Webly Supervised Learning of Convolutional Networks

Xinlei Chen
Carnegie Mellon University
xinleic@cs.cmu.edu

Abhinav Gupta
Carnegie Mellon University
abhinavg@cs.cmu.edu

Abstract

We present an approach to utilize large amounts of web data for learning CNNs. Specifically inspired by curriculum learning, we present a two-step approach for CNN training. First, we use easy images to train an initial visual representation. We then use this initial CNN and adapt it to harder, more realistic images by leveraging the structure of data and categories. We demonstrate that our two-stage CNN outperforms a fine-tuned CNN trained on ImageNet on Pascal VOC 2012. We also demonstrate the strength of webly supervised learning by localizing objects in web images and training a R-CNN style [19] detector. It achieves the best performance on VOC 2007 where no VOC training data is used. Finally, we show our approach is quite robust to noise and performs comparably even when we use image search results from March 2013 (pre-CNN image search era).

1. Introduction

With an enormous amount of visual data online, web and social media are among the most important sources of data for vision research. Vision datasets such as ImageNet [41], PASCAL VOC [14] and MS COCO [29] have been created from Google or Flickr by harnessing human intelligence to filter out the noisy images and label object locations. The resulting clean data has helped significantly advance performance on relevant tasks [16, 24, 19, 59]. For example, training a neural network on ImageNet followed by fine-tuning on PASCAL VOC has led to the state-of-the-art performance on the object detection challenge [24, 19]. But human supervision comes with a cost and its own problems (*e.g.* inconsistency, incompleteness and bias [52]). Therefore, an alternative, and more appealing way is to learn visual representations and object detectors from the web data directly, without using any manual labeling of bounding boxes. But the big question is, can we actually use millions of images online without using any human supervision?

In fact, researchers have pushed hard to realize this dream of learning visual representations and object detectors from web data. These efforts have looked at different aspects of webly supervised learning such as:

- **What are the good sources of data?** Researchers have tried various search engines ranging from text/image search engines [5, 56, 54, 17] to Flickr images [33].
- **What types of data can be exploited?** Researchers have tried to explore different types of data, like images-only [27, 9], images-with-text [5, 43] or even images-with-n-grams [13]).
- **How do we exploit the data?** Extensive algorithms (*e.g.* probabilistic models [17, 27], exemplar based models [9], deformable part models [13], self organizing map [20] *etc*.) have been developed.
- **What should we learn from web data?** There has been lot of effort ranging from just cleaning data [15, 57, 33] to training visual models [27, 53, 28], to even discovering common-sense relationships [9].

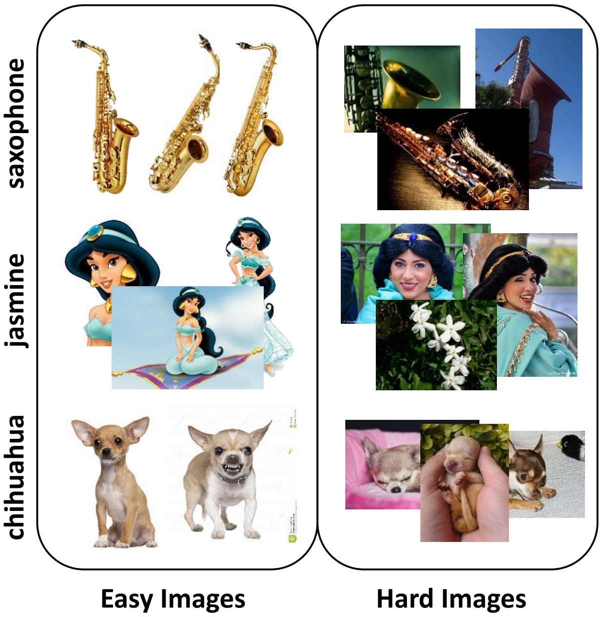

Figure 1. We investigate the problem of training a webly supervised CNN. Two types of visual data are available online: image search engine results (left) and photo-sharing websites (right). We train a two-stage network bootstrapping from clean examples retrieved by Google, and enhanced by noisier images from Flickr.

Nevertheless, while many of these systems have seen orders of magnitudes larger number of images, their performance has never matched up against contemporary methods that receive extensive supervision from humans. Why is that?

Of course the biggest issue is the data itself: 1) it contains noise, and 2) is has bias - image search engines like Google usually operate in the high-precision low-recall regime and tend to be biased toward images where a single object is centered with a clean background and a canonical viewpoint [30, 4, 29]. But is it just the data? We argue that it is not just the data itself, but also the ability of algorithms to learn from large data sources and generalize. For example, traditional approaches which use hand-crafted features (*e.g.* HOG [9]) and classifiers like support vector machines [13] have very few parameters (less capacity to memorize) and are therefore unlikely to effectively use large-scale training data. On the other hand, memory based nearest neighbors classifiers can better capture the distribution given a sufficient amount of data, but are less robust to the noise. Fortunately, Convolutional Neural Networks (CNNs) [24] have resurfaced as a powerful tool for learning from large-scale data: when trained with ImageNet [41] ($\sim$1M images), it is not only able to achieve state-of-the-art performance for the same image classification task, but the learned representation can be readily applied to other relevant tasks [19, 59].

Attracted by their amazing capability to harness large-scale data, in this paper, we investigate webly supervised learning for CNNs (See Figure 1). Specifically, 1) we present a two-stage webly supervised approach to learning CNNs. First we show that CNNs can be readily trained for easy categories using images retrieved by search engines. We then adapt this network to hard (Flickr style) web images using the relationships discovered in easy images. 2) We show webly supervised CNNs also generalize well to relevant vision tasks, giving state-of-the-art performance compared to ImageNet pretrained CNNs if there is enough data. 3) We show state-of-the-art performance on VOC data for the scenario where not a single VOC training image is used - just the images from the web. 4) We also show competitive results on scene classification. We believe this paper opens up avenues for exploitation of web data to achieve next cycle of performance gain in vision tasks (and at no human labeling costs!).

