Architectural Style Classification using Multinomial Latent Logistic Regression

Zhe Xu[1,2], Dacheng Tao[2], Ya Zhang[1], Junjie Wu[3], and Ah Chung Tsoi[4]

[1] Shanghai Key Laboratory of Multimedia Processing and Transmissions, Shanghai Jiao Tong University, Shanghai, China
[2] Centre for Quantum Computation & Intelligent Systems and Faculty of Engineering and Information Technology, University of Technology, Sydney, NSW, Australia
[3] School of Economics and Management, Beihang University, Beijing, China
[4] Faculty of Information Technology, Macau University of Science and Technology, Macau, China

Abstract. Architectural style classification differs from standard classification tasks due to the rich inter-class relationships between different styles, such as re-interpretation, revival, and territoriality. In this paper, we adopt Deformable Part-based Models (DPM) to capture the morphological characteristics of basic architectural components and propose Multinomial Latent Logistic Regression (MLLR) that introduces the probabilistic analysis and tackles the multi-class problem in latent variable models. Due to the lack of publicly available datasets, we release a new large-scale architectural style dataset containing twenty-five classes. Experimentation on this dataset shows that MLLR in combination with standard global image features, obtains the best classification results. We also present interpretable probabilistic explanations for the results, such as the styles of individual buildings and a style relationship network, to illustrate inter-class relationships.

Keywords: Latent Variable Models, Architectural Style Classification, Architectural Style Dataset

1 Introduction

Buildings can be classified according to architectural styles, where each style possesses a set of unique and distinguishing features [5]. Some features, especially the façade and its decorations, enable automatic classification using computer vision methods. Architectural style classification has an important property that styles are not independently and identically distributed. The generation of architectural styles evolves as a gradual process over time, where characteristics such as territoriality and re-interpretation lead to complicated relationships between different architectural styles.

Most of existing architectural style classification algorithms focus on efficient extraction of discriminative local-based patches or patterns [1, 3, 4, 8, 14]. In a four-style classification problem, Chu *et al.* [3] extracted visual patterns by modeling spatial configurations to address object scaling, rotation, and deformation.

Goel *et al.* [8] achieved nearly perfect results on published datasets by mining word pairs and semantic patterns, and therefore tested this approach further on a more challenging five-class dataset collected from the internet. Zhang *et al.* [20] used "blocklets" to represent basic architectural components and adopted hierarchical sparse coding to model these blocklets. However, as argued in [17], some patches that look totally different can be very close in the feature space, which degrades the performance of local patches in understanding detail-rich architecture images. One recent study showed the possibility of cross-domain matching from sketches to building images [15], and this inspired us to employ sketch-like features to represent the building façades.

The Deformable Part-based Model (DPM) [6] is a popular scheme that employs sketch-like Histogram of Oriented Gradient (HOG) features. DPM models both global and local cues and enables flexible configuration of local parts by introducing so-called deformation costs. By adopting a latent SVM (LSVM) algorithm for training, the DPM-LSVM framework produces hard assignments to the labels as classification results. However, in order to enable rational explanation of the gradual transition and mixture of architectural styles, it would be preferable to provide soft assignments and introduce the concept of probability into the model.

In this paper, we propose an algorithm that introduces latent variables into logistic regression, which we term "*Multinomial Latent Logistic Regression*" (MLLR). MLLR is similar to latent SVM but provides efficient probabilistic analysis and straightforward multi-class extension using a multinomial model. MLLR overcomes some of the drawbacks of LSVM, such as dealing with imbalanced training data and the multi-class problem, while producing soft assignment results.

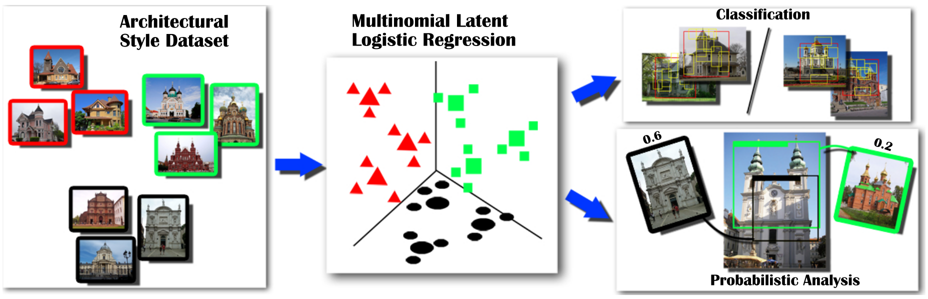
Architectural Style Dataset
Multinomial Latent Logistic Regression
Classification
0.6
0.2
Probabilistic Analysis

Fig. 1. Schematic illustration of architectural style classification using Multinomial Latent Logistic Regression (MLLR). Given a new large-scale architectural style dataset, we model the façade of buildings using deformable part-based models. In the middle figure, the smaller marks that surround the larger marks represent possible latent values for an object. The proposed latent variable algorithm simultaneously trains all the style models given a total objective function. The resulting classifiers can provide probabilistic analysis along with the standard classification results.

