Do We Need More Training Data?

Xiangxin Zhu · Carl Vondrick · Charless C. Fowlkes · Deva Ramanan

Abstract Datasets for training object recognition systems are steadily increasing in size. This paper investigates the question of whether existing detectors will continue to improve as data grows, or saturate in performance due to limited model complexity and the Bayes risk associated with the feature spaces in which they operate. We focus on the popular paradigm of discriminatively trained templates defined on oriented gradient features. We investigate the performance of mixtures of templates as the number of mixture components and the amount of training data grows. Surprisingly, even with proper treatment of regularization and “outliers”, the performance of classic mixture models appears to saturate quickly (∼10 templates and ∼100 positive training examples per template). This is not a limitation of the feature space as compositional mixtures that share template parameters via parts and that can synthesize new templates not encountered during training yield significantly better performance. Based on our analysis, we conjecture that the greatest gains in detection performance will continue to derive from improved representations and learning algorithms that can make efficient use of large datasets.

Keywords Object detection · mixture models · part models

1 Introduction

Much of the impressive progress in object detection is built on the methodologies of statistical machine learning, which make use of large training datasets to tune model parameters. Consider the benchmark results of the well-known PASCAL VOC object challenge (Fig. 1). There is a clear trend of increased benchmark performance over the years as new methods have been developed. However, this improvement is also correlated with increasing amounts of training data. One might be tempted to simply view this trend as another case of the so-called “effectiveness of big-data”, which posits that even very complex problems in artificial intelligence may be solved by simple statistical models trained on massive datasets (Halevy et al, 2009). This leads us to consider a basic question about the field: will continually increasing amounts of training data be sufficient to drive continued progress in object recognition absent the development of more complex object detection models?

To tackle this question, we collected a massive training set that is an order of magnitude larger than existing collections such as PASCAL (Everingham et al, 2010). We follow the dominant paradigm of scanning-window templates trained with linear SVMs on HOG features (Dalal and Triggs, 2005; Felzenszwalb et al, 2010; Bourdev and Malik, 2009; Malisiewicz et al, 2011), and evaluate detection performance as a function of the amount of training data and the model complexity.

Challenges: We found there is a surprising amount of subtlety in scaling up training data sets in current systems. For a fixed model, one would expect performance to generally increase with the amount of data and eventually saturate (Fig. 2). Empirically, we often saw the bizarre result that off-the-shelf implementations show decreased performance with additional data! One would also expect that to take advantage of additional train-
The best reported performance on PASCAL VOC challenge has shown marked increases since 2006 (top). This could be due to various factors: the dataset itself has evolved over time, the best-performing methods differ across years, etc. In the bottom row, we plot a particular factor — training data size — which appears to correlate well with performance. This begs the question: has the increase been largely driven from the availability of larger training sets?

We show that the apparent performance ceiling is not a consequence of HOG+linear classifiers. We provide an analysis of the popular deformable part model (DPM), showing that it can be viewed as an efficient way to implicitly encode and score an exponentially-large set of rigid mixture components with shared parameters. With the appropriate sharing, DPMs produce substantial performance gains over standard non-parametric mixture models. However, DPMs have fixed complexity and still saturate in performance with current amounts of training data, even when scaled to mixtures of DPMs. This difficulty is further exacerbated by the computational demands of non-parametric mixture models, which can be impractical for many applications.

Proposed solutions: In this paper, we offer explanations and solutions for many of these difficulties. First, we found it crucial to set model regularization as a function of training dataset using cross-validation, a standard technique which is often overlooked in current object detection systems. Second, existing strategies for discovering sub-category structure, such as clustering aspect ratios [Felzenszwalb et al. 2010], appearance features [Divvala et al. 2012], and keypoint labels [Bourdev and Malik 2009], may not suffice. We found this was related to the inability of classifiers to deal with “polluted” data when mixture labels were improperly assigned. Increasing model complexity is thus only useful when mixture components capture the “right” sub-category structure.

To efficiently take advantage of additional training data, we introduce a non-parametric extension of a DPM which we call an exemplar deformable part model (EDPM). Notably, EDPMs increase the expressive power of DPMs with only a negligible increase in computation, making them practically useful. We provide evidence that suggests that compositional representations of mixture templates provide an effective way to help target the “long-tail” of object appearances by sharing local part appearance parameters across templates.

Extrapolating beyond our experiments, we see the striking difference between classic mixture models and the non-parametric compositional model (both mixtures of linear classifiers operating on the same feature space) as evidence that the greatest gains in the near future will not be had with simple models+bigger data, but rather through improved representations and learning algorithms.

We introduce our large-scale dataset in Sec. 2, describe our non-parametric mixture models in Sec. 3, present extensive experimental results in Sec. 4, and conclude with a discussion in Sec. 5 including related work.

2 Big Detection Datasets

Throughout the paper we carry out experiments using two datasets. We vary the number of positive training examples, but in all cases keep the number of negative training images fixed. We found that performance was relatively static with respect to the amount of negative
training data, once a sufficiently large negative training set was used.

**PASCAL:** Our first dataset is a newly collected data set that we refer to as PASCAL-10X and describe in detail in the following section. This dataset covers the 11 PASCAL categories (see Fig.) and includes approximately 10 times as many training examples per category as the standard training data provided by the PASCAL detection challenge, allowing us to explore the potential gains of larger numbers of positive training instances. We evaluate detection accuracy on the 11 PASCAL categories from the PASCAL 2010 trainval dataset (because test annotations are not public), which contains 10000+ images.

