Capturing long-tail distributions of object subcategories

Xiangxin Zhu
University of California, Irvine
xzhu@ics.uci.edu

Dragomir Anguelov
Google Inc.
dragomir@google.com

Deva Ramanan
University of California, Irvine
dramanan@ics.uci.edu

Abstract

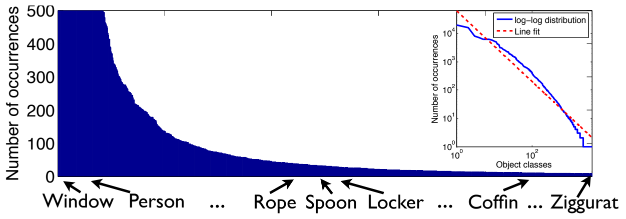
We argue that object subcategories follow a long-tail distribution: a few subcategories are common, while many are rare. We describe distributed algorithms for learning large-mixture models that capture long-tail distributions, which are hard to model with current approaches. We introduce a generalized notion of mixtures (or subcategories) that allow for examples to be shared across multiple subcategories. We optimize our models with a discriminative clustering algorithm that searches over mixtures in a distributed, "brute-force" fashion. We used our scalable system to train tens of thousands of deformable mixtures for VOC objects. We demonstrate significant performance improvements, particularly for object classes that are characterized by large appearance variation.

1. Introduction

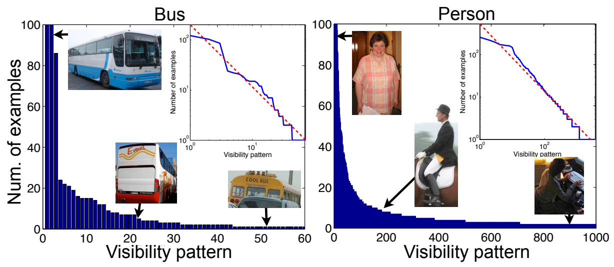
It is well-known that the frequency of object occurrence in natural scenes follows a long-tail distribution [26]: for example, people and windows are much more common than coffins and ziggurats (Fig. 1a). Long-tails complicate analysis because rare cases from the tail still collectively make up a significant portion of the data and so cannot be ignored. Many approaches try to minimize this phenomenon by working with balanced datasets of objects categories [10]. But long-tails still exist for object *sub*categories: most people tend to stand, but people can assume a large number of unusual poses (Fig.1b). We believe that current approaches may capture iconic object appearances well, but are still limited due to inadequate modeling of the tail.

In theory, multi-mixture or subcategory models should address this, with possibly large computational costs: train a separate model for different viewpoints, shape deformation, etc. Empirically though, these approaches tend to saturate early in performance after a modest number of mixtures [34, 20, 12, 16, 14].

We argue that the long-tail raises three major challenges that current mixture models do not fully address: (1) The "right" criteria for grouping examples into subcategories is not clear. Various approaches have been suggested (including visual similarity [12], geometric similarity [5], semantic ontologies [10]), but the optimal criteria remains unknown. (2) Even given the optimal criteria, it is not clear how to algorithmically optimize for it. Typical methods employ some form of clustering, but common algorithms (e.g., k-means) tend to report clusters of balanced sizes, while we hope to get long-tail distributions. (3) Even given the optimal clustering, how does one learn models for rare subcategories (small clusters) with little training data?



(a) The number of examples by object class in SUN dataset



(b) Distributions of the visibility patterns for bus and person

Figure 1: Long tail distributions exist for both object categories and subcategories. (a) shows the number of examples by object class in the SUN dataset. The blue curve in the inset show a log-log plot, along with a best-fit line in red. This suggests that the distribution follows a long-tail power law. (b) shows the distributions of the keypoint visibility patterns for bus and person from PASCAL (using the manual annotations of [6]), which also follow a long-tail. We describe methods for automatically discovering long-tail distributions of subcategories with a distributed, "brute-force" search without using additional annotations.

In our work, we address all three challenges: (1) We

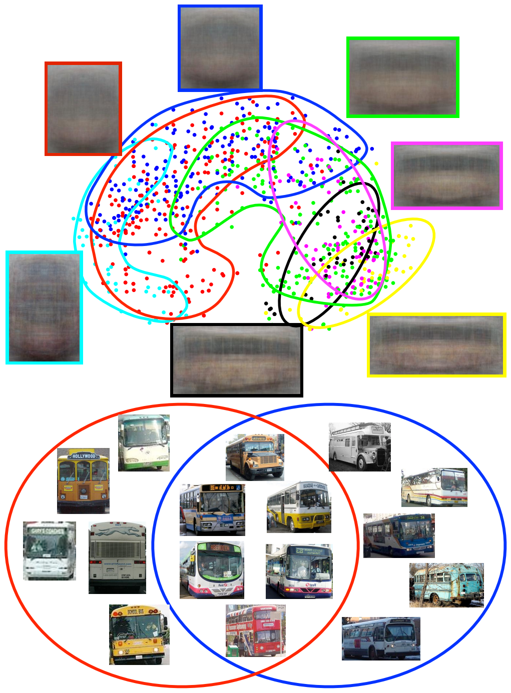


Figure 2: We describe overlapping subcategory models that allow for training data to belong to multiple clusters with a large variation in size. For example, frontal (red) and side-view (blue) buses may share a large number of $\frac{3}{4}$-view examples, and both are much more common than multi-body articulated buses (yellow). We show that such models better characterize objects with "long-tail" appearance distributions.

