This ICCV2013 paper is the Open Access version, provided by the Computer Vision Foundation.
The authoritative version of this paper is available in IEEE Xplore.

NEIL: Extracting Visual Knowledge from Web Data

Xinlei Chen   Abhinav Shrivastava   Abhinav Gupta
Carnegie Mellon University
www.neil-kb.com

Abstract

We propose NEIL (Never Ending Image Learner), a computer program that runs 24 hours per day and 7 days per week to automatically extract visual knowledge from Internet data. NEIL uses a semi-supervised learning algorithm that jointly discovers common sense relationships (e.g., "Corolla is a kind of/looks similar to Car", "Wheel is a part of Car") and labels instances of the given visual categories. It is an attempt to develop the world's largest visual structured knowledge base with minimum human labeling effort. As of 10th October 2013, NEIL has been continuously running for 2.5 months on 200 core cluster (more than 350K CPU hours) and has an ontology of 1152 object categories, 1034 scene categories and 87 attributes. During this period, NEIL has discovered more than **1700 relationships** *and has labeled more than* **400K** *visual instances.*

1. Motivation

Recent successes in computer vision can be primarily attributed to the ever increasing size of visual knowledge in terms of labeled instances of scenes, objects, actions, attributes, and the contextual relationships between them. But as we move forward, a key question arises: how will we gather this structured visual knowledge on a vast scale? Recent efforts such as ImageNet [8] and Visipedia [30] have tried to harness human intelligence for this task. However, we believe that these approaches lack both the richness and the scalability required for gathering massive amounts of visual knowledge. For example, at the time of submission, only 7% of the data in ImageNet had bounding boxes and the relationships were still extracted via Wordnet.

In this paper, we consider an alternative approach of automatically extracting visual knowledge from Internet scale data. The feasibility of extracting knowledge automatically from images and videos will itself depend on the state-of-the-art in computer vision. While we have witnessed significant progress on the task of detection and recognition, we still have a long way to go for automatically extracting the semantic content of a given image. So, is it really possible to use existing approaches for gathering visual knowledge directly from web data?

1.1. NEIL – Never Ending Image Learner

We propose NEIL, a computer program that runs 24 hours per day, 7 days per week, forever to: (a) semantically understand images on the web; (b) use this semantic understanding to augment its knowledge base with new labeled instances and common sense relationships; (c) use this dataset and these relationships to build better classifiers and detectors which in turn help improve semantic understanding. NEIL is a constrained semi-supervised learning (SSL) system that exploits the big scale of visual data to automatically extract common sense relationships and then uses these relationships to label visual instances of existing categories. It is an attempt to develop the *world's largest visual structured knowledge base* with minimum human effort – one that reflects the factual content of the images on the Internet, and that would be useful to many computer vision and AI efforts. Specifically, NEIL can use web data to extract: (a) Labeled examples of object categories with bounding boxes; (b) Labeled examples of scenes; (c) Labeled examples of attributes; (d) Visual subclasses for object categories; and (e) Common sense relationships about scenes, objects and attributes like "Corolla is a kind of/looks similar to Car", "Wheel is a part of Car", *etc*. (See Figure 1).

We believe our approach is possible for three key reasons:

(a) Macro-vision vs. Micro-vision: We use the term "micro-vision" to refer to the traditional paradigm where the input is an image and the output is some information extracted from that image. In contrast, we define "macro-vision" as a paradigm where the input is a large collection of images and the desired output is extracting significant or interesting patterns in visual data (*e.g.*, car is detected frequently in raceways). These patterns help us to extract common sense relationships. Note, the key difference is that macro-vision does not require us to understand every image in the corpora and extract all possible patterns. Instead, it relies on understanding a few images and statistically combine evidence from these to build our visual knowledge.

(b) Structure of the Visual World: Our approach exploits the structure of the visual world and builds constraints for detection and classification. These global constraints are represented in terms of common sense relationships be-