**Faces:** In addition to examining performance on PASCAL object categories, we also trained models for face detection. We found faces to contain more structured appearance variation, which often allowed for more easily interpretable diagnostic experiments. Face models are trained using the CMU MultiPIE dataset (Gross et al., 2010), a well-known benchmark dataset of faces spanning multiple viewpoints, illumination conditions, and expressions. We use up to 900 faces across 13 viewpoints. Each viewpoint was spaced 15° apart spanning 180°. 300 of the faces are frontal, while the remaining 600 are evenly distributed among the remaining viewpoints. For negatives, we use 1218 images from the INRIAPerson database (Dalal and Triggs, 2005). Detection accuracy of face models are evaluated on the annotated face in-the-wild (AFW) (Zhu and Ramanan, 2012), which contains images from real-world environments and tend to have cluttered backgrounds with large variations in both face viewpoint and appearance.

### 2.1 Collecting PASCAL-10X

In this section, we describe our procedure for building a large, annotated dataset that is as similar as possible to the PASCAL 2010 for object detection. We found faces to contain more structured appearance variation, which often allowed for more easily interpretable diagnostic experiments. Face models are trained using the CMU MultiPIE dataset (Gross et al., 2010), a well-known benchmark dataset of faces spanning multiple viewpoints, illumination conditions, and expressions. We use up to 900 faces across 13 viewpoints. Each viewpoint was spaced 15° apart spanning 180°. 300 of the faces are frontal, while the remaining 600 are evenly distributed among the remaining viewpoints. For negatives, we use 1218 images from the INRIAPerson database (Dalal and Triggs, 2005). Detection accuracy of face models are evaluated on the annotated face in-the-wild (AFW) (Zhu and Ramanan, 2012), which contains images from real-world environments and tend to have cluttered backgrounds with large variations in both face viewpoint and appearance.

### Table 1

| Category    | PASCAL 2010 | Our Data Set |
|-------------|-------------|--------------|
| Bicycle     | 471         | 614          | 3,927 | 7,401 |
| Bus         | 353         | 498          | 3,405 | 4,919 |
| Cat         | 1,005       | 1,132        | 12,204 | 13,998 |
| Cow         | 248         | 464          | 3,194 | 6,909 |
| Dining Table| 415         | 468          | 3,905 | 5,651 |
| Horse       | 425         | 621          | 4,086 | 6,488 |
| Motorbike   | 453         | 611          | 5,674 | 8,666 |
| Sheep       | 290         | 701          | 2,351 | 6,018 |
| Sofa        | 406         | 451          | 4,018 | 7,448 |
| Train       | 453         | 524          | 6,403 | 7,648 |
| TV Monitor  | 490         | 683          | 5,053 | 7,808 |
| Totals      | 4,609       | 6,167        | 50,772 | 81,075 |

### Table 2

| Attributes | PASCAL 2010 | Our Data Set |
|------------|-------------|--------------|
| Truncated  | 30.8        | 31.5          | 15.8  |
| Occluded   | 5.9         | 8.6           | 7.1   |
| Jumping    | 4.0         | 4.3           | 5.4   |
| Standing   | 69.9        | 68.8          | 54.6  |
| Trotting   | 23.5        | 24.9          | 20.6  |
| Sitting    | 2.0         | 1.4           | 0.7   |
| Other      | 0.0         | 0.5           | 0     |
| Person Top | 21.4        | 29.1          | 57.5  |
| Person Besides | 8.8  | 10.0          | 8.6   |
| No Person  | 66.0        | 59.8          | 33.8  |

### Table 1 PASCAL 2010 trainval and our data set for select categories. Our data set is an order of magnitude larger.

### Table 2 Frequencies of attributes (percent) across images in our 10x horse data set compared to the PASCAL 2010 trainval data set. Bolded entries highlight significant differences relative to our collected data. Our dataset has similar attribute distribution to the PASCAL 2010, but differs significantly from 2007, which has many more sporting events.

**Collection:** We downloaded over one hundred thousand large images from Flickr to build our dataset. We took care to directly mimic the collection procedure used by the PASCAL organizers. We begin with a set of keywords (provided by the organizers) associated with each object class. For each class, we picked a random keyword, chose a random date since Flickr’s launch, selected a random page on the results, and finally took a random image from that page. We repeat this procedure until we had downloaded an order of magnitude larger number of images for each class.

**Filtering:** The downloaded images from Flickr did not necessarily contain objects for the category that we were targeting. We created MTurk tasks that asked workers to classify the downloaded images on whether they contained the category of interest. Our user interface in Fig. gave workers instructions on how to handle special cases and this resulted in acceptable annotation quality without finding agreement between workers.

---

1 The dataset can be downloaded from [http://vision.ics.uci.edu/datasets/](http://vision.ics.uci.edu/datasets/)
2.2 Data Quality

To verify the quality of our annotations, we performed an in-depth diagnostic analysis of a particular category (horses). Overall, our analysis suggests that our collection and annotation pipeline produces high-quality training data that is similar to PASCAL.

Attribute distribution: We first compared various distributions of attributes of bounding boxes from PASCAL-10X to those from both PASCAL 2010 and 2007 trainval. Attribute annotations were provided by manual labeling. Our findings are summarized in Tab. 2. Interestingly, horses collected in 2010 and 2007 vary significantly, while 2010 and PASCAL-10X match fairly well. Our images were on average twice the resolution as those in PASCAL so we scaled our images down to construct our final dataset.