However, one reason why few previous studies focus on this problem is a lack of well-organized, large-scale datasets. Here, we collect a new and challenging dataset containing 25 architectural styles. The dataset possesses several preferred properties enclosing in architectural style classification, including multiple classes, inter-class relationships, hierarchical structure, change of views and scales. Our new dataset provides an improved platform for evaluating the performance of existing classification algorithms, and encourages the design of new ones. Fig. 1 summarizes the framework of the paper and illustrates the dataset, algorithm, and applications.

The contributions of this paper are:

– we create a multi-class and large-scale architectural style dataset and design several challenges based on this dataset. The rich set of inter-relationships between different architectural styles distinguish this dataset from standard scene classification datasets;
– we propose the MLLR algorithm, which introduces latent variables to logistic regression, analogous to latent SVM. The algorithm simultaneously trains classifiers for all classes. Thus, the detriment of the multi-class problem and unbalanced training data is minimized;
– by introducing the concept of "probability" to architectural style analysis, we analyze the inter-relationships of architectural styles probabilistically. Specifically, besides classification accuracies, the algorithm outputs a style relationship network and provides architectural style analysis for individual buildings.

2 Architectural Style Dataset

An architectural style is a specific construction, characterized by its notable features. For instance, unique features, such as pointed arches, rib vaults, rose windows and ornate façades, make it possible to distinguish the *Gothic* style from other styles. Architectural history has dictated that there are complicated inter-relationships between different styles, including rebellion, special territoriality, revivals, and re-interpretations. As a consequence, it is difficult to strictly classify two styles using a standard criterion.

In order to study architectural styles and model their underlying relationships, we collected a new architectural style dataset from Wikimedia[5]. We obtained the initial list by querying with the keyword *"Architecture_by_style"*, and downloaded images from subcategories following Wikimedia's hierarchy using the depth-first search strategy. The crawled images were manually filtered to exclude images of non-buildings, interior decorations, or part of a building. Therefore, the remaining images contained only the exterior façade of buildings. Styles with too few images were discarded, resulting in a total of 25 styles. The number of

[5] From Wikimedia commons.
http://commons.wikimedia.org/wiki/Category:Architecture_by_style.

images in each style varies from 60 to 300, and altogether the dataset contains approximately 5, 000 images[6].

We propose several challenges to extensively exploit the data size and rich relationships between different architectural styles in this dataset. Fig. 2 illustrates the dataset.

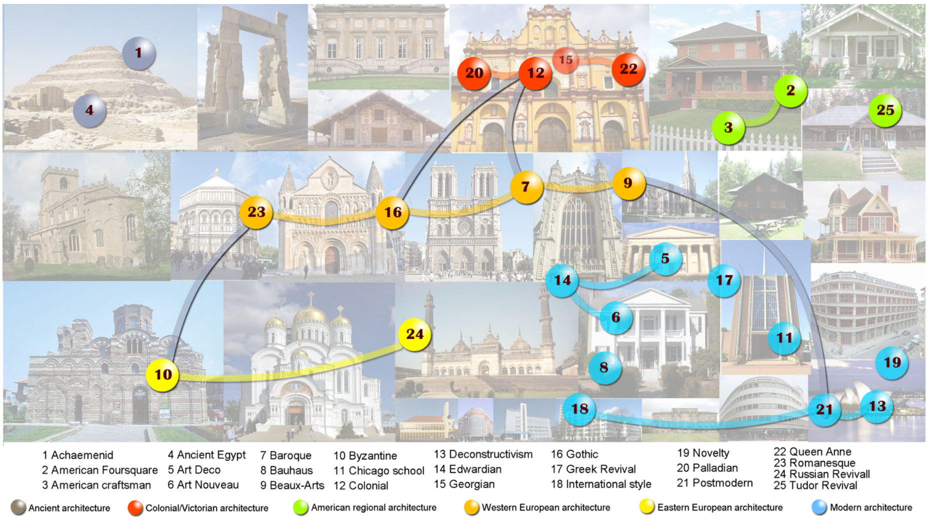

Fig. 2. Illustration of the architectural style dataset. Each of the 25 styles is represented by a circle with the respective number in the middle, where different colors indicate broad concepts, such as modern architecture and medieval architecture. The styles are arranged according to time order, where newer ones are placed in the right of ancient ones. Various inter-class relationships exist between the styles, e.g., lines between circles stand for following relationships; smaller circles around large ones indicate sub-categories. Typical images of the styles are shown in the background. Better viewed in color.