1.1. Why Webly Supervised?

Driven by CNNs, the field of object detection has seen a dramatic churning in the past two years, which has resulted in a significant improvement in the state-of-the-art performance. But as we move forward, how do we further improve performance of CNN-based approaches? We believe there are two directions. The first and already explored area is designing deeper networks [45, 50]. We believe a more promising direction is to feed more data into these networks (in fact, deeper networks would often need more data to train). But more data needs more human labeling efforts. But data labeling in terms of bounding boxes can be very cumbersome and expensive. Therefore, if we can exploit web data for training CNNs, it would help us move from million to billion image datasets in the future. In this paper, we take the first step in demonstrating: 1) CNNs can be trained effectively by just exploiting web data at much larger scales; 2) competitive object detection results can be obtained without using a single bounding box labels from humans.

2. Related Work

Mining high-quality visual data and learning good visual representation for recognition from the web naturally form two aspects of a typical chicken-and-egg problem in vision. On one hand, clean and representative seed images can help build better and more powerful models; but on the other hand, models that recognize concepts well are crucial for indexing and retrieving image sets that contain the concept of interest. How to attack this problem has long been attractive to both industry and academia.

From Models to Data: Image retrieval [47, 46] is a classical problem in this setting. It is not only an active research topic, but also fascinating to commercial image search engines and photo-sharing websites since they would like to better capture data streams on the Internet and thus better serve user's information need. Over the years, various techniques (*e.g.* click-through data) have been integrated to improve search engine results. Note that, using pretrained models (*e.g.* CNN [57]) to clean up web data also falls into this category, since extensive human supervision has already been used.

From Data to Models: A more interesting and challenging direction is the opposite - can models automatically discover the hidden structures in the data and be trained directly from web data? Many people have pushed hard in this line of research. For example, earlier work focused on jointly modeling images and text and used text based search engines for gathering the data [5, 43, 42]. This tends to offer less biased training pairs, but unfortunately such an association is often too weak and hard to capture, since visual knowledge is usually regarded as common sense knowledge and too obvious to be mentioned in the text [9]. As the image search engines became mature, recent work focused on using them to filter out the noise when learning visual models [18, 56, 54, 53, 28, 13, 20]. But using image search engines added more bias to the gathered data [7, 30, 29]. To combat both noise and data bias, recent approaches have taken a more semi-supervised approach. In particular, [27, 9] proposed iterative approaches to jointly learn models and find clean examples, hoping that simple examples learned first can help the model learn harder, more complex examples [3, 25]. However, to the best of our knowledge, human supervision is still a clear winner in performance, regardless of orders of magnitudes more data seen by many of these web learners.

Our work is also closely related to another trend in

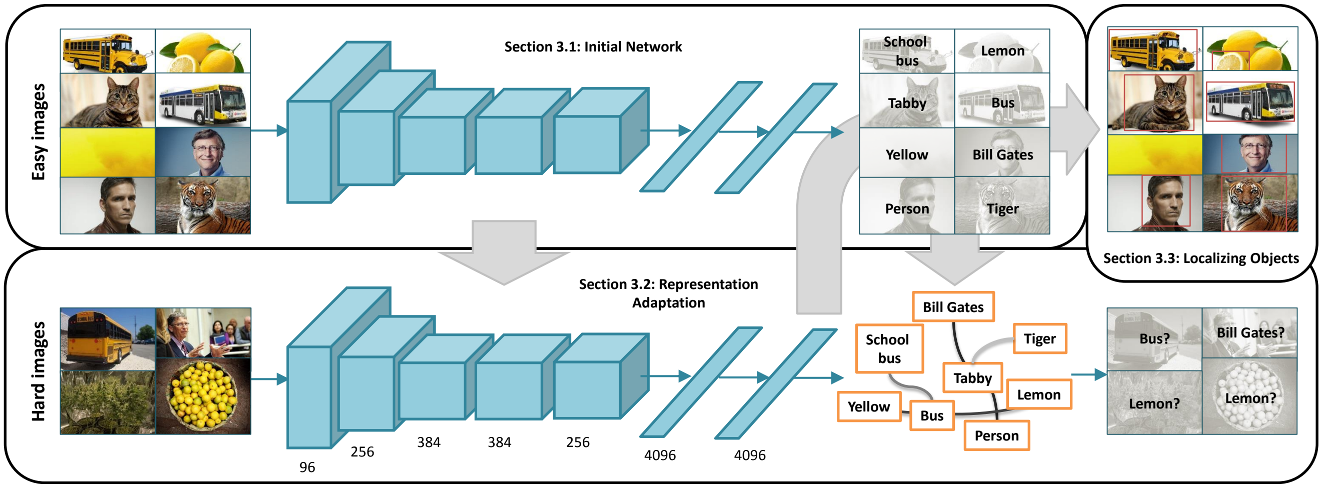

Figure 2. Outline of our approach. We first train a CNN using easy images from Google (above). This CNN is then used to find relationships and initialize another CNN (below) for harder images. The learned representations are in turn used to localize objects and clean up data.

computer vision: learning and exploiting visual representation via CNNs [24, 19, 51, 21]. However, learning these CNNs from noisy labeled data [49, 40] is still an open challenge. Following the recent success of convolutional networks and curriculum learning [3, 25, 26], we demonstrate that, while directly training CNNs with high-level or fine-grained queries (*e.g*. random proper nouns, abstract concepts) and noisy labels (*e.g*. Flickr tags) can still be challenging, a more learning approach might provide us the right solution. Specifically, one can bootstrap CNN training with easy examples first, followed by a more extensive and comprehensive learning procedure with similarity constraints to learn visual representations. We demonstrate that visual representations learned by our algorithm performs very competitively as compared to ImageNet trained CNNs.