posit that the optimal grouping criteria is simply **recognition accuracy**. But this presumably requires a "brute-force" search over all possible clusterings, and an evaluation of the recognition accuracy of each grouping, which appears hopeless. (2) We introduce a **discriminative clustering** algorithm that accomplishes this through distributed computation, making use of massively-parallel architectures. We show that long-tail cluster sizes *naturally emerge* from our algorithm. (3) To address the lack of training data for small clusters, we allow rare subcategories to share training examples with dominant ones, introducing a notion of a **overlapping subcategories**. Such fluid definitions of categories are common in psychology [25]. For example, a sport utility vehicle could be equally classified as a truck or a car. Overlapping subcategories allow for cluster label assignment to decouple across subcategories, crucial for our distributed optimization.

Noteably, our clustering algorithm does not explicitly enforce long-tail distributions as priors. Rather, our underlying hypothesis is that long-tail distributions are an *emergent property* of the "optimal" clustering, when measured with respect to recognition accuracy. We verify this hypothesis experimentally. It is possible that brute-force clustering of other data types with respect to other criteria may not produce long-tails. Rather, our experimental results reflect an empirical property of our visual world, much like the empirical analysis of Fig. 1.

We review related work in Sec. 2, introduce our generalized subcategory model and discriminative optimization algorithm in Sec. 3, and present results in Sec. 4. We demonstrate that our long-tail mixture-models significantly outperform prior work on benchmark detection datasets, in some cases achieving a factor of 2 improvement.

2. Related work

Subcategory discovery: Estimating subcategories is surprisingly hard; clustering based on keypoints [16, 5] and appearance [11, 3] have provided only modest performance increases [34]. [13] uses combined appearance, shape, and context information to discover a small number of common subcategories for object classification, however the rare cases are thrown away as "outliers". One attractive approach is to use a *discriminative* model to re-rank and identify other nearby training examples for clustering. This is often implemented through latent mixture assignment in a max-margin model [14] or discriminative k-means [30]. In practice, such methods are sensitive to initialization and can suffer from miscalibration [27]. We describe a discriminative clustering algorithm that searches over all initializations in a distributed fashion without ever comparing scores across different models during training. Finally, our models allow for overlapping clusters. This differs from soft assignment in that the total contribution of an example need not be 1; indeed, we show that certain examples are much more dominant than others (consistent with Rosch's prototype theory [24]).

Sharing across categories: There has been much work on sharing information between object category models [26, 4, 18]. Most related is [18], which allows an object class to borrow examples from similar categories, e.g. some armchairs can be used to train sofa models. While this approach yields modest performance gains (1.4%AP), we produce larger gains presumably due to our brute-force optimization over subcategory labels and sharing. Another attractive formalism is that of attributes, or generic properties shared across object categories [17, 15, 29]. One could interpret binary subcategory labels as a binary attribute vector; indeed, we perform multi-dimensional scaling on such a vector to generate the visualization in Fig. 2. Our approach differs from much past work in that our "attributes" are latently inferred in a data-driven manner.

Sharing across subcategories: Various approaches have also explored sharing across subcategories. For example, it is common to share local features across view-

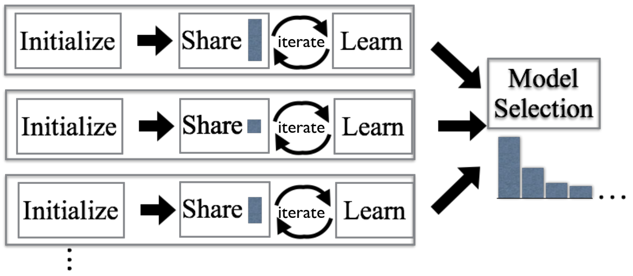


Figure 3: Our overall pipeline. We learn a massive number of candidate subcategory models in parallel, each *initialized* with its own training example (an exemplar) and particular cluster size. We train each subcategory with a discriminative clustering algorithm that iterates between selecting examples for *sharing* and *learning* detectors given those examples. Finally, we *select* a subset of candidate subcategory detectors for each object class as to maximize recognition accuracy. We show that this selection naturally produces subcategories with long-tail distribution of sizes.

based mixtures of an object [28, 33]. Typically, subcategory mixtures are supervised, but not always [21]. We share global examples rather an local parts, as the former is more amenable to brute-force distributed optimization.

