Joint Feature Selection for Object Detection and Recognition

Jerod J. Weinman, Allen Hanson, and Erik Learned-Miller
{weinman,hanson,elm}@cs.umass.edu
Department of Computer Science
University of Massachusetts-Amherst
Amherst, MA 01002

Technical Report UM-CS-2006-54
October 19, 2006

Abstract

Object detection and recognition systems, such as face detectors and face recognizers, are often trained separately and operated in a feed-forward fashion. Selecting a small number of features for these tasks is important to prevent over-fitting and reduce computation. However, when a system has such related or sequential tasks, selecting features for these tasks independently may not be optimal. We propose a framework for choosing features to be shared between object detection and recognition tasks. The result is a system that achieves better performance by joint training and is faster because some features for identification have already been computed for detection. We demonstrate with experiments in text detection and character recognition for images of scenes.

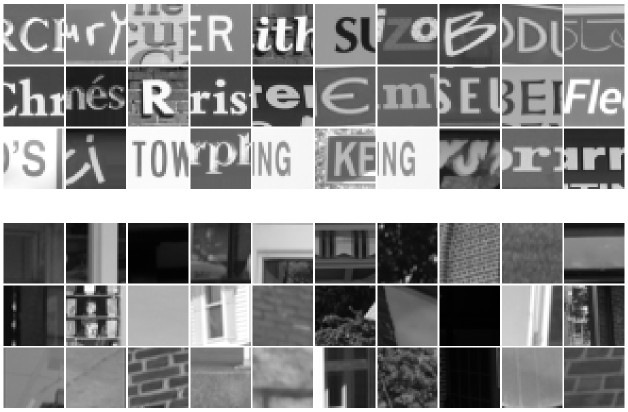

Figure 1. The detection task must only discriminate characters (top) from background patches (bottom), while the recognition task must identify the centered character.

1. Introduction

Many real-world problems must solve multiple classification tasks simultaneously or have tasks that are organized hierarchically or sequentially. For example, a vision system may need to discriminate between cars, people, text, and background as generic classes, while also recognizing particular cars, people, and letters. We shall define the detection task as determining whether an image region corresponds to an object from a class of interest (e.g., characters) or not. The recognition task is defined as discriminating among members of that class (e.g., if this is a character, is it a `p` or a `q`?). Often the detection and recognition tasks are treated in a hierarchical or sequential manner by first running a detector and then feeding detections into an appropriate recognizer. This work seeks to knit these processes more tightly by considering them jointly.

Constructing a classifier for a task involves many issues, including ascertaining the quality and necessary quantity of any training data and deciding which features or observations are relevant to the decision making process. Two reasons for limiting the number of features involved in classification include preventing over-fitting and reducing the amount of computation needed to reach a decision. Models with too many features irrelevant to a classification task are prone to poor generalization performance since they are fit to unnecessary constraints. Even when a problem with over-fitting is not manifest, if certain features are redundant or unnecessary for reaching a decision, the classification process can be expedited by eliminating the need to compute them.

Feature selection may be important for both detection and recognition, the primary difference being the generality of the classification tasks. However, if these problems are treated in isolation, we may not achieve a feature selection that is optimal—in computational or accuracy terms—for the *joint* detection-recognition problem.

We propose a framework for jointly considering the more generic object class detection and more specific object recognition tasks when selecting features. While some features will undoubtedly be useful primarily for detecting object classes and others will have the greatest utility for recognizing objects in a particular class, there may be some features with utility for both tasks. When this is the case, a method accounting for overlap in utility may have two advantages. First, a feature useful for object recognition may boost detection rates for the class by incorporating more object-specific information in the search. Second, if such dual-use features have already been computed for the purposes of detection, they may subsequently be utilized for recognition, effectively reducing the amount of computation necessary to make a classification.

2. Related Work

Several general frameworks exist for selecting features. The two most basic are greedy *forward* and *backward* schemes. Forward schemes incrementally add features to a model based on some criterion of feature utility. Examples of this include work by Viola and Jones [16], who use single-feature decision stumps as weak learners in a boosting framework and add features with the lowest weighted error to the ensemble. A similar forward method by Berger et al. [2] involves adding only those candidates that most increase the likelihood of a probabilistic model. Backward schemes, by contrast, selectively prune features from a model. The Laplacian (ℓ_1) prior for neural networks, maximum entropy models, logistic regression, etc. [17] belongs to this category. In this scheme, features are effectively eliminated from a model during training by fixing their corresponding weights to zero. Many other variants for selecting a subset of features are possible; see Blum and Langley [3] for a more thorough review.