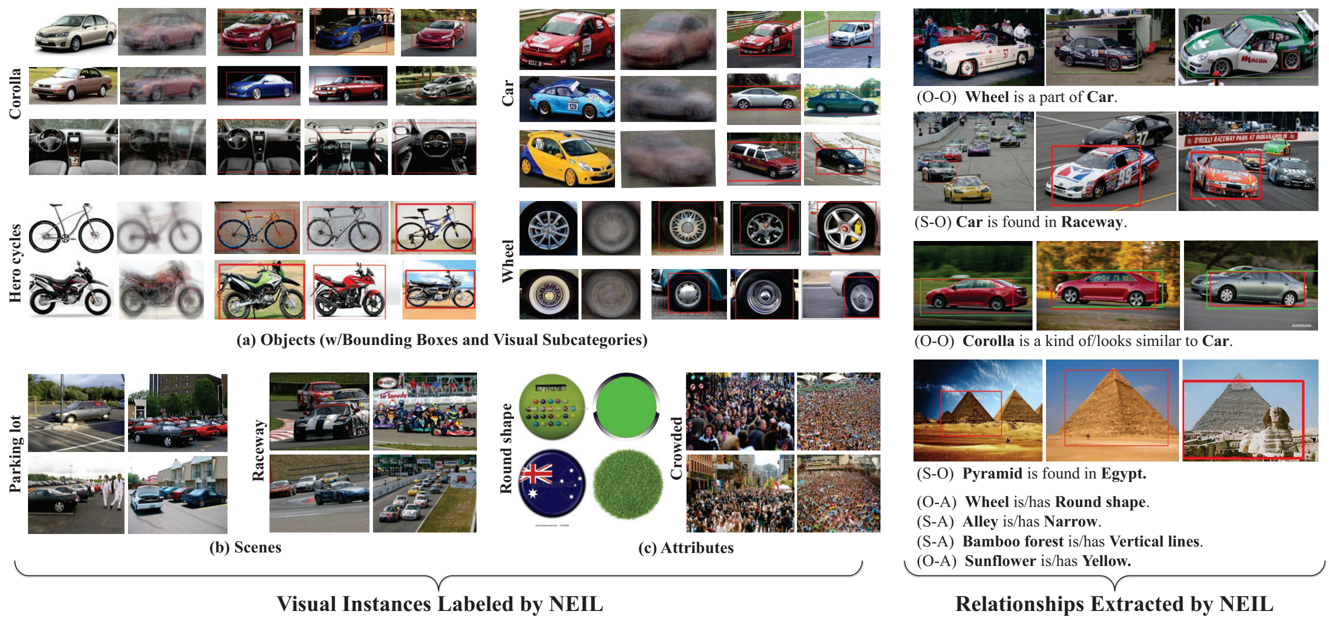

Figure 1. NEIL is a computer program that runs 24 hours a day and 7 days a week to gather visual knowledge from the Internet. Specifically, it simultaneously labels the data and extracts common sense relationships between categories.

tween categories. Most prior work uses manually defined relationships or learns relationships in a supervised setting. Our key insight is that at a large scale one can simultaneously label the visual instances and extract common sense relationships in a joint semi-supervised learning framework.
(c) Semantically driven knowledge acquisition: We use a semantic representation for visual knowledge; that is, we group visual data based on semantic categories and develop relationships between semantic categories. This allows us to leverage text-based indexing tools such as Google Image Search to initialize our visual knowledge base learning.
Contributions: Our main contributions are: (a) We propose a never ending learning algorithm for gathering visual knowledge from the Internet via macro-vision. NEIL has been continuously running for 2.5 months on a 200 core cluster; (b) We are automatically building a large visual structured knowledge base which not only consists of labeled instances of scenes, objects, and attributes but also the relationships between them. While NEIL's core SSL algorithm works with a fixed vocabulary, we also use noun phrases from NELL's ontology [5] to grow our vocabulary. Currently, our growing knowledge base has an ontology of 1152 object categories, 1034 scene categories, and 87 attributes. NEIL has discovered more than **1700 relationships** and labeled more than **400K** visual instances of these categories. (c) We demonstrate how joint discovery of relationships and labeling of instances at a gigantic scale can provide constraints for improving semi-supervised learning.

2. Related Work

Recent work has only focused on extracting knowledge in the form of large datasets for recognition and classification [8, 23, 30]. One of the most commonly used approaches to build datasets is using manual annotations by motivated teams of people [30] or the power of crowds [8, 40]. To minimize human effort, recent works have also focused on active learning [37, 39] which selects label requests that are most informative. However, both of these directions have a major limitation: annotations are expensive, prone to errors, biased and do not scale.

An alternative approach is to use visual recognition for extracting these datasets automatically from the Internet [23, 34, 36]. A common way of automatically creating a dataset is to use image search results and rerank them via visual classifiers [14] or some form of joint-clustering in text and visual space [2, 34]. Another approach is to use a semi-supervised framework [42]. Here, a small amount of labeled data is used in conjunction with a large amount of unlabeled data to learn reliable and robust visual models. These seed images can be manually labeled [36] or the top retrievals of a text-based search [23]. The biggest problem with most of these automatic approaches is that the small number of labeled examples or image search results do not provide enough constraints for learning robust visual classifiers. Hence, these approaches suffer from semantic drift [6]. One way to avoid semantic drift is to exploit additional constraints based on the structure of our visual data. Researchers have exploited a variety of constraints such as those based on visual similarity [11, 15], semantic similarity [17] or multiple feature spaces [3]. However, most of these constraints are weak in nature: for example, visual similarity only models the constraint that visually-similar images should receive the same labels. On the other hand, our visual world is highly structured: object categories share parts and attributes, objects and scenes have strong contextual relationships, *etc*. Therefore, we need a way to capture the rich structure of our visual world and exploit this structure during semi-supervised learning.

