementation of Multi-Class SVMs. JMLR, Vol 2, pp. 265-292.

Crammer, K.; Dekel, O.; Keshet, J.; Shalev-Shwartz, S.; Singer, Y. 2006. Online Passive-Aggressive Algorithms. JMLR, Vol 7.

Deng, J.; et al. 2009. ImageNet: A Large-Scale Hierarchical Image Database. IEEE Computer Vision and Pattern Recognition.

Divvala, S.K.; et al. 2014. Learning Everything about Anything: Webly-Supervised Visual Concept Learning. CVPR.

Griffin, G.; Holub, A.; and Perona, P.. 2007. Caltech-256 object category dataset. Technical Report 7694, Caltech.

Hall, K. B.; Gilpin, S.; Mann, G. 2010. MapReduce/Bigtable for Distributed Optimization. NIPS Workshop on Learning on Cores, Clusters, and Clouds.

Hua, X.-S.; et al. Clickage: Towards Bridging Semantic and Intent Gaps via Mining Click Logs of Search Engines. 2013. ACM MM.

Hsu, W. H.; et a;. 2007. Video Search Reranking through Random Walk over Document-Level Context Graph. ACM Multimedia.

Jing, Y. S.; and Baluja, S. 2008. VisualRank: Applying PageRank to Large-Scale Image Search. Transon Pattern Analysis and Machine Intelligence.

Krizhevsky, A.; Sutskever, I.; Hinton , G.E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. NIPS.

Le, Q.; Ranzato, M.; Monga, R.; Devin, M.; Chen, K.; Corrado, G.; Dean, J.; Ng, A. 2012. Building high-level features using large scale unsupervised learning. ICML.

Li, X. R.; Snoek, C. G. M.; and Worring, M. 2008. Learning Tag Relevance by Neighbor Voting for Social Image Retrieval. In Proceedings of the ACM ICMR.

Liu, D.; Hua, X.-S.; Yang, L.; Wang, M.; Zhang, H.-J. 2009. Tag Ranking. WWW.

Liu, D.; et al. 2011. Image Retagging via Collaborative Tag Propagation. IEEE Transaction on Multimedia (TMM), 13(1):82-91.

Russell, B.; et al. 2007. LabelMe: a database and web-based tool for image annotation. International Journal of Computer Vision.

Tang, J.; Yan, S.; Hong, R.; Qi, G.-J.; and Chua, T.-S. 2009. Inferring semantic concepts from community-contributed images and noisy tags. ACM Multimedia.

Tian, X.; Yang, L.; Wang, J.; Yang, Y.; Wu, X.; Hua, X.-S. 2008. Bayesian Video Search Reranking. ACM Multimedia.

Torralba, A.; Fergus, R.; and Freeman, W. 2008. 80 million tiny images: A large data set for nonparametric object and scene recognition. PAMI, 30(11):1958–1970.

Wang, X.-J.; Zhang, L.; Jing, F.; Ma, W.-Y. 2007. AnnoSearch: Image Auto-Annotation by Search. CVPR.

Winder, S.; Brown, M. 2007. Learning Local Image Descriptors. IEEE CVPR.

Weston, J. ; Makadia, A.; and Yee H. 2013. Label Partitioning For Sublinear Ranking. ICML.

Weston, J.; Bengio, S.; Usunier, N. 2011. Wsabie: Scaling Up To Large Vocabulary Image Annotation. Intl. Joint Conference on Artificial Intelligence (IJCAI).