Feature selection for object detection and recognition schemes generally involve one of a few variants. The Viola-Jones object detector [16] employs outputs of simple image difference features, which are similar to wavelets. There are many possible filters, only some of which are discriminative, so a selection process is required primarily for computational efficiency. Other methods use image fragments or patches as feature descriptors. These patches may be taken directly from the image [15, 1], or an intermediate wavelet-based representation [11]. These high-dimensional features can be densely sampled and vector quantized to create a discrete codebook representation. Winn et al. [18] iteratively merge code words that do not contribute to discrimination. Alternately, LeCun et al. [7] learn (rather than select for) a discriminative intermediate feature representation. These models are related to the Fukushima's Neocognitron [6], a model with hierarchical processing for invariant recognition based on successive stages of local template matching and spatial pooling.

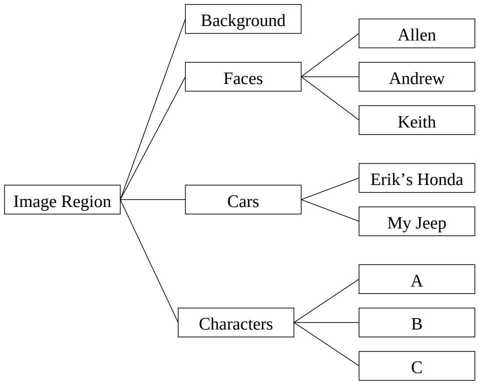

Figure 2. An example object class hierarchy for images. Object class detection is finding instances in the second column, while recognition is identifying instances in the third column.

Torralba et al. [14] have shown that jointly selecting features for detecting several object classes generalizes better and reduces the requisite the number of features. Our work synthesizes many of these ideas, adding the object recognition task to the competition for feature resources.

3. Detection and Recognition Model

Our underlying classification and feature selection scheme is probabilistic. Given an observation vector the goal is to determine whether it belongs to some general class of interest, and if so, to recognize it as a particular known object. Let $\mathbf{x}$ represent the input vector and $y \in \mathcal{Y}$ the associated label. In the simplest case, there is one class of interest to detect (e.g., characters), so the label space is partitioned into labels from that object class and "background," $\mathcal{Y} = \mathcal{Y}_{\mathsf{c}} \cup \{\mathsf{b}\}$. This generalizes easily to multiple classes (See Figure 2).

We use a discriminative maximum entropy model [2] for classification:

$$p\left(y \mid \mathbf{x}; \boldsymbol{\theta}, F\right) \equiv \frac{1}{Z} \exp\left(\boldsymbol{\theta}\left(y\right) \cdot F\left(\mathbf{x}\right)\right), \; y \in \mathcal{Y}, \qquad (1)$$

where F is a vector of features calculated on the input observation $\mathbf{x}$, parameters $\boldsymbol{\theta}$ are class-specific weights on these features, Z is a normalizing constant ensuring the expression is a proper probability. Given a labeled set of independent examples $\mathcal{D} = \left\{\left(y^{(i)}, \mathbf{x}^{(i)}\right)\right\}_i$, the parameters of the model may be optimized by a *maximum a posteriori* (MAP) estimate. The corresponding objective function

$$\mathcal{L}\left(\boldsymbol{\theta}; F, \mathcal{D}\right) \equiv \log p\left(\boldsymbol{\theta} \mid \alpha\right) + \sum_i \boldsymbol{\theta}\left(y^{(i)}\right) \cdot F\left(\mathbf{x}^{(i)}\right) - \log Z \qquad (2)$$

is convex when the prior on the model parameters $p\left(\boldsymbol{\theta} \mid \alpha\right)$

is convex. Thus a global maximum $\widehat{\boldsymbol{\theta}}$ can be found via convex optimization.

