Bayesian Integrate and Shift (BIAS) Model: Incorporating Deep Learning Features for Unsupervised Learning and Single-Example Learning of Object Parts

Brian Hu

Johns Hopkins University, Baltimore MD 21218

Abstract

During visual perception of complex objects, humans shift their gaze to different salient regions of a particular object in order to gain more information about that object. For example, when looking at a face, humans will often fixate on the eyes, the nose, and the mouth, which are particularly informative regions for recognizing a face. The Bayesian Integrate and Shift (BIAS) model is a biologically inspired model for learning visual object categories that is modeled after the process of human visual perception.

Previous works have demonstrated the ability of the BIAS model to learn new object categories from only a few training examples by integrating information from within and across different fixations. One limitation of these previous models is the requirement for supervised learning of object parts using regions selected by a human teacher. In this report, we extend the BIAS model to include more complex features taken from deep learning along with unsupervised learning methods and show that we are also able to learn object parts in an unsupervised manner.

Keywords: Unsupervised Learning, Deep Learning, Object Parts

1. Introduction

The human visual system is unmatched in its ability to learn visual object categories from only a few examples. This is accomplished starting in early visual areas through an array of neurons that are selective for specific features and whose receptive fields (RFs) are spatially organized in a specific way. The size of RFs near the fixation point are small whereas the size of RFs that cover more peripheral regions increases with the distance from the fixation point. Although we perceive the world around us as a continuous stream of high resolution images, our visual system somehow knows how to “discard” certain image regions that are not informative (the background) while seamlessly integrating information from the different regions or parts that make up objects of interest. If we are not able to recognize an object after a single fixation, then we often make additional saccades, combine the information obtained from these different fixations, and as a result improve our perception of the object.

In contrast to human visual processing, most computer vision systems process images at a uniform resolution and hence do not require this saccade-like image exploration. Futhermore, object categories, if learned in a supervised way, usually do not contain information about the structure of the object or object “parts”. Despite significant advances in computer vision in recent years, many artificial intelligence systems designed for object recognition (including those resulting from the recent explosion of interest in deep learning) also require large numbers of training examples (hundreds to thousands or more).

As an alternative to these approaches, the Bayesian Integrate and Shift (BIAS) model is a biologically-inspired, feature-based model for learning visual object categories [1, 2]. The model mimics the human visual system by using an array of RFs for visual processing and integrating information across RFs and eye movements. In the BIAS model, object categories are learned by training the system on various points of view of an object in a “parts-based” manner. In the following sections, we will describe the model as well as our improvements in making the system able to learn objects parts in an unsupervised manner.

2. The Model

2.1. Receptive Fields

The RFs in the BIAS model form a fixed grid of concentric rings arranged with hexagonal packing, and are parameterized by an overlap factor and scaling factor, which determine the location and size of RFs in the different rings of the model. There is a central fixation point which is centered on different points in the image, and the locations of all other RFs are defined relative to this central RF. Typically, we used models with a total of eight RFs for each ring, evenly spaced at intervals of $\pi/4$ within a ring. We also used between four to five rings, including the central RF. The grid of RFs can be shifted to different locations in the image, representing saccade-like shifts in the viewer's gaze.

2.2. Feature Detectors

In the original BIAS model, each RF is associated with a set of feature detectors. Traditionally, Gabor filters which serve as edge detectors of various orientations and sizes were used to model the responses of simple and complex cells found in early visual cortex. For the initial experiments that used Gabor filters, a total of four orientations ($\theta = 0, \pi/4, \pi/2, 3\pi/4$) and scales ($\sigma = 2, 4, 6, 8$) were used. For each feature detector, we introduce relative position invariance by taking the max response of each feature detector within a RF. As a result, each feature detector indicates the maximum filter response over the entire RF, which allows the feature detectors to be more robust to the specific location of the feature within the RF. There is also a tradeoff in RFs further away from the central fixation point, as they have larger sizes which can tolerate more uncertainty, but as a result, are also less specific. We introduce the deep learning feature detectors that we used to extend the model in Section 3.

2.3. Learning and Classification

We represent the feature detector outputs as a mixture of Gaussians, and during the learning phase, the BIAS model learns the model parameters for a specific object view or part and of the image background (the means and variances of the Gaussian distributions) by making random fixations within the view or background in a number of training examples. We typically used 50-100 training images except for in the last section, where we show we are also able to do single-example learning (Section 4). We use online update rules for the mean and variance, as described in [1, 2].

To classify whether a point in a new image belongs to an object view, the posterior probability of the point belonging to the object view is compared to the posterior probability of the point belonging to the background. This gives a likelihood ratio, and we set a threshold on this ratio in order to do the classification. This threshold minimizes the mean error rate at the threshold equilibrium point where false positives and false negatives are equal. Again, the exact details about Bayesian inference in the BIAS model are not discussed here, and we refer the reader to previous work [1, 2].