3. Learning long-tail subcategory models

In this section, we describe our approach for learning long-tail subcategory models. We model each subcategory with a single-mixture deformable part model (DPM) [14]. Our overall pipeline is summarized in Fig. 3. We explain each step in detail in the following.

3.1. Initialization

We begin by training a large "overcomplete" set of thousands to tens of thousands of candidate subcategory models in parallel. This large set of models will later be pruned. We initialize our subcategory models by learning a discriminative template for each positive example using exemplar SVMs [20]. We visualize exemplar root templates for cars in Fig. 4. In terms of category detection accuracy, they perform reasonably well (25% AP). But because it is easy to overfit to a single example, many templates include noisy features from the background.

Sharing as regularization: To help learning more reliable templates for the rare examples, we retrain subcategory model m with the n_m highest-scoring positive examples under the exemplar model. We consider the sharing as a form of "regularization" that prevents overfitting to noisy gradients. To demonstrate the effect of sharing, we visualize the exemplar templates and the retrained templates for $n_m = 50$ in Fig. 4. The templates "regularized" by shared examples have less noisy gradients and almost double performance, producing an AP of 42%. Indeed, "averaging" across n_m similar training examples maybe more natural than penalizing the squared norm of a template, as is typically done to prevent overfitting. This motivating example suggests that subcategory clusters need not be mutually exclusive and may overlap. In fact, we find that some positive examples are shared by many subcategories, a phenomenon that we will investigate later in Fig. 9.

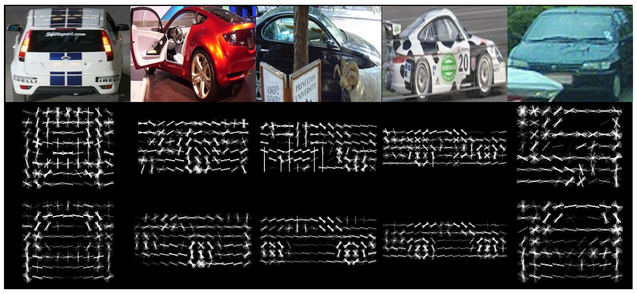


Figure 4: We visualize examples training images on the **top**. We show initial exemplar models trained with them in the **middle**. These templates perform well (25% AP on VOC2007), but sometimes emphasize incorrect gradients, such as the foreground tree in the center image. Retraining with the $n_m = 50$ highest-scoring examples (**bottom**) smooths out the template, de-emphasizing such noisy gradients (since they tend not be found in the n_m neighbors). This significantly improves performance to 42%. This suggests that optimal subcategory clusters may be overlapping, and maybe computed independently for each subcategory.

Iteration: We make two further observations. First, one can iterate the procedure and find the n_m highest scoring examples with the retrained subcategory model and repeat. The optimal choice of neighbors for one cluster is independent of the choice of another cluster, suggesting these iterations can be performed independently and in parallel. We show in Sec. 3.2 that such a distributed, iterative algorithm is guaranteed to converge since it can be formalized as joint optimization of a well-defined (discriminative) objective function.

Cluster-size: Selecting the optimal cluster size n_m is tricky. We want large n_m for common cases. Rare clusters are particularly hard to model; from one perspective, they should use a small n_m so that learned detectors aren't polluted by visually dissimilar examples. On the other hand, models learned from very small clusters may tend to overfit because they are trained with less data. As argued above, we treat n_m as a subcategory-specific regularization parameter that is tuned on validation data. Specifically, we learn models for a log-linear range of $n_m \in N = \{50, 100, 200, 400, 800, 1600\}$ values. Given a dataset of positives P , we learn a large set of candidate subcategories mixtures M ($|M| = |N||P|$) *in parallel*, spanning both examples and cluster sizes. After training this large redundant set, we select a subset on validation data.

3.2. Discriminative clustering with sharing

We formalize the iterative algorithm introduced in the previous subsection. We do so by writing a objective function for jointly training *all* $|M|$ subcategory models, and describe a coordinate descent optimization that naturally decouples across subcategories.

Let us write a mixture of templates (can be part models or simply rigid templates) as

$$f(x) = \max_m w_m \cdot x \quad \text{where} \quad m \in M \tag{1}$$

where m indicates a subcategory mixture component. M is the set of all mixture components . w_m is the template for m. x is an example. Given a training dataset (x_i, y_i) where $y_i \in \{-1, 1\}$ (for the detection problem), we explicitly write the non-convex learning objective as a function of both mixture models $\{w_m\}$ and binary latent variables $z_{im} \in \{0, 1\}$ that take on the value 1 if the i^{th} positive belongs to mixture component m. [23] refers to the $|P| \times |M|$ binary matrix of $Z = [z_{im}]$ as the "latent matrix", where $|P|$ is the number of positive examples:

$$\begin{aligned} L(w, Z) = \sum_{m \in M} \Big[\frac{1}{2}||w_m||^2 \\ + C \sum_{i \in pos} z_{im} \max(0, 1 - w_m \cdot x_i) \\ + C \sum_{i \in neg} \max(0, 1 + w_m \cdot x_i)\Big]. \end{aligned} \tag{2}$$

A standard latent SVM problem without sharing in [14] can be written as a joint minimization over (w, Z) subject to the constraints the rows of Z sum to 1 ($\sum_m z_{im} = 1$): each positive example i is assigned to exactly one mixture component.