Two separate classification mechanisms are often used for detection and recognition problems. Formally, this means first optimizing a detection model with parameters $\boldsymbol{\theta}_{\mathsf{D}}$ where the label $w \in \{\mathsf{c}, \mathsf{b}\}$ is either the generic character class c or background b. Then, given a detection of some class, ($w = \mathsf{c} \Rightarrow y \in \mathcal{Y}_{\mathsf{c}}$) a query is made to a recognition model for that class with parameters $\boldsymbol{\theta}_{\mathsf{R}}$ to assign a character label $y \in \mathcal{Y}_{\mathsf{c}}$. Model (1) discriminates among all character labels and backgrounds simultaneously. Alternatively, the probability may be factorized by introducing the detection variable w:

$$p\left(w, y \mid \mathbf{x}; \boldsymbol{\theta}, F\right) \equiv p\left(w \mid \mathbf{x}; \boldsymbol{\theta}_{\mathsf{D}}, F_{\mathsf{D}}\right) p\left(y \mid w, \mathbf{x}; \boldsymbol{\theta}_{\mathsf{R}}, F_{\mathsf{R}}\right), \tag{3}$$

where $w \in \{\mathsf{c}, \mathsf{b}\}$, $y \in \mathcal{Y}$ and the joint model parameters $\boldsymbol{\theta}$ and features F are the result of concatenating the detection and recognition models' parameters and features. The first term is the probability for detection, while the second term, conditioned on the detection result, is the probability for recognition. The value of y determines w. Thus, logical implication dictates the probability for $y = \mathsf{b}$ is unity when $w = \mathsf{b}$, and zero for all other y. Conversely, the probability for $y = \mathsf{b}$ is zero and when $w = \mathsf{c}$.

Detection and recognition models may be trained in one of three fashions. An integrated model like (1) may be trained. The model may be factorized like (3) and trained jointly. In this case, the objective function would have the form

$$\mathcal{L}_{\mathsf{J}}\left(\boldsymbol{\theta}; F, \mathcal{D}\right) = \mathcal{L}\left(\boldsymbol{\theta}_{\mathsf{D}}; F_{\mathsf{D}}, \mathcal{D}\right) + \mathcal{L}\left(\boldsymbol{\theta}_{\mathsf{R}}; F_{\mathsf{R}}, \mathcal{D}\right) \tag{4}$$

Finally, the detection and recognition components of the factorized model may be trained independently. The wealth of literature focusing strictly on detection or recognition schemes indicates this is the most common approach.

One disadvantage of the factorized independent scheme is that there may be intra-class variations that are hard to capture with a generic detector for some object classes; allowing explicit consideration of individual known objects at the detection stage could ameliorate the issue. Additionally, if detector training is independent of the identification problem, and vice-versa, the features used to make a detection decision may not overlap with the features used for identification, possibly increasing the total amount of computation. In the next section, we elaborate on our proposed method for joint training and feature selection.

4. Feature Selection

Our algorithm for selecting features follows that of Berger et al. [2]. It is a greedy forward method that incrementally adds the feature providing the greatest increase in the objective function (2). A set of candidate features $G\left(\mathbf{x}\right)$ are proposed with corresponding parameters $\boldsymbol{\alpha}$. Concatenating these features and parameters with the original model's F and $\boldsymbol{\theta}$, yields the model

$$p\left(y \mid \mathbf{x}; \boldsymbol{\theta}', F'\right) = \frac{1}{Z} \exp\left(\boldsymbol{\theta}\left(y\right) \cdot F\left(\mathbf{x}\right) + \boldsymbol{\alpha}\left(y\right) \cdot G\left(\mathbf{x}\right)\right), \tag{5}$$

where $F' = \left[\begin{array}{cc} F & G \end{array}\right]$ and $\boldsymbol{\theta}' = \left[\begin{array}{cc} \boldsymbol{\theta} & \boldsymbol{\alpha} \end{array}\right]$.

The the gain of the features is assessed with the optimal parameter values for the model with new features

$$\mathcal{G}\left(G; \mathcal{D}\right) = \mathcal{L}\left(\widehat{\boldsymbol{\theta}'}; F', \mathcal{D}\right) - \mathcal{L}\left(\widehat{\boldsymbol{\theta}}; F, \mathcal{D}\right). \tag{6}$$

This is equivalent to a likelihood-ratio test of the two models $\left(F, \widehat{\boldsymbol{\theta}}\right)$ and $\left(F', \widehat{\boldsymbol{\theta}'}\right)$. Thus, a model is built by iteratively adding the highest-gain feature until the increase in log-likelihood is negligible or some maximum number of features is reached.

Since many candidate features may need to be examined at every iteration, approximations are helpful for speeding the process. First, only the parameters $\boldsymbol{\alpha}$ for the candidate features G are optimized [2], leaving $\widehat{\boldsymbol{\theta}}$ fixed in the gain calculation (6). This can greatly reduce the search space. Second, we calculate the gains on a representative subset of the training data, and then calculate the gains of only the top features on the full data.