Finally, our paper is also related to learning from weak or noisy labels [11, 34, 12, 48, 55]. There are some recent works showcasing that CNNs trained in a weakly supervised setting can also develop detailed information about the object intrinsically [44, 32, 36, 6, 35]. However, different from the assumptions in most weakly-supervised approaches, here our model is deprived of clean human supervision altogether (instead of only removing the location or segmentation). Most recently, novel loss layers have also been introduced in CNNs to deal with noisy labels [49, 40]. On the other hand, we assume a vanilla CNN is robust to noise when trained with simple examples, from which a relationship graph can be learned, and this relationship graph provides powerful constraints when the network is faced with more challenging and noisier data.

3. Approach

Our goal is to learn deep representations directly from the massive amount of data online. While it seems that CNNs are designed for big data - small datasets plus millions of parameters can easily lead to over-fitting, we found it is still hard to train a CNN naively with random image-text/tag pairs. For example, most Flickr tags correspond to meta information and specific locations, which usually results in extremely high intra-tag variation. One possibility is to use commercial text-based image search engine to increase diversity in the training data. But if thousands of query strings are used some of them might not correspond to a visualizable concept and some of the query strings might be too fine grained (*e.g*. random names of a person or abstract concepts). These non-visualizable concepts and fine-grained categories incur unexpected noise during the training process[1]. One can use specifically designed techniques [9, 13] and loss layers [49, 40] to alleviate some of these problems. But these approaches are based on estimating the empirical noise distribution which is non-trivial. Learning the noise distribution is non-trivial since it is heavily dependent on the representation, and weak features (*e.g*. HOG or when the network is being trained from scratch) often lead to incorrect estimates. On the other hand, for many basic categories commonly used in the vision community, the top results returned by Google image search are pretty clean. In fact, they are so clean that they are biased towards iconic images where a single object is centered with a clean background in a canonical viewpoint [30, 38, 4, 29]. This is good news for learning algorithm to quickly grasp the appearance of a certain concept, but a representation learned from such data is likely biased and less generalizable. So, what we want is an approach that can learn visual representation from Flickr-like images.

Inspired by the philosophy of curriculum learning [3, 25, 26], we take a two-step approach to train CNNs from the web. In curriculum learning, the model is designed to learn the easy examples first, and gradually adapt itself to harder examples. In a similar manner, we first train our CNN model from scratch using easy images downloaded

[1] We tried to train a CNN with Google results of ∼7000 noun phrases randomly sampled from the web (∼5M images), but it does not converge.

from Google image search. Once we have this representation learned we try to feed harder Flickr images for training. Note that training with Flickr images is still difficult because of noise in the labels. Therefore, we apply constraints during fine-tuning with Flickr images. These constraints are based on similarity relationships across different categories. Specifically, we propose to learn a relationship graph and initial visual representation from the easy examples first, and later during fine-tuning, the error can back-propagate through the graph and get properly regularized. The outline of our approach is shown in Figure 2.

3.1. Initial Network

As noted above, common categories used in vision nowadays are well-studied and search engines give relatively clean results. Therefore, instead of using random noun phrases, we obtained three lists of categories from ImageNet Challenge [41], SUN database [58] and NEIL knowledge base [9]. ImageNet syn-sets are transformed to its surface forms by just taking the first explanation, with most of them focusing on object categories. To better assist querying and reducing noise, we remove the suffix (usually correspond to attributes, *e.g.* indoor/outdoor) of the SUN categories. Since NEIL is designed to query search engines, its list is comprehensive and favorable, we collected the list for objects and attributes and removed the duplicate queries with ImageNet. The category names are directly used to query Google for images. Apart from removing unreadable images, no pre-processing is performed. This leave us with $\sim$600 images for each query. All the images are then fed directly into the CNN as training data.

For fair comparison, we use the same architecture (besides the output layer) as the BLVC reference network [23], which is a slight variant of of the original network proposed by [24]. The architecture has five convolutional layers followed by two fully connected layers. After seventh layer, another fully connected layer is used to predict class labels.

3.2. Representation Adaptation with Graph

After converging, the initial network has already learned favorable low-level filters to represent the "visual world" outlined by Google image search. However, as mentioned before, this "visual world" is biased toward clean and simple images. For example, it was found that more than 40% of the cars returned by Google are viewed from a 45 degree angle [30]. Moreover, when a concept is a product, lots of the images are wallpapers and advertisements with artificial background, with the product centered and pictured from the best selling view. On the other hand, photo-sharing websites like Flickr have more realistic images since the users upload their own photos. Though photographic bias still exists, most of the images are closer-looking to the visual world humans experience everyday. Datasets constructed from them are shown to generalize better [52, 29]. Therefore, as a next step, we aim to narrow the gap by fine-tuning our representation on Flickr images [2].

For fine-tuning the network with hard Flickr images, we again feed these images as-is for training, with the tags as class labels. While we are getting more realistic images, we did notice that the data becomes noisier. Powerful as CNNs, they are still likely to be diluted by the noisy examples over the fine-tuning process[3]. In an noisy open-domain environment, mistakes are unavoidable. But humans are more intelligent: we not just learn to recognize concepts independently, but also build up interconnections and develop theories to help better understand the world [8]. Inspired by this, we want to train CNNs with such relationships - with their simplest form being pair-wise look-alike ones [9, 13]. Such a relationship graph can provide more information of the class and regularize/constrain the network training. A motivating example is "iphone". While Google mostly returns images of the product, on Flickr it is often used to specify the device a photo is taken with - as a result, virtually any image can be tagged as "iphone". Knowing similar-looking categories to "iphone" can intuitively help here.

One way to obtain relationships is through extra knowledge sources like WordNet [31]. However, they are not necessarily developed for the visual domain. Instead, we take a data-driven approach to discover relationships in our data: we assume the network will intrinsically develop connections between different categories when clean examples are offered, and all we have to do is to distill the knowledge out.

We take a simple approach by just testing our network on the training set, and take the confusion matrix as the relationships. Mathematically, for any pair of concepts i and j, the relationship R_{ij} is defined as:

$$R_{ij} = P(i|j) = \frac{\sum_{k \in C_i} CNN(j|I_k)}{|C_i|}, \qquad (1)$$

where C_i is the set of indexes for images that belong to concept i, $|\cdot|$ is the cardinality function, and given pixel values I_k, $CNN(j|I_k)$ is the network's belief on how likely image k belongs to concept i. We want our graph to be sparse, therefore we just used the top K ($K = 5$ in our experiments) and re-normalized the probability mass.

