Fine-Grained Recognition as HSnet Search for Informative Image Parts

Michael Lam, Behrooz Mahasseni†, Sinisa Todorovic
Oregon State University
Corvallis, OR
{lamm,sinisa}@oregonstate.edu †behrooz.mahasseni@gmail.com

Abstract

This work addresses fine-grained image classification. Our work is based on the hypothesis that when dealing with subtle differences among object classes it is critical to identify and only account for a few informative image parts, as the remaining image context may not only be uninformative but may also hurt recognition. This motivates us to formulate our problem as a sequential search for informative parts over a deep feature map produced by a deep Convolutional Neural Network (CNN). A state of this search is a set of proposal bounding boxes in the image, whose "informativeness" is evaluated by the heuristic function (H), and used for generating new candidate states by the successor function (S). The two functions are unified via a Long Short-Term Memory network (LSTM) into a new deep recurrent architecture, called HSnet. Thus, HSnet (i) generates proposals of informative image parts and (ii) fuses all proposals toward final fine-grained recognition. We specify both supervised and weakly supervised training of HSnet depending on the availability of object part annotations. Evaluation on the benchmark Caltech-UCSD Birds 200-2011 and Cars-196 datasets demonstrate our competitive performance relative to the state of the art.

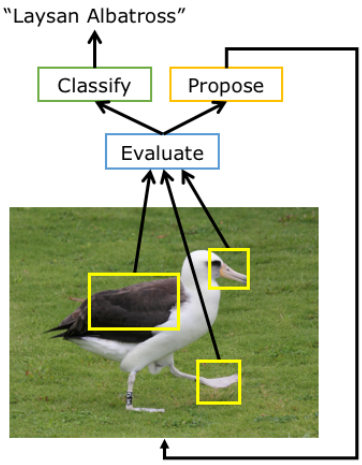


Figure 1: Overview of our approach. Given an image, we use HSnet to sequentially search for discriminative bounding boxes in the image, and fuse all uncovered image parts for fine-grained recognition. HSnet provides a unified framework to jointly learn the heuristic function, which evaluates the search states, and the successor function, which proposes new states in the search space.

1. Introduction

This paper addresses the problem of fine-grained object recognition. Recent work has made significant progress in terms of improving accuracy on an increasing number of object classes [26, 21, 16, 38]. In contrast to general object recognition, where contextual cues are widely considered important, fine-grained recognition has been shown to benefit from identifying critical object parts and learning only off of these parts to discriminate among similar classes [4, 5, 44, 45, 8]. In this paper, we continue this research direction by introducing and evaluating a new deep search-based framework.

It seems that our line of work stands somewhat isolated, but still rather necessary, in the context of recent advances in deep learning for various vision problems, including object tracking [34], activity recognition [27], as well as fine-grained object recognition [22, 38]. All these approaches demonstrate significant performance improvements when the initial training dataset is augmented with additional noisy data – e.g., for learning a tracker, by random sampling around ground truth trajectories, or for learning a fine-grained object detector, by downloading noisy results of Google searches on the Internet for images of fine-grained classes. Thus, the recent findings support and motivate a flurry of new research on how to obtain more training data from various multimodal sources, as this typically leads to better performance of deep methods. However, in a wide range of applications requiring fine-grained recognition, it is very difficult (if not impossible) to obtain additional ground truth or "noisy" annotations (e.g. military, bi-

ological images of fossils). To address these applications, in this paper, we focus on how to more optimally manipulate existing data so as to extract most discriminative features and remove background for reliable fine-grained recognition.

Our approach rests on the assumption that subtle differences among very similar but distinct object classes usually take the form of object parts. Thus, these parts are bound to produce the most discriminative features for fine-grained recognition. Since the objects considered are similar, it follows that the remainder of the objects' spatial extents are shared among the classes, and thus are likely to produce confusing features for fine-grained recognition. As the total number, locations, shapes, and often semantic meaning of these parts are not known a priori, we develop a search-based approach that

- sequentially uncovers discriminative image parts, and
- reasons over the entire search trace for recognition.

Fig. 1 shows an overview of our approach. An image in our approach defines a search space of the image's bounding boxes, represented by deep features of a convolutional neural network (CNN). In this search space, we run a search algorithm, which for a given search state proposes and moves to a new state, until a time bound. A search state at a given time is defined by bounding box proposals visited until that time. The search is defined by two functions. The successor function for the current state proposes a new state in the search space. The heuristic function scores the states, i.e., all bounding box visited in the image, and in this way guides the search toward the best image parts for recognition. When the search time expires, a classifier over the last state is used for recognition.

Our main contribution is a formulation of the new deep architecture, called HSnet, for computing the above heuristic and successor functions of our sequential search in the image. HSnet is grounded via the CNN to an image, and consists of the three components: H-layer for computing the heuristic function, S-layer for realizing the successor function, and Long Short-Term Memory (LSTM) [14] for capturing long-range dependencies along the search trajectory. Thus, the role of HSnet is twofold: to evaluate bounding box candidates and propose new bounding box candidates. Since LSTM has memory, our sequential search is not greedy. That is, the LSTM's memory enables our cumulative definition of a search state as a set of all bounding boxes visited before that state. Consequently, HSnet has a built-in robustness mechanism for handling uncertainty (e.g. occlusion, missing parts, shape deformations), as recognition does not hinge entirely on the very last set of bounding boxes uncovered when the search ends.

