Ensembles of Strong Learners for Multi-cue Classification

Zoltan-Csaba Marton, Florian Seidel, Ferenc Balint-Benczedi, Michael Beetz

Intelligent Autonomous Systems, Technische Universität München, Munich, Germany

Abstract

Real world heterogeneous scenes contain objects of a large variety of forms, surfaces, colors and textures, thus multi-modal approaches are needed to deal with their challenges. A promising method of combining various sources of information are ensemble methods which allow on the fly integration of classification modules, specific to a single sensor modality, into a classification process. These modular and extensible approaches have the advantage that they do not require that a single method copes with every eventuality, but combine existing specialized methods to overcome their weaknesses.

In addition, the rapid growth of the perception field means that comparing, evaluating, sharing and combining the available approaches becomes increasingly relevant. In this article we describe a novel training strategy for ensembles of strong learners that not only outperform the best member but also the best classifier trained on the concatenation of features.

The method was evaluated using a large RGBD dataset containing Kinect scans of 300 objects and special use-cases are presented that highlight how ensemble learning can be used to improve classification results.

Keywords: strong learner, ensemble, multi-modal, object recognition

Email address: {marton, seidelf, balintbe, beetz}@cs.tum.edu (Zoltan-Csaba Marton, Florian Seidel, Ferenc Balint-Benczedi, Michael Beetz)

URL: http://ias.cs.tum.edu (Zoltan-Csaba Marton, Florian Seidel, Ferenc Balint-Benczedi, Michael Beetz)

1. Introduction

Perception capabilities are an important component of cognitive systems, with object recognition preceding many high-level functions for example in the area of robotics and human-machine interaction. Intelligent systems operating in human environments have a huge variety of objects to deal with, and some of them present special problems (texture-less, shiny, etc). There are multiple approaches that have been shown to be able to classify some of the objects that might appear in such settings. There are, however, inherent limitations in these approaches, and as described by Kragic and Vincze (2009), there is no robust and large-scale solution yet.

As we will discuss, these object classification tasks can be made significantly more robust if multiple sources of information are used. Similarly, surveillance tasks in smart environments could benefit from exploiting information coming from multiple modalities, or from different sources. Since each perception method captures only some aspect of the objects, the situation is similar to the old story about the six blind men trying to describe an elephant based on a single touch. Clearly, a correct combination of different sensor modalities for classification would improve results.

Combining multiple sensor modalities to improve detection can be done in general either by combining multiple features in a single classification pipeline, or by separate processing pipelines for each modality whose results are combined. The former approach is pursued by Lai et al. (2011b), where a combination of visual and depth cues is used. We explored the latter approach in (Marton et al., 2011), highlighting the limitations of the different sensors, and exploiting that not all features need to be checked if there is a subset of them that uniquely describes the object. This modular approach, with the right processing units, allowed an incremental learning of new objects, but relied on some assumptions that do not always hold.

While both approaches use different features from different sensing modalities, each of them has inherent drawbacks. Combining different features by concatenation produces very high-dimensional features that produce excessively long training (and possibly classification) times. Additionally, any change in the used features requires a complete re-training of the whole classifier. On the other hand, it is a simple way of obtaining a very accurate classifier that considers all the useful information. The sequential method is very modular, and some features can be left out if needed (e.g. those from camera images in low light conditions) or new ones added without the need

to re-train the whole system, but the errors get accumulated from each step, and correlations can not be exploited. Ideally, one would like to have a separate classifier for each feature, evaluate what it is good at and what mistakes it makes, and then produce a final classification by considering the results of all the individual ones.

This is exactly the idea behind ensemble methods in machine learning and pattern recognition, which combine a number of different models, trained for the same task, in such a way that the performance of the ensemble surpasses the performance of the best of its members. In this article we add another requirement to the ensemble, namely to outperform the classifier that uses the concatenation of the features, thus to bring the best properties of the two approaches together for robotic object recognition.

During our evaluations we found that it is quite difficult to surpass the results obtained by concatenating all the features, but that it is possible with the right training strategy and ensemble method. More specifically, we explored a novel technique for creating uncorrelated ensemble members that is substantially different from previously used methods. The technique consists of training strong learners on subsets of a set of features extracted from different sensor modalities and to combine them using either simple heuristics or stacking. This is in sharp contrast to many other ensemble methods which combine weak learners on a indivisible set of features by perturbing, for example, the data distribution by re-sampling or re-weighting.

We also gave special consideration to ensemble methods that do not need to be trained together. While this approach produces slightly inferior results, its modularity makes it especially advantageous for large systems integrating multiple sensors, where (with the improvements in hardware and addition of new sensors) novel features are being developed that need to be integrated.

Furthermore, in the area of autonomous robotics, the ability to direct gaze only with certain sensors or changing environmental conditions can have the effect that some of the sensing modalities can not be used for observing an object (e.g. low light conditions for cameras, or objects with specular, transparent or reflective surfaces for 3D sensors), thus online modularity is an additional plus.