After constructing the relationship graph, we put this graph (represented as a matrix) on top of the seventh layer of the network, so that now the soft-max loss function becomes:

$$L = \sum_k \sum_i R_{il_k} \log(CNN(i|I_k)), \qquad (2)$$

where l_k is the class label. In this way, the network is trained

[2] Flickr images are downloaded using tag search. We use the same query strings as used in Google image search.

[3] In our experiments, we find with the same $\sim$1500 categories and close-to-uniform label distribution, a CNN converged on Google images yields an entropy $\sim$2.8, whereas Flickr gives $\sim$4.0. Note that complete random noise will give $\sim$log(1500)=7.3 and perfectly separable signal close to 0.0.

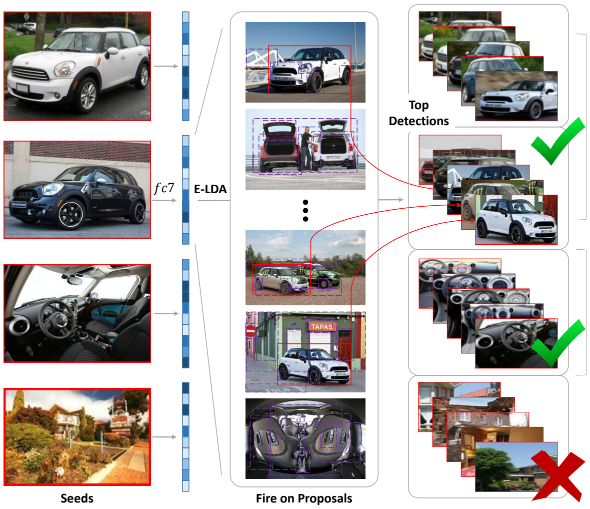

Figure 3. Our pipeline of object localization (for "countryman"). E-LDA detectors [22] trained on $fc7$ features of the seed images are fired on EdgeBox proposals (purple boxes) from other images for nearest neighbors (red boxes), which are then merged to form subcategories. Noisy subcategories are purged with density estimation [10].

to predict the context of a category (in terms of relationships to other categories), and the error is back-propagated through the relationship graph to lower layers. Note that, this extra layer is similar to [49], in which R_{ij} is used to characterize the label-flip noise. Different from them, we do not assume all the categories are *mutually exclusive*, but instead *inter related*. For example, "cat" is a hyper-class of "Siamese cat", and it is reasonable if the model believes some examples of "Siamese cat" are more close to the average image of a "cat". Please see Section 4 for our empirical validation of this assumption. For fear of semantic drift, in this paper we keep the initially learned graph structure fixed, but it would be interesting to see how updating the relationship graph performs (like [9]).

3.3. Localizing Objects

Until now, we have focused on learning a webly-supervised CNN representation based on classification loss. In order to train a webly-supervised object detector we still need to clean the web data and localize the objects in those images to train a detector like R-CNN [19]. Note that this is a non-trivial task, since: 1) the CNN is only trained to distinguish a closed set of classes, unnecessarily aware of all the negative visual world, *e.g*. background clutter; 2) the classification loss encourages the representation to be spatially *invariant* (*e.g*., the network should output "orange" regardless of where it exists in the image or how many there are), which can be a serious issue for localization.

We now describe our subcategory discovery based approach similar to [9] to clean data and localize objects. The whole process is illustrated in Figure 3.

Seeds: We use the full images returned by Google as seed bounding boxes. This is based on Google's bias toward images with a single centered object and a clean background.
Nearest Neighbor Propagation: For each seed, we train an Exemplar-LDA [22] detector using our trained $fc7$ features. Negative statistics for E-LDA are computed over all the downloaded images. This E-LDA detector is then fired on the remaining images to find its top k nearest neighbors. For efficiency, instead of checking all possible windows on each image, we use EdgeBox [60] to propose candidate ones, which also reduces background noise. We set k=10 in our experiments.
Clustering into Subcategories: We then use a publicly-available variant of agglomerative clustering [10] where the nearest neighbor sets are merged iteratively from bottom up to form the final subcategories based on E-LDA similarity scores and density estimation. Note that this is different from [9], but gives similar results while being much more efficient. Some example subcategories are shown in Figure 5.

Finally, we train a R-CNN [19] detector for each category based on all the clustered bounding boxes. Random patches from YFCC [1] are used as negatives. The naive approach would be using the positive examples as-is. Typically, hundreds of instances per category are available for training. While this number is comparable to the VOC 2007 trainval set [14], we also tried to increase positive bounding boxes using two strategies:

EdgeBox Augmentation (EA): We follow [19] to augment the positive training examples. We again use EdgeBox [60] to propose regions of interest on images. Whenever a proposal has a $\geq$0.5 overlapping (measured by intersection over union) with any of the positive bounding box, we add it for training.
Category Expansion (CE): One big advantage of Internet is its nearly infinite data limit. Here we again use the relationship graph to look for similar categories for more training examples. After verification the semantic-relatedness with WordNet [31], we add the examples into training dataset. We believe the extra examples should allow better generalization.

Note both these strategies are only used to increase the amount of positive data for the final SVM to be trained in R-CNN. We do not re-train our CNN representations using these strategies.