We now replace the hard-assignment constraint $\sum_m z_{im} = 1$ with $\sum_i z_{im} = n_m$. Now we sum over i instead of m. This means that rather than forcing each positive to be assigned to exactly one mixture component, we force the mixture component m to consist of n_m examples. This constraint allows a single positive example i to be used to learn multiple mixtures, which provides a natural form of sharing between mixtures.

We describe a coordinate descent optimization algorithm for optimizing (2) subject to our new linear constraints:

1. **(Sharing)** $\min_Z L(w, Z)$: Compute $w_m \cdot x_i$ for $i \in$ pos. Sort scores and set z_{im} for the n_m highest values to 1.

2. **(Learning)** $\min_w L(w, Z)$: Learn w_m with a convex program (SVM) using n_m positives and all negatives.

Step 1 optimizes latent assignment Z while keeping template weights w fixed. In detail, this step assigns n_m positive examples to each template w_m so as to minimize the hinge loss. The loss is exactly minimized by assigning the n_m highest scoring positives to m. Step 2 optimizes w while fixing Z. This step is a standard SVM problem. As both steps decrease the loss in (2), it is guaranteed to converge to a local optima.

Unlike other clustering methods such as k-means, both of the above steps can be performed independently and *in parallel* for all $|M|$ clusters.

3.3. Greedy model selection

For each object class, we generate a pool of $|M|$ candidate subcategory models. Typically, $|M|$ is in the thousands to tens of thousands. Many of these subcategory models will be redundant due to sharing. We want to compress the models by selecting a subset $S \subseteq M$.

We cast this as a combinatorial optimization problem: for a possible subset S, compute its average precision performance $AP(S)$ on a validation set, and select the subset that maximizes AP. To evaluate $AP(S)$, we run each subcategory model $m \in S$ on the validation set, eliminate overlapping detections from the subset S with non-maximum suppression (NMS), and compute a precision-recall curve.

The search over all powersets of M is clearly intractable, but we find a greedy strategy to work quite well. Initialize $S = \{\}$ and repeatedly find the next-best mixture m to turn "on":

1. $m^* := \operatorname*{argmax}_m AP(S \cup m)$
2. $S := S \cup m^*$

A natural stopping condition is an insufficient increase in AP. In practice, we stop after instantiating a fixed number ($|S| = 50$) of subcategories. The first such instantiated subcategory tends to model a dominant cluster trained with a large n_m, while latter instantiations tend to consist of rare cases with small n_m.

To ensure that subcategory scores are comparable during NMS, we first calibrate the scores across models by mapping them to object class probabilities [22]. We use the same set of validation images for calibration and model selection. Calibration and selection is fast because the computationally-demanding portion (training a large pool of detectors and running them on validation images) is parallelized across all $|M|$ subcategory models. We visualize instantiated mixtures in Fig. 5 and Fig. 9.

4. Experimental results

Map-reduce: Our approach requires training and evaluating tens of thousands of DPMs across large numbers of object categories. In order to manage learning at this scale, we have implemented an in-house map-reduce version of the DPM codebase [1] . Map-reduce [7, 8] is an architecture for parallel computing. In our system, we use mappers