User assessment: We also gauged the quality of our bounding boxes compared to PASCAL with a user study. We flashed a pair of horse bounding boxes, one from PASCAL-10X and one from PASCAL 2010, on a screen and instructed a subject to label which appeared to be better example. Our subject preferred the PASCAL 2010 data set 49% of the time and our data set 51% of the time. Since chance is 50%-50% and our subject operated close to chance, this further suggests PASCAL-10X matched well with PASCAL. Qualitatively, the biggest difference observed between the two datasets was that PASCAL-10X bounding boxes tend to be somewhat “looser” than the (hand curated) PASCAL 2010 data.

Redundant annotations: We tested the use of multiple annotations for removing poorly labeled positive examples. All horse images were labeled twice, and only those bounding boxes that agreed across the two annotation sessions were kept for training. We found that training on these cross-verified annotations did not significantly affect the performance of the learned detector.

3 Mixture models

To take full advantage of additional training data, it is vital to grow model complexity. We accomplish this by adding a mixture component to capture additional “sub-category” structure. In this section, we describe various approaches for learning and representing mixture models. Our basic building block will be a mixture of linear classifiers, or templates. Formally speaking, we compute the detection score of an image window $I$ as:

$$S(I) = \max_m \left[ w_m \cdot \phi(I) + b_m \right]$$  \hspace{1cm} (1)
Fig. 4 We compare supervised versus automatic (k-means) approaches for clustering by displaying the average RGB image of each cluster. The supervised methods use viewpoint labels to cluster the training data. Because our face data is relatively clean, both obtain reasonably good clusters. However, at some levels of the hierarchy, unsupervised clustering does seem to produce suboptimal partitions - for example, at $K = 2$. There is no natural way to group multi-view faces into two groups. Automatically selecting $K$ is a key difficulty with unsupervised clustering algorithms.

where $m$ is a discrete mixture variable, $\Phi(I)$ is a HOG image descriptor [Dalal and Triggs 2005], $w_m$ is a linearly-scored template, and $b_m$ is an (optional) bias parameter that acts as a prior that favors particular templates over others.

3.1 Independent mixtures

In this section, we describe approaches for learning mixture models by clustering positive examples from our training set. We train independent linear classifiers $(w_m, b_m)$ using positive examples from each cluster. One difficulty in evaluating mixture models is that fluctuations in the (non-convex) clustering results may mask variations in performance we wish to measure. We took care to devise a procedure for varying $K$ (the number of clusters) and $N$ (the amount of training data) in such a manner that would reduce stochastic effects of random sampling.

Unsupervised clustering: For our unsupervised baseline, we cluster the positive training images of each category into 16 clusters using hierarchical k-means, recursively splitting each cluster into $k = 2$ subclusters. For example, given a fixed training set, we would like the cluster partitions for $K = 8$ to respect the cluster partition of $K = 4$. To capture both appearance and shape when clustering, we warp an instance to a canonical aspect ratio, compute its HOG descriptor (reduce the dimensionality with PCA for computational efficiency), and append the aspect ratio to the resulting feature vector.

Partitioned sampling: Given a fixed training set of $N_{max}$ positive images, we would like to construct a smaller sampled subset, say of $N = \frac{N_{max}}{2}$ images, whose cluster partitions respect those in the full dataset. This is similar in spirit to stratified sampling and attempts to reduce variance in our performance estimates due to “binning artifacts” of inconsistent cluster partitions across re-samplings of the data.

To do this, we first hierarchically-partition the full set of $N_{max}$ images by recursively applying k-means. We then subsample the images in the leaf nodes of the
In this section, we describe various architectures for compositional mixture models that share information between mixture components. We share local spatial regions of templates, or parts. We begin our discussion by reviewing standard architectures for deformable part models (DPMs), and show how they can be interpreted and extended as high-capacity mixture models.

**Deformable Part Models (DPMs):** We begin with an analysis that shows that DPMs are equivalent to an exponentially-large mixture of rigid templates Eqn. (1). This allows us to analyze (both theoretically and empirically) under what conditions a classic mixture model will approach the behavior of a DPM. Let the location of part $i$ be $(x_i, y_i)$. Given an image $I$, a DPM scores a configuration of $P$ parts $(x, y) = \{(x_i, y_i) : i = 1..P\}$ as:

$$S_{DPM}(I) = \max_{x,y} S(I, x, y)$$

where

$$S(I, x, y) = \sum_{i=1}^{P} \sum_{(u,v) \in W_i} \alpha_i [u, v] \cdot \phi(I, x_i + u, y_i + v) + \sum_{ij \in E} \beta_{ij} \cdot \psi(x_i - x_j - a_{ij}^x, y_i - y_j - a_{ij}^y) \quad (2)$$

where $W_i$ defines the spatial extent (length and width) of part $i$. The first term defines a local appearance score, where $\alpha_i$ is the appearance template for part $i$ and $\phi(I, x_i, y_i)$ is the appearance feature vector extracted from location $(x_i, y_i)$. The second term defines a pairwise deformation model that scores the relative placement of a pair of parts with respect to an anchor position $(a_{ij}^x, a_{ij}^y)$. For simplicity, we have assumed all filters are defined at the same scale, though the above can be extended to the multi-scale case. When the associated relational graph $G = (V, E)$ is tree-structured, one can compute the best-scoring part configuration $\max_{(x,y) \in \Omega} S(I, x, y)$ with dynamic programming, where $\Omega$ is the space of possible part placements. Given that each of $P$ parts can be placed at one of $L$ locations, $|\Omega| = L^P \approx 10^{20}$ for our models.