4. Experimental Results

We now describe our experimental results. Our goal is to demonstrate that the visual representation learned using two-step webly supervised learning is meaningful. For this, we will do four experiments: 1) First, we will show that our learned CNN can be used for object detection. Here, we use the approach similar to R-CNN [19] where we will fine-tune our learned CNN using VOC data. This is followed by learning SVM-detectors using CNN features. 2)

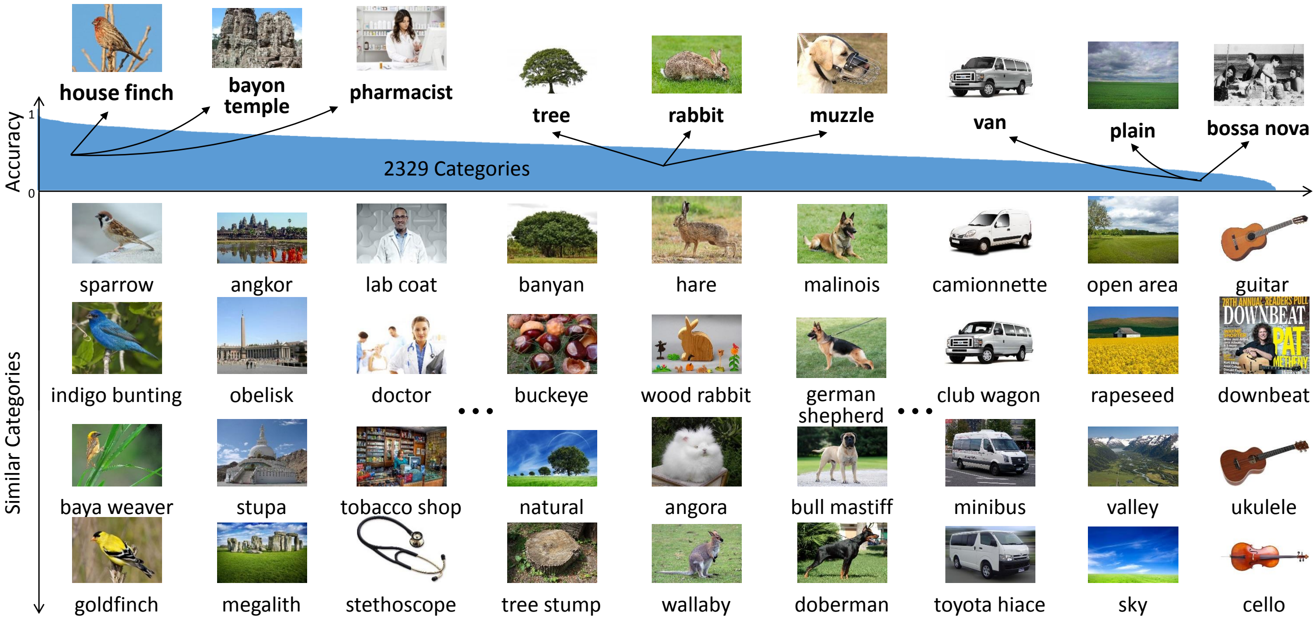

Figure 4. Visualization of the relationships learned from the confusion matrix. The horizontal axis is for categories, which are ranked based on CNN's accuracy. Here we show random examples from three parts of the distribution: top, middle, bottom. It can be seen that the relationships are reasonable: at the top of the distribution the network can recognize well, but when it gets confused, it gets confused to similar categories. Even for bottom ones where the network gets heavily confused, it is confusing between semantically related categories. Somewhat to our surprise, for noisy classes like "bossa nova", the network can figure out it is related to musical instruments.

We will also show that our CNN can be used to clean up the web data: that is, discover subcategories and localize the objects in web images. 3) We will train detectors using the cleaned up web data and evaluate them on VOC data. Note in this case, we will not use any VOC training images. We will only use web images to train both the CNN and the subsequent SVMs. 4) Finally, we will show scene classification results to further showcase the usefulness of the trained representation.

All the networks are trained with the Caffe Toolbox [23]. In total we have 2,240 objects, 89 attributes, and 874 scenes. Two networks are trained on Google: 1) The object-attribute network (GoogleO), where the output dimension is 2,329, and 2) All included network (GoogleA), where the output dimension is 3,203. For the first network, $\sim$1.5 million images are downloaded from Google image search. Combining scene images, $\sim$2.1 million images are used in the second network. We set the batch size as 256 and start with a learning rate of 0.01. The learning rate is reduced by a factor of 10 after every 150K iterations, and we stop training at 450K iterations. For two-stage training, GoogleO is then fine-tuned with $\sim$1.2 million Flickr images. We tested both with (FlickrG) and without (FlickrF) the relationship graph as regularization. Fine-tuning is performed for a total of 100K iterations, with a step size of 30K. As baseline, we also report numbers for CNN learned using Flickr images alone (FlickrS) and combined Google+Flickr images (GFAll). Note in case of GFAll, neither two stage learning or relationship graph constraint is used.

Is Confusion Matrix Informative for Relationships? We first want to show if the network has learned to discover the look-alike relationships between concepts in the confusion matrix. To verify the quality of the network, we take the GoogleO net and visualize the top-5 most confusing concepts (including self) to some of the categories. To ensure our selection has a good coverage, we first rank the diagonal of the confusing matrix (accuracy) in the descending order. Then we randomly sample 3 categories from the top-100, bottom-100, and middle-100 from the list. The visualization and explanations can be found in Figure 4. We can see that the top relationships learned are indeed reasonable.

4.1. PASCAL VOC Object Detection

Next, we test our webly trained CNN model for object detection on the PASCAL VOC. Following the R-CNN pipeline, two sets of experiments are performed on VOC 2007. First, we directly test the generalizability of CNN-representations learned without fine-tuning on VOC data. Second, we fine tune the CNN by back-propagating the error end-to-end using PASCAL trainval set. The fine-tuning procedure is performed 100K iteration, with a step size of 20K. In both cases, $fc7$ features are extracted to represent patches, and a SVM is learned to produce the final score.

We report numbers for all the CNNs on VOC 2007 data in Table 1. Several interesting notes:

- Despite the search engine bias and the noise in the data, our two-stage CNN with graph regularization is on par with ImageNet-trained CNN.

	VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	pers	plant	sheep	sofa	train	tv	**mAP**
	ImageNet [19]	**57.6**	57.9	38.5	**31.8**	23.7	51.2	58.9	**51.4**	20.0	**50.5**	**40.9**	**46.0**	**51.6**	55.9	**43.3**	23.3	**48.1**	35.3	**51.0**	**57.4**	**44.7**
w/o VOC FT	GoogleO [Obj.]	57.1	59.9	35.4	30.5	21.9	53.9	59.5	40.7	18.6	43.3	37.5	41.9	49.6	57.7	38.4	22.8	45.2	37.1	48.0	54.5	42.7
	GoogleA [Obj. + Sce.]	54.9	58.2	35.7	30.7	22.0	54.5	59.9	44.7	19.9	41.0	34.5	40.1	46.8	56.2	40.0	22.2	45.8	36.3	47.5	54.2	42.3
	FlickrS [Flickr Obj.]	50.0	55.9	29.6	26.8	18.7	47.6	56.3	34.4	14.5	35.9	33.3	34.2	43.2	52.2	36.7	21.5	43.3	31.6	48.5	48.4	38.1
	GFAll [All Obj., 1-stage]	52.1	57.8	38.1	25.6	21.2	47.6	56.4	43.8	19.6	42.6	30.3	37.6	45.1	50.8	39.3	22.9	43.5	34.2	48.3	52.2	40.5
	FlickrF [2-stage]	53.9	60.7	37.0	31.6	23.8	**57.7**	60.8	44.1	20.3	46.5	31.5	39.8	49.7	59.0	41.6	23.0	44.4	36.2	49.9	56.2	43.4
	FlickrG [2-stage, Graph]	55.3	**61.9**	**39.1**	29.5	**24.8**	55.1	**62.7**	43.5	**22.7**	49.3	36.6	42.7	48.9	**59.7**	41.2	**25.4**	47.7	**41.9**	48.8	56.8	**44.7**
w/ VOC FT	VOC-Scratch [2]	49.9	60.6	24.7	23.7	20.3	52.5	64.8	32.9	20.4	43.5	34.2	29.9	49.0	60.4	47.5	28.0	42.3	28.6	51.2	50.0	40.7
	ImageNet [19]	64.2	**69.7**	**50.0**	**41.9**	**32.0**	62.6	71.0	**60.7**	**32.7**	58.5	46.5	**56.1**	60.6	66.8	54.2	31.5	52.8	48.9	57.9	**64.7**	**54.2**
	GoogleO	**65.0**	68.1	45.2	37.0	29.6	65.4	73.8	54.0	30.4	57.8	48.7	51.9	**64.1**	64.7	54.0	**32.0**	**54.9**	44.5	57.0	64.0	53.1
	GoogleA	64.2	68.3	42.7	38.7	26.5	65.1	72.4	50.7	28.5	**60.9**	**48.8**	51.2	60.2	65.5	**54.5**	31.1	50.5	**48.5**	56.3	60.3	52.3
	FlickrG	63.7	68.5	46.2	36.4	30.2	**68.4**	**73.9**	56.9	31.4	59.1	46.7	52.4	61.5	**69.2**	53.6	31.6	53.8	44.5	**58.1**	59.6	53.3

Table 1. Results on VOC 2007 (PASCAL data used). Please see Section 4.1 for more details.

	VOC 2012 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	**mAP**
w/ VOC FT	ImageNet [19]	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	**66.2**	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6	49.6
	ImageNet-TV	**73.3**	67.1	46.3	31.7	30.6	59.4	61.0	**67.9**	27.3	**53.1**	39.1	64.1	**60.5**	70.9	57.2	26.1	**59.0**	40.1	56.2	**54.9**	52.3
	GoogleO	72.2	67.3	46.0	**32.3**	**31.6**	**62.6**	62.5	66.5	27.3	52.1	38.9	64.0	59.1	71.6	58.0	27.2	57.6	41.3	56.3	53.7	52.4
	FlickrG	72.7	**68.2**	**47.3**	32.2	30.6	62.3	**62.6**	65.9	**28.1**	52.2	**39.5**	65.1	60.0	**71.7**	**58.2**	**27.3**	58.0	**41.5**	**57.2**	53.8	**52.7**

Table 2. Results on VOC 2012. Since [19] only fine-tuned on the train set, we also report results on trainval (ImageNet-TV) for fairness.

- Training a network directly on noisy and hard Flickr images hurt the learning process. For example, FlickrS gives the worst performance and in fact when a CNN is trained using all the images from Google and Flickr it gives a mAP of 40.5, which is substantially lower than our mAP.
- The proposed two-stage training strategy effectively takes advantage of the more realistic data Flickr provides. Without graph regularization we achieve a mAP of 43.4 (FlickrF). However, adding the graph regularization brings our final FlickrG network on par with ImageNet (mAP = 44.7).

We use the same CNNs for VOC 2012 and report results in Table 2. In this case, our networks outperform the ImageNet pretrained network even after fine-tuning (200K iterations, 40K step size). Note that the original R-CNN paper fine-tuned the ImageNet CNN using train data alone and therefore reports lower performance [19]. For fairness, we fine-tuned both ImageNet network and our networks on combined trainval images (ImageNet-TV). In both VOC 2007 and 2012, our webly supervised CNNs tend to work better for vehicles, probably because we have lots of data for cars and other vehicles ($\sim$500 classes). On the other hand, ImageNet CNN seems to outperform our network on animals [41] (*e.g.* cat). This is probably because ImageNet has a lot more data for animals. It also suggests our CNNs can potentially benefit from more animal categories.

Does web supervision work because the image search engine is CNN-based? One possible hypothesis can be that our approach performs comparably to ImageNet-CNN because Google image search itself uses a trained CNN. To test if this hypothesis is true, we trained a separate CNN using NEIL images downloaded from Google before March 2013 (pre-CNN based image search era). Despite the data being noisier and less ($\sim$450 per category), we observe $\sim$1% performance fall compared to a CNN trained with November 2014 data on the same categories. This indicates that the underlying CNN in Google image search has minimal effect on the training procedure and our network is quite robust to noise.

