Semantic Scene Segmentation using Random Multinomial Logit

Ananth Ranganathan
http://www.ananth.in

Honda Research Institute, USA
Cambridge, MA, USA

Abstract

We introduce Random Multinomial Logit (RML), a general multi-class classifier based on an ensemble of multinomial logistic regression models, and apply it to the task of semantic image segmentation. The algorithm is simple, can be trained efficiently, and has near realtime runtime performance. RML combines the desirable properties of multinomial logistic regression, being stable and theoretically sound, with those of bagging, which is noise-resistant and applicable to large feature spaces. As a second major contribution, we describe a feature selection algorithm for RML based on statistical properties of logistic regression that ensures that computation is not wasted on statistically insignificant features.

We evaluate RML on an extremely challenging real-world video dataset of traffic scenes with large illumination, perspective, and intra-class variation, as well as on the 20-class VOC 2008 dataset. Comparisons with recent techniques on the latter dataset demonstrate RML as being state of the art, and advancing it in many cases.

1 Introduction

We are interested in the task of classifying street scenes for use in intelligent transportation systems. This involves detecting the road, other vehicles, and pedestrians to alert the user in potentially dangerous situations. For instance, the detection of fast moving vehicles, such as motorbikes, would help decrease the number of accidents significantly.

The challenge in this application, in addition to the inherent difficulty of generic street scene recognition, is the need to detect and localize even tiny objects in the image. Hence, image-level features and statistics are insufficient. At the same time, real-time processing is also of crucial importance.

In this paper, we address these challenges by introducing Random Multinomial Logit (RML), a general purpose classifier, and applying it to the task of texture-based scene segmentation. RML consists of an ensemble of multinomial logistic regression models, each of which operates on a randomly selected subset of features and outputs a probability distribution on the label of the pixel corresponding to the features. The use of a bagging framework overcomes the inability of multinomial logistic regression to operate in large feature spaces.

RML exhibits exceedingly good scaling properties with respect to amount of training data and has near-realtime runtime performance. It is also tolerant to significant levels of label noise and is resistant to over-fitting due to the use of a bagging framework [3].

We describe the use of goodness-of-fit statistics for feature selection during the learning phase. Feature selection is performed to improve the quality of each regression model

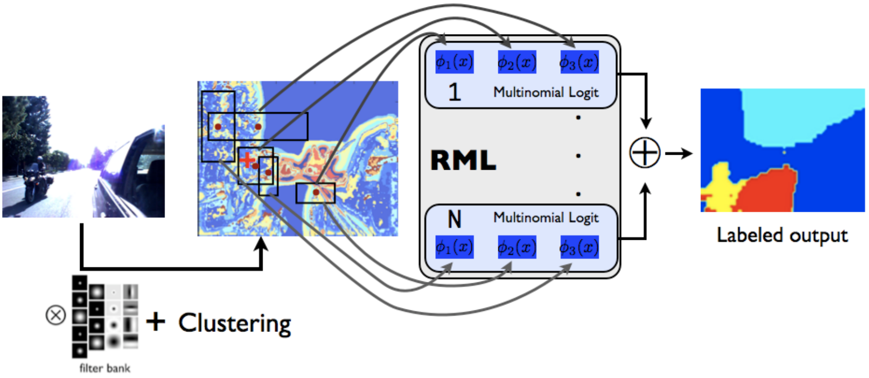


Figure 1: An overview of our scene segmentation system using RML: training images are convolved with a filter bank and clustered to yield texton images, on which texture-layout features operate. Five texture-layout features are shown in the second box from the left. Subsets of such features are randomly selected and passed into an ensemble of multinomial logistic regression models (here having three features each), the outputs from which are averaged to yield the final semantic segmentation.

and reduce the total number of models required. We use texture-layout features [23] in conjunction with RML to perform scene segmentation in this paper. Our motivation for using these features is that they are simple, fast to compute, and capture local context of the image. However, in general, any texture-based feature can be used. An overview of our scene segmentation system is shown in Figure 1.