– **Multi-class classification**. To the best of our knowledge, this dataset is the largest publicly available dataset for architectural style classification. Other popular datasets related to buildings do exist, such as the Oxford Landmark dataset [14]. However, their main purpose is for the retrieval of individual landmark buildings rather than classification of architectural styles. A discussion of the difference between "style" and "content" can be found in [7]. There are some researches on different type of art styles, such as painting [18, 21] and car designing [10], which may also provide cross-domain knowledge from other aspects of art styles.
– **Modeling inter-class relationships between styles**. Various relationships exist between the 25 architectural styles, e.g., following, revival, and

[6] https://sites.google.com/site/zhexuutssjtu/projects/arch

against. Styles can be roughly classified into broad concepts, such as ancient architecture, medieval architecture and modern architecture, and thus further be arranged in a hierarchical structure. For reference, we summarize the relationships between different styles verified by Wikipedia. It is of interest to explore whether computer vision algorithms can efficiently extract the underlying inter-class relationships.

– **Modeling intra-class variance within a style**. The establishment of an architectural style is a gradual process. When styles spread to other locations, each location develops its own unique characteristics. On the other hand, each building is unique due the personalities of different architects. Therefore, it is challenging to find common features within a style, as well highlighting the specific design of an individual building.
– **Style analysis for an individual building**. When designing a building, an architect sometimes integrates several different style elements. The building can therefore be represented as a mixture of styles. An algorithm should be able to model this phenomenon, e.g., show that the window is inspired by style I and the arch by style II.

In this paper, we develop a probabilistic latent model algorithm to tackle some of these challenges, including classification, inter-class relationships, and individual building style analysis. Other challenges such as how to build a hierarchy of styles and how to deal with different shooting angles remain open issues for future study.

3 Model Description

A Deformable Part-based Model (DPM) [6] describes an image by a multi-scale HOG feature pyramid. The model consists of three parts: (i) a root filter that captures the outline of the object; (ii) a set of part filters that are applied to the image with twice the resolution of the root, conveying detailed information of the object; (iii) deformation costs that penalize deviations of the parts from their default locations with respect to the root.

In object detection tasks, a hypothesis x represents location of the root in the feature pyramid, and the part locations with respect to the root are treated as latent variables z. The filter response is given by the dot product of the HOG features and the model parameters, i.e., in the form of a score function:

$$s_\beta(x) = \max_{z \in Z(x)} \beta \cdot f(x, z), \tag{1}$$

where β is the vector of DPM parameters, $Z(x)$ stands for all possible relative positions between the root and parts. To introduce deformation costs, the model is parameterized by a concatenation of all the filters and deformation weights, and $f(x, z)$ is the concatenation of the HOG features and the part displacements.

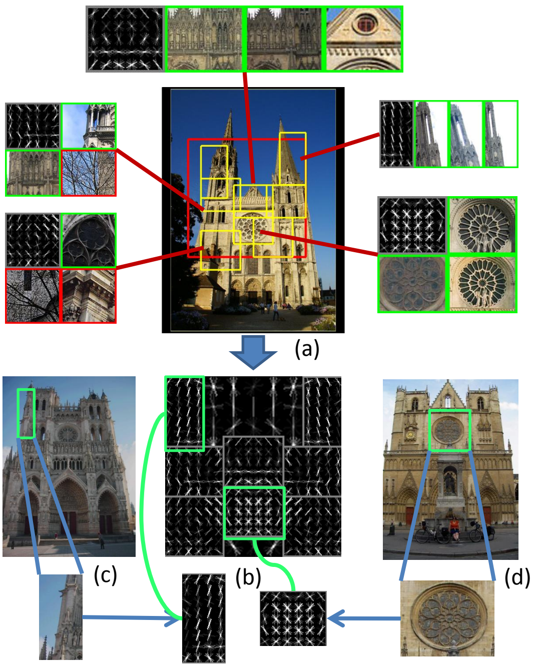

Fig. 3. Visualization of the use of DPM in architectural style classification. (a)(c)(d) show detection results for different testing images. The trained model for *Gothic* architectural style is shown in (b). The root model shows typical façade outline of *Gothic* style buildings, and the part filters captures discriminative architectural elements such as rose windows. Independently using each part filter cannot obtain convincing results, i.e., red boxes indicate incorrect detections with high scores given a part filter.

To train the model parameters β, a latent SVM algorithm is adopted, whose objective function is defined analogically to classical SVMs as:

$$L(D) = \frac{1}{2}||\beta||^2 + C\sum_{i=1}^{N} \max(0, 1 - y_i s_\beta(x_i)), \tag{2}$$

where $\max(0, 1 - y_i s_\beta(x_i))$ is the standard hinge loss and C is the soft margin parameter, which controls the weight of the regularization term. Due to the non-convex training objective function, latent SVM is solved using a coordinate descent framework.

Following [6], Pandey *et al.* [13] use DPM in a scene recognition and a weakly supervised object localization task. They point out that scene recognition can also be viewed as a "part-based" problem, where the root captures the entire image and the parts encompass moveable "regions of interest" (ROIs). In their experiments, they find that when the model is trained using the entire image as the root, the resulting performance is not as good as expected. They remark that the root filter should be allowed to move, and regarding the position of the root filter as another latent variable. We follow this setup, i.e., use a square root filter and restrict it to have at least 40% overlap with the image.