In recent years, there have been huge advances in modeling the rich structure of our visual world via contextual relationships. Some of these relationships include: Scene-Object [38], Object-Object [31], Object-Attribute [12, 22, 28], Scene-Attribute [29]. All these relationships can provide a rich set of constraints which can help us improve SSL [4]. For example, scene-attribute relationships such as amphitheaters are circular can help improve semi-supervised learning of scene classifiers [36] and Wordnet hierarchical relationships can help in propagating segmentations [21]. But the big question is: how do we obtain these relationships? One way to obtain such relationships is via text analysis [5, 18]. However, as [40] points out that the visual knowledge we need to obtain is so obvious that no one would take the time to write it down and put it on web.

In this work, we argue that, at a large-scale, one can jointly discover relationships and constrain the SSL problem for extracting visual knowledge and learning visual classifiers and detectors. Motivated by a never ending learning algorithm for text [5], we propose a never ending visual learning algorithm that cycles between extracting global relationships, labeling data and learning classifiers/detectors for building visual knowledge from the Internet. Our work is also related to attribute discovery [33, 35] since these approaches jointly discover the attributes and relationships between objects and attributes simultaneously. However, in our case, we only focus on semantic attributes and therefore our goal is to discover semantic relationships and semantically label visual instances.

3. Technical Approach

Our goal is to extract visual knowledge from the pool of visual data on the web. We define visual knowledge as any information that can be useful for improving vision tasks such as image understanding and object/scene recognition. One form of visual knowledge would be labeled examples of different categories or labeled segments/boundaries. Labeled examples helps us learn classifiers or detectors and improve image understanding. Another example of visual knowledge would be relationships. For example, spatial contextual relationships can be used to improve object recognition. In this paper, we represent visual knowledge in terms of labeled examples of semantic categories and the relationships between those categories. Our knowledge base consists of labeled examples of: (1) Objects (*e.g*., Car, Corolla); (2) Scenes (*e.g*., Alley, Church); (3) Attributes (*e.g*., Blue, Modern). Note that for objects we learn detectors and for scenes we build classifiers; however for the rest of the paper we will use the term detector and classifier interchangeably. Our knowledge base also contains relationships of four types: (1) Object-Object (*e.g*., Wheel is a part of Car);(2) Object-Attribute (*e.g*., Sheep is/has White); (3) Scene-Object (*e.g*., Car is found in Raceway); (4) Scene-Attribute (*e.g*., Alley is/has Narrow).

The outline of our approach is shown in Figure 2. We use Google Image Search to download thousands of images for each object, scene and attribute category. Our method then uses an iterative approach to clean the labels and train detectors/classifiers in a semi-supervised manner. For a given concept (*e.g*., car), we first discover the latent visual sub-categories and bounding boxes for these sub-categories using an exemplar-based clustering approach (Section 3.1). We then train multiple detectors for a concept (one for each sub-category) using the clustering and localization results. These detectors and classifiers are then used for detections on millions of images to learn relationships based on co-occurrence statistics (Section 3.2). Here, we exploit the fact the we are interested in macro-vision and therefore build co-occurrence statistics using only confident detections/classifications. Once we have relationships, we use them in conjunction with our classifiers and detectors to label the large set of noisy images (Section 3.3). The most confidently labeled images are added to the pool of labeled data and used to retrain the models, and the process repeats itself. At every iteration, we learn better classifiers and detectors, which in turn help us learn more relationships and further constrain the semi-supervised learning problem. We now describe each step in detail below.

3.1. Seeding Classifiers via Google Image Search

The first step in our semi-supervised algorithm is to build classifiers for visual categories. One way to build initial classifiers is via a few manually labeled seed images. Here, we take an alternative approach and use text-based image retrieval systems to provide seed images for training initial detectors. For scene and attribute classifiers we directly use these retrieved images as positive data. However, such an approach fails for training object and attribute detectors because of four reasons (Figure 3(a)) – (1) Outliers: Due to the imperfectness of text-based image retrieval, the downloaded images usually have irrelevant images/outliers; (2) Polysemy: In many cases, semantic categories might be overloaded and a single semantic category might have multiple senses (*e.g*., apple can mean both the company and the fruit); (3) Visual Diversity: Retrieved images might have high intra-class variation due to different viewpoint, illumination *etc*.; (4) Localization: In many cases the retrieved image might be a scene without a bounding-box and hence one needs to localize the concept before training a detector.