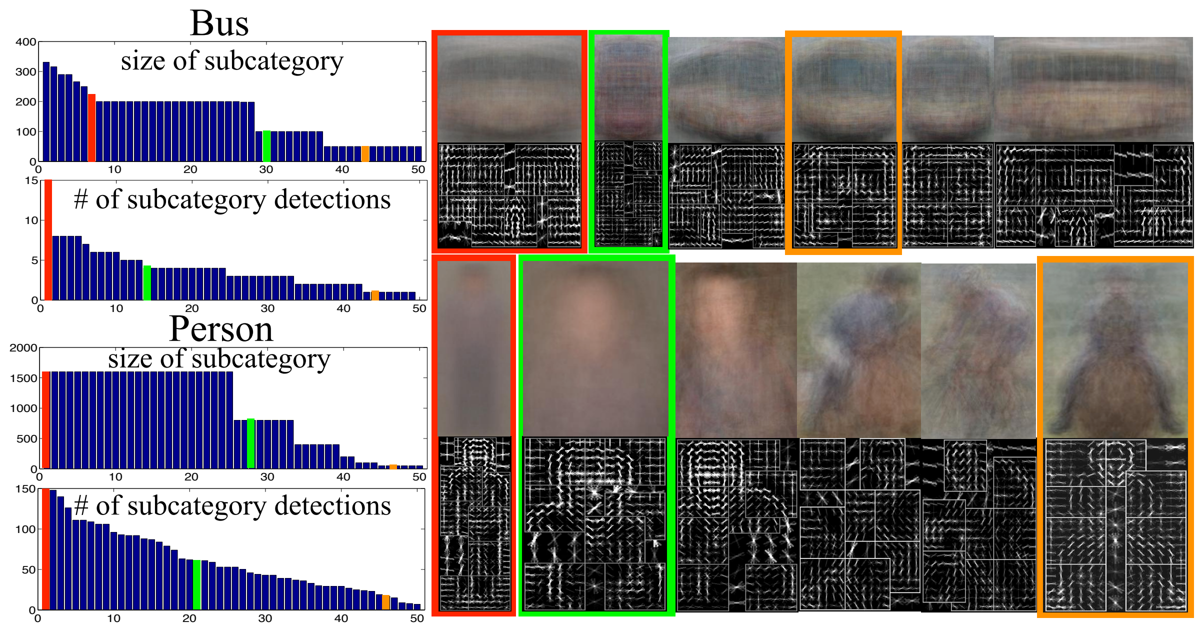


Figure 5: For two categories (bus and person), we plot the size of each subcategory cluster, as well as the number of true positive detections of that subcategory on test images (**left**). Both tend to follow a long-tail distribution; some subcategories are dominant, while many are rare. We visualize average images and templates associated with select subcategories on the **right**. We find that bus subcategories correspond to bus viewpoint, while person subcategories correspond to different truncations, poses, and interactions with objects such as horses and bicycles.

to collect positive examples and negative images, and use mixture-specific reducers to learn the mixture models. This distributed architecture allows us to learn mixtures *in parallel* according to the formulation of (2).

Computation: Because our distributed training algorithm can be parallelized across $|M|$ cores, each iteration of our learning algorithm is no slower than training a single-mixture DPM. We perform a fixed number (6) of discriminative clustering iterations, but find that cluster labels tend to stabilize after 3 iterations. At test-time, our long-tail subcategory models are equivalent to running $|S| = 50$ single-mixture DPMs in parallel, which takes about 1 second per image in our in-house implementation.

Benchmarks: Following much past work, we evaluate our detection system on PASCAL VOC 2007. Additionally, in order to test our implementation of DPMs (DistDPMs), we evaluate our DistDPMs on the Columbia Dog Breed Dataset [19] and the CUHK Cat Head Dataset [31]. The Columbia Dog dataset consist of 8000 dog images obtained from various repositories, while the CUHK Cat Dataset consists of 10000 Flickr images of cats. We used their standard training/test split: roughly 50%/50% for dogs and 70%/30% for cats.

Distributed DPMs: Before evaluating our proposed long-tail subcategory (LTS) models, we first verify that our map-reduce DPM implementation can reproduce the state-of-the-art voc-release4 models of [1, 14], when we use the same number of mixtures and the same latent hard assignment strategy (no sharing) as in [1]. We compare to the raw detectors without any post-processing (bounding-box prediction or contextual rescoring) in Table 1. Our implementation produces an average AP of 28.1%, compared to 31.8% from [1]. The 3% drop is mainly due to the fact that [1] uses max-regularization over all mixtures and learns 3 pairs of mirrored mixtures rather than the 6 separate mixtures in our implementation. Both of these are hard to decouple and parallelize, but known to help the empirical performance.

Even given this shortcoming, we will show our codebase can be used to construct state-of-the-art subcategory models. We also verify that our re-implementation produces state-of-the-art performance on other benchmark datasets in Fig. 7.

Long-tail subcategories (LTS). We use a subset of the VOC2011 training set (that does not overlap with the 2007 dataset) as validation data to greedily select our long-tail subcategory models (Sec. 3.3). Interestingly, we find that greedy selection on the original training data works well, producing only a 2% drop from the 50-LTS numbers in Table 1. We compare to previously published results that

Category	plane	bicycle	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	total
voc-rel4[1]	29.6	57.3	10.1	17.1	25.2	47.8	55.0	18.4	**21.6**	24.7	23.3	11.2	57.6	46.5	42.1	12.2	18.6	31.9	44.5	40.9	31.8
voc-rel5[2]	32.4	57.7	**10.7**	15.7	25.3	51.3	54.2	17.9	21.0	24.0	25.7	11.6	55.6	**47.5**	**43.5**	14.5	22.6	34.2	44.2	41.3	32.5
LEO[32]	29.4	55.8	9.4	14.3	28.6	44.0	51.3	21.3	20.0	19.3	25.2	12.5	50.4	38.4	36.6	15.1	19.7	25.1	36.8	39.3	29.6
DPM-WTA[9]	19	48	03	10	16	41	44	9	15	19	23	10	52	34	20	10	16	28	34	34	24
ESVM[20]	20.8	48.0	7.7	14.3	13.1	39.7	41.1	5.2	11.6	18.6	11.1	3.1	44.7	39.4	16.9	11.2	22.6	17.0	36.9	30.0	22.7
MCI+MCL[3]	33.3	53.6	9.6	15.6	22.9	48.8	51.5	16.3	16.3	20.0	23.8	11.0	55.3	43.8	36.9	10.7	22.7	23.5	38.6	41.0	29.8
Our 6-DistDPM	26.6	54.3	9.4	14.4	24.8	46.5	50.5	12.8	14.1	26.3	14.0	3.2	**57.7**	43.2	35.2	10.1	16.9	18.2	41.6	41.2	28.1
Our 50-LTS	**34.1**	**61.2**	10.1	**18.0**	**28.9**	**58.3**	**58.9**	**27.4**	21.0	**32.3**	**34.6**	**15.7**	54.1	47.2	41.2	**18.1**	**27.2**	**34.6**	**49.3**	**42.2**	**35.7**