While we focused our considerations on object recognition for robotics or distributed systems, the presented findings and the used framework can be applied in multiple fields, with the following main contributions:

- comparing different features and classifiers and testing their scalability

for multi-modal object recognition;

- evaluation of a multitude of approaches for creating classification ensembles based on strong learners, and a method for improving on the concatenation of features;

In the next section we review related approaches for multi-modal perception. Next, Section 3 briefly describes the ensemble methods used and the experimental results are presented in Section 4. We discuss our findings in Section 5 and conclude in Section 6 giving our insight for future work.

2. Related Work

In the following subsections we present related work and show the relevance of our experiments for robotic object recognition.

2.1. Multi-Modal Perception

Most of the perception systems rely either on color/black&white camera or 3D information, although image processing techniques can be applied on different image sources as well (e.g. thermal cameras). The SIFT (Lowe, 2004) descriptor, which is one of the best-known keypoint descriptors and detectors, makes use of detected keypoints in scenes and compares them with referenced objects in order to identify the objects currently being observed. In the 3D domain, the VFH (Rusu et al., 2010) descriptor was recently developed as the latest, viewpoint dependent, extension of of PFH (Rusu et al., 2008). The feature's discriminative power is increased by the inclusion of the viewpoint, which, however, also represents a deficiency in that the feature becomes orientation variant.

There are approaches which combine geometry and color descriptors into a single feature, however, properly balancing these two different properties is difficult, as discussed in (Kanezaki et al., 2011).

Inspired by earlier work based on developmental psychology (Griffith et al., 2009), object categorization using multiple modalities is explored by Sinapov and Stoytchev (2011). While the authors base their approach on psychological findings that suggest that a single sensory modality is often not enough, they leave out the most descriptive modality, vision, and focus on proprioceptive and auditory feedback (Lynott and Connell, 2009).

Lai et al. (2011b) validate the use of different visual modalities, using spin images for describing shape and SIFT and texton histogram features to

capture the visual appearance. They showed that color-based cues are more important for instance recognition, while geometric ones are better suited for categorization, and that their combination improves on both. This finding is also supported by Sun et al. (2010).

2.2. Ensemble Learning

While single classifiers, or sequential applications of them are what was typically used for object recognition so far, a combination of different classifiers and features using ensemble learning seems like a promising approach as it has been successfully applied to challenging machine learning problems in other domains.

Lam and Suen (1995) investigated how to optimally combine classifiers for character recognition and found that simple voting is often the most robust choice. They found, that the combination methods which are trainable exhibited better performance than voting on the partition of the dataset on which they were trained, but performed worse on a second partition of the dataset.

An approach which has been particularly successful in the Netflix contest (Sill et al., 2009) is stacking, introduced by Wolpert (1992). In stacking the outputs of so called level-0 classifiers and also meta-data is combined to form the input for classifiers on level-1 which are trained to improve generalization accuracy on a validation set.

Madry et al. (2011) evaluated several linear and non-linear methods for combining classifiers trained on shape and appearance features. They used SIFT (on grayscale and the opponent color channels) and histograms of oriented gradients (HOG) for 2D appearance and 2D shape, together with the Fast PFH version for 3D shape. As base classifiers they used (multi-class) support vector machines (SVM) with χ^2 kernel. The ensemble methods evaluated were the max confidence rule, the product rule using confidences and the confidence weighted voting rule. They also tested stacking using SVM with the RBF kernel, the histogram intersection kernel and the χ^2 kernel trained on the confidences of the base classifiers. On a dataset of 11 object categories (10 objects each, 16 views per object), they found that the voting rule outperforms the other rules as well as the SVM.

In our experiments we tested a large number of scans falling into 51 categories, and evaluated a large number of base classifiers, simple rules, voting and stacking methods. Apart from considering only the base classifier confidences, we also used a separate partitioning of the data to evaluate their

performance, and used the obtained accuracies as weight factors as well. This accuracy weighted voting was found to be the best ensemble method among the simple combination rules, but in our large scale experiments it didn't outperform the stacking approach.

3. Ensemble Methods for Multi-modal Object Recognition

In this section we will present the ensemble methods we implemented and evaluated to find an alternative to concatenating features. As Figure 1 summarizes, the key idea is to create separate classifiers for different features and to combine their output for the final classification.

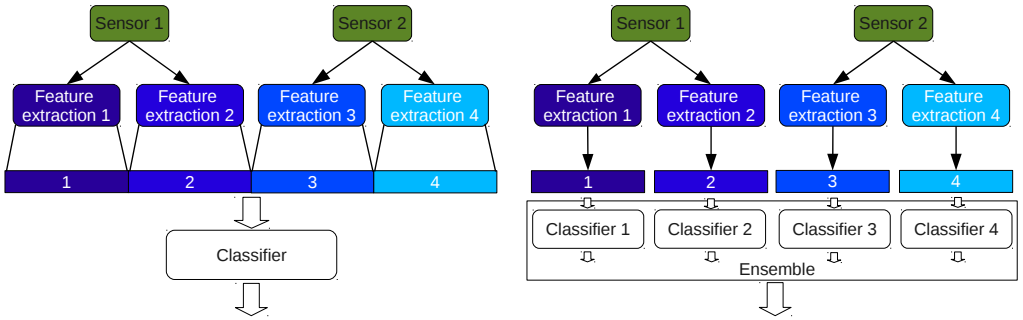


Figure 1: Technique of concatenation (left) compared to ensembles (right).