Most of the current approaches handle these problems via clustering. Clustering helps in handling visual diversity [9] and discovering multiple senses of retrieval (polysemy) [25]. It can also help us to reject outliers based on distances from cluster centers. One simple way to cluster

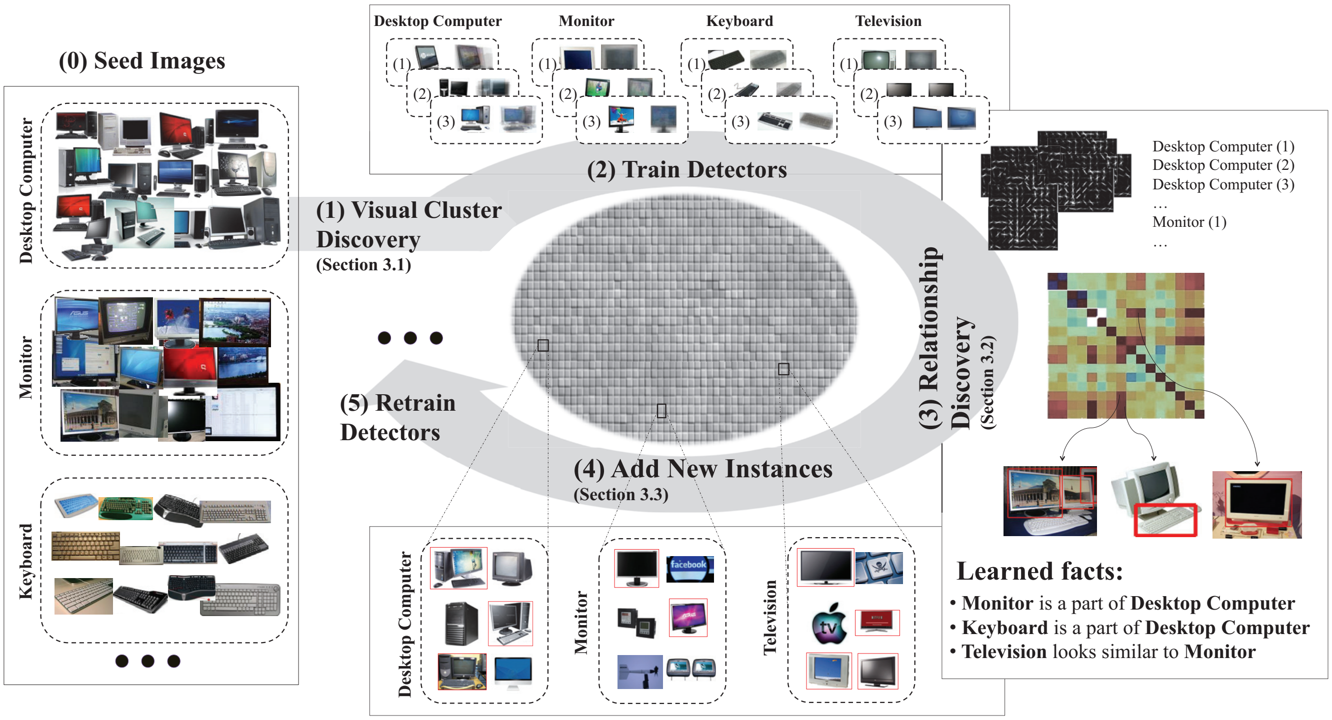

Figure 2. Outline of Iterative Approach

would be to use K-means on the set of all possible bounding boxes and then use the representative clusters as visual sub-categories. However, clustering using K-means has two issues: (1) High Dimensionality: We use the Color HOG (CHOG) [20] representation and standard distance metrics do not work well in such high-dimensions [10]; (2) Scalability: Most clustering approaches tend to partition the complete feature space. In our case, since we do not have bounding boxes provided, every image creates millions of data points and the majority of the datapoints are outliers. Recent work has suggested that K-means is not scalable and has bad performance in this scenario since it assigns membership to every data point [10].

Instead, we propose to use a two-step approach for clustering. In the first step, we mine the set of downloaded images from Google Image Search to create candidate object windows. Specifically, every image is used to train a detector using recently proposed exemplar-LDA [19]. These detectors are then used for dense detections on the same set of downloaded images. We select the top K windows which have high scores from multiple detectors. Note that this step helps us prune out outliers as the candidate windows are selected via representativeness (how many detectors fire on them). For example, in Figure 3, none of the tricycle detectors fire on the outliers such as circular dots and people eating, and hence these images are rejected at this candidate widow step. Once we have candidate windows, we cluster them in the next step. However, instead of using the high-dimensional CHOG representation for clustering, we use the detection signature of each window (represented as a vector of seed detector ELDA scores on the window) to create a $K \times K$ affinity matrix. The (i, j) entry in the affinity matrix is the dot product of this vector for windows i and j. Intuitively, this step connects candidate windows if the same set of detectors fire on both windows. Once we have the affinity matrix, we cluster the candidate windows using the standard affinity propagation algorithm [16]. Affinity propagation also allows us to extract a representative window (prototype) for each cluster which acts as an iconic image for the object [32] (Figure 3). After clustering, we train a detector for each cluster/sub-category using three-quarters of the images in the cluster. The remaining quarter is used as a validation set for calibration.