When two separate classifiers are trained in a pipelined framework, the gain of a feature is only measured with respect to a particular task, detection or recognition. However, the entire end-to-end task of detection *and* recognition yields a different ranking of the features.

5. Text Detection and Recognition

Our goal is to detect and recognize text in unconstrained images. Here we describe the image features we use and the data used to evaluate the method.

5.1. Features

Several authors have demonstrated that edges and textural features are useful for text detection [5, 4, 19, 13]. Most of these systems do a preliminary layout analysis, and then pass the detection regions to commercial recognition systems. However, Chen et al. [4] and Thillou et al. [13] describe character isolation strategies and employ their own character classifiers (a Fisher linear discriminant and neural network, respectively).

Our features are mainly derived from the steerable pyramid wavelet basis [12], which roughly models the "simple cells" in an initial layer of processing in mammalian visual systems. The wavelet coefficients are complex, representing outputs from even and odd paired filters. Taking complex magnitudes yields phase invariant responses, similar to complex cells in biological systems.

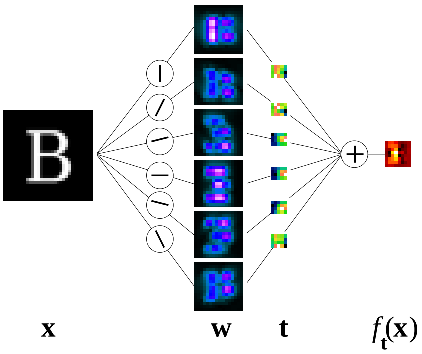

Figure 3. Patch feature map computation. There is a normalization and downsampling between $\mathbf{w}$ and $\mathbf{t}$, and another dilation between $\mathbf{t}$ and f.

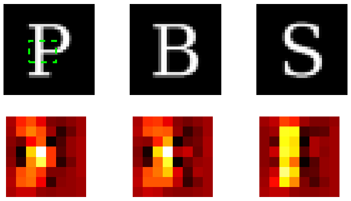

Figure 4. Sample patch template and feature map outputs.

One pool of features intended primarily to facilitate text detection are a set of image and wavelet statistics originally crafted for texture synthesis [9]. These include image statistics (four central moments plus min and max) at several scales, means of wavelet channel magnitudes, and local auto- and cross-correlation of wavelet channels. Although originally intended to be computed globally over an image of ergodic texture, we compute them locally over a small image region, which may be efficiently achieved by convolution.

Chen et al. [4] train a subsequent character classifier directly on the local wavelet features. However, such a model may not be robust to image deformations. Indeed, research in cognitive psychology by Rehling [10] indicates that two mechanisms operate in human character recognition: an initial “flat” recognizer like Chen’s that is fast, and a secondary hierarchical parts-based model like LeCun’s convolutional network [7] that is slower but more accurate. Following this hierarchical framework, we add template features to the candidate pool.

First, the wavelet magnitudes are locally normalized by a process similar to that of SIFT descriptors [8]. At each location, all the wavelet magnitudes in a local window are normalized to a unit ℓ_2 norm, clipped at a threshold (0.2 in our experiments), and re-normalized, keeping the normalized values of the center location. To decrease spatial and phase sensitivity, the image’s wavelet magnitudes are downsized by taking the maximum over a small window within each channel (a simple morphological dilation). Template features $\mathbf{t}$ are small patches extracted from these subsampled wavelet magnitudes and subsequently normalized to have zero mean and unit ℓ_1 norm.

Feature maps are computed from these templates by convolving the image’s normalized, downsampled wavelet band magnitudes $\mathbf{w}$ with the corresponding channels from the template $\mathbf{t}$ and summing the output over all channels c. Let $\mathbf{t}^c$ represent the normalized wavelet coefficient magnitudes of some channel (e.g., scale and orientation) c for a template, then the corresponding feature map calculation for an image $\mathbf{x}$ having wavelet coefficient magnitudes $\mathbf{w}$ (normalized and downsampled) is

$$f_{\mathbf{t}}\left(\mathbf{x}\right) = \sum_c \mathbf{t}^c * \mathbf{w}^c \qquad (7)$$

where $*$ is the convolution operator. The feature map $f_{\mathbf{t}}$ is then subject to another downsampling operation for even further spatial pooling and dimensionality reduction. An illustration of the image to feature map calculation is given in Figure 3. Resulting template feature map outputs (Figure 4) may be transformed to a vector and added to the classification model (1) as entries in $F\left(\mathbf{x}\right)$. The goal will then be to select the templates most useful for a particular task, be it detection, identification, or both.