Evaluation on the benchmark Caltech-UCSD Birds 200-2011 and Cars-196 datasets demonstrate our competitive performance relative to the state of the art.

In the following, Sec. 2 places our approach in the context of prior work, Sec. 3 specifies our approach, and Sec. 4 presents our results.

2. Related Work

Fine-Grained Recognition. There is a a wide range of methods that have been developed for fine-grained object recognition [9, 40, 41, 39, 4, 5, 44, 45, 8, 26, 21]. These approaches seek to distinguish subtle differences among similar classes typically by identifying and reasoning about the layout structure of object parts present in fine-grained classes [4, 5, 44, 45, 8].

Our approach is related to existing work aimed at finding object parts using little or no supervision of parts [9, 11, 7, 16]. For example, recent work [21] combines alignment and co-segmentation to generate parts without annotations. Also, in [16], informative object parts are learned without needing part annotations by augmenting an existing CNN architecture with a differentiable spatial transformation module. In contrast to these methods, our HSnet has built-in refinement mechanism to search for increasingly more informative parts and thus improve recognition, as well as robustness mechanism against wrongly identified parts during inference.

Object Detection. Our work is most similar to recent object detection methods [12, 31, 30]. Object detection has been applied to fine-grained classification in prior work, where R-CNN [12, 31, 44] is trained to detect object parts. In contrast to these works, which predict object parts to classify an image in one shot, we employ sequential reasoning leveraging LSTM to search for object parts in order to classify an image. Additionally, we cannot directly use these approaches since their object proposals are based on objectness, whereas we need object parts, and parts are not objects.

Search. There is a host of search-based methods in vision [13, 10, 19, 29, 32]. For example, minimizing energy of graphical models has been addressed using Monte-Carlo Markov Chain (MCMC), which in turn can be viewed as a search [3]. Our approach is closely related to those methods that seek to learn the heuristic and successor functions of the search on training data, instead of using heuristics [2, 35, 18, 32, 25]. These methods typically define the heuristic and successor functions as separate modules that are trained disjointly. In contrast, we parameterize our heuristic and successor functions such that they have the same predictor for evaluating and proposing search candidates. Moreover, we specify a unified end-to-end learning of the heuristic and successor functions.

Attention Models. Our approach is also similar to meth-

ods for estimating visual attention. Attention models are aimed at identifying discriminative image parts that are most responsible for recognition [37, 28, 6, 43]. While the majority of attention models focus on one bounding box or one part of an image at a time (e.g. [6]), our HSnet identifies and reasons about multiple parts of the image at a time. Our approach is closest to Jaderberg et al. [16], since their method can be interpreted as a multi-attention estimation; however, we also employ sequential reasoning and search.

3. Technical Approach

3.1. Search Overview

This section formulates our search framework. Search is defined in a space of *search states* $s \in \mathscr{S}$, where $\mathscr{S}$ may be computationally intractable or non-enumerable, as is our case. A search algorithm $\mathcal{A}$ is an iterative adaptive program that yields a trajectory from a given initial state s_0 to an end state s_τ: $[s_0, s_1, \ldots, s_\tau]$. $\mathcal{A}$ is typically guided by two functions, called the heuristic and successor function.

Each state s can be assigned a score using the heuristic function $\mathcal{H} : s \mapsto \mathbb{R}$. There are many ways to define $\mathcal{H}$. For example, when the goal state is known, A* search uses the heuristic function specified in terms of a distance to the goal state. Recent work seeks to learn $\mathcal{H}$ on training data [2, 35, 18, 32, 25]. The end state s_τ may be reached when the search time expires, or alternatively when the score $\mathcal{H}(s_\tau)$ is greater than a threshold. The literature also presents other more sophisticated specifications of when to stop the search.

In the case of intractable $\mathscr{S}$, search requires the successor function $\mathcal{S}$ for partially constructing the search space. $\mathcal{S}$ "expands" a given state s to its "neighbors" $\{s_1, s_2, ..., s_k\} \subseteq \mathcal{S}$, i.e., constructs a finite set of new states that can be reached by search from s. The specifics of deciding what and when to expand are defined by a particular search algorithm. For instance, in greedy search, the neighbor with the highest $\mathcal{H}$ score is the next one to expand for a given state.

In the next section, we formulate fine-grained recognition as search.

3.2. Search in the Space of Image Bounding Boxes

We perform search over a deep feature map, produced by a CNN, to find the most informative bounding boxes in the image for recognition. Our search space $\mathscr{S}$ is thus defined over bounding box configurations in the deep feature map. Fig. 2 illustrates a sample search trajectory, and provides an overview of how our search-based inference works.

The CNN maps an image to a deep feature map, $x \in \mathbb{R}_{\geq 0}^{H \times W \times C}$, where H is the height, W is the width, and C is the number of channels. The convolutional layers late in deep architectures have been shown to produce informative object-characteristic features [42] and thus we will use this.