4.2. Object Localization

In this subsection, we are interested to see if we can detect objects without using a single PASCAL training image. We believe this is possible since we can localize objects automatically in web images with our proposed approach (see Section 3.3). Please refer to Figure 5 for the qualitative results on the training localization we can get with $fc7$ features. Compared to [9], the subcategories we get are less homogeneous (*e.g.* people are not well-aligned, objects in different view points are clustered together). But just because of this more powerful representation (and thus better distance metric), we are able to dig out more signal from the training set - since semantically related images can form clusters and won't be purged as noise when an image is evaluated by its nearest neighbors.

Using localized objects, we train R-CNN based detectors to detect objects on the VOC 2007 test set. We compare our results against [13], who used Google n-grams to expand the categories (*e.g.* "horse" is expanded to "jumping horse", "racing horse" *etc.*) and the models were also directly trained from the web. The results are shown in Table 3. For our approach, we try five different settings: 1) GoogleO: Features are based on GoogleO CNN and the bounding boxes are also extracted only on easy Google im-

VOC 2007 test	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	**mAP**
LEVAN [13]	14.0	36.2	12.5	10.3	9.2	35.0	**35.9**	8.4	**10.0**	17.5	6.5	12.9	**30.6**	27.5	6.0	1.5	18.8	10.3	23.5	16.4	17.1
GoogleO	30.2	34.3	16.7	13.3	6.1	43.6	27.4	22.6	6.9	16.4	10.0	21.3	25.0	35.9	7.6	9.3	21.8	17.3	31.0	18.1	20.7
GoogleA	29.5	38.3	15.1	14.0	9.1	44.3	29.3	24.9	6.9	15.8	9.7	22.6	23.5	34.3	9.7	12.7	21.4	15.8	33.4	19.4	21.5
FlickrG	32.6	42.8	19.3	13.9	9.2	46.6	29.6	20.6	6.8	17.8	10.2	22.4	26.7	40.8	11.7	**14.0**	19.0	19.0	34.0	21.9	22.9
FlickrG-EA	**32.7**	**44.3**	17.9	14.0	**9.3**	**47.1**	26.6	19.2	8.2	18.3	10.0	22.7	25.0	42.5	12.0	12.7	**22.2**	20.9	35.6	18.2	23.0
FlickrG-CE	30.2	41.3	**21.7**	**18.3**	9.2	44.3	32.2	**25.5**	9.8	**21.5**	**10.4**	**26.7**	27.3	**42.8**	**12.6**	13.3	20.4	**20.9**	**36.2**	**22.8**	**24.4**

Table 3. Webly supervised VOC 2007 detection results (No PASCAL data used). Please see Section 4.2 for more details.

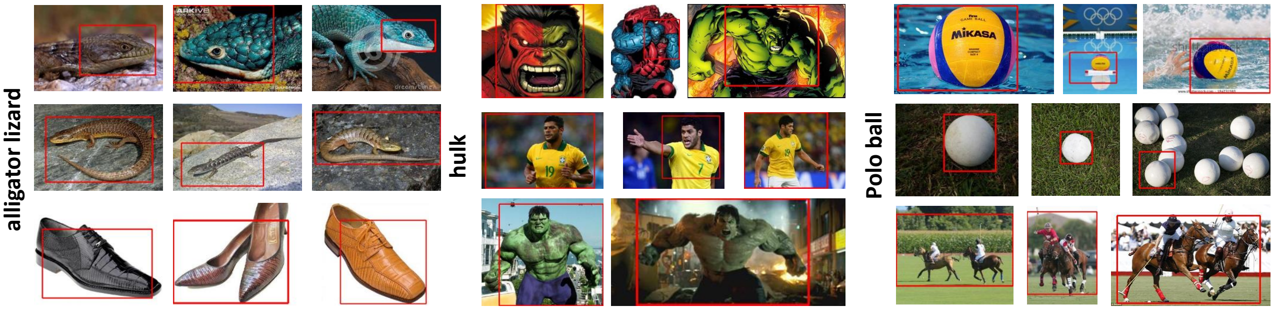

Figure 5. We use the learned CNN representation to discover subcategories and localize positive instances for different categories [9].

Indoor-67	**Accuracy**
ImageNet [59]	56.8
OverFeat [39]	58.4
GoogleO [Obj.]	58.1
FlickrG [Obj.]	59.2
GoogleA [Obj. + Sce.]	**66.5**

Table 4. Scene classification results on MIT Indoor-67. Note that GoogleA has scene categories for training but others do not.

ages; 2) GoogleA: Using GoogleO to extract features instead; 3) FlickrG: Features are based on FlickrG instead; 4) FlickrG-EA: The same Flickr features are used but with EdgeBox augmentation; 5) FlickrG-CE: The Flickr features are used but the positive data includes examples from both original and expanded categories. From the results, we can see that in all cases the CNN based detector boosts the performance a lot.

This demonstrates that our framework could be a powerful way to learn detectors for arbitrary object categories without labeling any training images. We plan to release a service for everyone to train R-CNN detectors on the fly. The code will also be released.

4.3. Scene Classification

To further demonstrate the usage of CNN features directly learned from the web, we also conducted scene classification experiments on the MIT Indoor-67 dataset [37]. For each image, we simply computed the $fc7$ feature vector, which has 4096 dimensions. We did not use any data augmentation or spatial pooling technique, with the only pre-processing step being normalizing the feature vector to unit ℓ_2 length [39]. The default SVM parameters (C=1) were fixed throughout the experiments.

Table 4 summarizes the results on the default train/test split. We can see our web based CNNs achieved very competitive performances: all the three networks achieved an accuracy at least on par with ImageNet pretrained models. Fine-tuning on hard images enhanced the features, but adding scene-related categories gave a huge boost to 66.5 (comparable to the CNN trained on Places database [59], 68.2). This indicates CNN features learned directly from the web are generic and quite powerful.

Moreover, since we can easily get images for semantic labels (*e.g.* actions, n-grams, *etc.*) other than objects or scenes from the web, webly supervised CNN bears a great potential to perform well on many relevant tasks - with the cost as low as providing a query list for that domain.