3.2. Extracting Relationships

Once we have initialized object detectors, attribute detectors, attribute classifiers and scene classifiers, we can use them to extract relationships automatically from the data. The key idea is that we do not need to understand each and every image downloaded from the Internet but instead understand the statistical pattern of detections and classifications at a large scale. These patterns can be used to select the top-N relationships at every iteration. Specifically, we extract four different kinds of relationships:

Object-Object Relationships: The first kind of relationship we extract are object-object relationships which include: (1) Partonomy relationships such as "Eye is a part of Baby"; (2) Taxonomy relationships such as "BMW 320 is a kind of Car"; and (3) Similarity relationships such as

(a) Google Image Search for "tricycle"

(b) Sub-category Discovery

Figure 3. An example of how clustering handles polysemy, intraclass variation and outlier removal (a). The bottom row shows our discovered clusters.

"Swan looks similar to Goose". To extract these relationships, we first build a co-detection matrix O_0 whose elements represent the probability of simultaneous detection of object categories i and j. Intuitively, the co-detection matrix has high values when object detector i detects objects inside the bounding box of object j with high detection scores. To account for detectors that fire everywhere and images which have lots of detections, we normalize the matrix O_0. The normalized co-detection matrix can be written as: $N_1^{-\frac{1}{2}} O_0 N_2^{-\frac{1}{2}}$, where N_1 and N_2 are out-degree and in-degree matrix and (i, j) element of O_0 represents the average score of top-detections of detector i on images of object category j. Once we have selected a relationship between pair of categories, we learn its characteristics in terms of mean and variance of relative locations, relative aspect ratio, relative scores and relative size of the detections. For example, the nose-face relationship is characterized by low relative window size (nose is less than 20% of face area) and the relative location that nose occurs in center of the face. This is used to define a compatibility function $\psi_{i,j}(\cdot)$ which evaluates if the detections from category i and j are compatible or not. We also classify the relationships into the two semantic categories (part-of, taxonomy/similar) using relative features to have a human-communicable view of visual knowledge base.

Object-Attribute Relationships: The second type of relationship we extract is object-attribute relationships such as "Pizza has Round Shape", "Sunflower is Yellow" *etc*. To extract these relationships we use the same methodology where the attributes are detected in the labeled examples of object categories. These detections and their scores are then used to build a normalized co-detection matrix which is used to find the top object-attribute relationships.

Scene-Object Relationships: The third type of relationship extracted by our algorithm includes scene-object relationships such as "Bus is found in Bus depot" and "Monitor is found in Control room". For extracting scene-object relationships, we use the object detectors on randomly sampled images of different scene classes. The detections are then used to create the normalized co-presence matrix (similar to object-object relationships) where the (i, j) element represents the likelihood of detection of instance of object category i and the scene category class j.

Scene-Attribute Relationships: The fourth and final type of relationship extracted by our algorithm includes scene-attribute relationships such as "Ocean is Blue", "Alleys are Narrow", *etc*. Here, we follow a simple methodology for extracting scene-attribute relationships where we compute co-classification matrix such that the element (i, j) of the matrix represents average classification scores of attribute i on images of scene j. The top entries in this co-classification matrix are used to extract scene-attribute relationships.

3.3. Retraining via Labeling New Instances

Once we have the initial set of classifiers/detectors and the set of relationships, we can use them to find new instances of different objects and scene categories. These new instances are then added to the set of labeled data and we retrain new classifiers/detectors using the updated set of labeled data. These new classifiers are then used to extract more relationships which in turn are used to label more data and so on. One way to find new instances is directly using the detector itself. For instance, using the car detector to find more cars. However, this approach leads to semantic drift. To avoid semantic drift, we use the rich set of relationships we extracted in the previous section and ensure that the new labeled instances of car satisfy the extracted relationships (*e.g.*, has wheels, found in raceways *etc*.)

Mathematically, let $\mathcal{R}_\mathcal{O}$, $\mathcal{R}_\mathcal{A}$ and $\mathcal{R}_\mathcal{S}$ represent the set of object-object, object-attribute and scene-object relationships at iteration t. If $\phi_i(\cdot)$ represents the potential from object detector i, $\omega_k(\cdot)$ represents the scene potential, and $\psi_{i,j}(\cdot)$ represent the compatibility function between two object categories i, j, then we can find the new instances of object category i using the contextual scoring function given below:

$$\phi_i(x) + \sum_{i,j \in \mathcal{R}_O \bigcup \mathcal{R}_A} \phi_j(x_l)\psi_{i,j}(x, x_l) + \sum_{i,k \in \mathcal{R}_S} \omega_k(x)$$

where x is the window being evaluated and x_l is the top-detected window of related object/attribute category. The above equation has three terms: the first term is appearance term for the object category itself and is measured by the

Figure 4. Qualitative Examples of Bounding Box Labeling Done by NEIL

score of the SVM detector on the window x. The second term measures the compatibility between object category i and the object/attribute category j if the relationship (i, j) is part of the catalogue. For example, if "Wheel is a part of Car" exists in the catalogue then this term will be the product of the score of wheel detector and the compatibility function between the wheel window (x_l) and the car window (x). The final term measures the scene-object compatibility. Therefore, if the knowledge base contains the relationship "Car is found in Raceway", this term boosts the "Car" detection scores in the "Raceway" scenes.