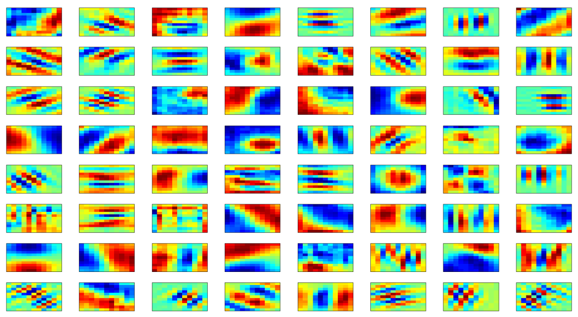


Figure 1: imagenet-vgg-f Layer 1 filters

3. Incorporating Deep Learning Features

3.1. Previous approaches.

Traditionally, a small set of Gabor features (4 orientations, 4 scales) has been used in a supervised manner to learn the different object categories (e.g. face, motorbike, etc.) This method required labeled bounding boxes around the different views or parts of an object. Here, we first relax these constraints by using a form of weakly-supervised learning, i.e. starting only with a bounding box around the entire object, we learn in an unsupervised manner the different views or parts of an object by sampling uniformly from points within the object. We explain below the features and methods used to implement this new type of learning.

3.2. Deep learning features.

Deep learning allows for end-to-end (usually supervised) learning of a set of complex representations of the input data useful for a pre-defined task (e.g. object categorization). Many pre-trained models are freely available, and these are often borrowed and re-configured for use in similar tasks. Here, we use the first layer of convolutional filters from the *imagenet-vgg-f* model [3]. Figure 1 shows example filters from this first layer.

3.3. General methods.

Our model is implemented in MATLAB. We use the MatConvNet toolbox [4] for interfacing the BIAS model with deep learning features. In particular, we used the *vl_nnconv* function for fast and efficient convolution.

Training images for each object category (face, motorbike, etc.) were taken from the Caltech-101 dataset [5]. Unless otherwise noted, we typically used a training set size of 100 images.

3.4. RF sizes correlate with filter sizes.

In our model, we used a total of five concentric rings, with each ring having a total of eight RFs (except for the first ring, which only had one RF). As a result, there were a total of 33 RFs in our model. RFs within subsequent rings increased in size by a scaling factor of 1.4 and were arranged with a hexagonal packing structure in order to maintain an overlap of 0.5. The smallest RF size was 5x5 pixels. In order to maintain scale invariance of the filters, we re-scaled the deep learning features (originally 11x11 pixels) in accordance with the scaling factor between rings of RFs. For the larger RFs, we did not include all scales of features, as the smaller-scaled features are inherently noisy when pooled over larger RFs. Instead, we discarded more of the smaller-scaled features as the RF sizes increased.

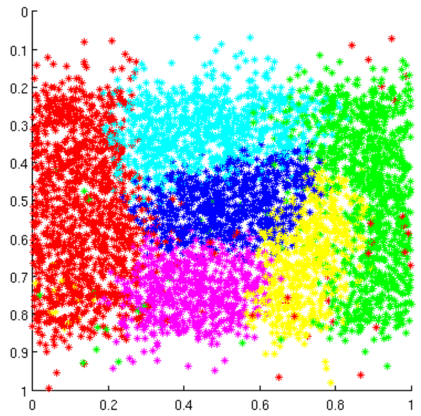
3.5. Image pre-processing.

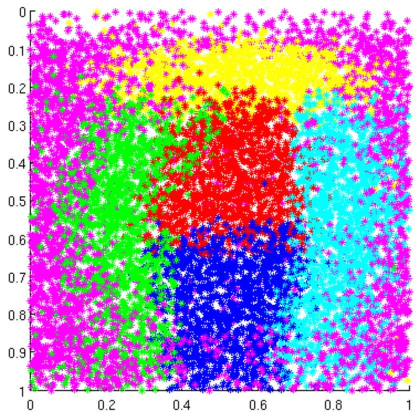
For the main results that we show, each input image was first re-scaled to 224x224 pixels, which was the input image size used for training the *imagenet-vgg-f* model. We also subtract out the "average" image, which is the average color image of all input images the network was trained on. Note that we re-scaled the labeled bounding boxes from the Caltech-101 dataset in order to account for this new (typically smaller) image size. For these results, the input to the model was a color image, while for previous BIAS results, the input to the model was grayscale images. We also test the effect of not re-scaling the deep learning features to match the RF sizes and using deep learning features with grayscale images as input.