Table 1: Average precision for different object categories in PASCAL 2007 dataset. We compare our approach to the existing methods on the same benchmark. The first two rows contain the results reported in the released code of [1, 2] without any post-processing. Our parallel implementation with the same number of mixtures (6-DistDPM) is shown in the second last row. The last row is our long-tail subcategory model (LTS), which significantly improves performance, particularly on difficult categories with large appearance variation (Fig. 6).

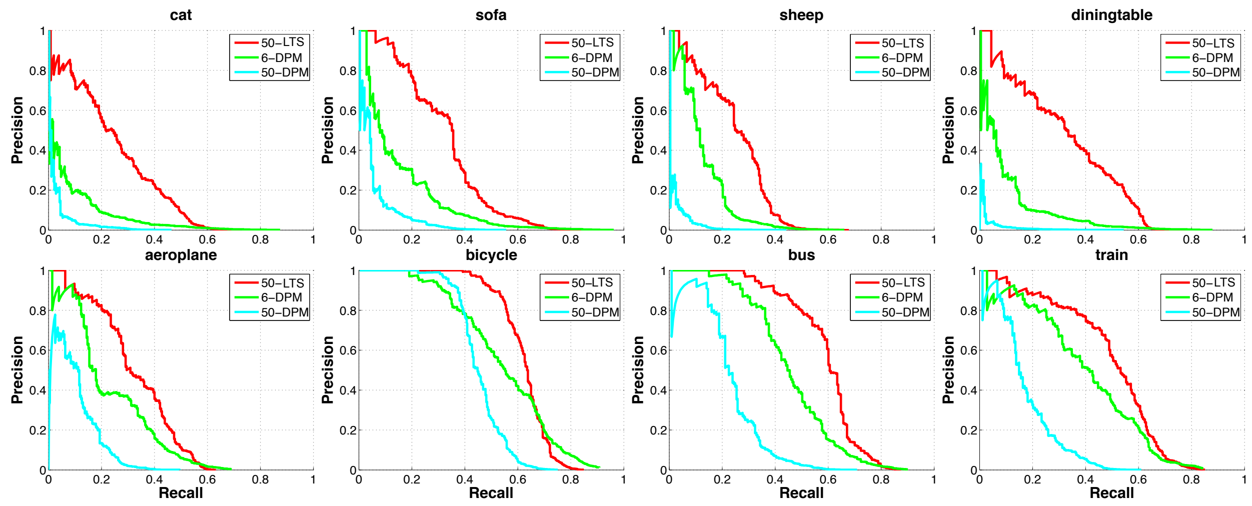


Figure 6: We plot precision-recall curves for PASCAL VOC 2007 categories. We show "hard" classes on the top row, where the baseline 6-mixture DPM (in green) performs poorly. These classes tend to have rich appearance variation (due to viewpoint, subclass structure, and occlusion from other objects). Scaling the baseline to 50-mixture DPM (in cyan) yields worse performance due to lack of training examples for each mixture component. Our 50-mixture long-tail subcategory model (LTS, in red) provides a significant improvement in such cases. The "easy" classes in the bottom row show similar trend. We further examine this performance increase in Fig. 10.

use the same feature set (HOG) without contextual post-processing, as the choice of features/post-processing is orthogonal to our work. We obtain the highest performance in 15/20 categories and the best overall performance. When compared to our in-house DPM implementation, our long tail models increase performance from 28.1% to 35.7%. We diagnose this performance increase (relative to our in-house implementation) in the following.

Hard vs easy classes: We split up the set of classes into hard vs easy, depending on baseline performance. We hypothesize that low baseline performance is indicative of a long tail (more varied appearances) that are not well modeled by the 6 mixtures of a standard DPM. We more than **double** the average precision of the baseline for such categories (Fig. 10). On "easier" classes, 50-LTS still perform best, but 6-DPM is a close second. We posit that a smaller number of mixtures sufficiently captures the limited appearance variation of such classes. We plot performance versus the number of mixtures on the validation set in Fig. 8. For "easy" classes such as car, we find that a few mixtures do well. For "hard" classes with more appearance variation (such as cats), we find that adding additional mixtures, even beyond 50, will likely improve performance.