At each iteration, we also add new instances of different scene categories. We find new instances of scene category k using the contextual scoring function given below:

$$\omega_k(x) + \sum_{m,k \in \mathcal{R}_{A'}} \omega_m(x) + \sum_{i,k \in \mathcal{R}_S} \phi_i(x_l)$$

where $\mathcal{R}_{A'}$ represents the catalogue of scene-attribute relationships. The above equation has three terms: the first term is the appearance term for the scene category itself and is estimated using the scene classifier. The second term is the appearance term for the attribute category and is estimated using the attribute classifier. This term ensures that if a scene-attribute relationship exists then the attribute classifier score should be high. The third and the final term is the appearance term of an object category and is estimated using the corresponding object detector. This term ensures that if a scene-object relationship exists then the object detector should detect objects in the scene.

Implementation Details: To train scene & attribute classifiers, we first extract a 3912 dimensional feature vector from each image. The feature vector includes 512D GIST [27] features, concatenated with bag of words representations for SIFT [24], HOG [7], Lab color space, and Texton [26]. The dictionary sizes are 1000, 1000, 400, 1000, respectively. Features of randomly sampled windows from other categories are used as negative examples for SVM training and hard mining. For the object and attribute section, we use CHOG [20] features with a bin size of 8. We train the detectors using latent SVM model (without parts) [13].

4. Experimental Results

We demonstrate the quality of visual knowledge by qualitative results, verification via human subjects and quantitative results on tasks such as object detection and scene recognition.

4.1. NEIL Statistics

While NEIL's core algorithm uses a fixed vocabulary, we use noun phrases from NELL [5] to grow NEIL's vocabulary. As of 10$^{\text{th}}$ October 2013, NEIL has an ontology of 1152 object categories, 1034 scene categories and 87 attributes. It has downloaded more than 2 million images for extracting the current structured visual knowledge. For bootstrapping our system, we use a few seed images from ImageNet [8], SUN [41] or the top-images from Google Image Search. For the purposes of extensive experimental evaluation in this paper, we ran NEIL on steroids (200 cores as opposed to 30 cores used generally) for the last 2.5 months. NEIL has completed 16 iterations and it has labeled more than 400K visual instances (including 300,000 objects with their bounding boxes). It has also extracted 1703 common sense relationships. Readers can browse the current visual knowledge base and download the detectors from: www.neil-kb.com

4.2. Qualitative Results

We first show some qualitative results in terms of extracted visual knowledge by NEIL. Figure 4 shows the extracted visual sub-categories along with a few labeled instances belonging to each sub-category. It can be seen from the figure that NEIL effectively handles the intra-class variation and polysemy via the clustering process. The purity and diversity of the clusters for different concepts indicate that contextual relationships help make our system robust to semantic drift and ensure diversity. Figure 5 shows some of the qualitative examples of scene-object and object-object relationships extracted by NEIL. It is effective in using a few confident detections to extract interesting relationships. Figure 6 shows some of the interesting scene-attribute and object-attribute relationships extracted by NEIL.