Random forests [3, 10] have recently attracted attention by demonstrating good results on the tasks of scene segmentation and image categorization. RML and Random forests are similar in the manner in which the individual models, multinomial logit and decision trees respectively, are put together in an ensemble. Thus, while both RML and Random forests share the advantages of being ensemble methods, RML has many exclusive properties arising from its use of multinomial logistic regression -

- RML learns specific regression functions for classification while Random forests find rectangular classification "regions" in input space [3]
- The number of feature computations is constant and can be pre-computed, as opposed to Random forests, where the number of features required depends on the depth of the tree and path taken through the tree.
- Feature selection in Random forests is usually done using data-agnostic methods such as information gain, whereas RML uses goodness-of-fit tests on the data.

2 Related Work

Scene segmentation [1, 25] has been an active area of research, and methods based on discretized representations such as codebooks of features [24] or textons [12, 16, 26] have proven quite powerful. These methods employ "bag of words" models [6, 29], which model the whole image or specific regions using histograms of visual words [4], with or without spatial context [15]. RML is different from these methods in that it uses a logistic regression model over local feature responses rather than a "bag of words" model. Hence, spatial context is not built into the representation but is modularized in the features used. We now focus on research using texture-based classifiers and randomized methods.

Textonboost [23], which also uses texture-layout features, introduced the use of boosting for selecting features that act on textons. Since the number of such features is very large, training is very slow and the number of features required for a constant level of performance increases rapidly with the size of the dataset and variation in object classes therein. Semantic texton forests [22] were introduced to avoid the problems of boosting through the use of Random forests, both for the creation of the texton codebook and for classification. However, in this and other works using Random forest classifiers [2, 9], the decision trees are only grown upto a pre-determined depth, say d, which is in direct violation of the extremely randomized forests algorithm [10] on which these methods are based. While a fixed depth random forest works well for most cases, its performance deteriorates with increasing intra-class variation and label bias in training data (Cf. Section 5.2). Good labeling performance is only achieved when the random forest is grown without pruning. However, this makes the runtime performance slow, since the depth of the tree increases with the amount of training data. In contrast, a Random Multinomial Logit (RML) classifier always has $O(1)$ runtime performance.

Csurka and Perronnin [5] give a novel method based on a Fisher vector representation of patches that is simple but requires one classifier per object category. He *et al.* [11] introduced multiscale conditional random fields (CRFs) for per-pixel segmentation while Kohli *et al.* [13] explore graph cuts on such random fields in a series of papers which yield accurate segmentation but are too slow for our purposes. Random Ferns [20] have been used for semantic segmentation [2] but require the the use of sophisticated discriminative features with built-in invariances, which unfortunately are slow to compute.

Other related object detection methods can only deal with a few labels [1, 27], or confine themselves to a single object category [9]. Moosmann and Triggs [17] use Random forests for clustering to create the "bag of words" model but only address image classification, and not pixel-level segmentation.

Random forests of multinomial logistic regressors were first introduced in [21] but did not include feature selection. Our extensions to the original algorithm and its use in computer vision are novel contributions.

3 Random Multinomial Logit

An RML classifier consists of N multinomial logistic regression models, each of which models the probability distribution of the label y given the input vector x as

$$\pi_{il} = p_i(y = l|\mathbf{x}, \beta_i) = \begin{cases} \exp\left(\beta_{il0} + \sum\limits_{f=1:M} \beta_{ilf}\phi_f(\mathbf{x})\right)/Z, & l = 1 : L-1 \\ 1/Z, & l = L \end{cases} \tag{1}$$

where i and l are indices into the model and label set respectively, and Z is the normalizing constant that makes the distribution sum to unity. The $\phi(.)$ are feature functions computed on the input vector $\mathbf{x}$, and β_{il} is the vector of coefficients of length $(L-1)$ that define the detection function for object category l. Stacking each of these vectors, we obtain the $(L-1) \times (M+1)$ matrix β_i of all the coefficients for the multinomial regression model.

Training for the RML classifier, which involves learning the β coefficients from data, proceeds in a manner similar to Random forests. The training set is sampled with replacement to get N smaller sets using which the individual regression models are learned. The features for the individual models are also selected randomly, with M features per model. The final output label distribution of the RML is computed by averaging over the output of the individual models

$$\hat{\pi}_l = \sum_{i=1:N} \pi_{il} \tag{2}$$

An illustration of the training process is given in Figure 1.

The coefficients β for the individual regression models are learned in a maximum a posteriori (MAP) framework with a L2 regularizing prior on the model parameters β. The function to be maximized is thus (dropping the index for the model number) -

$$L(\beta|\{\mathbf{x},y\}) = \sum_{\{\mathbf{x},y\}} \log \pi_y - \alpha\beta^2 \tag{3}$$

where $\{\mathbf{x},y\}$ is the training data, α is the regularization strength, and π_y is as in (1).