6. Experiments

In this section, we compare three training and feature selection strategies for detection and recognition: the joint, integrated all-way classifier, a factored but jointly trained classifier, and independently trained classifiers.

6.1. Data

To test our hypothesis that joint feature selection can improve speed and accuracy, we need data with labels for background and characters. A set of 300 images taken from scenes around a downtown area have had text regions masked, and square patches of various scales from the non-text regions are extracted and labeled as background. Examples are in the bottom of Figure 1.

Rather than manually crop and label individual characters from actual image regions, we generate similar synthetic character images. There are 62 characters (26 upper case, 26 lower case, 10 digits) in our alphabet to be recognized $\mathcal{Y}_{\mathrm{c}}$. The characters were rendered in 954 fonts at a pixel height of 25 (roughly 12.5px x-height) and centered in a 32x32 pixel window. Neighboring characters were sampled from bigrams learned on a corpus of English text and placed with uniform random kerning. The trigram image was then subject to a random distortion involving contrast, brightness, polarity, scale, shear, and rotation, followed by zero-mean Gaussian noise. The degree of noise and distortions are modelled after the text from our scene images.

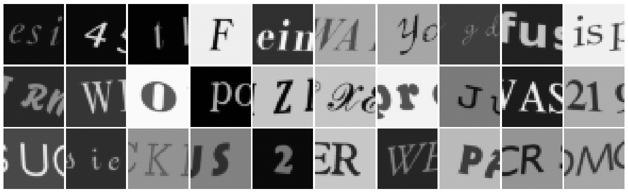

Figure 5. Synthetic character images used for experiments.

Adding these factors to the data set allows the classifier to learn them and provides a reasonable test bed without having to manually ground truth individual characters in many images. The label of these character windows is the center character. Examples characters are shown in Figure 5, and may be compared to characters from actual images of scene text in Figure 1. Note that the recognition task involves no character segmentation—the character in the center of the window must be recognized in the presence of neighboring character "clutter."

For the non-sign background class, our training set consists of roughly 65,000 windows at multiple scales from 77 images of outdoor scenes. The foreground character class consists of nearly 30,000 character windows (each of 62 characters in 467 fonts). Since text is actually rarer in natural scenes, we weight all the data instances in training and test evaluation such that characters have a class prior of $1e-4$; in other words, the ratio of text to background is almost one to ten-thousand. The test set is roughly the same size but comes from a different set of scene images and fonts. (Indeed, if we use the same fonts for testing even with different distortions applied, the recognition results are much higher.)

As shown in Figure 3, the wavelet transform of a given 32×32 patch is downsized to 16×16 and the resulting feature map is downsized to 4×4 for a very compact representation of responses for each feature.

In all cases, a Laplacian prior $p(\boldsymbol{\theta} \mid \alpha) \propto \exp(-\alpha \|\boldsymbol{\theta}\|_1)$ was used, and the value of the hyperparameter α was chosen by cross-validation. The training set was split in two, half was used for training, and the value of α that yielded the highest likelihood on the the other half was then used on the entire training set. All of the features were included for cross-validation, since we do not a priori know which might be useful. However, a slightly smaller portion of the training data was used since all features for all instances exceeded memory limits. Since less training data is available, this likely results in a stronger prior than necessary. Conversely, only a few features are actually used in the initial training stages, so a strong prior may be of some value.