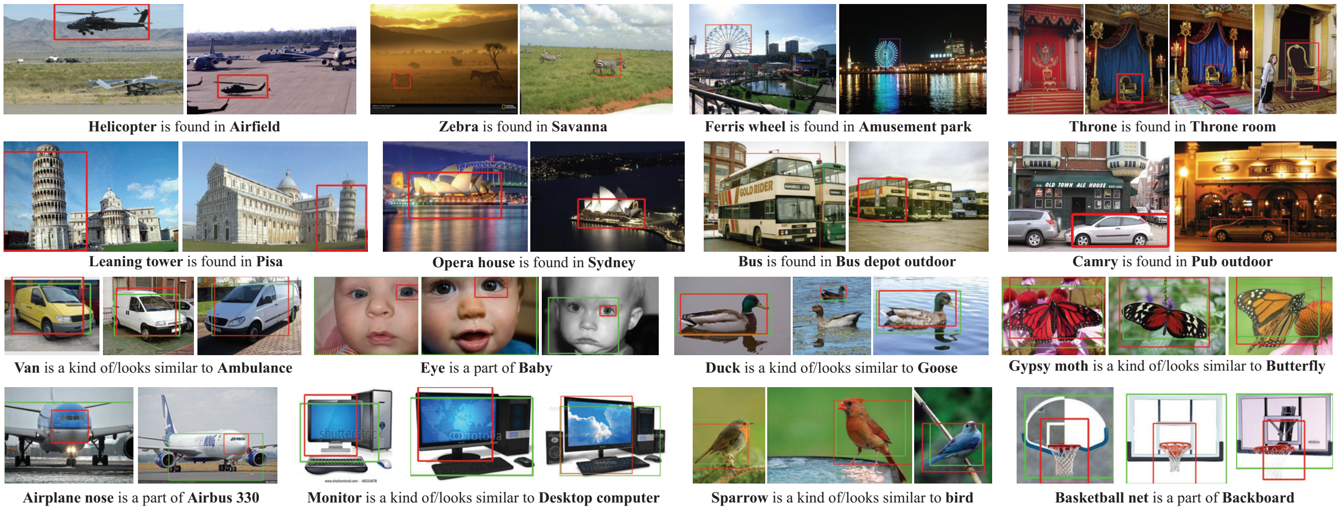

Figure 5. Qualitative Examples of Scene-Object (rows 1-2) and Object-Object (rows 3-4) Relationships Extracted by NEIL

4.3. Evaluating Quality via Human Subjects

Next, we want to evaluate the quality of extracted visual knowledge by NEIL. It should be noted that an extensive and comprehensive evaluation for the whole NEIL system is an extremely difficult task. It is impractical to evaluate each and every labeled instance and each and every relationship for correctness. Therefore, we randomly sample the 500 visual instances and 500 relationships, and verify them using human experts. At the end of iteration 6, 79% of the relationships extracted by NEIL are correct, and 98% of the visual data labeled by NEIL has been labeled correctly. We also evaluate the per iteration correctness of relationships: At iteration 1, more than 96% relationships are correct and by iteration 3, the system stabilizes and 80% of extracted relationships are correct. While currently the system does not demonstrate any major semantic drift, we do plan to continue evaluation and extensive analysis of knowledge base as NEIL grows older. We also evaluate the quality of bounding-boxes generated by NEIL. For this we sample 100 images randomly and label the ground-truth bounding boxes. On the standard intersection-over-union metric, NEIL generates bounding boxes with 0.78 overlap on average with ground-truth. To give context to the difficulty of the task, the standard Objectness algorithm [1] produces bounding boxes with 0.59 overlap on average.

4.4. Using Knowledge for Vision Tasks

Finally, we want to demonstrate the usefulness of the visual knowledge learned by NEIL on standard vision tasks such as object detection and scene classification. Here, we will also compare several aspects of our approach: (a) We first compare the quality of our automatically labeled dataset. As baselines, we train classifiers/detectors directly on the seed images downloaded from Google Image Search. (b) We compare NEIL against a standard bootstrapping approach which does not extract/use relationships. (c) Finally, we will demonstrate the usefulness of relationships by detecting and classifying new test data with and without the learned relationships.

Scene Classification: First we evaluate our visual knowledge for the task of scene classification. We build a dataset of 600 images (12 scene categories) using Flickr images. We compare the performance of our scene classifiers against the scene classifiers trained from top 15 images of Google Image Search (our seed classifier). We also compare the performance with standard bootstrapping approach without using any relationship extraction. Table 1 shows the results. We use mean average precision (mAP) as the evaluation metric. As the results show, automatic relationship extraction helps us to constrain the learning problem, and so the learned classifiers give much better performance. Finally, if we also use the contextual information from NEIL relationships we get a significant boost in performance.

Table 1. mAP performance for scene classification on 12 categories.

	mAP
Seed Classifier (15 Google Images)	0.52
Bootstrapping (without relationships)	0.54
NEIL Scene Classifiers	0.57
NEIL (Classifiers + Relationships)	**0.62**

Object Detection: We also evaluate our extracted visual knowledge for the task of object detection. We build a dataset of 1000 images (15 object categories) using Flickr data for testing. We compare the performance against object detectors trained directly using (top-50 and top-450) images from Google Image Search. We also compare the performance of detectors trained after aspect-ratio, HOG clustering and our proposed clustering procedure. Table 2 shows the detection results. Using 450 images from Google image search decreases the performance due to noisy retrievals. While other clustering methods help, the gain by our clustering procedure is much larger. Finally, detectors trained using NEIL work better than standard bootstrapping.

Monitor is found in **Control room**
Washing machine is found in **Utility room**
Siberian tiger is found in **Zoo**
Baseball is found in **Butters box**
Bullet train is found in **Train station platform**
Cougar looks similar to **Cat**
Urn looks similar to **Goblet**
Samsung galaxy is a kind of **Cellphone**
Computer room is/has **Modern**
Hallway is/has **Narrow**
Building facade is/has **Check texture**
Trading floor is/has **Crowded**
Umbrella looks similar to **Ferris wheel**
Bonfire is found in **Volcano**

Figure 6. Examples of extracted common sense relationships.

