Image classification using ROIs and Multiple Kernel Learning - Bosch, Zisserman & Munoz

Essay Cognitive Robotics – by Willem Stuursma (1393138)

Image classification

Image classification is the process of classifying an image by the object category that it contains. The authors investigate how far a combination of good features and good learning can go towards complete classification of existing data sets. Two existing data sets are used: the Caltech-101 data set consists of images from 101 object categories, and contains from 31 to 800 image per category. The Caltech-256 data set consists of images from 256 object categories and is an extension of the Caltech-101 data set. The release of challenging data sets with ever more categories is forcing the development of image representations that can cope with multiple classes and of algorithms that are efficient in training and testing.

Describing the images

In order to classify and image, a description of the image is required. This description can then be compared with a description of each object class, after which the image can be classified as the object class that most matches the image description. The images in the data sets are described by two types of descriptors and a representation of spatial layout.

The descriptors consist of visual words computed on a dense grid, which is called the appearance, and of image gradients, again on a dense grid, which is called the local shape. The visual words are computed by vector quantification of SIFT descriptors which are computed for circular patches in a grid of the image. These descriptors are rotation invariant. Each image is then represented by a histogram of word occurrences. Two variants are used: both gray level and color representations
are tested. A vocabulary of 300 words is used. This histogram is called the Pyramid Histogram Of visual Words (PHOW) and is normalized to sum to unity taking into account all the pyramid levels.

The local shape is represented by a histogram of edge orientations within an image subregion quantized into $K$ bins. Each bin in the histogram represents the number of edges that have orientations within a certain angular range. Edge contours are extracted using the Canny edge detector. Again, two variants are used: one that ignores the contrast sign for edges (referred to as Shape$_{180}$) and one that does use all orientation information as in the original SIFT descriptor (referred to as Shape$_{360}$).

For the representation of the spatial layout, an image pyramid representation is used. The image is tiled into regions at multiple resolutions. A histogram of the descriptors is computed for each tile and at each resolution level, and the representation consists of a weighted concatenation of these histograms into a single vector. This pyramid is called the Pyramid Histogram of Oriented Gradients. The distance between the pyramids of two images reflects the extent to which the images contain similar shapes and the extend to which the shapes correspond in their spatial layout. The PHOG is normalized to sum to unity taking into account all the pyramid levels. This ensures that images with more edges are not weighted more strongly than others. Three resolutions are used for creating the pyramids.

Each image descriptor at a pyramid level is a feature. Thus, each image is described by 18 features (three pyramid levels with four appearance and two shape descriptors).

**Weights for each class**

The weights for the concatenation of the pyramid levels are learned for each class. This required because some classes can better be distinguished by their shape and other classes by their appearance. Each pyramid level is represented by a kernel, and the weights used to combine these kernels are learned from training data. These are learned by minimizing the classification error on a validation set. This allows more weight to be given to more the discriminative features of each class.
Weights are learned for each class separately by optimizing classification performance for that class using a one vs. the rest classification. This means that it is necessary to learn one parameter per pyramid per class. This is computationally much more expensive than learning weights for all classes combined, which would require the computation of only one weight per pyramid level. The advantage of learning class-specific feature-weights is that classes then have the freedom to adapt if there is more or less intra-class spatial variation or if one feature is much better than another for a specific class. The disadvantage is that the solution is suboptimal since performance is not optimized over all categories simultaneously.

**Learning the regions of interest**

Instead of using the entire image the learn the model, an alternative is to focus on the object instance. This is necessary because objects in images are not constrained in pose or may have significant background clutter. Therefore, for training sets it is necessary to “home in” to the object instance in order to learn its visual description from the whole image. This is done by automatically by learning the region of interest (ROI) in each image.

Between a subset of images from the same class there will be regions with high visual similarity. These regions typically contain the object instance. These regions can be identified by measuring the similarity using the spatial pyramid representation. The training data can then be represented only by the contents of the ROI instead of the entire image. Thus, first the regions of interest are learnt for each training image, then the other parameters of the classifier are learned from these.

These regions are found in the training images by creating a window to fit the image, and then decreasing the size of this window and moving the window over the images in small steps. The cost function is then optimized. The process is then repeated within the previous window. Thus, the window containing the ROI continually shrinks until the ROI stops changing, or after a preset limit of iterations have been reached.
The classification process

For each class, images are selected as a training and validation set. The training set is disjoint from the test set, which is used to measure the performance of the classifier. Parameter learning is carried out by optimization of the training set classification error. The weights for each class are learned by minimizing the error on the training set. First, the level weights for each descriptor are optimized. When the level weights for each descriptor are optimized, the weights of the different descriptors can be optimized. When the training is complete, the classification process can begin.

An image is classified by a “sliding window” classifier over a range of translations and scales. It is assumed that there is one object instance per image. This holds for almost all images in the data sets. An image is classified as class $c$ which gives the maximum distance from the hyperplane given by the support vector machine (SVM) classifier for each class.

Hierarchical classifier

Finally, when working with many categories, similar categories can easily be confused with each other, for example the categories tennis shoe and sneaker or beer mug and coffee mug. Once the system is trained, the validation set can be used to find the most confused categories, by classifying the images in this set and obtaining a confusion matrix. Categories that are often confused with each other are merged and images are classified using the merged categories. The idea is to merge the confused categories on a first hierarchical level, and then learn the features that best separate the classes into a second hierarchical level. This is especially useful when there are many categories, because when there are many categories, it is probable that some of them will be ambiguous.

This hierarchical system has the capability to learn which are the best features to distinguish between two specific categories instead of trying to learn the best weights to distinguish amongst all the categories. No human intervention is required: the hierarchy is automatically discovered.
**Performance**

Performance score is computed as the mean recognition rate per class. The classification process is repeated ten times, changing the training and test sets, and the average score is reported.

The combination of good features and good learning can go a long way towards the complete classification of data sets such as the Caltech data sets described earlier. Using all the features mentioned above, the Caltech-101 data set was classified with a 98.2 per cent correct classification. This is the highest classification performance for this data set to date. The Caltech-256 data set was classified with 69.8 per cent correct classification. Again, this is the highest classification performance for this data set to date.

**Discussion**

The classification system and methods proposed by Bosch, Zisserman and Munoz are clearly compelling and offer a high performance. All techniques they use have a straightforward theory to back them up. However, the images in the Caltech data sets are probably not realistic for real life image data that a camera would capture. Nearly all images contain exactly one object, many images show this object on a white background. This is not realistic for the cluttered environment that the world is. This is also reflected in the final errors that remain in the classification of the Caltech-256 data set: the erroneous classification of images containing multiple objects.

Also, all images contain a complete object instance. No image contains half an object, again, this is not realistic for real life imagery. A system that is to be applied in the real world should be able to deal with such imagery.

The authors also fail to mention the computational requirements of their system. Looking at the number of weights that have to be found, and optimizations that have to be solved, one can only assume that this system requires immense computational power. Real-time image classification would probably only be possible using a very sophisticated computer system.