5. Conclusion

We have presented a two-stage approach to train CNNs using noisy web data. First, we train CNN with easy images downloaded from Google image search. This network is then used to discover structure in the data in terms of similarity relationships. Then we fine-tune the original network on more realistic Flickr images with the relationship graph. We show that our two-stage CNN comes close to the ImageNet pretrained-CNN on VOC 2007, and outperforms on VOC 2012. We report state-of-the-art performance on VOC 2007 without using any VOC training image. Finally, we will like to differentiate webly supervised and unsupervised learning. Webly supervised learning is suited for semantic tasks such as detection, classification (since supervision comes from text). On the other hand, unsupervised learning is useful for generic tasks which might not require semantic invariance (*e.g.*, 3D understanding, grasping).

Acknowledgments: This research is supported by ONR MURI N000141010934, Yahoo-CMU InMind program and a gift from Google. AG and XC were partially supported by Bosch Young Faculty Fellowship and Yahoo Fellowship respectively. The authors would also like to thank Yahoo! for a computing cluster and Nvidia for Tesla GPUs.

References

[1] YFCC dataset. `labs.yahoo.com/news/yfcc100m/`.

[2] P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In *ECCV*. 2014.

[3] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *ICML*, 2009.

[4] T. L. Berg and A. C. Berg. Finding iconic images. In *CVPRW*, 2009.

[5] T. L. Berg and D. A. Forsyth. Animals on the web. In *CVPR*, 2006.

[6] A. Bergamo, L. Bazzani, D. Anguelov, and L. Torresani. Self-taught object localization with deep networks. *arXiv:1409.3964*, 2014.

[7] A. Bergamo and L. Torresani. Exploiting weakly-labeled web images to improve object classification: a domain adaptation approach. In *NIPS*, 2010.

[8] P. Carruthers and P. K. Smith. *Theories of theories of mind*. Cambridge Univ Press, 1996.

[9] X. Chen, A. Shrivastava, and A. Gupta. NEIL: Extracting visual knowledge from web data. In *ICCV*, 2013.

[10] X. Chen, A. Shrivastava, and A. Gupta. Enriching visual knowledge bases via object discovery and segmentation. In *CVPR*, 2014.

[11] D. J. Crandall and D. P. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *ECCV*. 2006.

[12] T. Deselaers, B. Alexe, and V. Ferrari. Weakly supervised localization and learning with generic knowledge. *IJCV*, 2012.

[13] S. K. Divvala, A. Farhadi, and C. Guestrin. Learning everything about anything: Webly-supervised visual concept learning. In *CVPR*, 2014.

[14] M. Everingham, L. VanGool, C. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV 10*.

[15] J. Fan, Y. Shen, N. Zhou, and Y. Gao. Harvesting large-scale weakly-tagged image databases from the web. In *CVPR*, 2010.

[16] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *TPAMI*, 2010.

[17] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from internet image searches. *Proceedings of the IEEE*, 2010.

[18] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for google images. In *ECCV*. 2004.

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[20] E. Golge and P. Duygulu. Conceptmap: Mining noisy web data for concept learning. In *ECCV*. 2014.

[21] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*. 2014.

[22] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*.

[23] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, 2014.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[25] M. P. Kumar, B. Packer, and D. Koller. Self-paced learning for latent variable models. In *NIPS*, 2010.

[26] Y. J. Lee and K. Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011.

[27] L.-J. Li and L. Fei-Fei. OPTIMOL: automatic online picture collection via incremental model learning. *IJCV*, 2010.

[28] Q. Li, J. Wu, and Z. Tu. Harvesting mid-level visual concepts from large-scale internet images. In *CVPR*, 2013.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*. 2014.

[30] E. Mezuman and Y. Weiss. Learning about canonical views from internet image collections. In *NIPS*, 2012.

[31] G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 1995.

[32] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Weakly supervised object recognition with convolutional neural networks. Technical report, 2014.

[33] V. Ordonez, G. Kulkarni, and T. L. Berg. Im2text: Describing images using 1 million captioned photographs. In *NIPS*, 2011.

[34] M. Pandey and S. Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV*, 2011.

[35] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille. Weakly- and semi-supervised learning of a dcnn for semantic image segmentation. *arXiv:1502.02734*, 2015.

[36] D. Pathak, E. Shelhamer, J. Long, and T. Darrell. Fully convolutional multi-class multiple instance learning. *arXiv:1412.7144*, 2014.

[37] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *CVPR*, 2009.

[38] R. Raguram and S. Lazebnik. Computing iconic summaries of general visual concepts. In *CVPRW*, 2008.

[39] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *CVPRW*, 2014.

[40] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv:1412.6596*, 2014.

[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv:1409.0575*, 2014.

[42] K. Saenko and T. Darrell. Unsupervised learning of visual sense models for polysemous words. In *NIPS*, 2009.

[43] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. *TPAMI*, 2011.

[44] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv:1312.6034*, 2013.

[45] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

[46] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.

[47] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *TPAMI*, 2000.

[48] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darrell. On learning to localize objects with minimal supervision. In *ICML*.

[49] S. Sukhbaatar and R. Fergus. Learning from noisy labels with deep neural networks. *arXiv:1406.2080*, 2014.

[50] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014.

[51] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014.

[52] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

[53] L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. In *ECCV*. 2010.

[54] S. Vijayanarasimhan and K. Grauman. Keywords to visual categories: Multiple-instance learning forweakly supervised object categorization. In *CVPR*, 2008.

[55] C. Wang, W. Ren, K. Huang, and T. Tan. Weakly supervised object localization with latent category learning. In *ECCV*. 2014.

[56] X.-J. Wang, L. Zhang, X. Li, and W.-Y. Ma. Annotating images by mining image search results. *TPAMI*, 2008.

[57] Y. Xia, X. Cao, F. Wen, and J. Sun. Well begun is half done: Generating high-quality seeds for automatic image dataset construction from web. In *ECCV*. 2014.

[58] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[59] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.

[60] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*. 2014.