50-LTS vs exemplars: We first compare to existing non-parametric models based on exemplar SVMs [20, 11]. One may hypothesize that rare subcategories (in the tail) are well modeled with a single exemplar. Our long-tail models with sharing considerably outperform such methods in Table 1,

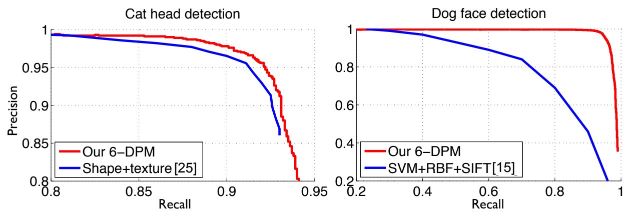


Figure 7: We run our implementation of 6-DistDPM on two public datasets of dog[19] and cat[31] face detection. Our 6-DistDPM outperforms the previously reported best methods on both datasets. Our improvement is particular large on the dog dataset, which arguably is more challenging due to the larger appearance variations of a dog face.

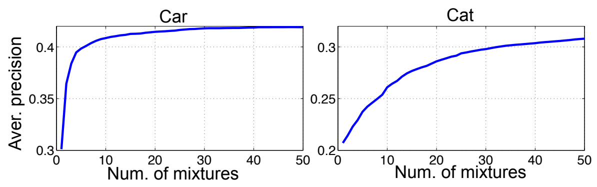


Figure 8: The performance as a function of the number of mixtures. For "easy" classes such as car, we find that a few mixtures do well. For "hard" classes with more appearance variation (such as cats), we find that adding additional mixtures, even beyond 50, improves performance.

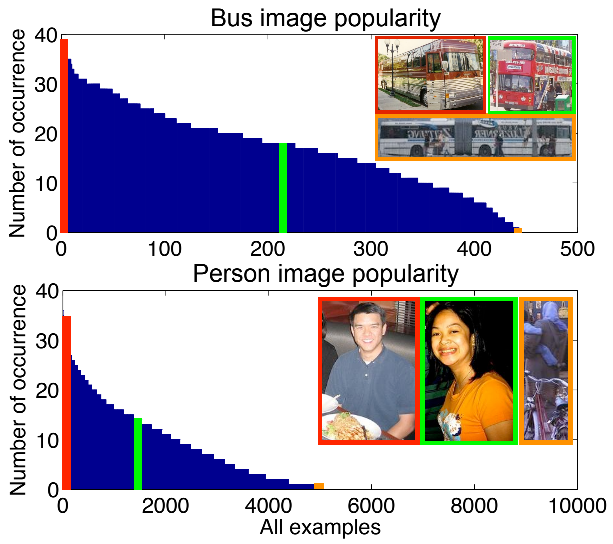


Figure 9: This plot reveals "popular" positive training images that are selected by many subcategories. Popular training images (in red) are prototypes that are representative of the category. Unpopular or rare images (such as multi-body articulated bus, in orange) are used by less subcategories.

suggesting that sharing is still crucial for the rare cases.

50-LTS vs 50-DPM: We also compare to the widely-used latent mixture-assignment algorithm of [14] for large number of mixtures. At $|S| = 50$, although it seems that 50-LTS and 50-DPM have the same capacity, our approach

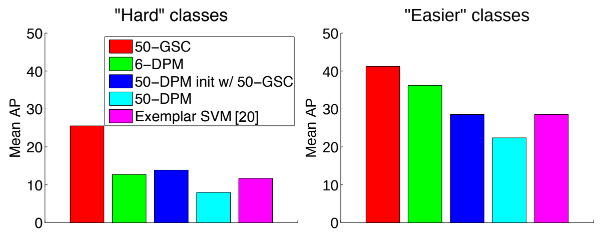


Figure 10: We separately analyze "hard" PASCAL classes with large appearance and viewpoint variations {*bird, cat, dog, sheep, plant, table, sofa and chair*}; these typically have lower average precision (AP) than other classes. On hard classes, our LTS **doubles** the AP of DPMs, increasing accuracy from 12% to 25%. On easier categories, our LTS model still outperforms 6-mixture DPMs by 5%. To analyze these improvements, we first scale DPMs to large (50) mixtures (50-DPM), which consistently underperform the DPMs with 6 mixtures (6-DPM). When 50-DPMs are initialized with our 50-LTS models, performance increases (suggesting that poor latent variable initialization is one reason to blame). When these models are allowed to share training examples, we see the largest performance increase, suggesting our **overlapping subcategory models are crucial for large-mixture representations.**

consistently performs significantly better. The improvement may arise from two factors: (1) Our mixtures are trained with a brute-force search that avoids the local minima of latent reassignment. (2) Our mixtures share training examples across clusters. We now analyze the improvement due to (1) vs (2). We focus on examining the improvement due to "hard" classes. The similar observations apply to the"easier" classes too.