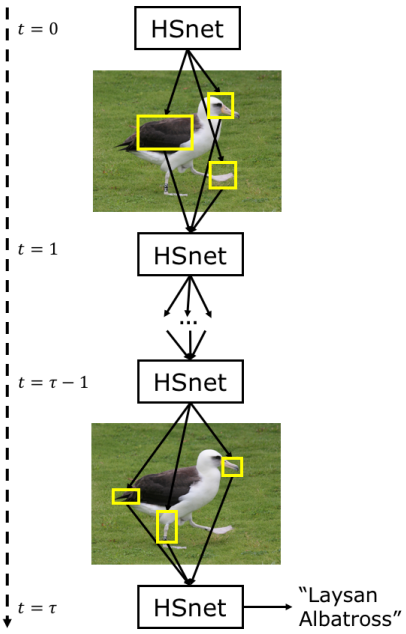


Figure 2: Our search framework. A search state at a given time is defined by bounding box proposals visited until that time in the image. The search is guided by the heuristic $\mathcal{H}$ and successor $\mathcal{S}$ functions, which are unified and jointly learned using HSnet. $\mathcal{S}$ proposes new states, and $\mathcal{H}$ scores the states, until the time bound τ. One component of HSnet is LSTM whose memory fuses all candidate bounding boxes visited along the search trajectory. The soft-max layer of HSnet outputs final recognition.

The location of a bounding box i is parameterized by a tuple $l^{(i)} = (\mathrm{x}^{(i)}, \mathrm{y}^{(i)}, w^{(i)}, h^{(i)})$ where $(\mathrm{x}^{(i)}, \mathrm{y}^{(i)})$ is the center and $(w^{(i)}, h^{(i)})$ is the width and height. The ranges of $l^{(i)}$ are normalized between 0 and 1. The deep features of bounding box i, denoted by $x^{(i)}$, can be deterministically identified in x of the entire image. A search state $s_t \in \mathscr{S}$ at time t consists of $K(t)$ bounding boxes visited before t, $s_t = (l_t, x_t)$ where $l_t = \{l^{(i)} : i = 1, \ldots, K(t)\}$ and similarly $x_t = \{x^{(i)} : i = 1, \ldots, K(t)\}$.

Given an initial state s_0, our search algorithm uncovers a trajectory $[s_0, s_1, \ldots, s_\tau]$ until time bound τ. Our search is guided by the heuristic $\mathcal{H}$ and successor $\mathcal{S}$ functions, which are unified and estimated by a single deep architecture, as shown in Fig. 2. Specifically, we parameterize $\mathcal{H}$ and $\mathcal{S}$ as HSnet such that they have the same predictor for evaluating and proposing new search states. Moreover, this allows us to specify a unified end-to-end learning of $\mathcal{H}$ and $\mathcal{S}$.

Given the current state s_t, $\mathcal{H}$ computes a vector of heuristic scores, $\mathcal{H}(s_t) = \phi_t$. The heuristic scores are passed to $\mathcal{S}$ to propose a set of $k \geq 1$ bounding boxes, $\mathcal{S}(\phi_t) = [(l^{(1)}, x^{(1)}) \ldots, (l^{(k)}, x^{(k)})]$. This "expands" s_t to the next

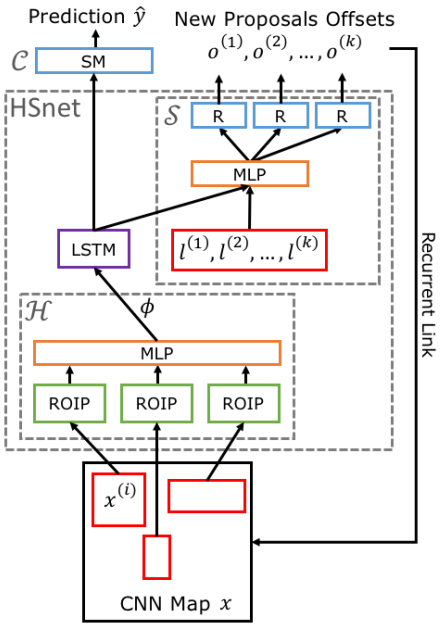


Figure 3: HSnet consists of H-layer, S-layer and LSTM. The CNN extracts a deep feature map x from the image. The H-layer implements $\mathcal{H}$. It computes heuristic scores ϕ from k current bounding boxes $[x^{(1)} \ldots x^{(k)}]$ (marked red). The S-layer implements $\mathcal{S}$. It takes ϕ, LSTM memory and locations of the bounding boxes $[l^{(1)} \ldots l^{(k)}]$ as input, and proposes k spatial offsets $[o^{(1)} \ldots o^{(k)}]$ relative to $[l^{(1)} \ldots l^{(k)}]$. This is fed back via the recurrent link to define new bounding boxes in the image. After τ search steps, the soft-max layer $\mathcal{C}$ is used for fine-grained recognition $\hat{y}$. MLP is multi-layer perceptron, ROIP is region of interest pooling, SM is softmax layer and R is regression.

state $s_{t+1} = (l_{t+1}, x_{t+1}) = [s_t, \mathcal{S}(\phi_t)]$, where the number of bounding boxes considered increases to $K(t+1) = K(t) + k$. In every search step, $\mathcal{S}(\phi_t)$ predicts k spatial displacements, also called offsets, of bounding boxes relative to the previous k predictions at time $t-1$. In our experiments, predicting offsets rather than absolute locations of bounding boxes have produced better performance.

Note that as k becomes larger, $\mathcal{H}$ and $\mathcal{S}$ increase the number of parameters, which in turn become harder to robustly learn.

Our approach is summarized in Alg. 1. In the following section, we specify HSnet.

3.3. HSnet

We parameterize $\mathcal{H}$ and $\mathcal{S}$ as HSnet. As illustrated in Fig. 3, HSnet takes the current state as input, and produces the next state. HSnet consists of three components: H-layer, S-layer and LSTM.