Fumera and Roli (2005) analyzed linear combinations of classifiers and found that the performance depends on the correlation of the outputs of the ensemble members. This is intuitively clear, since if two classifiers often make the same prediction little additional information is gained by observing the output of both. This is obviously also true for all other ensemble methods.

A lot of the best performing ensemble methods have some scheme for re-weighting or re-sampling the dataset for training the ensemble members to achieve this lack of correlation. One example is bagging, which stands for bootstrap aggregation, in which a random subset of the dataset is used for training each member. Another method is the popular AdaBoost in which the classifiers are trained in sequence and the training data is re-weighted for training a new classifier in the sequence. When using neural networks the random initialization of the weights leads to different local minima and it is then possible to combine these networks to committees.

In the next subsections we present the various ensemble methods we used grouped into four categories, starting with the simpler "max rules" and finishing with stacking.

We implemented the various feature extraction modules and classifiers in C++, using the communication interface of the Robot Operating System (`ros.org`), where standard service requests were defined that deal generally with a large number of classifiers. Wrapping other existing libraries is on our agenda, but up to now we focused on integrating LIBSVM (Chang and Lin, 2011), FLANN (Muja and Lowe, 2009), and one-against-all boosting approaches based on decision trees from OpenCV (Bradski, 2000). We will make the code and examples available on the project homepage at:
`http://code.in.tum.de/indefero/index.php/p/ias-cf/`.

The accuracy of the classifiers are estimated from a validation partition of the dataset. In some of the rules it is quite likely to get a tie between two decisions. The ties are broken by making a random decision. To keep things simple this is not stated explicitly for each rule for which it applies.

3.1. Max and Product Rules

The max accuracy rule is to choose the classifier whose classification decision has the highest class-conditional classification accuracy. Similar to this, the max confidence rule is to choose the classifier which has the highest confidence in it's decision. The combined max confidence and accuracy rule is to choose the classifier whose classification decision has the highest class conditional classification accuracy multiplied by the classifiers confidence. These are simple ways of combining classifiers that do not require a common training and consider the result of only one selected classifier. Another well known method we explore that does not need optimization of parameters but does merge all the available information is the product rule.

3.2. Weighted Voting

The weighted votes method assigns an additional weight to every classifier. We experimented with various voting approaches that do not require combined training and are thus modular. In accuracy weighted voting each classifiers vote counts with with the accuracy of the classifier. Confidence weighted voting is equivalent to simple voting with the addition of having a vote count with the classifiers confidence. Confidence and accuracy rated voting is equivalent to simple voting with the addition of having a vote count with the classifiers confidence multiplied with it's class conditional accuracy.

3.3. Error Correlation

For regression problems it is possible to find a closed form solution for the weights of a linearly combined ensemble of classifiers if the expected sum of squares loss is to be minimized. Bishop (2006) analyzes this approach and a closed form solution is given and shows that for all convex loss functions this method's expected performance is higher than simple averaging and can not produce an increase in the expected error. He discusses the topic in the context of regression but it is also possible to use this approach for classifier ensembles. The derivation goes through unchanged for the classification case.

3.4. Stacking

Stacking or stacked generalization as it was originally called by Wolpert (1992) is an ensemble method in which classifiers are stacked to achieve improved classification performance. Stacking classifiers is done by collecting the outputs of so called level-0 classifiers into a new dataset and to train one or more level-1 classifiers on the outputs of the level- 0 classifiers. When doing this it is of importance to build the dataset on which the level-1 classifiers will be trained from training data that wasn't used to to train the level-0 classifiers, since what we are interested in is to improve the performance on the test set, not on the training set, in short to improve generalization.

This is usually done by separating the dataset into n folds, removing one, training one of the classifiers on the remaining folds and evaluating the classifier on the fold which was left out. This is repeated for all the folds. Together with the labels this process creates the level-1 training dataset, which has the same size as the level-0 training set. Formally, the level-0 training set D_0 is split up into $m = |E|$ equally sized folds $P_1, .., P_m$. By P_{-i} we denote $D \backslash P_i$. Each $e_i \in E$ is trained on P_{-i} and evaluated on P_i to yield a part of D_1, the level-1 training set. Then the level-1 classifier is trained on D_1. The accuracy of the stack is evaluated on a separate test dataset. Stacking is illustrated schematically in Figure 2.