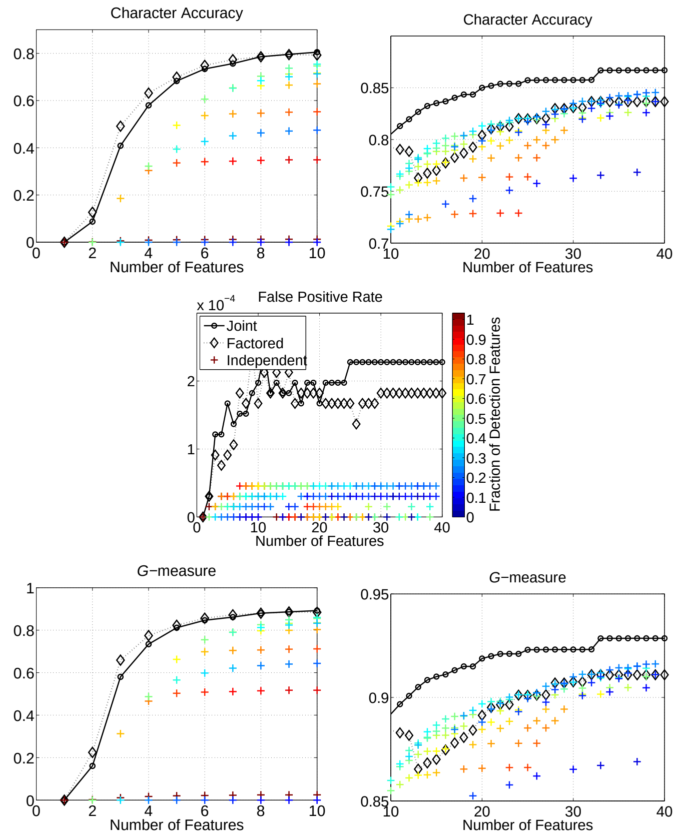

Figure 6. Comparison of feature selection strategies. See text for details. (Note: This figure is best viewed in color.)

6.2. Results

In this section, we present the results of our experiments. Section 6.3 contains an analysis and discussion of these results in greater detail.

Figure 6 shows the comparative results of the feature selection strategies. Character accuracy is the accuracy on all test character instances—to positively contribute, an instance must not only be detected as text but also be correctly identified. Character detection curves are qualitatively similar between the three methods, but are much higher since an instance must only be classified as *some* character to be correct. The false positive rate is given for the relative weighting of the instances described above, though the actual numbers are quite small: each "row" of pluses constitutes one absolute false positive, so that the most for the independent method is three, while the joint method yields sixteen at its peak.

The independent method has lower character accuracies, but also a lower false positive rate. Therefore, we need a single measure that accounts for the detection rate/false positive trade-off and incorporates accuracy. The information retrieval community encounters the same issue with the precision and recall metrics, unifying them in the so-called

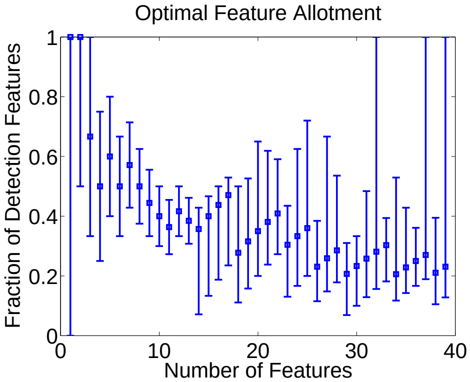

Figure 8. Optimal allotment of detection and recognition features versus total number of features with independent subtask selection. For total features greater than 12, the space of possible allotments is sub-sampled; bars show the fractions tested that are nearest to the optimal.

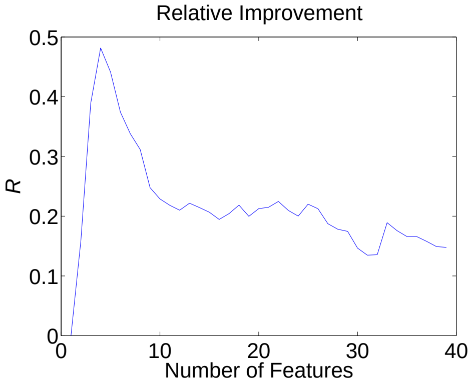

Figure 9. Relative improvement (see Eq. 8) of joint feature selection over independent feature selection given optimal feature allotment.

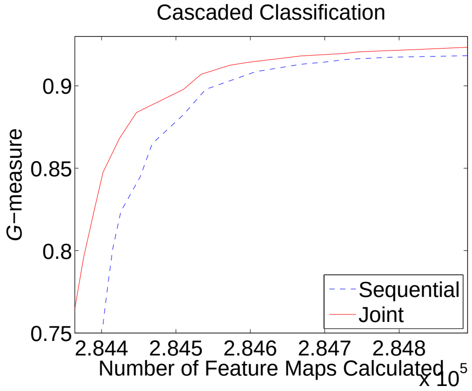

Figure 10. Comparison of feature selection strategies for cascaded classification.

"F-measure," which is their harmonic mean. Unlike the arithmetic mean, the harmonic mean is much more sensitive to differences between the values being averaged and will hew closer to the "outliers." This can more accurately reflect poor performance along a particular dimension by preventing it from being averaged out. To unify the character accuracy and false positive rate in our system, we propose to use the harmonic mean of character accuracy and the true negative rate (or, one minus the false positive rate). In an ROC curve, this is a type of outlier-sensitive distance measure from a particular point on the curve (given by the classifier), to the upper-left corner of the unit cube, which an optimal ROC curve would pass through.