LSTM [14] is a recurrent neural network with a memory cell. LSTMs have been successfully used for solving a wide range of vision problems cast as sequential decision making. In this paper, we use a basic 1-layer LSTM architecture [14]. Note that our definition of a cumulative search state over all bounding boxes visited is enabled by the LSTM memory.

Algorithm 1 Search-based Fine-Grained Recognition

Input: Initial State s_0, Time Bound τ
Output: Prediction $\hat{y}$
Timer $t := 0$
while $t < \tau$ **do**
    Heuristic Features $\phi_t := \mathcal{H}(s_t)$
    Next State $s_{t+1} := s_t + \mathcal{S}(\phi_t)$
    $t := t + 1$
end while
Predict $\hat{y} := \mathcal{C}(s_\tau)$

The H-layer implements $\mathcal{H}$. The H-layer takes deep features of k bounding boxes $[x^{(1)} \ldots x^{(k)}]$ proposed in the previous search step, and outputs a vector of heuristic scores, ϕ. In the R-CNN literature [12, 31], these bounding boxes are also called regions of interest (ROIs). Each ROI is passed to a region of interest pooling layer (ROIP) to obtain a fixed-size vector representation. All the ROIs are then concatenated and passed through a multi-layer perceptron (MLP) to produce ϕ as output.

The S-layer implements $\mathcal{S}$. As shown in Fig. 3, as input, the S-layer takes ϕ, along with the contents of LSTM memory and locations of the k bounding boxes $[l^{(1)} \ldots l^{(k)}]$. This input is passed to a multi-layer perceptron for predicting k spatial offsets $[o^{(1)} \ldots o^{(k)}]$ of new bounding boxes relative to $[l^{(1)} \ldots l^{(k)}]$.

The prediction of offsets from the output of the S-layer is fed back via the recurrent link to define new bounding boxes in the image. After τ search steps, the soft-max layer $\mathcal{C}$ is used to predict the fine-grained class $\hat{y}$.

Note that our complexity is lower than that of beam search (employed in much of prior work) since our $\mathcal{H}$ and $\mathcal{S}$ jointly processes all k bounding boxes. Our complexity is also on the order of standard LSTM processing of video sequences since our $\mathcal{H}$ and $\mathcal{S}$ are relatively "shallow".

3.4. Learning HSnet

In this paper, we consider learning HSnet in two settings: (1) access to annotations of part locations is available, and (2) part annotations are not provided in training data. In both settings, end-to-end learning of all three components in HSnet is performed using the gradient-based backpropagation through time (BPTT), commonly used for training LSTMs.

The BPTT backpropagates the standard cross entropy loss incurred on training data when the search reaches time

bound τ, and the soft-max output of HSnet is used for predicting class label $\hat{y}$. This classification loss is regularized by additional loss functions, defined differently for each of the above two settings.

With Part Annotations. When part annotations are available, we are able to regularize learning of HSnet to predict locations of bounding boxes such that they align better with ground truth part annotations. Specifically, we regularize learning with the Euclidean distance between a predicted bounding box and closest ground truth part. For k parts, the regularization is a sum of k Euclidean distances. We compute this sum at each search step t, and weight it with a regularization parameter λ_t. Thus, our regularized loss in this setting is defined as

$$L = -\log p(y) + \sum_{t=1}^{\tau} \lambda_t \sum_{i=1}^{k} \|l^{(i)} - \hat{l}_t^{(l)}\|^2, \qquad (1)$$

where the first term is the cross entropy loss and the second term is regularization. In (1), y denotes ground truth class label, $p(y)$ denotes the soft-max score of HSnet for the ground truth class, $l^{(i)}$ is the ground truth location of part i, $\hat{l}_t^{(l)}$ is the location prediction of the bounding box nearest to $l^{(i)}$ (done greedily) at search step t, and λ_t is a regularization hyperparameter at search step t.

Without Part Annotations. When ground truth part annotations are not provided in training data, we seek to regularize learning of HSnet to predict locations of bounding boxes such that they are visually diverse. To this end, we regularize the cross entropy loss with a term characterized by the determinantal point processes (DPP). The DPP has been widely used for regularization in learning [24]. Our regularized loss in this setting is defined as

$$L(\hat{y}, y) = -\log p(y) - \sum_{t=1}^{\tau} \lambda_t \log P_t \qquad (2)$$

where the first term is the cross entropy loss and the second term is DPP regularization. The hyperparameters λ_t control the magnitude of DPP regularization. P_t is the probability of having diverse bounding boxes at search step t, defined as $P_t = \det|\Omega_k| / \det|\Omega + I|$. Ω is a positive semi-definite kernel matrix of affinities between all possible bounding boxes, and Ω_k denotes the restriction of Ω to the k selected bounding boxes. The affinities are specified as inverse Euclidean distances between locations. The determinant $\det|\Omega_k|$ quantifies the diversity of k locations. Hence, the higher the diversity, the higher P_t.

Even though we have no access to part locations in this setting, we are still able to regularize the positions of bounding boxes. DPP discourages trivial solutions of learning only a single object part. Without DPP or some other training signal on the predicted part locations, it would be much more difficult to train with just a classification objective.