Optimization of (3) is done using standard algorithms such as gradient descent or second order methods. The gradient of (3) for a specific coefficient is given as

$$\frac{\partial L}{\partial \beta_{lf}} = \sum_{\{\mathbf{x},y\}} \phi_f(\mathbf{x})\left(I(l=y) - \pi_{il}\right) - 2\alpha\beta_{lf} \tag{4}$$

where I(.) is the indicator function which yields unity if its argument is true. We compared stochastic gradient descent, Newton's method, and L-BFGS for this purpose and found L-BFGS to give the best trade-off between training time and performance. Stochastic gradient descent is slower to converge for large training sets but can be considered for online applications where updates for β based on individual training points is necessary.

3.1 RML for Scene Analysis

Our algorithm for scene analysis is based on textons [16]. Textons are discretized texture words, which are learnt by clustering the output of a filter bank applied to the training set. Clustering is performed using hierarchical K-means clustering. We considered supervised clustering using Random forests [17] but this produces large codebooks and the exact number of clusters cannot be controlled.

Texture-layout features [23], which we use in this paper, consist of a rectangle r and a texton word t. For every pixel p, the feature computes the proportion of the texton word t inside the rectangle r, where r has been translated to be in the coordinate system with p as the origin. It can be seen that texture-layout features capture local textural context in the image, for instance the relationship that a boat is usually surrounded by water. In addition, this contextual relationship, expressed as a linear combination of multiple texture-layout feature values, is sufficient to do pixel-wise scene labeling. TextonBoost [23] does exactly this but does not scale well with the size of the training set and number of labels since the total pool of features from which the selection has to be made also increases correspondingly.

To learn the RML based on texture-layout features, we first pre-select N_r rectangular regions randomly, so that the total number of possible features is $N_r \times N_t$, where N_t is the number of texton words in the codebook. Subsequently, for each multinomial regression model in the RML, a set of $M \ll N_r \times N_t$ features are selected randomly to create the distributions given in (1). We limit the number of features M to be small, typically 10-20, so reasons of computational efficiency. In practice, this number suffices to produce good results.

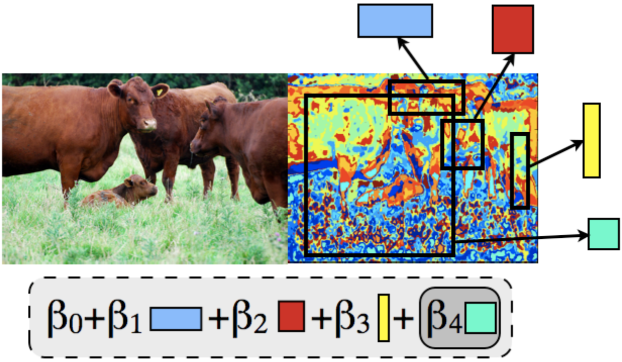


Figure 2: Feature selection: Texture-layout features consist of a rectangular shape and a texton word (denoted here by color). Features are statistically insignificant for labeling when their values are not indicative of the pixel label. This may be because the shape of texture-layout feature is too large or too small. Here, for instance, the 4th feature in the multinomial regression model is statistcally insignificant and should be discarded through feature selection.

The RML is learned in a supervised manner using pixel-wise labeled data. The feature values evaluated at a pixel along with its label constitute a training instance. Randomly selected subsets of this training data are used to learn the individual regression models. During runtime, the features in the regression models are evaluated on the test image and passed through the regression models to get the output labeling as in (2).

4 Feature Selection

The vanilla RML algorithm described above can be improved upon through feature selection. An impetus for doing this is provided by the fact that many of the features picked randomly will not have a bearing on the labeling decision. For example, the shape of a texture-layout filter may be too big or too small to provide any useful information as shown in Figure 2. Hence, evaluating the feature response in these cases will merely result in wasted computation.

Feature selection is performed by swapping a feature in the model with a randomly selected new feature. If statistically insignificant features exist in the current model, these are preferentially swapped out, otherwise the selection is made randomly and the better feature, as pointed out by the likelihood ratio test, is retained. Statistical significance of coefficients and selection using the likelihood ratio test are explained below.