Table 2. mAP performance for object detection on 15 categories.

	mAP
Latent SVM (50 Google Images)	0.34
Latent SVM (450 Google Images)	0.28
Latent SVM (450, Aspect Ratio Clustering)	0.30
Latent SVM (450, HOG-based Clustering)	0.33
Seed Detector (NEIL Clustering)	0.44
Bootstrapping (without relationships)	0.45
NEIL Detector	0.49
NEIL Detector + Relationships	**0.51**

Acknowledgements: This research was supported by ONR MURI N000141010934 and a gift from Google. The authors would like to thank Tom Mitchell and David Fouhey for insightful discussions. We would also like to thank our computing clusters `warp` and `workhorse` for doing all the hard work!

References

[1] B. Alexe, T. Deselares, and V. Ferrari. What is an object? In *TPAMI*, 2010. 7

[2] T. Berg and D. Forsyth. Animals on the web. In *CVPR*, 2006. 2

[3] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, 1998. 2

[4] A. Carlson, J. Betteridge, E. R. H. Jr., and T. M. Mitchell. Coupling semi-supervised learning of categories and relations. In *NAACL HLT Workskop on SSL for NLP*, 2009. 3

[5] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, 2010. 2, 3, 6

[6] J. R. Curran, T. Murphy, and B. Scholz. Minimising semantic drift with mutual exclusion bootstrapping. In *Pacific Association for Computational Linguistics*, 2007. 2

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 6

[8] J. Deng, W. Dong, R. Socher, J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2, 6

[9] S. Divvala, A. Efros, and M. Hebert. How important are 'deformable parts' in the deformable parts model? In *ECCV, Parts and Attributes Workshop*, 2012. 3

[10] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. Efros. What makes Paris look like Paris? *SIGGRAPH*, 2012. 4

[11] S. Ebert, D. Larlus, and B. Schiele. Extracting structures in image collections for object recognition. In *ECCV*, 2010. 2

[12] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 3

[13] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *TPAMI*, 2010. 6

[14] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for Google images. In *ECCV*, 2004. 2

[15] R. Fergus, Y. Weiss, and A. Torralba. Semi-supervised learning in gigantic image collections. In *NIPS*. 2009. 2

[16] B. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 2007. 4

[17] M. Guillaumin, J. Verbeek, and C. Schmid. Multimodal semi-supervised learning for image classification. In *CVPR*, 2010. 2

[18] A. Gupta and L. S. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *ECCV*, 2008. 3

[19] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*. 2012. 4

[20] S. Khan, F. Anwer, R. Muhammad, J. van de Weijer, A. Joost, M. Vanrell, and A. Lopez. Color attributes for object detection. In *CVPR*, 2012. 4, 6

[21] D. Kuettel, M. Guillaumin, and V. Ferrari. Segmentation propagation in ImageNet. In *ECCV*, 2012. 3

[22] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009. 3

[23] L.-J. Li, G. Wang, and L. Fei-Fei. OPTIMOL: Automatic object picture collection via incremental model learning. In *CVPR*, 2007. 2

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 6

[25] A. Lucchi and J. Weston. Joint image and word sense discrimination for image retrieval. In *ECCV*, 2012. 3

[26] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *PAMI*, 2004. 6

[27] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 2001. 6

[28] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, 2011. 3

[29] G. Patterson and J. Hays. SUN attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012. 3

[30] P. Perona. Visions of a Visipedia. *Proceedings of IEEE*, 2010. 1, 2

[31] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, 2007. 3

[32] R. Raguram and S. Lazebnik. Computing iconic summaries of general visual concepts. In *Workshop on Internet Vision*, 2008. 4

[33] M. Rastegari, A. Farhadi, and D. Forsyth. Attribute discovery via predictable discriminative binary codes. In *ECCV*, 2012. 3

[34] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. In *ICCV*, 2007. 2

[35] V. Sharmanska, N. Quadrianto, and C. H. Lampert. Augmented attribute representations. In *ECCV*, 2012. 3

[36] A. Shrivastava, S. Singh, and A. Gupta. Constrained semi-supervised learning using attributes and comparative attributes. In *ECCV*, 2012. 2, 3

[37] B. Siddiquie and A. Gupta. Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *CVPR*, 2010. 2

[38] E. Sudderth, A. Torralba, W. T. Freeman, and A. Wilsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV*, 2005. 3

[39] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. In *CVPR*, 2011. 2

[40] L. von Ahn and L. Dabbish. Labeling images with a computer game. In *SIGCHI*, 2004. 2, 3

[41] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba. SUN database: Large scale scene recognition from abbey to zoo. In *CVPR*, 2010. 6

[42] X. Zhu. Semi-supervised learning literature survey. Technical report, CS, UW-Madison, 2005. 2