In our implementation, we exploit DPM as the detection model and introduce MLLR as the learning algorithm. A visualization of DPMs in architectural style classification is shown in Fig. 3.

4 Multinomial Latent Logistic Regression

In this section, we first review the concept and notation of logistic regression and then show the form of posterior probabilities using latent variables. By proving the semi-concavity property of the resulting objective function, we provide a gradient ascent solution analogous to latent SVM [6]. The training procedure has a clear explanation based on probabilistic analysis.

4.1 Logistic Regression

Given a training set of $D = \{(x_1, y_1), ..., (x_N, y_N)\}$, where $y_i \in \mathcal{Y} = [1, 2, ..., K]$, the logistic regression models the posterior probabilities of K classes via linear functions in x. The posterior probability has the form:

$$Pr(Y = k|X = x) = \frac{exp(\beta_k^T x)}{\sum_{l=1}^{K} exp(\beta_l^T x)}, \tag{3}$$

where β_k are the model parameters of the k-th class. We omit the bias term in the model for brevity. Denote the entire parameter set as $\theta = \{\beta_1^T, ..., \beta_K^T\}$. The probability of an example x belonging to a class k given model parameters θ is defined as $Pr(Y = k|X = x) = p_k(x; \theta)$.

Logistic regression models are typically fitted by maximizing the log-likelihood function, defined as:

$$l(\theta) = \sum_{i=1}^{N} \log p_{y_i}(x_i; \theta). \tag{4}$$

To maximize the log-likelihood function, gradient-based methods are generally used, such as the Newton-Raphson algorithm.

4.2 Latent variables

In MLLR, each input example x is associated with a latent variable z. Let $f(x, z)$ be the feature vector of an example x with a latent variable z, where $z \in Z(x)$, and the $Z(x)$ define the possible latent value set for an example x.

Consider a score function of the form:

$$s(x; \beta_k) = \max_{z \in Z(x)} \beta_k \cdot f(x, z). \tag{5}$$

The score is obtained by finding the optimized latent value z that gives the highest score to example x given a model β_k. An example x is called a positive example with respect to model β_k if the respective training label $y = k$, and called a negative example with respect to model β_k if the label $y \neq k$.

The optimized latent value can then be denoted as:

$$z(\beta_k) = \arg\max_{z \in Z(x)} \beta_k \cdot f(x, z). \tag{6}$$

Analogous to standard logistic regression, we rewrite the posterior probability of the k-th class given an example x as:

$$p_k(x; \theta) = \frac{exp(s(x; \beta_k))}{\sum_{l=1}^{K} exp(s(x; \beta_l))}. \tag{7}$$

Given that the parameter space has a large volume, the training data might be easily overfit by MLLR; we therefore add a lasso regularizer to avoid overfitting. The log-likelihood function then becomes:

$$\begin{aligned} l(\theta) &= \sum_{i=1}^{N} \log p_{y_i}(x_i; \theta) - \lambda \sum_{l=1}^{K} |\beta_l| \\ &= \sum_{i=1}^{N} \log \frac{exp(s(x_i; \beta_{y_i}))}{\sum_{l=1}^{K} exp(s(x_i; \beta_l))} - \lambda \sum_{l=1}^{K} |\beta_l| \\ &= \sum_{i=1}^{N} s(x_i; \beta_{y_i}) - \sum_{i=1}^{N} \log \sum_{l=1}^{K} exp(s(x_i; \beta_l)) - \lambda \sum_{l=1}^{K} |\beta_l|, \end{aligned} \tag{8}$$

4.3 Semiconcavity

Maximizing the log-likelihood function of the standard logistic regression model in (4) leads to a concave optimization problem. However, after introducing latent variables to the log-likelihood function, the function is no longer concave with respect to the model β_k, due to the maximum operator in $s(x_i; \beta_{y_i})$. Similar to LSVM, MLLR has a semi-concavity property. Fixing the latent value for positive examples, the first term in (8) is reduced to a linear function of β_k, which is concave. We thus employ gradient ascent method based on the semi-concavity property to maximize the likelihood function.

Define $l(\theta, Z_p)$ as an auxiliary function that bounds the exact likelihood function by fixing the latent variable for each positive example, where $Z_p = \{z_i, i = 1, ..., N\}$ is a set of latent values specifying the positive configuration for all the N training examples. In particular, the auxiliary function is defined as:

$$l(\theta, Z_p) = \sum_{i=1}^{N} s'(x_i; \beta_{y_i}) - \sum_{i=1}^{N} \log \sum_{l=1}^{K} exp(s'(x_i; \beta_l)) - \lambda \sum_{l=1}^{K} |\beta_l|, \tag{9}$$

where

$$s'(x_i;\beta_l) = \begin{cases} \beta_l^T f(x_i, z_l), \, y_i = l. \\ s(x_i;\beta_l), \quad\;\; y_i \neq l. \end{cases}$$

Considering that $l(\theta, Z_p)$ is a concave function, we maximize $l(\theta)$ using the coordinate ascent method, as follows.