By defining index variables in image coordinates $u' = x_i + u$ and $v' = y_i + v$, we can rewrite Eqn. (2) as:

$$S(I, x, y) = \sum_{u', v'} \sum_{i=1}^{P} \alpha_i [u' - x_i, v' - y_i] \cdot \phi(I, u', v') + \sum_{ij \in E} \beta_{ij} \cdot \psi(x_i - x_j - a_{ij}^x, y_i - y_j - a_{ij}^y) = \sum_{u', v'} w(x, y) [u', v'] \cdot \phi(I, u', v') + b(x, y)$$

$$= w(x, y) \cdot \phi(I) + b(x, y) \quad (3)$$
where \( w(x, y) | u', v' \) = \( \sum_{i=1}^{P} \alpha_i | u' - x_i, v' - y_i | \). For notational convenience, we assume parts templates are padded with zeros outside of their default spatial extent.

From the above expression it is easy to see that the DPM scoring function is formally equivalent to an exponentially-large mixture model where each mixture component \( m \) is indexed by a particular configuration of parts \((x, y)\). The template corresponding to each mixture component \( w(x, y) \) is constructed by adding together parts at shifted locations. The bias corresponding to each mixture component \( b(x, y) \) is equivalent to the spatial deformation score for that configuration of parts.

DPMs differ from classic mixture models previously defined in that they (1) share parameters across a large number of mixtures or rigid templates, (2) extrapolate by “synthesizing” new templates not encountered during training, and finally, (3) use dynamic programming to efficiently search over a large number of templates.

**Exemplar Part Models (EPMs):** To analyze the relative importance of part parameter sharing and extrapolation to new part placements, we define a part model that limits the possible configurations of parts to those seen in the \( N \) training images, written as

\[
S_{EPM}(I) = \max_{(x, y) \in \Omega_N} S(I, x, y) \quad \text{where} \quad \Omega_N \subseteq \Omega. \tag{4}
\]

We call such a model an Exemplar Part Model (EPM), since it can also be interpreted as set of \( N \) rigid exemplars with shared parameters. EPMs are not to be confused with exemplar DPMs (EDPMs), which we will shortly introduce as their deformable counterpart. EPMs can be optimized with a discrete enumeration over \( N \) rigid templates rather than dynamic programming. However, by caching scores of the local parts, this enumeration can be made quite efficient even for large \( N \). EPMs have the benefit of sharing, but cannot synthesize new templates that were not present in the training data. We visualize example EPM templates in Fig. 6.

To take advantage of additional training data, we would like to explore non-parametric mixtures of DPMs. One practical issue is that of computation. We show that with a particular form of sharing, one can construct non-parametric DPMs that are no more computationally complex than standard DPMs or EPMs, but considerably more flexible in that they extrapolate multi-modal shape models to unseen configurations.

**Exemplar DPMs (EDPMs):** To describe our model, we first define a mixture of DPMs with a shared appearance model, but mixture-specific shape models. In the extreme case, each mixture will consist of a single training exemplar. We describe an approach that shares both the part filter computations and dynamic programming messages across all mixtures, allowing us to eliminate almost all of the mixture-dependant computation. Specifically, we consider mixture of DPMs of the form:

\[
S(I) = \max_{m \in \{1...M\}, z \in \Omega} \left[ w(z) \cdot \phi(I) + b_m(z) \right] \tag{5}
\]

where \( z = (x, y) \) and we write a DPM as an inner maximization over an exponentially-large set of templates indexed by \( z \in \Omega \), as in Eqn. 5. Because the appearance model does not depend on \( m \), we can write:

\[
S(I) = \max_{z \in \Omega} \left[ w(z) \cdot \phi(I) + b(z) \right] \tag{6}
\]

where \( b(z) = \max_m b_m(z) \). Interestingly, we can write the DPM, EPM, and EDPM in the form of Eqn. 6 by simply changing the shape model \( b(z) \):

\[
b_{DPM}(z) = \sum_{ij \in E} \beta_{ij} \cdot \psi(z_i - z_j - a_{ij}) \tag{7}
\]

\[
b_{EDPM}(z) = \max_{m \in \{1...M\}} \sum_{ij \in E} \beta_{ij} \cdot \psi(z_i - z_j - a^{m}_{ij}) \tag{8}
\]

\[
b_{EPM}(z) = b_{DPM}(z) + b_{EDPM}(z) \tag{9}
\]

where \( a^{m}_{ij} \) is the anchor position for part \( i \) and \( j \) in mixture \( m \). We write \( b^{*}_{EDPM}(z) \) to denote a limiting case of \( b_{EDPM}(z) \) with \( \beta_{ij} = -\infty \) and thus takes a value of 0 when \( z \) has the same relative part locations as some exemplar \( m \) and \(-\infty\) otherwise.

While the EPM only considers \( M \) different part configurations to occur at test time, the EDPM extrapolates away from these shape exemplars. The spring parameters \( \beta \) in the EDPM thus play a role similar to the kernel width in kernel density estimation. We show a visualization of these shape models as probabilistic priors in Fig. 7.
Fig. 6 Classic exemplars vs EPMs. On the top row, we show three rigid templates trained as independent exemplar mixtures. Below them, we show their counterparts from an exemplar part model (EPM), along with their corresponding training images. EPMs share spatially-localized regions (or “parts”) between mixtures. Each rigid mixture is a superposition of overlapping parts. A single part is drawn in blue. We show parts on the top row to emphasize that these template regions are trained independently. On the right, we show a template which is implicitly synthesized by a DPM for a novel test image on-the-fly. In Fig. 15, we show that both sharing of parameters between mixture components and implicit generation of mixture components corresponding to unseen part configurations contribute to the strong performance of a DPM.