The procedure described above is the classical formulation. We take a slightly different approach which is motivated by the abundance of data we have, the multi-modal nature of our features and the goal to easily try feature combinations without having to retrain a classifier on the combined feature vector every time. Let $D_1, ..., D_N$ be N different datasets obtained from N different feature extraction processes. The datasets are split up into three parts each (D_i^0 to D_i^2), the level-0 training sets, level-0 evaluation sets and level-1 evaluation sets. The classifiers $e_i \in E$ with $|E| = N$ are trained on D_i^0

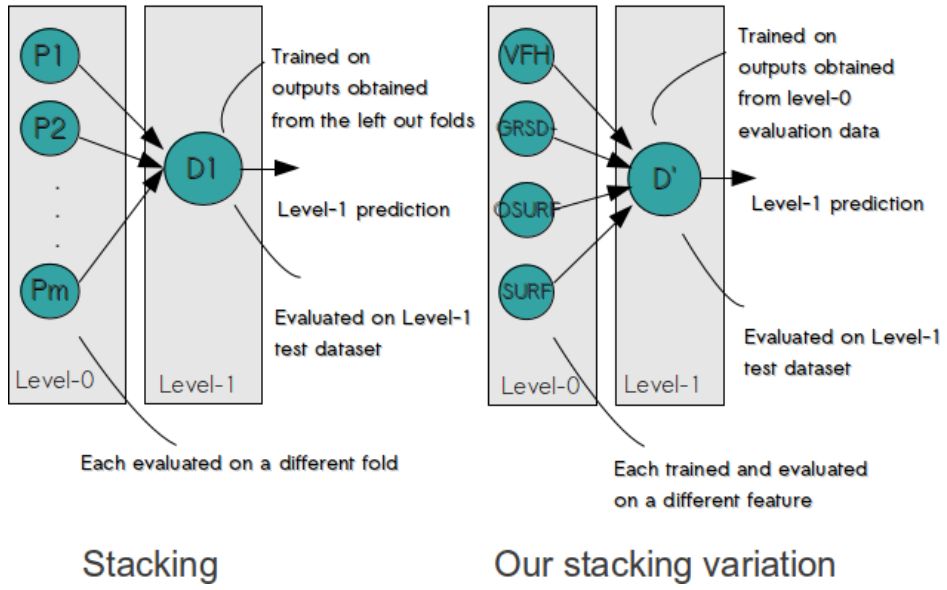


Figure 2: Original stacking compared to our stacking version.

and evaluated on D_i^1 to produce D_i' the level-1 training-set parts which are combined to form D' the full level-1 training-set on which a level-1 classifier is then trained. The whole stack is then evaluated on the D_i^2 datasets.

So far we only spoke of the "outputs" of the classifiers without defining them more precisely. Ting and Witten (1997) evaluated stacking on four different classifiers and their combinations and found that for stacking to work the level-0 classifiers have to output a confidence rating for each label. Since our AdaBoost multi-class version outputs confidence ratings we were able to use it as a level-0 classifier. The SVM implementation we use is also capable of training a probability model, unfortunately the training process is very time consuming and so we were not able to try SVMs as level-0 classifiers.

4. Experimental Results

The following subsections present the results we got during the evaluation of the presented method. The results will be discussed in the next section.

4.1. Dataset preparation and feature extraction

The full dataset presented in Lai et al. (2011a) contains roughly 250,000 scans of the 300 objects organized in 51 categories[1]. This is a lot of data and handling datasets of this size poses a challenge in itself. Therefore, like the original authors as well, we decided on using only every 5th sample in the dataset. This still leaves us with about 1 day of extraction time for the whole dataset, but it assures that enough views of the objects are considered in order to make a proper evaluation.

After extracting the VFH, GRSD-, SURF and OpponentSURF features, the Bag of Words (BoW) model is used to create a global descriptor for the objects when using OpenCV features. Then the datasets were split up further by removing every second instance to obtain the training partitions of the dataset. Of the removed instances every second was used to form a validation set for the level-0 classifiers and the remaining instances were used to form a validation set for the level-1 classifiers. Table 1 contains the number of instances for each partition of the datasets.

We use PCL (Rusu and Cousins, 2011) to extract the VFH and GRSD- features. The point clouds in the dataset are very dense so to speed up normal estimation the point clouds were down-sampled using a voxel grid filter with 0.001 meters grid size. The local neighborhood for estimating the normals is determined using a 0.02 meter radius search. In our experiments we use the OpenCV 2.2 implementation of SURF. We detect features using the dynamic SURF detector, an extension which lowers the detection threshold until a sufficient number of points have been found. We also scaled each dimension of the VFH and GRSD- features to the range $[0, 1]$ by subtracting the minimum value along a dimension and dividing by the maximum along that dimension (determined after subtracting the minimum). The SURF BoW feature vectors were linearly scaled so make the bins of the histogram sum to one.

As a next step the datasets were concatenated to yield six datasets containing the concatenations of each group of two features, four datasets containing the combinations of each group of three features, and one combination containing concatenations of all four features. We call the single features (VFH, GRSD-, SURF and OpponentSURF) "singles", the concatenations of two features "doubles", of three features "triples", and of all features "all"

[1] `http://www.cs.washington.edu/rgbd-dataset/`

Table 1: Number of instances in the datasets containing 20 classes (left) and all of the 51 classes (right).

Number of classes	20
Training instances	3600
# eval level-0	1800
# eval level-1	1800
Total scans	7200

Number of classes	51
Training instances	20738
# eval level-0	10369
# eval level-1	10369
Total scans	41476

in the following.