1. *Optimize positive examples*: Optimize $l(\theta, Z_p)$ over Z_p. For each example x_i, find the optimized latent value for its respective model β_{y_i} using (5), $z_i^* = \arg\max_{z \in Z(x_i)} \beta_{y_i} \cdot f(x_i, z)$ and set $Z_p = \{z_i^*, i = 1, ..., N\}$.

2. *Optimize model parameters θ*. Optimize the concave function $l(\theta, Z_p)$ over θ and all possible latent values for negative examples.

4.4 Gradient Ascent

This process is to optimize model parameters θ and the latent value for negative examples. Although the lasso regularization term is non-differentiable, we can compute a subgradient of (8) with respect to β_k as:

$$\nabla l(\beta_k) = \sum_{i=1}^{N} f(x_i, z_i(\beta_k)) \cdot h(x_i, \beta_k) - \lambda \cdot sgn(\beta_k), \tag{10}$$

where

$$h(x_i, \beta_k) = \begin{cases} 1 - p_k(x_i;\theta) \; ,x_i \text{ is positive for class } k. \\ -p_k(x_i;\theta) \quad ,x_i \text{ is negative for class } k. \end{cases}$$

The gradient ascent procedure iteratively updates model parameters and latent variables for negative examples, as follows:

1. In the $(t+1)$-th iteration, for all training examples x_i and all class models β_k, let $z_i(\beta_k^{(t)}) = \arg\max_{z \in Z(x_i)} \beta_k^{(t)} \cdot f(x_i, z)$, where $y_i \neq k$, and $z_i(\beta_k^{(t)}) = z_i^*$ if $y_i = k$, where $z_i^* \in Z_p$

2. For all class models, set $\beta_k^{(t+1)} = \beta_k^{(t)} + \alpha_t \cdot [\sum_{i=1}^{N} f(x_i, z_i(\beta_k^{(t)})) \cdot h(x_i, \beta_k^{(t)}) - \lambda \cdot sgn(\beta_k^{(t)})]$.

The form of $h(x_i, \beta_k)$ has a clear probabilistic explanation. Similar to the perceptron algorithm, the gradient ascent method repeatedly pushes the model β_k towards positive examples and away from negative examples. By adding a probabilistic multiplier, the algorithm assigns a larger penalization on "hard negative", where $p_k(x_i;\theta)$ is large. For positive examples, a "hard positive" indicates an example that has a smaller probability with respect to the current model, and plays a more important role in updating the model parameters.

The training procedure is outlined in Algorithm 1.

4.5 Comparison to the latent SVM

MLLR and LSVM have many shared characteristics, such as the form of latent variables and the semi-convexity property. These similarities enable MLLR to incorporate existing latent variable models trained by LSVM, such as deformation part-based models. However, there are some notable differences between

Algorithm 1 Multinomial Latent Logistic Regression Training

```
Input: Training examples $\{(x_1, y_1), ..., (x_N, y_N)\}$
  Initial models $\theta = \{\beta_1, ..., \beta_K\}$
Output: New models $\theta$
  for posLoop:=1 to numPosLoop do
    {Relabel positive examples}
    for $i$:=1 to $N$ do
      Optimize $z_i^* = \arg\max_{z \in Z(x_i)} \beta_{y_i} \cdot f(x_i, z)$.
    end for
    {Gradient Ascent}
    for t:=1 to numGradientAscentLoop do
      {Relabel negative examples}
      for $i$:=1 to $N$ do
        for $k$:=1 to $K$ and $k \neq y_i$ do
          Optimize $z_i(\beta_k^{(t)}) = \arg\max_{z \in Z(x_i)} \beta_k^{(t)} \cdot f(x_i, z)$.
        end for
      end for
      {Update model parameters}
      for $k$:=1 to $K$ do
        Update $\beta_k^{(t+1)} = \beta_k^{(t)} + \alpha_t \cdot [\sum_{i=1}^{N} f(x_i, z_i(\beta_k^{(t)})) \cdot h(x_i, \beta_k^{(t)}) - \lambda \cdot sgn(\beta_k^{(t)})]$.
      end for
    end for
  end for
```

MLLR and LSVM, which make MLLR more suitable for the architectural style classification task.

First, it is argued that SVM tends to underperform when using imbalanced training data where negative examples far outnumber positive examples [19]. The vast "background" class in the object detection framework introduces a serious imbalance between positive and negative examples. Moreover, in LSVM, the training process needs to fix the latent value for positive examples, while keeping all possible latent values for negative examples. This process makes the imbalance problem even more severe.

Second, the dominant method for solving multi-class problems using SVM has been based on reducing a single multi-class problem to multiple binary problems. However, since each binary problem is trained independently, adopting this strategy is problematic because it cannot capture correlations between different classes. As a result, the output decision values are not comparable, and this is known as the "calibration" problem. MLLR trains all classes simultaneously by introducing a unified objective function and in this way does not suffer from the hazard of different biases occurring with the multi-class problem and imbalanced training data.