Inference: We now show that inference on EDPMs (Eqn. 5) can be quite efficient. Specifically, inference on a star-structured EDPM is no more expensive than a EPM built from the same training examples. Recall that EPMs can be efficiently optimized with a discrete enumeration of $N$ rigid templates with “intelligent caching” of part scores. Intuitively, one computes a response map for each part, and then scores a rigid template by looking up shifted locations in the response maps. EDPMs operate in a similar same manner, but one convolves a “min-filter” with each response map before looking up shifted locations. To be precise, we explicitly write out the message-passing equations for a star-structured EDPM below, where we assume part $i = 1$ is the root without loss of generality:

$$S_{EDPM}(I) = \max_{z_1,m} \left[ \alpha_1 \cdot \phi(I, z_1) + \sum_{j>1} m_j(z_1 + \alpha_{1j}) \right]$$ (10)

$$m_j(z_1) = \max_{z_j} \left[ \alpha_j \cdot \phi(I, z_j) + \beta_{1j} \cdot \psi(z_1 - z_j) \right]$$ (11)

The maximization in Eqn. (11) needs only be performed once across mixtures, and can be computed efficiently with a single min-convolution or distance transform (Felzenszwalb and Huttenlocher 2012). The resulting message is then shifted by mixture-specific anchor positions $\alpha_{1j}$ in Eqn. (10). Such mixture-independent messages can be computed only for leaf parts, because internal parts in a tree will receive mixture-specific messages from downstream children. Hence star EDPMs are essentially no more expensive than a EPM (because a single min-convolution per part adds a negligible amount of computation). In our experiments, running a 2000-mixture EDPM is almost as fast as a standard 6-mixture DPM. Other topologies beyond stars might provide greater flexibility. However, since EDPMs encode shape non-parametrically using many mixtures, each individual mixture may need not deform too much, making a star-structured deformation model a reasonable approximation (Fig. 7).

4 Experiments

Armed with our array of non-parametric mixture models and datasets, we now present an extensive diagnostic
analysis on 11 PASCAL categories from the 2010 PASCAL trainval set and faces from the Annotated Faces in the Wild test set (Zhu and Ramanan, 2012). For each category, we train the model with varying number of samples ($N$) and mixtures ($K$). To train our independent mixtures, we learn rigid HOG templates (Dalal and Triggs, 2005) with linear SVMs (Chang and Lin, 2011). We calibrated SVM scores using Platt scaling (Platt, 1999). Since the goal is to calibrate scores of mixture components relative to each other, we found it sufficient to train scaling parameters using the original training set rather than using a held-out validation set. To train our compositional mixtures, we use a locally-modified variant of the codebase from (Felzenszwalb et al., 2010).

To show the uncertainty of the performance with respect to different sets of training samples, we randomly re-sample the training data 5 times for each $N$ and $K$ following the partitioned sampling scheme described in Sec. 3. The best regularization parameter $C$ for the SVM was selected by cross validation. For diagnostic analysis, we first focus on faces and buses.

**Evaluation:** We adopt the PASCAL VOC precision-recall protocol for object detection (requiring 50% overlap), and report average precision (AP). While learning theory often focuses on analyzing 0-1 classification error rather than AP (McAllester, 1999), we experimentally verified that AP typically tracks 0-1 classification error and so focus on the former in our experiments.

4.1 The importance of proper regularization

We begin with a rather simple experiment: how does a single rigid HOG template tuned for faces perform when we give it more training data $N$? Fig. 8 shows the surprising result that additional training data can decrease performance! For imbalanced object detection datasets with many more negatives than positives, the hinge loss appears to grow linearly with the amount of positive training data; if one doubles the number of positives, the total hinge loss also doubles. This leads to overfitting. To address this problem, we found it crucial to cross-validate $C$ across different $N$. By doing so, we do see better performance with more data (Fig. 8). While cross-validating regularization parameters is a standard procedure when applying a classifier to a new dataset, most off-the-shelf detectors are trained using a fixed $C$ across object categories with large variations in the number of positives. We suspect other systems based on standard detectors (Felzenszwalb et al., 2010; Dalal and Triggs, 2005) may also be suffering from sub-optimal regularization and might show an improvement by proper cross-validation.

4.2 The importance of clean training data

Although proper regularization parameters proved to be crucial, we still discovered scenarios where additional training data hurt performance. Fig. 9 shows an experiment with a fixed set of $N$ training examples where we train two detectors: (1) *All* is trained with all $N$ examples, while (2) *Frontal* is trained with a smaller, “clean” subset of examples containing frontal faces. We cross-validate $C$ for each model for each $N$. Surprisingly, *Frontal* outperforms *All* even though it is trained with less data.

This outcome cannot be explained by a failure of the model to generalize from training to test data. We examined the training loss for both models, evaluated on the full training set. As expected, *All* has a lower SVM
In (a), we compare the performance of a single HOG template trained with $N$ multi-view face examples, versus a template trained with a subset of those $N$ examples corresponding to frontal faces. The frontal-face template (b) looks “cleaner” and makes fewer classification errors on both testing and training data. The fully-trained template (c) looks noisy and performs worse, even though it produces a lower SVM objective value (when both (b) and (c) are evaluated on the full training set). This suggests that SVMs are sensitive to noise and benefit from training with “clean” data.

The single bicycle template (marked with red) alone achieves $ap=29.4\%$, which is almost equivalent to the performance of using all 8 mixtures ($ap=29.7\%$). Both models strongly outperform a single-mixture model trained on the full training set. This suggests that these additional mixtures are useful during training to capture outliers and prevent “noisy” data from polluting a “clean” template that does most of the work at test time.