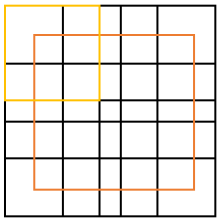


Figure 4: Positions and sizes of bounding boxes for the initial state of search with 10 boxes (as used in Cars-196). We use a regular grid of 9 boxes except each box is slightly larger to overlap with its neighbors. Yellow denotes the top-left-most box out of the mentioned 9 boxes for clarity. The 10th box is in the center with larger size, denoted by orange.

4. Experiments

4.1. Setup

Datasets. We evaluate on CUB-2011 [36] and Cars-196 [23] datasets. CUB-2011 contains 11,788 images of 200 species of birds and is generally considered one of the most competitive datasets for fine-grained recognition. Cars-196 has 16,185 images of 196 car types. Both datasets have a single bounding box annotation in each image (for the entire object, not each part), and CUB-2011 moreover contains rough segmentations and 15 keypoints annotated per image. We do not use the bounding box or segmentation annotations in our experiments.

Evaluation Setup. For CUB-2011 and Cars-196, we follow the train and test splits as provided by [36, 23].

Metrics. Our evaluation metric is top-1 accuracy, where a correct classification is defined as when the ground truth label is present in the top 1 most confident predictions.

Initial Search State. Our initial search state contains k bounding boxes centered at prior locations in the image. We set $k = 15$ for CUB-2011 because there are 15 bird parts available for supervision. We set $k = 10$ for Cars-196 since we empirically observed that $k = 10$ had the best tradeoff of accuracy and speed. We designed the initial state in such a way that the initial boxes are in an overlapping grid. Fig. 4 shows an example for $k = 10$ bounding boxes, where nine boxes are arranged in a grid and the tenth box is at the center of the image. We find that this performs better than random initialization, which is harder to train. We also find that it is better to cover the entire image at first with multiple bounding boxes to obtain an "overall impression" before refining the proposals to focus on parts.

Number of Iterations. We empirically determined that $\tau = 15$ worked best for CUB-2011 and $\tau = 10$ worked best for Cars-196. We experimented with $\tau = 1, 2, 5, 10, 15, 20, 25, 50$ and determined the best trade-off of accuracy and computation time for each dataset. It turns out that roughly setting $\tau = k$ yields the best performance.

One possible explanation is that each time step can focus on one part even if multiple bounding boxes are being refined simultaneously. We also set regularization hyperparameter λ_t to a linear schedule, with the most weight when $t = \tau$.

Implementation Details. For our CNN, we employed a GoogLeNet architecture with batch normalization [15] pre-trained on ImageNet [33]. Since ImageNet contains several images from our other datasets for evaluation, we removed them from training. We use Caffe [17] for extracting feature maps from images and TensorFlow [1] for implementing HSnet. The MLP (multi-layer perceptron) layer after ROIP (region of interest pooling) contains two fully connected layers of size 4096. The MLP layer after the LSTM contains one layer of size 2048. The LSTM contains 2048 hidden units. We train our framework with Adam optimizer using the default parameters.

4.2. Baselines

We define the following baseline methods.

B1. CNN: Given an image, a CNN directly predicts the class. We fine-tune a pre-trained model as done in [16].

B2. CNN with ground truth bounding boxes: Given an image, a CNN produces a feature map, then a neural network predicts the class based only on the contents of k bounding boxes, initialized to the ground truth part locations. This can be thought of as one time step of search with an initial state set to the ground truth parts. We want this "cheating" baseline to demonstrate that k boxes are enough to classify an image with the absence of context. Note that this baseline is only available for CUB-2011, which contain annotated part locations. Since only the part locations are given and not the bounding box sizes, we empirically determined that 64 was the best size out of 16, 32, 64, 128.

B3. HSnet with one ground truth bounding box: In this baseline, HSNet accepts as input one bounding box instead of k bounding boxes. HSnet is run for a fixed sequence (predetermined before search begins) of k time steps, where at each time step a ground truth bounding box is provided based on the k part annotations. HSnet's proposals are not used. Additionally, our loss function only contains the classification objective since the "proposed" bounding boxes from search are already ground truth. In this "cheating" baseline, we want to show that sequential reasoning of object parts works reasonably. Note that this baseline is also only available for CUB-2011. Again, we empirically determined that 64 was the best box size.

B4. HSnet with one bounding box: This baseline is similar to B3 except that instead of using ground truth bounding boxes, the next bounding box predicted by HSnet is used. This baseline still only uses one bounding box rather than k boxes. The initial box is the center box in Fig. 4. For each sequence, we only focus on one part. We train on all object parts for each image. Note that using k boxes results in our proposed approach.

Table 1: Quantitative results on CUB-2011 Birds Dataset. Annotations used during training time are also specified: "GT" denotes ground truth category labels, "BB" denotes bounding box annotations, "parts" denotes part annotations and "web" denotes augmenting dataset with web data.