Finally, SVM does not provide an effective probabilistic analysis with a soft boundary. Given an input example, the corresponding output of SVM is called the decision value, which is the distance from the example to the decision boundary. A previous work [12], in which a normalization process was proposed to convert the decision values of SVM to probabilistic outputs, does not provide a genuine probabilistic explanation. MLLR produces comparable classification results for multiple classes, and more reasonably turns them into probabilities.

5 Experiment

The experiment is presented in three steps. In the first step, we choose ten architectural styles that are relatively distinguishable by their façades and have lower intra-class variance. As a result, using sketch-like HOG features, DPM can clearly demonstrate the characteristics of these styles. Second, we evaluate the effect of a more extensive multi-class problem and larger intra-class variance using the full dataset. Given the probabilistic results, we formulate inter-class relationships using a style relationship map. The third part illustrates individual building style analysis of MLLR.

5.1 Classification Task

A ten-class sub-dataset is exploited for the first classification task, most of which have prominent façade or decoration features, such as pointed arches, the ribbed vaults and the flying buttresses characteristics of *Gothic* architecture. For each class, 30 images are randomly chosen as training images and the remaining images are used for testing, $1,716$ in total. We run a ten-fold experiment. The proposed algorithm is denoted by DPM-MLLR. Table 1 compares the classification accuracy of DPM-MLLR with other algorithms, including GIST [16], Spatial Pyramid (SP) [9], Object Bank [11], and DPM-LSVM [13]. DPM-MLLR outperforms LSVM in terms of overall accuracy. It is noted that DPM and local patch-based algorithms, such as Spatial Pyramid, have complementary properties. We therefore combine their results using a naive softmax function and achieve the best result with nearly 70% accuracy.

Table 1. Results on the architectural style classification dataset. MLLR consistently outperforms LSVM. Multiple features are combined by adopting the softmax function on classifier outputs.

	GIST	SP	OB-Partless	OB-Part	DPM-LSVM	DPM-MLLR	MLLR+SP
10 classes	30.74	60.08	62.26	63.76	65.67	67.80	**69.17**
25 classes	17.39	44.52	42.50	45.41	37.69	42.55	**46.21**

Fig. 4 shows the trained models and typical detection results of MLLR. Close inspection of the results reveals that the models capture discriminative features of the styles. For instance, the model representing *American Queen Anne* architecture detects twin gables and allows them to move within limits (in the top of the root). Thus, the model is robust to slight view changes and intra-class variance.

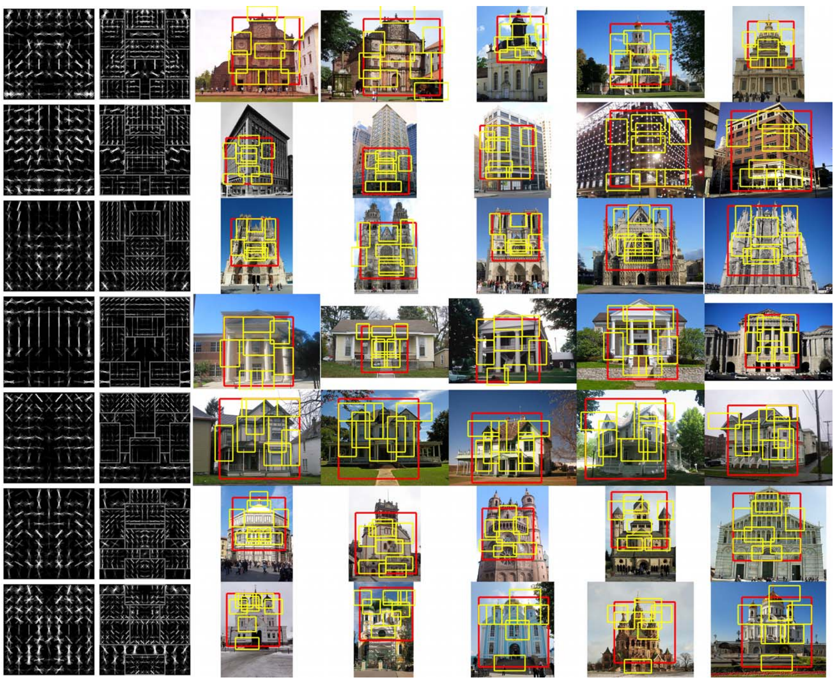

Fig. 4. Testing results for the ten architectural styles. The first two columns visualize the result root and part filters for each model. From top to bottom: *Baroque, Chicago school, Gothic, Greek Revival, Queen Anne, Romanesque* and *Russian Revival* architecture. Detected root filters are displayed in red, and part filters are shown in yellow. Better viewed in color.