The importance of “clean” training data suggests it could be fruitful to correctly cluster training data into mixture components where each component is “clean”. We evaluated the effectiveness of providing fully supervised clustering in producing clean mixtures. In Fig. 11, we see a small 2% to 5% increase for manual clustering. In general, we find that unsupervised clustering can work reasonably well but depends strongly on the category and features used. For example, the DPM implementation of (Felzenszwalb et al, 2010) initializes mixtures based on aspect ratios. Since faces in different viewpoint share similar aspect ratios, this tends to produce “unclean” mixtures compared to our non-latent clustering.

4.3 Performance of independent mixtures

Given the right regularization and clean mixtures trained independently, we now evaluate whether performance asymptotes as the amount of training data and the model complexity increase.

Fig. 12 shows performance as we vary $K$ and $N$ after cross-validating $C$ and using supervised clustering. Fig. 12a demonstrates that increasing the amount of training data yields a clear improvement in performance...
at the beginning, and the gain quickly becomes smaller later. Larger models with more mixtures tend to perform worse with fewer examples due to over fitting, but eventually win with more data. Surprisingly, improvement tends to saturate at ~100 training examples per mixture and decreases when we split the training data too much so that each mixture only has few samples.

We performed similar experiments for all 11 PASCAL object categories in our PASCAL-10X dataset shown in Fig. 13. We evaluate performance on the PASCAL 2010 trainval set since the testset annotations are not public. We cluster the training data into \( K = \{1, 2, 4, 8, 16\} \) mixture components, and \( N = \{50, 100, 500, 1000, 3000, N_{\text{max}}\} \) training samples, where \( N_{\text{max}} \) is the number of training samples collected for the given category. For each \( N \), we select the best \( C \) and \( K \) through cross-validation. Fig. 13a appears to suggest that performance is saturating across all categories as we increase the amount of training data. However, if we plot performance on a log scale (Fig. 13b), it appears to increase roughly linearly. This suggests that the required training data may need to grow exponentially to produce a fixed improvement in accuracy. For example, if we extrapolate the steepest curve in Fig. 13b (motorbike), we will need \( 10^{12} \) motorbike samples to reach 95% AP!

Of course 95% AP may not be an achievable level of performance. There is some upper-bound imposed by the Bayes risk associated with the HOG feature space which no amount of training data will let us surpass. Are classic mixtures of rigid templates approaching the Bayes optimal performance? Of course we cannot compute the Bayes risk so this is hard to answer in general. However, the performance of any system operating on the same data and feature space provides a lower bound on the optimal performance. We next analyze the performance of compositional mixtures to provide much better lower bound on optimal performance.

4.4 Performance of compositional mixtures

We now perform a detailed analysis of compositional mixture models, including DPMs, EPMs, and EDPMs. We focus on face detection and Pascal buses. We consider the latent star-structured DPM of [Felzenszwalb et al. 2010] as our primary baseline. For face detection, we also compare to the supervised tree-structured DPM of [Zhu and Ramanan 2012], which uses facial landmark annotations in training images as supervised part locations. Each of these DPMs makes use of different parts, and so can be used to define different EPMs and EDPMs. We plot performance of faces in Fig. 15 and buses in Fig. 16.

**Supervised DPMs:** For face detection, we first note that a supervised DPM can perform quite well (91% AP) with less than 200 example faces. This represents a lower bound on the maximum achievable performance with a mixture of linear templates given a fixed training set. This performance is noticeably higher than that of our cross-validated rigid mixture model, which maxes out at an AP of 76% with 900 training examples. By extrapolation, we predict that one would need \( N = 10^{10} \) training examples to achieve the DPM performance. To analyze where this performance gap is coming from, we now evaluate the performance of various compositional mixtures models.

**Latent parts:** We begin by analyzing the performance of compositional mixtures defined by latent parts, as they can be constructed for both faces and Pascal buses. Recall that EPMs have the benefit of sharing parameters between rigid templates, but they cannot extrapolate to new shape configurations not seen among the \( N \) training examples. EPMs noticeably improve performance over independent mixtures, improving AP
Fig. 13 We plot the best performance at varying amount of training data for 11 PASCAL categories on PASCAL 2010 trainval set. (a) shows that all the curves look saturated with a relatively small amount of training data; but in log scale (b) suggests a diminishing return instead of true saturation. However the performance increases so slow that we will need more than $10^{12}$ examples per category to reach 95% AP if we keep growing at the same rate.

from 76% to 78.5% for faces and improving AP from 56% to 64% for buses. In fact, for large $N$, they approach the performance of latent DPMs, which is 79% for faces and 63% for buses. For small $N$, EPMs somewhat underperform DPMs. This makes sense: with very few observed shape configurations, exemplar-based methods are limited. But interestingly, with a modest number of observed shapes ($\approx 1000$), exemplar-based methods with parameter sharing can approach the performance of DPMs. This in turn suggests that extrapolation to unseen shapes is may not be crucial, at least in the latent case. This is further evidenced by the fact that EDPMs, the deformable counterpart to EPMs, perform similarly to both EPMs and DPMs.

**Supervised parts:** The story changes somewhat for supervised parts. Here, supervised EPMs outperform independent mixtures 85% to 76%. Perhaps surprisingly, EPMs even outperform latent DPMs. However, supervised EPMs still underperform a supervised DPM. This suggests that, in the supervised case, the performance gap (85% vs 91%) stems from the ability of DPMs to synthesize configurations that are not seen during training. Moreover, the reduction in relative error due to extrapolation is more significant than the reduction due to part sharing. (Zhu and Ramanan, 2012) point out that a tree-structured DPM significantly outperforms a star-structured DPM, even when both are trained with the same supervised parts. One argument is that trees better capture nature spatial constraints of the model, such as the contour-like continuity of small parts. Indeed, we also find that a star-structured DPM does a “poorer” job of extrapolation. In fact, we show that an EDPM does as well a supervised star model, but not quite up to the performance of a tree DPM.