Statistical Significance A feature does not contribute in model (1) if the column of coefficients corresponding to it are all extremely small. A simple, scale-independent test for determining this is to ascertain the statistical significance of the β values by comparing them with their standard deviation. If $|\beta_{lf}| \leq 2\sigma_{lf}, \ \forall l \in [1 : L-1]$, where σ_{lf} are the corresponding standard deviations, the feature ϕ_f is dropped from the model and another feature is randomly selected in its place. The regression model is then re-learnt with the current coefficients being used as initial values for the optimization. Since the discarded feature was not statistically significant, the coefficient values for the other features generally

Algorithm 1 RML Feature Selection

```
Input: Current multinomial logistic regression model $ML = \{\phi_{1:M}, \beta\}$, coefficient standard deviations
  $\sigma$, rounds of feature selection $S$
  $LL \Leftarrow$ log-likelihood of $ML$
  for $i = 1$ to $S$ do
    $B \Leftarrow$ set of features $\phi_{\{f\}}$ s.t. $|\beta_{lf}| \leq 2\sigma_{lf}\ \forall l \in 1 : L-1$
    if $B$ is empty then
      Swap a randomly selected feature from current model for a randomly selected new feature $\phi'$
    else
      Swap a randomly selected feature from $B$ for a randomly selected new feature $\phi'$
    end if
    $ML' \Leftarrow$ new model learnt using maximum likelihood
    $LL' \Leftarrow$ log-likelihood of $ML'$
    if $LL' > LL$ then
      $ML \Leftarrow ML'$, $LL \Leftarrow LL'$
    end if
  end for
  return $ML$
```

do not change much [1] ,and the re-learning proceeds efficiently.

The standard deviation of the coefficient estimates can be computed from the Hessian of the log-posterior function (3)

$$\frac{\partial^2 L}{\partial \beta_{chlf}^2} = \sum_{\{\mathbf{x},y\}} -\phi_h(\mathbf{x})\phi_f(\mathbf{x})\pi_c\left(I(c=l)-\pi_l\right) - 2\alpha \tag{5}$$

where c,l and h,f are indices into the label and feature sets respectively. The inverse of the Hessian is the covariance matrix of β, from which the standard deviations can be obtained.

Randomized Feature Selection When all the features in a multinomial logistic regression model are statistically significant, the model is improved by randomized feature selection based on the likelihood ratio test. The quantity $-2\log L$, where L is the log-likelihood of the model, follows a chi-squared statistic and is smallest for the best-fitting model. Hence, for two models differing by a single feature, the model with the lower statistic is retained. The feature selection algorithm is summarized in Algorithm 1.

5 Evaluation

The RML was implemented in Matlab and all our experiments were run on a modern, quad-core linux machine.

5.1 Implementation

Textonization We first pre-process all the images so that they are histogram equalized and have zero mean, unit standard deviation. Subsequently, the training set is convolved with a 17 dimensional filter bank, consisting of Gaussians at scales k, $2k$, $4k$, derivatives of Gaussians along the x and y axes at scales $2k$, $4k$, and Laplacians of Gaussians at scales k, $2k$, $4k$, $8k$,

[1] though correlation effects can cause large changes, albeit rarely

where k is a parameter. The Gaussians are computed on all three channels of the CIELab color space while the rest of the filters are only applied to the luminance channel. This filter bank is the same as the one used in [28]. The 17-D vectors corresponding to the pixels are clustered using hierarchical k-means clustering with Elkan's method [7] to speed it up.

Texture-layout features are computed efficiently using integral images [23]. Moreover, since the features to be computed are known before-hand, they need to be computed just once, as opposed to Random forests.

Learning the multinomial logit models We learn the multinomial logistic regression models by maximizing (3) using Nocedal's L-BFGS [18], which is a low memory, quasi-Newton optimization algorithm. The regression models in RML are learnt using a booststrapping approach in the sense that the training data for successive models are biased towards instances that the existing models have trouble classifying correctly. We also found that this reduces overlap between the training data subsets used, and hence, the correlation between the regression models. The regularization parameter α is set manually to get optimal performance on the training set.

Implementations for comparison We implemented Textonboost and Random forests for comparison purposes. Both algorithms learn texture models and perform texture-based pixel-wise classification.