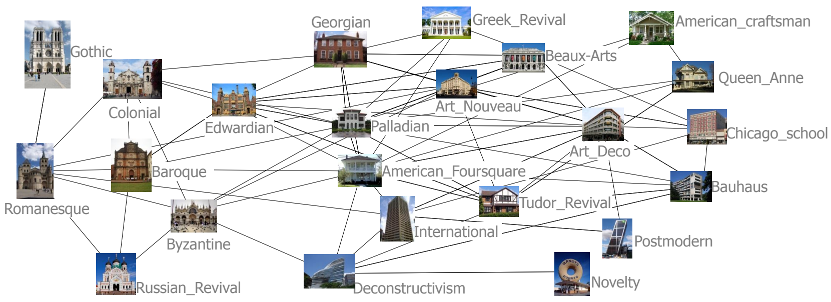

Fig. 5. An architectural style relationship map generated by the proposed algorithm. The confusion probability between style A and B is obtained by summing the probabilities with regard to B for all images labeled by A. Only links whose weight exceeds a given threshold are shown in the figure. Modern styles, such as *Postmodern* and *International* style, are connected, while the links between modern and medieval styles are weak. The figure is drawn using NetDraw [2].

5.2 Inter-class relationships between styles

This part of the experiment is implemented on the full dataset. The 25-class dataset has stronger intra-class invariance and is harder to distinguish purely by the façades. The results show that algorithms that take the features of the entire image into consideration, i.e., Spatial Pyramid and Object Bank, achieve superior performance (Table 1). DPM-MLLR has slightly lower accuracy. However, compared to the result of the ten-class problem, MLLR outperforms LSVM by a larger margin due to the increased number of classes. Again, the combined MLLR-SP algorithm achieves the best result.

Despite classification accuracies, the proposed algorithm provides a probabilistic style distribution for each building image. By summing the probabilities, we obtain a probabilistic confusion matrix, which is further decomposed into a style inter-relationship network by assigning an edge between two styles whose confusion probability exceeds a given threshold. Fig. 5 shows the resulting relationship map of the 25-class dataset. According to the set of relationships between styles collected from Wikipedia, the proposed algorithm gets a recall of 0.66, and the average precision $AP@10$ is 0.51.

Unlike hard-margin confusion matrices, large values can occur in the probabilistic confusion matrix under two occasions. The first is when two styles are similar to each other, making them hard to distinguish, and the second is when two styles appear on different parts of the same building, which is most likely to happen when the styles spread to a same place and start to mix. We try to distinguish these two scenarios by considering whether the optimized detecting bounding boxes of the two styles frequently appear at the same location. Experimental results show that the averaging bounding box intersection ratio of the *Queen Anne* and *American Craftsman* styles is higher than that of the *Baroque* and *Colonial* style (0.56 vs. 0.46), which means that the first two styles have a similar façade and should therefore be more dependent on local parts for classification. This phenomenon is in accordance with architectural history.

5.3 Individual building analysis

MLLR makes it possible to analyze the architectural style of a building probabilistically. Fig. 6 shows two typical situations in which the algorithm gives comparable scores for at least two styles, which correspond to the two scenarios discussed in the previous subsection. The first is when different architectural styles share similar features, such as pear-shaped domes in both *Baroque* and *Russian Revival* architecture. The second scenario appears when architects design new buildings that combine several different architectural styles. For instance, Fig. 6(b) shows a failed classification case in which the main body of the building follows the *Queen Anne* style, while the terrace shows a strong *Greek* sense. MLLR mistakenly classifies the building as *Greek Revival* style due to the unusual shooting angle, which places the main body in side view. However, MLLR discovers interesting patterns in the building that indicates a combination of different styles, and assigns probabilities for each style according to the training set.

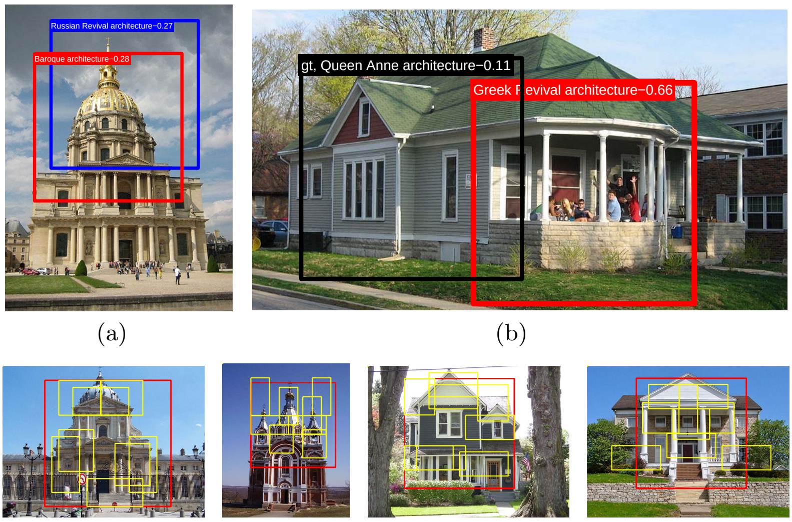
Russian Revival architecture−0.27
Baroque architecture−0.28
gt, Queen Anne architecture−0.11
Greek Revival architecture−0.66
(a)
(b)
(c)
(d)
(e)
(f)