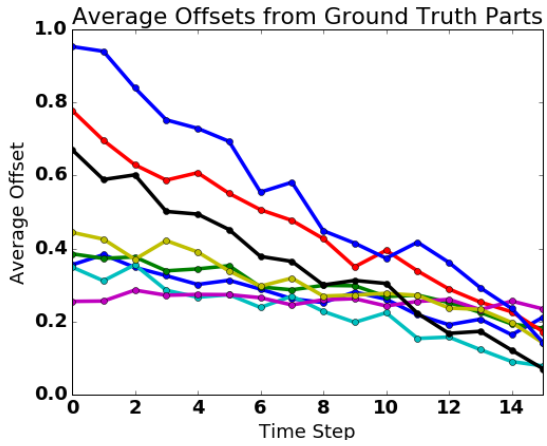
Method	Annotations Used	Accuracy
Krause et al. [21]	GT+BB	82.8
Jaderberg et al. [16]	GT	84.1
Xu et al. [38]	GT+BB+parts+web	84.6
Lin et al. [26]	GT+BB	85.1
B1	GT	82.3
B2	GT+parts	83.1
B3	GT+parts	86.2
B4	GT+parts	85.7
HSnet	GT+parts	87.5

Table 2: Quantitative results on Cars-196 Dataset. Annotations used during training time are also specified: "GT" denotes ground truth category labels, "BB" denotes bounding box annotations and "parts" denotes part annotations.

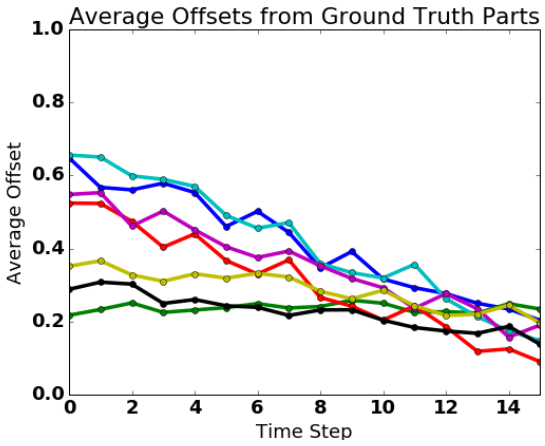
Method	Annotations Used	Accuracy
Deng et al. [8]	GT+BB	63.6
Krause et al. [23]	GT+BB	67.6
Krause et al. [20]	GT+BB	73.9
Lin et al. [26]	GT	91.3
Krause et al. [21]	GT+BB	92.6
B1	GT	88.5
B4	GT+parts	92.2
HSnet	GT+parts	93.9

4.3. Quantitative Results

Table 1 compares our main result and baselines with prior work on CUB-2011. Our results on CUB-2011 are competitive with the state of the art with about a 3% boost. Baselines B1 and B2 are comparable, which suggests that removing some context does not hurt recognition. B3 and B4 yield higher accuracies than B1 and B2, suggesting that sequential reasoning does help. B4 is slightly worse than B3 since ground truth is not present in B4. Finally, our complete framework performs better than all the baselines, which suggests that multiple proposals are better than one proposal at a time. We also believe that our approach performs better than B3 because our model observes multiple observations of different-sized bounding boxes whereas B3

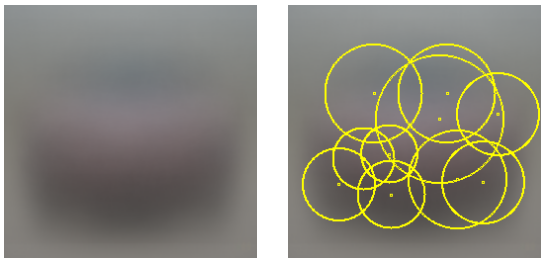


(a) Left Leg, Right Leg, Belly, Throat, Left Wing, Breast, Tail, Nape



(b) Left Eye, Right Eye, Forehead, Right Wing, Back, Crown, Bill

Figure 5: Plot of average offsets as a function of time steps on the CUB-2011 dataset, where offset is the distance between the ground truth part location and predicted location. Different colors indicate different parts, for a total of 15 parts split across two plots. The offsets decrease over time, indicating that the bounding boxes are converging close to the ground truth parts.



(a) Average Image (b) Parts Clusters

Figure 6: (a) Average image computed from cars images. This shows that most images of a car are taken from the front. (b) Map showing clusters of parts. The centers represent the mean locations of parts and the circles represent the range of those centers. The part locations align mostly align with the front of the car and the center of the image, where most of the car is present on average.

only uses one bounding box with a fixed size.