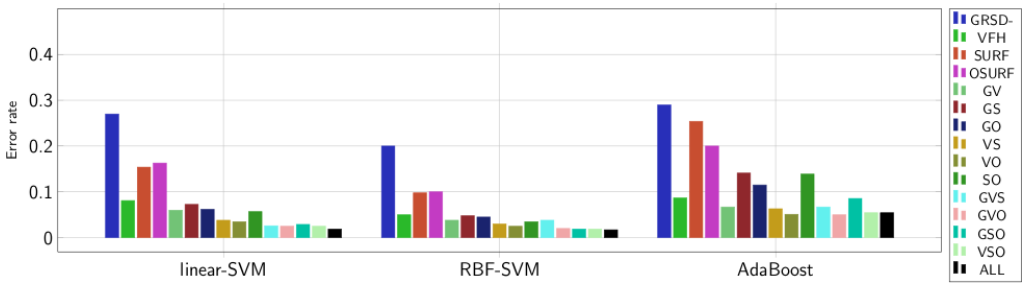
In the next subsections we present the comparisons of results obtained by the different classifiers and ensembles, and consider the simple features as well as their combinations. As we will discuss in Section 5, outperforming the classifier trained on the concatenation of all the features requires the use of pairwise combinations even with the most complicated ensemble method. Nonetheless, this solution leaves us with a relatively good modularity, as the cost of adding a new feature to the list of level-0 classifiers is linear in the number existing ones (training the pairwise combinations), and only the level-1 classifier has to be re-trained.

4.2. Evaluation of Features and Classifiers

Throughout the paper we will present performance using the error rate, i.e. the fraction of object scans to which a false label was assigned. As shown in Figure 3 AdaBoost is not very good with the concatenations and/or is very bad with the BoW model. Regarding the SVM classifier, the linear kernel is as good as the RBF kernel for high dimensional features. This implies that AdaBoost is inferior to SVM. For the 51 class problem the accuracy is roughly halved for both methods, but SVM scales marginally better. On the other hand, as it can be seen from the classification times, the AdaBoost classifier can be trained in less time and the time for classification with AdaBoost is independent of the feature length. These findings were to be expected given the generally known behaviors of the used classifiers. Therefore, for some tests, using SVM with RBF kernel was attempted, but was found to be prohibitively time-consuming and was left out.

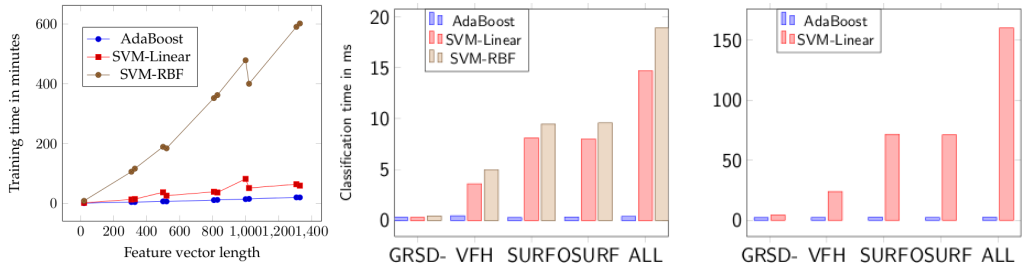
4.3. Simple Ensemble Methods

We tested the simple ensemble methods described in Section 3 using several feature-classifier combinations, as shown in Figures 4 and 5. The pro-



(a) Error rate for the 20 classes problem

(b) Error rate for the 51 classes problem



(c) Feature vector length vs. training and classification time (for 20 classes)

Figure 3: Results using AdaBoost compared to those using SVM.

vided error rate for the concatenation and the best member being the error rates achieved with the respective classifier. Due to time constraints we did not train a probability model for the SVM and consequently all the ensemble methods involving confidences were left out.

4.4. Stacking Compared to Simple Ensembles

In this experiment we evaluate performance of our variation of stacking as described in subsection 3.4. We used Real AdaBoost as level-0 classifiers and Real AdaBoost, LogitBoost and Gentle Boost as well as a linear SVM