Figure 7 compares the gains of some features under the different selection strategies during a round of feature selection. In each graph, five features for a particular task are uniformly sampled from best to worst, and the gains of these features under all the tasks is shown. To normalize the gains for comparison, each is divided by the maximum gain in the round for its particular selection strategy.

Given a fixed total number of features, we calculate the number of features alloted to detection and recognition tasks (according to their independent feature selections) that results in an optimal G-measure, as shown in the top of Figure 8. Figure 9 shows the relative improvement

$$R = 1 - \frac{1 - G_{\mathsf{J}}}{1 - G_{\mathsf{I}}} \tag{8}$$

(or reduction in "error") between the G-measure of the joint selection G_{J} and the optimal independent selection G_{I}.

We may also take a cascaded approach to detection and recognition. The independently trained classifiers operate in a sequential fashion—first detection, then recognition—while the joint model evaluates the entire hypothesis space. We iteratively compute features until the posterior entropy of the task at hand (detection, recognition, or both in the case of the joint model) decreases to some pre-specified fraction of the initial entropy, at which point the classification is accepted. For instance, with no features the initial entropy is a function of the number of classes in the task:

$$\begin{aligned} H_D^0 &= -\log 2 \\ H_{\mathsf{R}}^0 &= -\log 62 \\ H_{\mathsf{J}}^0 &= -\log 63. \end{aligned} \tag{9}$$

When we add the top k features to the model, the probability distribution changes, yielding a new entropy H^k. We thus add the top features until

$$\frac{H^k}{H^0} < \tau \tag{10}$$

The independent classifiers do this sequentially, first for the detection task using H_{D}, the posterior entropy of w), then, if necessary, fixing w and using H_{R}, the posterior entropy of y. Figure 10 shows the sum of k on the entire test set while varying thresholds τ for both the joint and independent models.

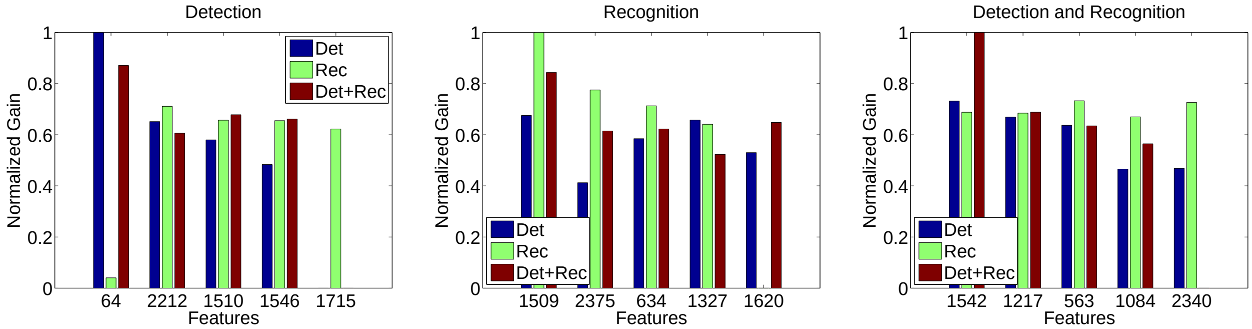

Figure 7. Relative gains of features for different tasks during the fifth round of forward selection. See text for a full explanation.

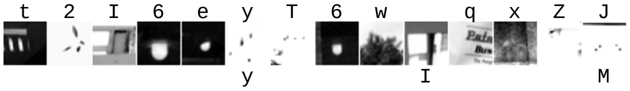

Figure 11. All false positive predictions for the joint model (top) and independent models (bottom).

6.3. Discussion

Our experimental results demonstrate the superiority of joint feature selection over the traditional independent methods in several ways. The first and most obvious way is that the G-measure of joint method dominates the independent method for any number of features. This is shown by the consistent 13-50% error reduction in Figure 9. Even if the relative improvement was more modest, the problem of determining the optimal allotment of features to the detection and recognition task would remain. If the purpose of feature selection is to minimize the amount of necessary computation, one must figure into the calculation how many features to use. The problem with the independent method is given a feature bandwidth a priori, the feature allotments will undoubtedly depend on the task. To determine the number of features that should be used for detection would require an additional level of optimization to find the maximum shown in Figure 8.