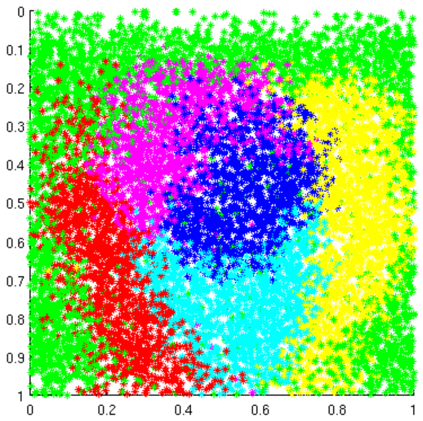
3.6. Convolution, normalization, and max pooling.

We convolve the layer 1 filters (Figure 1) with the input image. Following this convolution, we use linear rectification (inputs below zero are set to zero, all other inputs remain the same). Following this rectification, we normalize the full set of filter responses (across features and scales) using a local response normalization operator, similar to the L_2 norm. This operation is performed independently at each spatial location and across features as follows:

$$y_{ijk} = x_{ijk}\Big(\kappa + \alpha \sum_{t \epsilon G(k)} x_{ijt}^2\Big)^{-\beta}$$


(a) Gabor filters


(b) Deep learning features


(c) Without re-scaling features

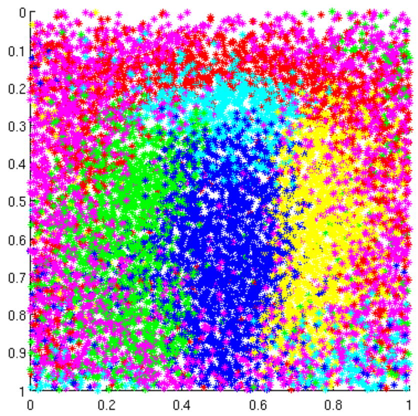

(d) Using grayscale images

Figure 2: Unsupervised learning of object parts using the EM algorithm

where $\kappa = 1 \times 10^{-6}$, $\alpha = 1$, $\beta = 0.5$, and t includes the set of all features and scales.

We then apply max-pooling within each RF (the size of the window used is based on the RF size), and concatenate the features from across different RFs to form a long feature vector, which is used for the subsequent EM learning, which is described below.

3.7. EM learning of clusters.

We first do dimensionality reduction of the features using PCA and we choose the number of PCA components based on a variance cutoff of 80%.

The MATLAB code used for the EM algorithm is based off of the “EmGm” package freely available on the MathWorks File Exchange. The EM algorithm converges if there is either a maximum of 500 iterations or a change in the log-likelihood from one iteration to the next that is less than a predefined tolerance (1e-6). Ten initializations of the EM algorithm were performed (each starting with random assignments), and the model with the highest log-likehood was chosen as the best model and used for futher analysis. We did EM clustering with a total of six clusters, where each cluster is assumed to represent a different object part after learning.

3.8. BIAS results with Gabor features.

We first show previous results from the standard BIAS model learned on the *face* category using standard Gabor features (4 orientations, 4 scales). Fig 2a shows learned cluster labels for the different face parts within a normalized bounding box. For example, the red labels might correspond to the left side of the face and background, the green labels might correspond to the right side of the face and background, and the cyan labels might correspond to the top of the face or forehead. For these results, 50 training images were used and the smallest RF size was 30x30 pixels. We should also note that these results were learned on grayscale images.

3.9. BIAS results with deep learning features.

We now show results obtained using the deep learning features. Fig 2b again shows learned cluster labels for the different face parts within a normalized bounding box. Note that the color labels are arbitrary, so they do not necessarily match up with those shown in Fig 2a. For these results, 100 training images were used and the smallest RF size was 5x5 pixels. With these new results, the symmetry of the face, as well as the consistent labeling of the background as a single cluster becomes much more apparent. However, we have not done any quantitative analysis of these results or compared them with previous results.

3.10. Effect of filter rescaling and color.

We also examined the various effects of our image pre-processing routine on the final results. First, we looked at whether re-scaling the filters with RF size in our model had an influence on the final results. In Fig 2c, we show the results of our model when we kept the original size of filters throughout each RF layer. The model is still able to identify different clusters within the face,