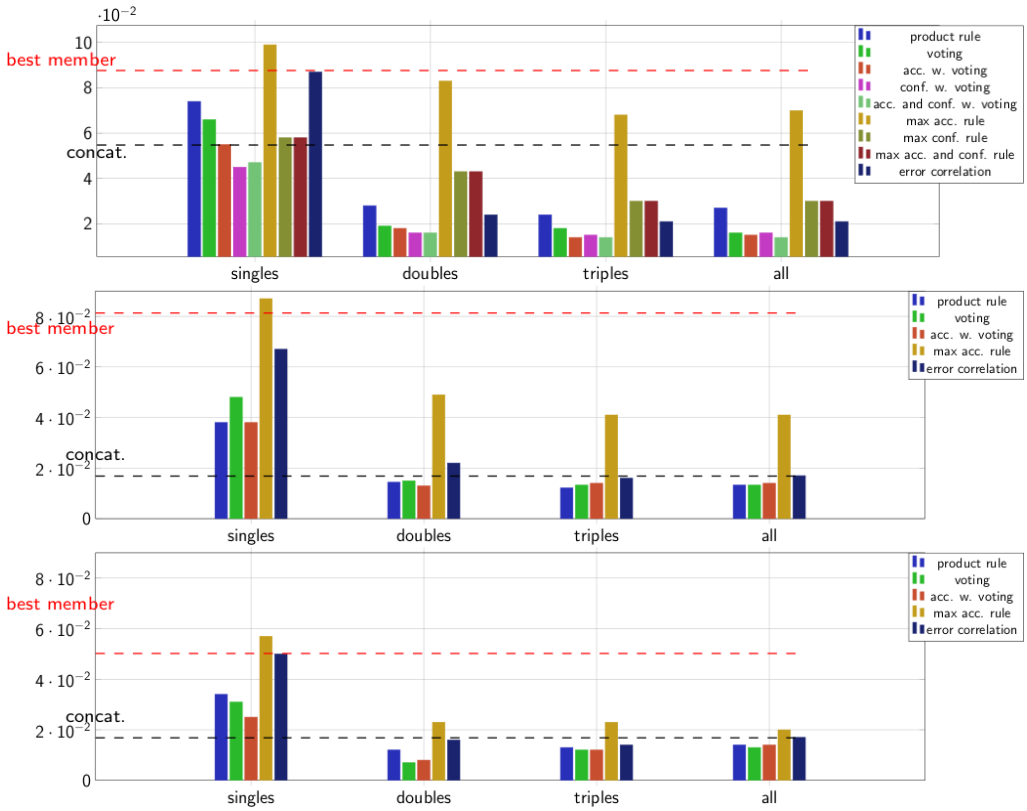


Figure 4: Error rate of all simple methods on the 20 classes problem with (from top to bottom) AdaBoost, linear SVM and SVM with RBF kernel as base classifier

and a SVM with Radial Basis Function kernel as level-1 classifiers. As for the simple ensemble methods we combine level-0 classifiers trained on the singles, doubles and triples for the 20 classes problem. For the 51 classes problem we combined the singles and doubles only due to excessively long run times. The results are shown in Figure 6, with the provided "concat." error rate as the best error rate achieved using the concatenation approach.

5. Discussion

To establish a baseline to which we can compare the performance of the ensemble methods we evaluated a small number of classifiers (linear-SVM, RBF-SVM and AdaBoost) on a number of features, which capture different aspects of the objects, and their concatenations. From the experiments

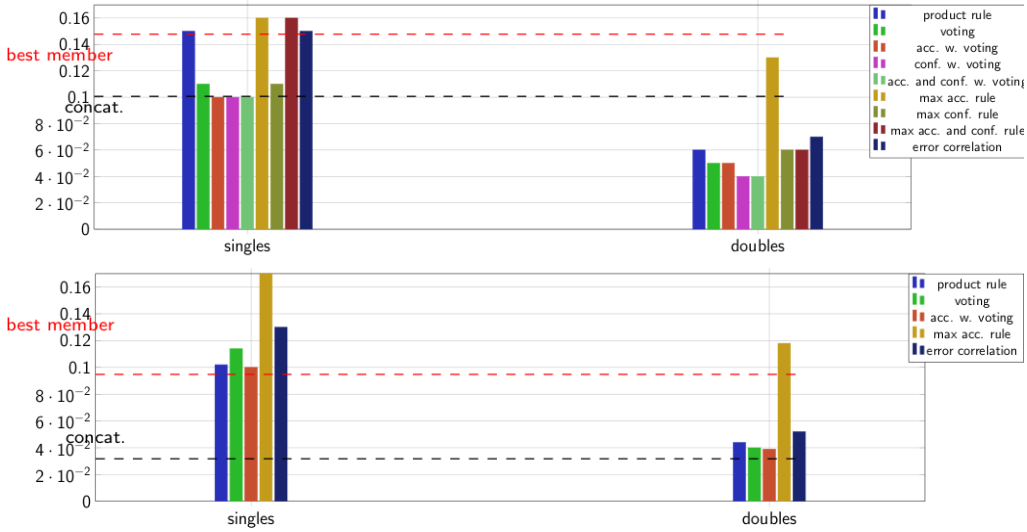


Figure 5: Error rate of all simple methods on the 51 classes problem with AdaBoost (top) and linear SVM (bottom) as base classifier

