Support Vector Machines

Support Vector Machines use a ‘kernel trick’ to linearly separate data in a high-dimensional ‘feature space’. The separating hyperplane is expressed as a linear combination of a subset of the images of training data (the support vectors), and for problems with large sets of data the decision function becomes expensive to evaluate.

The decision function of a support vector machine is

\[ f(x) = \sum_{i=1}^{N} y_i \alpha_i K(x_i, x) + b \]

where \( K(x, y) = \phi(x) \cdot \phi(y) \) is the kernel function and \( y_i \) and \( \alpha_i \) are labels and weights of training data vectors \( x_i \).

The hyperplane of the SVM decision function is defined by

\[ \Psi = \sum_{i=1}^{N} y_i \alpha_i \phi(x_i) \]

SVM approximation

One can approximate \( \Psi \) by finding a smaller set of vectors defining

\[ \tilde{\Psi} = \sum_{i=1}^{M} \beta_i \phi(x_i) \]

so that the distance \( p = ||\Psi - \tilde{\Psi}||^2 \) is minimized.

We explored two approximation techniques. As suggested by Burges, a chosen number of vectors and their weights could be optimized using a gradient method to minimize the distance \( p \).

An alternative approach by Romdhani suggests finding a single best approximating vector with the above technique; then taking the difference between the approximation and the target and finding the second vector that best approximates this difference. Iterating the last step results in a greedy (less accurate) algorithm which however works brilliantly when building classifier cascades.

SVM cascades

In object detection, classifier cascades are a powerful tool. A row of classifiers of increasing accuracy can be biased to confidently discard non-objects and linked so that only potential matches are passed on to the next, more accurate and complex decision level.

A set of SVM approximations with 1, 2, 3, ..., \( N \) vectors works very well as a cascade. We compared cascades built using Burges and Romdhani methods. The greedy cascade, while discarding slightly less non-objects at early steps, is much cheaper to evaluate: it only adds one new vector to every level of the cascade.

We suggest a hybrid approach that combines the higher accuracy of vectors optimized in a group (Burges) with the sharing of vectors between the classifiers in the cascade (Romdhani). We optimized approximating vectors in pairs, effectively trading lower accuracy of odd levels for higher accuracy of even ones.

Results

In our experiments we built cascades for eye detecting SVM classifier that was trained on several thousands of 20x20 image patches containing human eyes. The Burges cascade was twice as slow as the Romdhani one in terms of average time per image, while our hybrid cascade showed further improvement of 9% in time taken.

On a set of 500px wide images the Burges cascade was on average twice as slow as the Romdhani one, while our hybrid cascade showed a speedup of another 9%.

---

Ignas Kukenyis
Brendan McCane