Textonboost was implemented as described in [23] with standard texture-layout features and random feature selection. We implemented a Random forest classifier based on texture-layout features for comparison purposes. Learning proceeds by taking all the pixels as training points and splitting this set at each node of the decision tree according to the function $v_{[r,t]}(p) > \theta$, where $v_{[r,t]}(.)$ is a randomly chosen texture-layout feature with shape r acting on texton t, and θ is a threshold that maximizes the information gain of the split out of a set of randomly sampled alternatives.

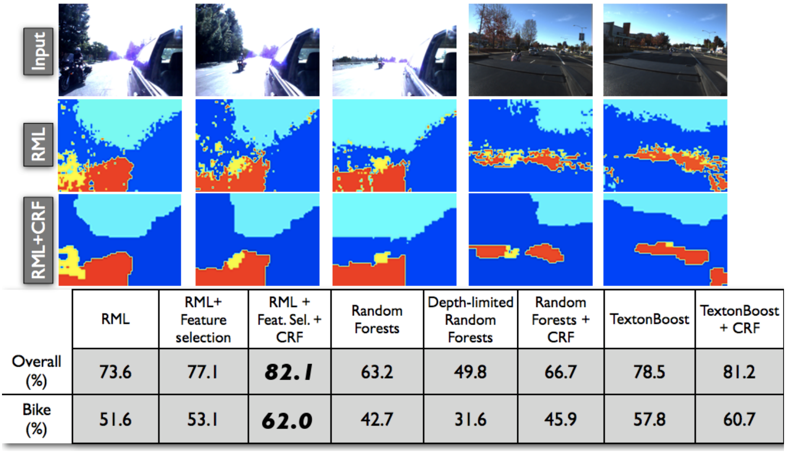
Finally, we implemented the Conditional Random Field (CRF) algorithm described in [23] to take the outputs of RML, Textonboost, and random forests respectively as texture potentials, and provide a smooth segmentation respecting edge boundaries. The color and edge models were the same as in [23] but CRF learning was done using L-BFGS instead of the piece-wise learning used therein.

5.2 Motorbike video dataset

We have gathered a challenging dataset of motorbikes on streets for validating and comparing algorithms in a realistic scenario. The dataset consists of video sequences obtained by mounting a camera pointed backwards from the window of a moving vehicle. The videos are similar to what a driver would see in a side rearview mirror. The videos differ in the types of bikes they contain, viewing perspective, the amount of clutter and lighting quality. Since perspective varies widely - the bikes in this dataset are mostly viewed head-on in a street setting - existing datasets that only contain side views of parked bikes cannot be used for training.

We manually labeled 63 frames from 6 video sequences containing three very different looking bikes and approximately 5800 frames in total. The labeling only contains 4 classes - bike, road, sky, and other, and hence, the scene segmentation task in this case is to segment these classes precisely. 43 of the labeled frames were used for training while the rest were used as test data for calculating error.

The 1024x768 training images were subsampled so that only every 4th pixel was used for training. The number of textons used was $N_t = 100$, and the number of rectangular shapes



	RML	RML+ Feature selection	RML + Feat. Sel. + CRF	Random Forests	Depth-limited Random Forests	Random Forests + CRF	TextonBoost	TextonBoost + CRF
Overall (%)	73.6	77.1	**82.1**	63.2	49.8	66.7	78.5	81.2
Bike (%)	51.6	53.1	**62.0**	42.7	31.6	45.9	57.8	60.7

Figure 3: Results on the motorbike dataset. **Top:** Representative results on a few frames of video with sky labeled in light blue, road in red, bike in yellow, and background in dark blue. Note the difficulty due to illumination, shadows, occlusions, and small size of the bikes. **Bottom:** Overall pixel-wise classification and Bike-only pixel-wise classification results for various algorithms.

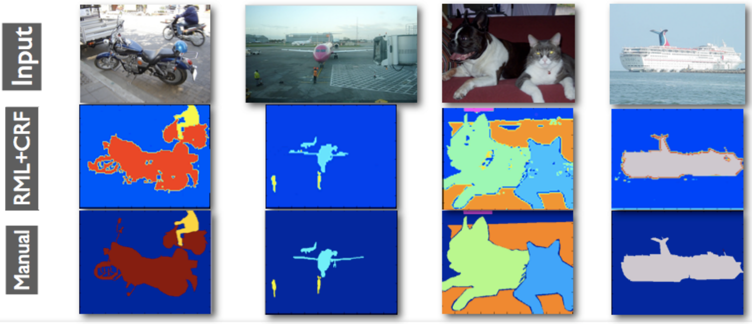
used to create texture-layout features was $N_r = 100$. RML was trained with $N = 15$ and $M = 15$, 25 decision trees were used in the random forest, and Textonboost was trained for 500 rounds of boosting (and had achieved zero training error by then).