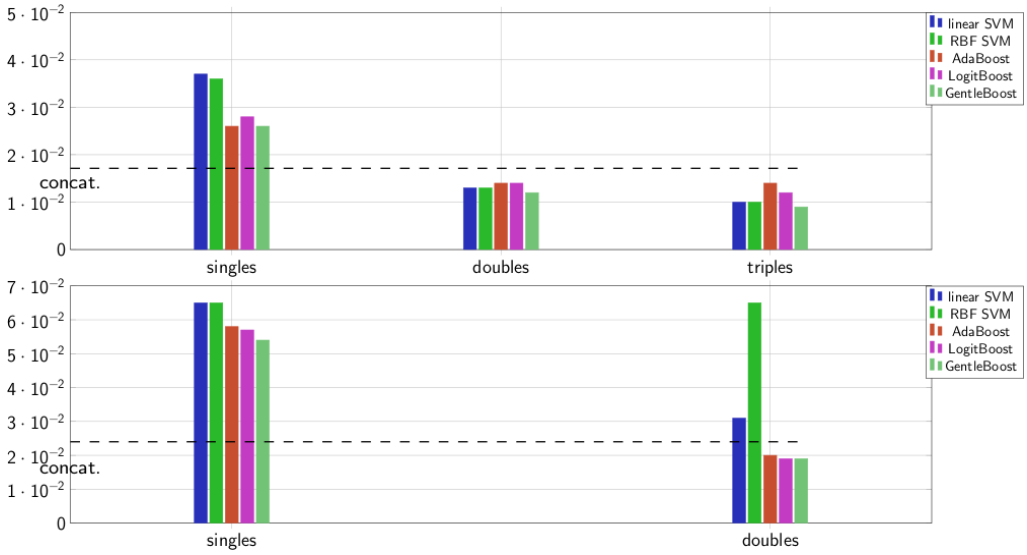


Figure 6: Stacking with AdaBoost as level-0 classifier and various level-1 classifiers, for 20 classes (top) and 51 classes (bottom).

we conclude that VFH is the most descriptive feature, among the features considered, for the task at hand. Not surprisingly, the accuracy of all the

classifiers and the time required for training increases with the number of features used. Furthermore, the direct comparison of the classifiers shows, that the SVMs can utilize the information contained in the diverse features better than AdaBoost and also that the linear-SVM approaches the RBF-SVMs accuracy for concatenations of features of increasing length – as suggested also in (Chang and Lin, 2011). Regarding the training time, we see an approximately linear increase with feature vector length for all classifiers.

In every ensemble method the goal is to produce a number of uncorrelated classifiers and the experiments described in the previous section provide ample evidence to suggest that our feature recombination method leads to uncorrelated classifiers. Most noteworthy is the significant increase in accuracy obtained over the single feature ensembles by the ensemble whose members were trained on two element subsets of the feature set. In virtually all cases for the 20 classes problem this setup was able to improve even on the classifiers trained on the concatenation of all the features, and for the 51 classes problem at least the stacking approaches were able to achieve this goal. However, adding further classifiers trained on larger subsets of the feature set to the ensemble did not lead to much improvement in most cases and to decreased accuracy in some cases. As shown in Figure 7 the results of the concatenated features were almost reached already by stacking the single original features.

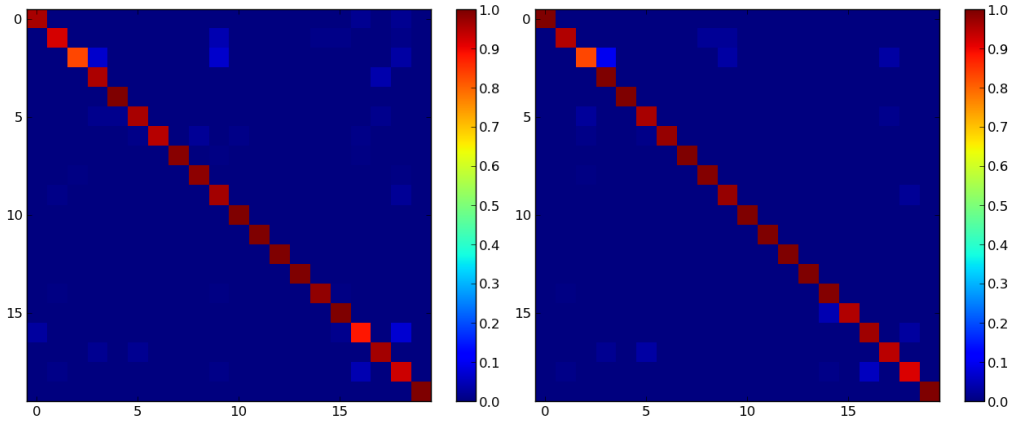


Figure 7: Confusion matrices for the AdaBoost ensemble (left) as compared to the classifiers of the concatenation of all features with a linear SVM classifier (right) for the 20 classes problem. Best viewed in color.

One important advantage of the simple rule ensembles over monolythic

classifiers and stacking is their modularity which, as we could show, comes at the cost of decreased accuracy. The hybrid model of the simple rule ensembles trained on two element subsets, however, is an approach balancing the conflicting goals of accuracy and modularity while incurring only an insignificant increase in the time needed for training. It therefore presents an attractive addition to the machine learning toolchest for applications which can take advantage of multiple data sources. In particular, accuracy and confidence based voting approaches seem to be the method of choice as they deliver the highest accuracy among the simple rules.