Effect of brute-force search: One might argue that discriminative latent-reassignment may also benefit from a brute-force search over initializations of the latent variables. To help answer this, we initialize latent reassignment of 50-DPM using our 50-LTS detectors. This ensures that latent reassignment is initialized with a set of 50 good models that produce an AP of 25% (on the "hard" classes). This better initialization increases performance of 50-DPM from 7% to 13% AP, suggesting that latent-reassignment in [14] does suffer from local-minima.

Effect of overlapping subcategories: The above experiment also points out a conspicuous *drop* in performance; when initializing with 50 good detectors (25%AP), hard latent reassignment (that eliminates sharing by forcing each positive example to belong to a single cluster) dramatically drops performance to 13%AP. This suggests that the gains due to ***sharing*** are even more important than the gains due to better initialization. In our algorithm, initialization and sharing are intimately intertwined.

Conclusion: Many current approaches to object recognition successfully model the dominant iconic appearances of objects, but struggle to model the long tail of rare appearances. We show that distributed optimization and example sharing partially address this issue. We introduce a discriminative clustering algorithm that naturally allows for example sharing and distributed learning. We use this algorithm to perform a "brute-force" search over subcategory initialization and subcategory size, and demonstrate that the resulting models significantly outperform past work on difficult objects with varied appearance. We posit that our performance is now limited by the lack of training data for the rare subcategories in the tail. We may need large training datasets to fully encounter the set of rare cases. Our analysis suggests that "big-rare-data", which focuses on rare examples not already seen, may be more beneficial than traditional "big-data".

Acknowledgements: Funding for this research was provided by NSF Grant 0954083 and ONR-MURI Grant N00014-10-1-0933.

References

[1] http://www.cs.brown.edu/pff/latent/voc-release4.tgz.

[2] http://www.cs.berkeley.edu/~rgb/latent/voc-release5.tgz.

[3] O. Aghazadeh, H. Azizpour, J. Sullivan, and S. Carlsson. Mixture component identification and learning for visual recognition. In *ECCV*, 2012.

[4] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV*. IEEE, 2011.

[5] H. Azizpour and I. Laptev. Object detection using strongly-supervised deformable part models. In *ECCV*, 2012.

[6] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *ICCV*, 2009.

[7] C. Chu, S. Kim, Y. Lin, Y. Yu, G. Bradski, A. Ng, and K. Olukotun. Map-reduce for machine learning on multicore. *NIPS*, 19:281, 2007.

[8] J. Dean and S. Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[9] T. Dean, M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik. Fast, accurate detection of 100,000 object classes on a single machine. In *CVPR*, 2013.

[10] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE, 2009.

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

[12] S. K. Divvala, A. A. Efros, and M. Hebert. Object instance sharing by enhanced bounding box correspondence. In *BMVC*, 2012.

[13] J. Dong, W. Xia, Q. Chen, J. Feng, Z. Huang, and S. Yan. Subcategory-aware object classification. In *CVPR*, 2013.

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

[15] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, Dec. 2007.

[16] C. Gu, P. Arbelaez, Y. Lin, K. Yu, and J. Malik. Multi-component models for object detection. In *ECCV*, 2012.

[17] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, pages 951–958. IEEE, 2009.

[18] J. J. Lim, R. Salakhutdinov, and A. Torralba. Transfer learning by borrowing examples for multiclass object detection. In *NIPS*, 2011.

[19] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur. Dog breed classification using part localization. In *ECCV*, 2012.

[20] T. Malisiewicz, A. Gupta, and A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, pages 89–96, 2011.

[21] P. Ott and M. Everingham. Shared parts for deformable part-based models. In *CVPR*, pages 1513 –1520, june 2011.

[22] J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press, 1999.

[23] N. Razavi, J. Gall, P. Kohli, and L. V. Gool. Latent hough transforms for object detection. In *ECCV*, 2012.

[24] E. Rosch. Prototype classification and logical classification: The two systems. *New trends in conceptual representation: Challenges to Piagets theory*, pages 73–86, 1983.

[25] E. Rosch and C. Mervis. Family resemblances: Studies in the internal structure of categories. *Cognitive psychology*, 7(4):573–605, 1975.

[26] R. Salakhutdinov, A. Torralba, and J. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *CVPR*, pages 1481–1488. IEEE, 2011.

[27] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, pages 73–86. Springer, 2012.

[28] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE TPAMI*, 29(5):854–869, 2007.

[29] Y. Wang and G. Mori. A discriminative latent model of object classes and attributes. In *ECCV*, pages 155–168. Springer, 2010.

[30] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. *NIPS*, 17:1537–1544, 2004.

[31] W. Zhang, J. Sun, and X. Tang. Cat head detection - how to effectively exploit shape and texture features. In *ECCV*, 2008.

[32] L. Zhu, Y. Chen, A. L. Yuille, and W. T. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010.

[33] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, 2012.

[34] X. Zhu, C. Vondrick, D. Ramanan, and C. Fowlkes. Do we need more training data or better models for object detection? In *BMVC*, 2012.