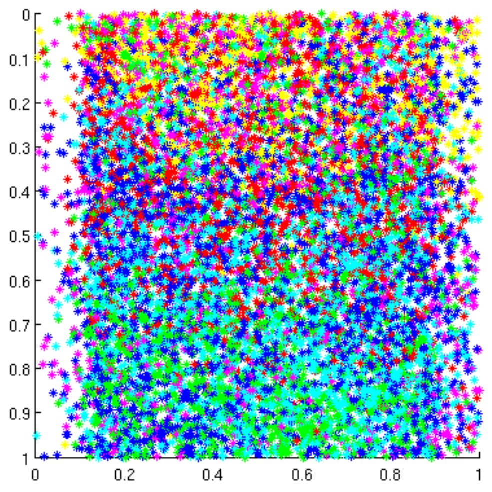
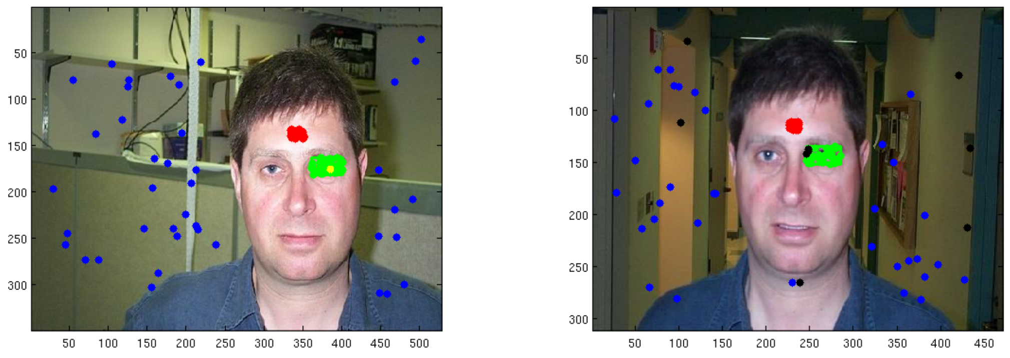


Figure 3: Poor results on the airplane class

although the symmetry is not as stark as in our original implementation. Next, we examined whether we could still perform object category learning on grayscale images. The original layer 1 deep learning filters operate on the three separate color channels of the input image. We chose one set of the layer 1 filters corresponding to the red channel (Fig 1) and applied them to grayscale images. In Fig 2d, we show the results of our model when we used grayscale instead of color images. We could not test the effect of image re-sizing, i.e. using the original image size instead of re-sizing to the standard 224x224 pixels, as storing filtered outputs at the original image size was prohibitively expensive in terms of memory (RAM).

3.11. Poor results on other object classes.

We also found that while we obtained relatively good results on the *face* class, we obtained much poorer results on other object classes, e.g. *airplane*. Fig 3 shows results of training our system on the airplane class. The final cluster labels are relatively random with no perceivable structure. We believe that this may be due to the relatively large amount of background that is included in the labeled bounding boxes of these object classes compared to the face, which has relatively tight bounding boxes. Because we sample uniformly from within the bounding box for our EM learning, many of the background points may be used when they contain no consistent structure.



(a) Training image (b) Test image

Figure 4: Single-example learning on an example object view, the right eye of a face. The model was trained on a single fixation point in the training image (yellow dot), and then tested on other points in the same object region (green points), same object and different region (red points), and the background (blue points). Correctly classified points are shown by the corresponding color, incorrectly classified points are shown in black.

This problem could be addressed by either: 1) looking at other object classes within the Caltech-101 dataset that have tighter bounding boxes, or 2) using a dataset that has much tighter segmentations (e.g. piecewise contours) instead of a rectangular bounding box. One example of such a dataset is the LabelMe database, although we would also have to take into account size invariance of the various instances of objects in these other datasets.

4. Single-Example Learning

We also demonstrate the ability of the BIAS model to learn an object part from a single fixation point using a single training image. Using the set of deep learning features and the RF structure defined above, we trained on a single fixation point within an object region and tested on other points within that object region, other points within the object (but a different object region), and also the background. We tested on a total of 40 randomly selected points within each region. We did this both for the same training image as well as for a new image to test the generalizability of our results. Our results for two example faces are shown in Figure 4.

Based on these preliminary results, we achieved a classification accuracy of greater than 90%. Our results show that our model is not biased or overtrained, despite learning from a single example, as we achieve relatively good performance even on new images. As can be seen in our results, there are still several points either on the object or on the background (shown as black dots in Figure 4) that are incorrectly classified as being part of the trained object region. We are currently investigating if there is a systematic reason why our system makes these errors and in doing so, hope to gain better insight into how our model is able to learn from a single example. We also believe that this insight will allow us to improve our model, and extend our results to other object classes and use cases. As for future directions of research, we are also interested in extending our model to be scale- and rotation-invariant, as well as creating an efficient search algorithm that allows for object recognition using multiple fixations on different object views.

5. References

[1] Predrag Neskovic, Liang Wu, and Leon N Cooper. Learning by integrating information within and across fixations. In *Artificial Neural Networks–ICANN 2006*, pages 488–497. Springer, 2006.

[2] Predrag Neskovic, Ian Sherman, Liang Wu, and Leon N Cooper. Learning faces with the bias model: On the importance of the sizes and locations of fixation regions. *Neurocomputing*, 72:2915–2922, 2009.

[3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[4] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. 2015.

[5] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106:59–70, 2007.