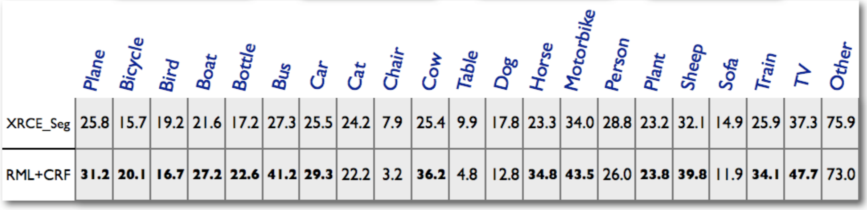
The resulting pixel-wise accuracies of the various algorithms are given in Figure 3. Also shown are representative images of results from RML. A video of these results has also been submitted as supplementary material. Since the motorbike label is frequently only a very small portion of the image, the pixel-wise accuracy for bikes is more revealing and is given separately. Textonboost and RML show very similar results but RML is much faster, especially during runtime (0.12 secs/frame compared to 6.03 sec/frame for TextonBoost), because it evaluates fewer features. However, this results in slightly noisier segmentations so that the improvement in RML through the use of the CRF is much greater than that in the other algorithms. Figure 3 also shows that the use of our feature selection algorithm results in an increase in performance as expected.

Results for two versions of the random forest algorithm are given - one in which the trees are only learned upto a maximum depth $d = 15$ and the second in which they are grown to full depth until the leaves only contain pixels from a single label. The depth limited version performs worse (figure 3) in labeling bikes. However, we found that this gap in performance was largely due to extreme label bias in the training set combined with large viewpoint variations. When the training data was resampled to reduce label bias and images differing most widely in viewpoint were discarded, the performance of the two versions of random forests became similar. This indicates that not limiting the tree depth results in a more powerful random forest classifier capable of coping with greater variation in the training data without the need for adding randomly transformed copies of the training images [22].

5.3 VOC 2008

The VOC dataset [8] contains 20 object classes and is considered extremely challenging. Parameters for RML were $N = 25$ and $M = 20$. We used the 'trainval' split in the dataset for





	Plane	Bicycle	Bird	Boat	Bottle	Bus	Car	Cat	Chair	Cow	Table	Dog	Horse	Motorbike	Person	Plant	Sheep	Sofa	Train	TV	Other
XRCE_Seg	25.8	15.7	19.2	21.6	17.2	27.3	25.5	24.2	7.9	25.4	9.9	17.8	23.3	34.0	28.8	23.2	32.1	14.9	25.9	37.3	75.9
RML+CRF	**31.2**	**20.1**	**16.7**	**27.2**	**22.6**	**41.2**	**29.3**	22.2	3.2	**36.2**	4.8	12.8	**34.8**	**43.5**	26.0	**23.8**	**39.8**	11.9	**34.1**	**47.7**	73.0

Figure 4: **Top:** Representative results from the VOC dataset giving a visual comparison of RML output and groundtruth. **Bottom:** Results on the VOC 2008 dataset given as pixel-wise percentage accuracy. XRCE_Seg is the method from [5] which performed best on this dataset at the PASCAL challenge [8]. RML improves on these results in most cases. Mean accuracy for the two methods are 25.8% for XRCE_Seg and 28.7% for RML+CRF.

training and testing. We compare against [5], which was the best method in the PASCAL challenge 2008. It can be seen from the results in Figure 4 that RML improves over [5] almost across the board and is almost as good when it does not. Exceptions are the chair and table classes, which are extremely hard, and get confused with other classes and background.

We also tested the effect of the number of models N, and the number of features M on this dataset. Figure 5 verifies that performance increases with N initially but then plateaus off as expected. It also demonstrates the need for regularization in multinomial regression models with large number of features, as without regularization performance initially increases with M but then starts decreasing beyond a particular value.

6 Discussion

We described a new multi-way classification algorithm based on multinomial logistic regression and demonstrated its usefulness for scene analysis through state of the art results on challenging datasets. Random Multinomial Logit (RML) combines small groups of randomly selected features in a theoretically sound and well-proven statistical framework. It is easy to implement and very computationally efficient.