**Analysis:** Our results suggest that part models can be seen as a mechanism for performing intelligent parameter sharing across observed mixture components and extrapolation to implicit, unseen mixture components. Both these aspects contribute to the strong performance of DPMs. However, with the “right” set of (supervised) parts and the “right” geometric (tree-structured) constraints, extrapolation to unseen templates has the potential to be much more significant. We see this as a consequence of the “long-tail” distribution of object shape (Fig. 17): many object instances can be modeled with a few shape configurations, but there exists of long tail of unusual shapes. Examples from the long tail may be difficult to observe in any finite training dataset, suggesting that extrapolation is crucial for recognizing these cases. Once the representation for sharing and extrapolation is accurately specified, fairly little training data is needed. Indeed, our analysis shows that one can train a state-of-the-art face detector (Zhu and Ramanan, 2012) with 50 face images.

**Relation to Exemplar SVMs:** In the setting of object detection, we were not able to see significant performance improvements due to our non-parametric compositional mixtures. However, EDPMs may be useful for other tasks. Specifically, they share an attractive property of exemplar SVMs (ESVMs) (Malisiewicz et al, 2011): each detection can be affiliated with its closest matching training example (given by the mixture index), allowing us to transfer annotations from a training example to the test instance. (Malisiewicz et al, 2011) argue that non-parametric label transfer is an effective
Do We Need More Training Data? 13

Fig. 14 We visualize detections using our exemplar DPM (EDPM) model. As opposed to existing exemplar-based methods [Malisiewicz et al, 2011], our model shared parameters between exemplars (and so is faster to evaluate) and can generalize to unseen shape configurations. Moreover, EDPMs returns corresponding landmarks between an exemplar and a detected instance (and hence an associated set of landmark deformation vectors), visualized on the top row of faces.

way of transferring associative knowledge, such as 3D pose, segmentation masks, attribute labels, etc. However, unlike ESVMs, EDPMs share computation among the exemplars (and so are faster), can generalize to unseen configurations (since they can extrapolate to new shapes), and also report a part deformation field associated with each detection (which maybe useful to warp training labels to better match the detected instance). We show example detections (and their matching exemplars) in Fig. 14.

5 Related Work

We view our study as complementary to other meta-analysis of the object recognition problem, such as studies of the dependence of performance on the number of object categories [Deng et al, 2010], visual properties [Hoiem et al, 2012], dataset collection bias [Torralba and Efros, 2011], and component-specific analysis of recognition pipelines [Parikh and Zitnick, 2011].

Object detection: Our analysis is focused on template-based approaches to recognition, as such methods are currently competitive on challenging recognition problems such as PASCAL. However, it behooves us to recognize the large body of alternate approaches including hierarchical or “deep” feature learning [Krizhevsky et al, 2012], local feature analysis [Tuyltelaars and Mikolajczyk, 2003], and flexible template matching [Liu et al, 2011].
We plot the number of distinct shape patterns in our training set of buses and faces. Each training example is “binned” into a discrete shape by quantizing a vector of part locations. The above histograms count the number of examples that fall into a particular shape bin. In both cases, the number of occurrences seems to follow a long-tail distribution: a small number of patterns are common, while there are a huge number of rare cases. Interestingly, there are less than 500 unique bus configurations observed in our PASCAL-10X dataset of 2000 training examples. This suggests that one can build an exemplar part model (EPM) from the “right” set of 500 training examples and still perform similarly to a DPM trained on the full dataset (Fig. 16).

Non-parametric models in vision: Most relevant to our analysis is work on data-driven models for recognition. Non-parametric scene models have been used for scene completion (Hays and Efros 2007), geolocation (Hays and Efros 2008), Exemplar-based methods have also been used for scene-labeling through label transfer (Liu et al. 2011 Tighé and Lazebnik 2010). Other examples include nearest-neighbor methods for low-resolution image analysis (Torralba et al. 2008) and image classification (Zhang et al. 2006 Boiman et al. 2008). The closest approach to us is (Malisiewicz et al. 2011), who learn exemplar templates for object detection. Our analysis suggests that it is crucial to share information between exemplars and extrapolate to unseen templates by re-composing parts to new configurations.

Scalable nearest-neighbors: We demonstrate that compositional part models are one method for efficient nearest-neighbor computations. Prior work has explored approximate methods such as hashing (Shakhnarovich et al. 2003, 2005) and kd-trees (Muja and Lowe 2009 Beis and Lowe 1997). Our analysis shows that one can view parts as tools for exact and efficient indexing into an exponentially-large set of templates. This suggests an alternative perspective of parts as computational entities rather than semantic ones.

6 Conclusion

We have performed an extensive analysis of the current dominant paradigm for object detection using HOG feature templates. We specifically focused on performance as a function of the amount of training data, and introduced several non-parametric models to diagnose the state of affairs.

To scale current systems to larger datasets, we find that one must get certain “details” correct. Specifically, (a) cross-validation of regularization parameters is mundane but crucial, (b) current discriminative classification machinery is overly sensitive to noisy data, suggesting that (c) manual cleanup and supervision or more clever latent optimization during learning may play an important role for designing high-performance detection systems. We also demonstrate that HOG templates have a relatively small effective capacity; one can train accurate HOG templates with 100-200 positive examples (rather than thousands of examples as is typically done (Dalal and Triggs 2005)).