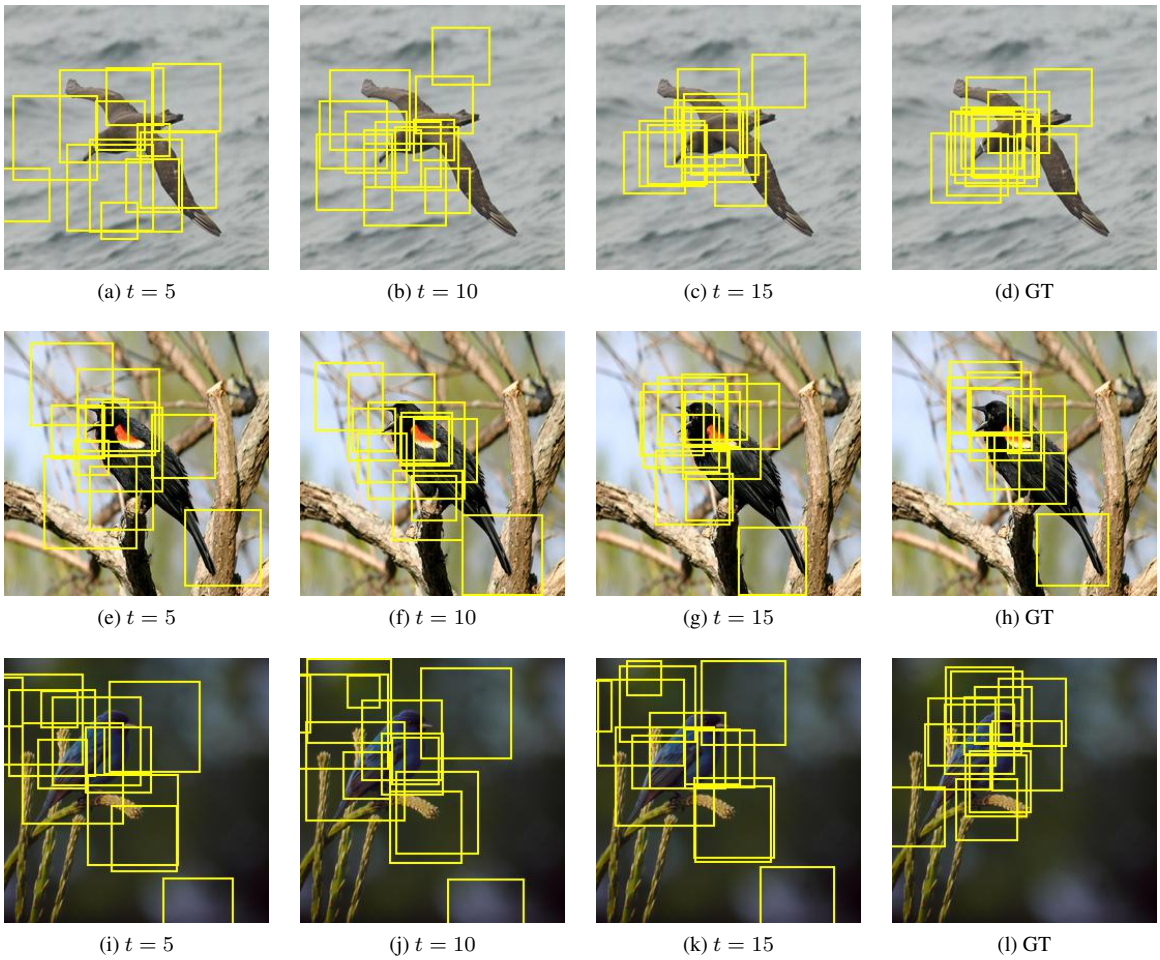
Table 2 compares our main result and baselines with prior work on Cars-196. Since part annotations are not available for Cars-196, we can only perform baselines B1 and B4. Notably, B4 performs significantly better than B1, which again supports that sequential reasoning performs better than recognition with a CNN in one shot. Overall, our complete framework performs better than the baselines and it is also competitive with the previous state of the art.

Fig. 5 plots the average offsets of predicted part locations to ground truth part locations as a function of search time step for CUB-2011. The plots show that as the time step increases, the average offsets are decreasing, indicating that our framework is learning to localize parts.

4.4. Qualitative Results

Fig. 7 shows the sequence of bounding boxes predicted for a few images of birds. We show two success cases and one failure case, where a success case is when the final predicted class is correct and otherwise a failure. We can see that as the time step increases, the boxes start to converge to the parts of the birds. These success cases make sense because the training objective takes into account the ground truth part locations. For the failure case, some of the bounding boxes do not converge to the ground truth parts. Nonetheless, some of the boxes still converge to the annotated parts. Although some bounding boxes do not fall on the object, LSTM has a robust mechanism of memory to memorize important parts. Thus classification does not critically depend on wrong detections at the final time step. This is also clear in the success case where some boxes could have been refined into better positions and sizes.

Since no part annotations are provided for Cars-196, we cannot quantitatively compare the part predictions to ground truth. Instead, we visualize the average part locations that are predicted by our algorithm. Fig. 6 shows the average image of the Cars-196 dataset and shows clusters of part locations predicted for the cars dataset. The average image shows that most images of a car are taken from the front, which means we can expect some average part locations to align with the front of the car. Indeed, on average most parts align with the front of the car. All parts are near the center of the image, where most cars are present



(a) $t = 5$ (b) $t = 10$ (c) $t = 15$ (d) GT

(e) $t = 5$ (f) $t = 10$ (g) $t = 15$ (h) GT

(i) $t = 5$ (j) $t = 10$ (k) $t = 15$ (l) GT

Figure 7: Sequence of bounding boxes predicted for a few images. The top two rows are success cases and the bottom row is a failure case, where a success case is a correct classification and a failure is an incorrect classification. We show for time steps $t = 5, 10, 15$, where $t = 15$ is the final time step used for classification. We also compare these bounding box locations with the ground truth locations (denoted GT), where the size of these boxes are fixed at 64×64. In the success cases, the sequences of bounding boxes are converging to the GT, demonstrating that our framework is learning to detect parts.

on average. Furthermore, the average part locations are relatively diverse, covering a majority of the average image rather than just a few locations. Finally, Fig. 6 shows that our approach discovers visually diverse parts that are also discriminative, as desired for fine-grained categorization.

5. Conclusion

We presented a search-based framework with deep architectures for fine-grained recognition that achieves competitive results. We proposed a search-based architecture where the search space is defined on a convolutional feature map of CNN, and the heuristic and successor functions are parameterized by a new deep network architecture called HSnet. HSnet is formulated with a built-in refinement mechanism to search for increasingly more informative parts and thus improve recognition, in addition to a robustness mechanism against wrongly identified parts during inference. We specified two training settings, one where part location annotations are available and one where they are not available, where the latter is addressed with a determinantal point process loss for obtaining diverse proposals. Finally, our experimental results on Caltech-UCSD Birds 200-2011 and Cars-196 datasets demonstrated that sequential reasoning about object parts and removing background context are effective for fine-grained recognition.