A limitation of RML is that a large label bias in data is hard to overcome when learning the regression models except through bootstrapping or biased sampling of the training data. RML is also noisier when used as a stand-alone classifier on difficult datasets, thus necessitating some form of smoothing of the output.

It is future work to explore the use of sparse multinomial regression methods [14] for speeding up RML since sparsification can be employed instead of feature selection. RML

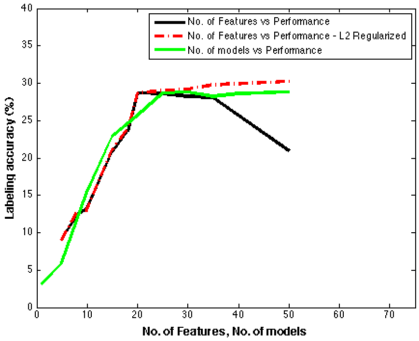


Figure 5: Effect of RML parameters on labeling accuracy: the black (dark) curve shows the effect of increasing the number of features M in a RML without regularization. The red (dashed) curve shows the same effect but with an L2-regularized RML while the green (light) curve demonstrates that performance levels off as the number of models N in the RML is increased beyond a certain limit.

can also quantify its confidence on its output labeling, based on confidence intervals from multinomial logistic regression. Use of this information can improve performance. Finally, it may be possible to fuse other cues based on similar representations, such as shape and contours [19], into a single RML-based framework for robust classification.

References

[1] S. M. Bileschi. *StreetScenes: Towards Scene Understanding in Still Images*. PhD thesis, Massachusetts Institute of Technology, May 2006.

[2] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *Intl. Conf. on Computer Vision (ICCV)*, pages 1–8, 2007.

[3] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[4] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[5] G. Csurka and F. Perronnin. A simple high performance approach to semantic segmentation. In *British Machine Vision Conf. (BMVC)*, 2008.

[6] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[7] C. Elkan. Using the triangle inequality to accelerate k-means. In *Intl. Conf. on Machine Learning (ICML)*, pages 147–153, 2003.

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results. http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html, 2008.

[9] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[10] P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 36(1):3–42, 2006. URL `http://www.montefiore.ulg.ac.be/services/stochastic/pubs/2006/GEW06a`.

[11] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labelling. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[12] B. Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290:91–97, March 1981.

[13] P. Kohli, L. Ladicky, and P. Torr. Robust higher order potentials for enforcing label consistency. *Intl. J. of Computer Vision*, 2009.

[14] B. Krishnapuram, M. Figueiredo, L. Carin, and A. Hartemink. Sparse multinomial logistic regression: Fast algorithms and generalization bounds. *IEEE Trans. Pattern Anal. Machine Intell.*, 27:957–968, June 2005.

[15] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[16] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *Intl. J. of Computer Vision*, 43:7–27, June 2001.

[17] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

[18] J. Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35:773–782, 1980.

[19] A. Opelt, A. Pinz, and A. Zisserman. Fusing shape and appearance information for object category detection. In *British Machine Vision Conf. (BMVC)*, 2006.

[20] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Trans. Pattern Anal. Machine Intell.*, 2009.

[21] A. Prinzie and D. Van den Poel. Random forests for multiclass classification: Random multinomial logit. *Expert Systems with Applications*, 35(3), 2008.

[22] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[23] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling appearance, shape and context. *Intl. J. of Computer Vision*, 81(1):2 –, 2009.

[24] Josef Sivic, Bryan Russell, Alexei A. Efros, Andrew Zisserman, and Bill Freeman. Discovering objects and their location in images. In *Intl. Conf. on Computer Vision (ICCV)*, 2005.

[25] E. Sudderth, A. Torralba, W. Freeman, and A. S. Willsky. Describing visual scenes using transformed objects and parts. *Intl. J. of Computer Vision*, March 2008.

[26] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *Intl. J. of Computer Vision*, 62:61–81, April 2005.

[27] J. Winn and A. Criminisi. Object class recognition at a glance. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[28] J. Winn, A. Criminisi, and T. Minka. Categorization by learned universal visual dictionary. In *Intl. Conf. on Computer Vision (ICCV)*, volume 2, pages 1800–1807, 2005.

[29] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *Intl. J. of Computer Vision*, 73:213–238, 2007.