From a broader perspective, an emerging idea in our community is that object detection might be solved with simple models backed with massive training sets. Our experiments suggest a slightly refined view. Given the size of existing datasets, it appears that the current state-of-the-art will need significant additional data (perhaps exponentially larger sets) to continue producing consistent improvements in performance. We found that larger gains were possible by enforcing richer constraints within the model, often through non-parametric compositional representations that could make better use of additional data. In some sense, we need “better models” to make better use of “big data”.

Another common hypothesis is that we should focus on developing better features, not better learning algorithms. While HOG is certainly limited, we still see substantial performance gains without any change in the features themselves or the class of discriminant functions. Instead, the strategic issues appear to be parameter sharing, compositionality, and non-parametric
encodings. Establishing and using accurate, clean correspondence amongst training examples (e.g., that specify that certain examples belong to the same sub-category, or that certain spatial regions correspond to the same part) and developing non-parametric compositional approaches that implicitly make use of augmented training sets appear the most promising directions.

References

Beis JS, Lowe DG (1997) Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In: Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, IEEE, pp 1000–1006

Boiman O, Shechtman E, Irani M (2008) In defense of nearest-neighbor based image classification. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE, pp 1–8

Bosch A, Zisserman A, Muoz X (2007) Image classification using random forests and ferns. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE, pp 1–8

Bourdev L, Malik J (2009) Poselets: Body part detectors trained using 3d human pose annotations. In: International Conference on Computer Vision

Chang C, Lin C (2011) LIBSVM: A library for support vector machines. ACM Transactions on Systems and Technology 2:27:1–27:27, software available at [http://www.csie.ntu.edu.tw/~cjlin/libsvm](http://www.csie.ntu.edu.tw/~cjlin/libsvm)

Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: CVPR 2005.

Deng J, Berg A, Li K, Fei-Fei L (2010) What Does Classifying More Than 10,000 Image Categories Tell Us? In: International Conference on Computer Vision

Divvala SK, Efros AA, Hebert M (2012) How important are deformable parts in the deformable parts model? In: European Conference on Computer Vision (ECCV), Parts and Attributes Workshop

Everingham M, Van Gool L, Williams C, Winn J, Zisserman A (2010) The PASCAL visual object classes (VOC) challenge. International Journal of Computer Vision 88(2):303–338

Felzenszwalb P, Huttenlocher D (2012) Distance transforms of sampled functions. Theory of Computing 8(19)

Felzenszwalb P, Girshick R, McAllester D, Ramanan D (2010) Object detection with discriminatively trained part-based models. IEEE TPAMI

Gross R, Matthews I, Cohn J, Kanade T, Baker S (2010) Multi-pie. Image and Vision Computing

Halevy A, Norvig P, Pereira F (2009) The unreasonable effectiveness of data. Intelligent Systems, IEEE 24(2):8–12

Hays J, Efros A (2007) Scene completion using millions of photographs. In: ACM Transactions on Graphics (TOG), ACM, vol 26, p 4

Hays J, Efros AA (2008) Im2gps: estimating geographic information from a single image. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE, pp 1–8

Hoiem D, Chodpathumvan Y, Dai Q (2012) Diagnosing error in object detectors. In: Computer Vision ECCV 2012, Springer Berlin Heidelberg, vol 7574, pp 340–353

Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25, pp 1106–1114

Liu C, Yuen J, Torralba A (2011) Nonparametric scene parsing via label transfer. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33(12):2368–2382

Malisiewicz T, Gupta A, Efros A (2011) Ensemble of exemplar-svms for object detection and beyond. In: International Conference on Computer Vision, IEEE, pp 89–96

McAllester DA (1999) Some pac-bayesian theorems. Machine Learning 37(3):355–363

Muja M, Lowe DG (2009) Fast approximate nearest neighbors with automatic algorithm configuration. In: International Conference on Computer Vision Theory and Applications (VISSAPP09), pp 331–340

Parikh D, Zitnick C (2011) Finding the weakest link in person detectors. In: Computer Vision and Pattern Recognition, IEEE, pp 1425–1432

Platt J (1999) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: ADVANCES IN LARGE MARGIN CLASSIFIERS, MIT Press, pp 61–74

Shakhnarovich G, Viola P, Darrell T (2003) Fast pose estimation with parameter-sensitive hashing. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on, IEEE, pp 750–757

Shakhnarovich G, Darrell T, Indyk P (2005) Nearest-neighbor methods in learning and vision: theory and practice, vol 3. MIT press Cambridge

Tighe J, Lazebnik S (2010) Superparsing: scalable non-parametric image parsing with superpixels. In: Computer Vision–ECCV 2010, Springer, pp 352–365

Torralba A, Efros A (2011) Unbiased look at dataset bias. In: Computer Vision and Pattern Recognition, IEEE, pp 1521–1528

Torralba A, Fergus R, Freeman WT (2008) 80 million tiny images: A large data set for nonparametric object
and scene recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on 30(11):1958–1970
Tuytelaars T, Mikolajczyk K (2008) Local invariant feature detectors: a survey. Foundations and Trends® in Computer Graphics and Vision 3(3):177–280
Vedaldi A, Gulshan V, Varma M, Zisserman A (2009) Multiple kernels for object detection. In: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, pp 606–613
Wu Y, Liu Y (2007) Robust truncated hinge loss support vector machines. Journal of the American Statistical Association 102(479):974–983
Zhang H, Berg AC, Maire M, Malik J (2006) Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, IEEE, vol 2, pp 2126–2136
Zhu X, Ramanan D (2012) Face detection, pose estimation, and landmark localization in the wild. In: Computer Vision and Pattern Recognition