Fig. 6. MLLR detects the optimized latent position for each class and outputs a global list of probabilities for each class. (a) Parts shared by different styles. (b) A building that combines several styles. (c)-(f) Typical detection results for the four styles appearing in (a) and (b), i.e., from left to right, *Baroque, Russian Revival, Queen Anne* and *Greek Revival.*

6 Conclusions

We introduce Multinomial Latent Logistic Regression (MLLR), a latent variable algorithm that uses log-likelihood as the objective function and simultaneously trains multi-class models. Experimental results using a new, large-scale architectural style dataset show that the Deformable Part-based Model (DPM)-MLLR algorithm achieves the best performance when the styles have highly distinguishable façades. The algorithm is also competitive comparing with other state-of-the-art algorithms, even when using a more challenging experimental setup with large intra-class variance. The probabilistic analysis of MLLR makes it possible to interpret inter-class relationships between architectural styles and the combination of multiple styles in an individual building.

Acknowledgement

The work is partially supported by the High Technology Research and Development Program of China (2012AA011702), National Natural Science Foundation of China (61003107), the Australian Research Council Projects FT-130101457 and ARC DP-120103730, the National Natural Science Foundation of China (71322104, 71171007), and the National High Technology Research and Development Program of China (863 Program) (SS2014AA012303).

References

1. Berg, A.C., Grabler, F., Malik, J.: Parsing images of architectural scenes. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on. pp. 1–8. IEEE (2007)
2. Borgatti, S.: Netdraw software for network visualization. Analytic Technologies (2002)
3. Chu, W.T., Tsai, M.H.: Visual pattern discovery for architecture image classification and product image search. In: Proceedings of the 2nd ACM International Conference on Multimedia Retrieval. p. 27. ACM (2012)
4. Doersch, C., Singh, S., Gupta, A., Sivic, J., Efros, A.A.: What makes paris look like paris? ACM Transactions on Graphics (TOG) 31(4), 101 (2012)
5. Dunlop, C.: Architectural Styles. Dearborn Real Estate (2003)
6. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. Pattern Analysis and Machine Intelligence, IEEE Transactions on 32(9), 1627 –1645 (sept 2010)
7. Freeman, W.T., Tenenbaum, J.B.: Learning bilinear models for two-factor problems in vision. In: Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on. pp. 554–560. IEEE (1997)
8. Goel, A., Juneja, M., Jawahar, C.: Are buildings only instances?: exploration in architectural style categories. In: Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing. p. 1. ACM (2012)
9. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. vol. 2, pp. 2169 – 2178 (2006)
10. Lee, Y.J., Efros, A.A., Hebert, M.: Style-aware mid-level representation for discovering visual connections in space and time. In: Computer Vision (ICCV), 2013 IEEE International Conference on. pp. 1857–1864. IEEE (2013)
11. Li, L.J., Su, H., Fei-Fei, L., Xing, E.P.: Object bank: A high-level image representation for scene classification & semantic feature sparsification. In: Advances in neural information processing systems. pp. 1378–1386 (2010)
12. Lin, H.T., Lin, C.J., Weng, R.C.: A note on platts probabilistic outputs for support vector machines. Machine learning 68(3), 267–276 (2007)
13. Pandey, M., Lazebnik, S.: Scene recognition and weakly supervised object localization with deformable part-based models. In: Computer Vision (ICCV), 2011 IEEE International Conference on. pp. 1307–1314. IEEE (2011)
14. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on. pp. 1 –8 (june 2007)
15. Shrivastava, A., Malisiewicz, T., Gupta, A., Efros, A.A.: Data-driven visual similarity for cross-domain image matching. In: ACM Transactions on Graphics (TOG). vol. 30, p. 154. ACM (2011)
16. Torralba, A., Murphy, K.P., Freeman, W.T., Rubin, M.A.: Context-based vision system for place and object recognition. In: Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on. pp. 273–280. IEEE (2003)
17. Vondrick, C., Khosla, A., Malisiewicz, T., Torralba, A.: Hoggles: Visualizing object detection features. ICCV (2013)
18. Watanabe, S.: Discrimination of painting style and quality: pigeons use different strategies for different tasks. Animal cognition 14(6), 797–808 (2011)

19. Wu, G., Chang, E.Y.: Class-boundary alignment for imbalanced dataset learning. In: ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC. pp. 49–56 (2003)
20. Zhang, L., Song, M., Liu, X., Sun, L., Chen, C., Bu, J.: Recognizing architecture styles by hierarchical sparse coding of blocklets. Information Sciences 254, 141–154 (2014)
21. Zujovic, J., Gandy, L., Friedman, S., Pardo, B., Pappas, T.N.: Classifying paintings by artistic genre: An analysis of features & classifiers. In: Multimedia Signal Processing, 2009. MMSP'09. IEEE International Workshop on. pp. 1–5. IEEE (2009)