Acknowledgements. This work was supported in part by DARPA XAI, NSF RI1302700 and NSF GRFP 1314109-DGE.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 6

[2] M. R. Amer, D. Xie, M. Zhao, S. Todorovic, and S.-C. Zhu. Cost-sensitive top-down/bottom-up inference for multiscale activity recognition. In *European Conference on Computer Vision*, pages 187–200. Springer, 2012. 2, 3

[3] A. Barbu and S.-C. Zhu. Generalizing swendsen-wang to sampling arbitrary posterior probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1239–1253, 2005. 2

[4] T. Berg and P. Belhumeur. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 955–962, 2013. 1, 2

[5] S. Branson, G. Van Horn, S. Belongie, and P. Perona. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952*, 2014. 1, 2

[6] J. C. Caicedo and S. Lazebnik. Active object localization with deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2488–2496, 2015. 3

[7] Y. Chai, V. Lempitsky, and A. Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 321–328, 2013. 2

[8] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2013. 1, 2, 6

[9] K. Duan, D. Parikh, D. Crandall, and K. Grauman. Discovering localized attributes for fine-grained recognition. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3474–3481. IEEE, 2012. 2

[10] P. F. Felzenszwalb and D. McAllester. The generalized a* architecture. *Journal of Artificial Intelligence Research*, 29:153–190, 2007. 2

[11] E. Gavves, B. Fernando, C. G. Snoek, A. W. Smeulders, and T. Tuytelaars. Fine-grained categorization by alignments. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1713–1720, 2013. 2

[12] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 2, 4

[13] P. Gupta, D. Doermann, and D. DeMenthon. Beam search for feature selection in automatic svm defect classification. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 2, pages 212–215. IEEE, 2002. 2

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2, 4

[15] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 6

[16] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 1, 2, 3, 6

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 6

[18] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert. Activity forecasting. In *European Conference on Computer Vision*, pages 201–214. Springer, 2012. 2, 3

[19] I. Kokkinos. Rapid deformable object detection using dual-tree branch-and-bound. In *Advances in Neural Information Processing Systems*, pages 2681–2689, 2011. 2

[20] J. Krause, T. Gebru, J. Deng, L.-J. Li, and F.-F. Li. Learning features and parts for fine-grained recognition. In *ICPR*, volume 2, page 8. Citeseer, 2014. 6

[21] J. Krause, H. Jin, J. Yang, and L. Fei-Fei. Fine-grained recognition without part annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5546–5555, 2015. 1, 2, 6

[22] J. Krause, B. Sapp, A. Howard, H. Zhou, A. Toshev, T. Duerig, J. Philbin, and L. Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. *arXiv preprint arXiv:1511.06789*, 2015. 1

[23] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013. 5, 6

[24] A. Kulesza and B. Taskar. Determinantal point processes for machine learning. *arXiv preprint arXiv:1207.6083*, 2012. 5

[25] M. Lam, J. Rao Doppa, S. Todorovic, and T. G. Dietterich. Hc-search for structured prediction in computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4923–4932, 2015. 2, 3

[26] T.-Y. Lin, A. RoyChowdhury, and S. Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1449–1457, 2015. 1, 2, 6

[27] B. Mahasseni and S. Todorovic. Regularizing Long Short Term Memory with 3D human-skeleton sequences for action recognition. In *CVPR*, 2016. 1

[28] V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014. 3

[29] N. Payet and S. Todorovic. Sledge: Sequential labeling of image edges for boundary detection. *International journal of computer vision*, 104(1):15–37, 2013. 2

[30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2015. 2

[31] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In

Advances in neural information processing systems, pages 91–99, 2015. 2, 4

[32] A. Roy and S. Todorovic. Scene labeling using beam search under mutex constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1178–1185, 2014. 2, 3

[33] 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. *International Journal of Computer Vision*, 115(3):211–252, 2015. 6

[34] R. Tao, E. Gavves, and A. Smeulders. Siamese instance search for tracking. In *CVPR*, 2016. 1

[35] O. Teboul, I. Kokkinos, L. Simon, P. Koutsourakis, and N. Paragios. Shape grammar parsing via reinforcement learning. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2273–2280. IEEE, 2011. 2, 3

[36] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 5

[37] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2(3):5, 2015. 3

[38] Z. Xu, S. Huang, Y. Zhang, and D. Tao. Augmenting strong supervision using web data for fine-grained categorization. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015. 1, 6

[39] S. Yang, L. Bo, J. Wang, and L. G. Shapiro. Unsupervised template learning for fine-grained object recognition. In *Advances in Neural Information Processing Systems*, pages 3122–3130, 2012. 2

[40] B. Yao, G. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3466–3473. IEEE, 2012. 2

[41] B. Yao, A. Khosla, and L. Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1577–1584. IEEE, 2011. 2

[42] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014. 3

[43] J. Zhang, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff. Top-down neural attention by excitation backprop. In *ECCV*, 2016. 3

[44] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014. 1, 2

[45] N. Zhang, R. Farrell, F. Iandola, and T. Darrell. Deformable part descriptors for fine-grained recognition and attribute prediction. In *2013 IEEE International Conference on Computer Vision*, pages 729–736. IEEE, 2013. 1, 2