One of the interesting properties of the joint method's performance is the great improvement over the independent method when there are fewer features available; the joint feature selection strategy ramps up much more quickly. The reason for this can be seen by examining the gains shown in Figure 7. In this round, the top feature selected for the detection task has almost no value for recognition. With this strategy, by the time a character is detected, the features that have been computed will be of little help in actually identifying the character. By contrast, the top feature for the joint task has modest value for both detection and recognition. Adding this feature to the classifier not only aids in detecting characters, but very early on the system is also able to identify many more characters as well.

The typical approach to detection and recognition is sequential. Under such a strategy, the independent detector selects 20 features before the model likelihood plateaus, while the independent recognizer selects 35 features. For any window detected as text, the detector will have calculated 20 features, and then an additional 35 features will be calculated for recognition. Since the prior probability for text is very small, the total additional computation is modest. However, as the number of object classes grows (as in Figure 2), the requisite number of queries to the class-specific recognizers gets much larger, and the impact of additional feature computation for recognition becomes non-negligible. For multiclass detection and recognition schemes to be feasible, the features learned or selected must consider the task in its entirety. Figure 10 demonstrates that even for a cascaded approach to classification, the joint feature selection strategy requires the computation far fewer feature maps for equivalent performance.

7. Conclusions

The typical approach to image understanding involves training system components individually. Unfortunately, errors propagating through sequential systems can have compounded negative effects. Furthermore, if resources (e.g., features) can be shared among the components, training components independently will make their resources too specialized to be useful for any other task. Therefore, we have proposed to extend the idea of shared feature selection to the the task of object class detection and a more specific object recognition.

We have laid out three frameworks for feature selection—the usual, which selects features independently for the detection and recognition task, and two others that jointly select features for the entire detection and recognition task, one being factorized. Our results show that consideration of the entire end-to-end task yields greater accuracy. In a system with limited computational resources, joint feature selection obviates the need to optimize feature allocation for different tasks.

In more general systems, there will be many detection and recognition tasks. The benefit of multi-purpose discriminative features for these systems will be even larger than demonstrated here. With more complex object classes to detect, knowledge of individual members can help boost detection rates, and having features that are useful for multiple tasks can greatly reduce the necessary amount of computation.

While recent research has focused on developing high accuracy, specialized systems for tasks such as face detection, our results indicate it may be time to consider returning to frameworks that allow joint training of these powerful new models on broader, end-to-end tasks.

Acknowledgments

This work was supported in part by The Central Intelligence Agency, the National Security Agency, and National Science Foundation under NSF grant #IIS-0326249.

References

[1] S. Agarwal, A. Awan, and D. Roth. Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1475–1490, 2004.

[2] A. L. Berger, S. A. D. Pietra, and V. J. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[3] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.

[4] X. Chen, J. Yang, J. Zhang, and A. Waibel. Automatic detection and recognition of signs from natural scenes. *IEEE Transactions on Image Processing*, 13(1):87–99, 2004.

[5] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 366–373, 2004.

[6] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 39:139–202, 1980.

[7] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, Nov 1998.

[8] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[9] J. Portilla and E. P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–71, 2000.

[10] J. A. Rehling. *Letter Spirit (Part Two): Modeling Creativity in a Visual Domain*. PhD thesis, Indiana University, July 2001.

[11] T. Serre, L. Wolf, and T. Poggio. Object recognition with features inspired by visual cortex. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 994–1000, 2005.

[12] E. P. Simoncelli and W. T. Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *IEEE International Conference on Image Processing*, volume 3, pages 444–447, 23-26 Oct. 1995, Washington, DC, USA, 1995.

[13] C. Thillou, S. Ferreira, and B. Gosselin. An embedded application for degraded text recognition. *Eurasip Journal on Applied Signal Processing*, 13:2127–2135, 2005.

[14] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, volume 2, pages 762–769, 264.

[15] S. Ullman, E. Sali, and M. Vidal-Naquet. A fragment-based approach to object representation and classification. In *International Workshop on Visual Form*, number 2059 in LNCS, pages 85–102, 2001.

[16] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004.

[17] P. M. Williams. Bayesian regularization and pruning using a Laplace prior. *Neural Computation*, 7:117–143, 1995.

[18] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *Proc. Intl. Conf. on Computer Vision*, pages 1800–1807, 2005.

[19] V. Wu, R. Manmatha, and E. M. Riseman. Finding text in images. In *DL'97: Proceedings of the 2nd ACM International Conference on Digital Libraries*, pages 3–12, 1997.