6. Conclusions and Future Works

All in all, we can say that multi-modal and multi-featured object classification has a clear advantage over single-featured one. In contrast to our earlier work where a sequential approach was used (Marton et al., 2011), first considering geometric, then texture based features, here all modalities are considered together. Ensemble methods are a viable way of combining the various sources of information, and they allow for a higher modularity and efficiency than simple concatenations of features. As we saw, with the right approach, we can improve not only over the best ensemble member, but also the concatenation.

There is growing evidence that human vision combines top-down (concept driven) and bottom-up (data driven) approaches (Frisby and Stone, 2010), thus extending classification systems with context information is a natural way of increasing performance. Similarly, geometric verification can help to reject at least some of the false positives, as discussed in (Mozos et al., 2011). Additionally, in (Marton et al., 2011) we reported on the improvement in accuracy of over 10% when geometric categorization is allowed to work with "internal" categories. This suggests that an unsupervised classification level followed by a mapping to human-defined labels, similarly as in (Mozos et al., 2011), would enable the classifiers to tune themselves to the specific feature space used.

Therefore we see the integration of all these different system into a single one as the next important step towards the realization of a general and robust object object detection and recognition system that is capable to reliably deal with a large number of objects and environmental conditions.

Acknowledgments

This work was supported by the DFG cluster of excellence CoTeSys (Cognition for Technical Systems) at the Technische Universität München; and by the PR2 Beta Program of Willow Garage, Menlo Park, CA. The authors would like to thank Kevin Lai for his help with the dataset.

References

Bishop, C. M., 2006. Pattern Recognition and Machine Learning. Springer.

Bradski, G., 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools.

Chang, C.-C., Lin, C.-J., 2011. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Frisby, J. P., Stone, J. V., 2010. Seeing: The Computational Approach to Biological Vision. MIT Press, Ch. 8: Seeing Objects, pp. 178–179.

Fumera, G., Roli, F., 2005. A theoretical and experimental analysis of linear combiners for multiple classifier systems. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 942–956.

Griffith, S., Sinapov, J., Miller, M., Stoytchev, A., 2009. Toward interactive learning of object categories by a robot: A case study with container and non-container objects. In: IEEE 8th International Conference on Development and Learning (ICDL).

Kanezaki, A., Marton, Z.-C., Pangercic, D., Harada, T., Kuniyoshi, Y., Beetz, M., September, 25–30 2011. Voxelized Shape and Color Histograms for RGB-D. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Workshop on Active Semantic Perception and Object Search in the Real World. San Francisco, CA, USA.

Kragic, D., Vincze, M., 2009. Vision for robotics. Foundations and Trends in Robotics 1 (1), 1–78.

Lai, K., Bo, L., Ren, X., Fox, D., 2011a. A large-scale hierarchical multi-view rgb-d object dataset. In: Proc. of International Conference on Robotics and Automation (ICRA).

Lai, K., Bo, L., Ren, X., Fox, D., 2011b. Sparse distance learning for object recognition combining rgb and depth information. In: Proc. of International Conference on Robotics and Automation (ICRA).

Lam, L., Suen, C. Y., 1995. Optimal combinations of pattern classifiers. Pattern Recognition Letters 16 (9), 945–954.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60, 91–110.

Lynott, D., Connell, L., 2009. Modality exclusivity norms for 423 object properties. Behavior Research Methods 41 (2), 558–564.

Madry, M., Song, D., Kragic, D., April 8 2011. 2D/3D Object Categorization for Task Based Grasping. In: RGB-D Workshop on 3D Perception in Robotics at the European Robotics (euRobotics) Forum. Västerås, Sweden.

Marton, Z. C., Pangercic, D., Blodow, N., Beetz, M., 2011. Combined 2D-3D Categorization and Classification for Multimodal Perception Systems. The International Journal of Robotics Research.

Mozos, O. M., Marton, Z. C., Beetz, M., 2011. Furniture Models Learned from the WWW – Using Web Catalogs to Locate and Categorize Unknown Furniture Pieces in 3D Laser Scans. Robotics & Automation Magazine 18 (2), 22–32.

Muja, M., Lowe, D. G., 2009. Fast approximate nearest neighbors with automatic algorithm configuration. In: International Conference on Computer Vision Theory and Application VISSAPP'09). INSTICC Press, pp. 331–340.

Rusu, R. B., Bradski, G., Thibaux, R., Hsu, J., October 2010. Fast 3d recognition and pose using the viewpoint feature histogram. In: Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Taipei, Taiwan.

Rusu, R. B., Cousins, S., May 9-13 2011. 3D is here: Point Cloud Library (PCL). In: IEEE International Conference on Robotics and Automation (ICRA). Shanghai, China.

Rusu, R. B., Marton, Z. C., Blodow, N., Beetz, M., 2008. Learning Informative Point Classes for the Acquisition of Object Model Maps. In: Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision (ICARCV), Hanoi, Vietnam, December 17-20.

Sill, J., Takács, G., Mackey, L., Lin, D., 2009. Feature-weighted linear stacking. CoRR abs/0911.0460.

Sinapov, J., Stoytchev, A., 2011. Object category recognition by a humanoid robot using behavior-grounded relational learning. In: IEEE International Conference on Robotics and Automation (ICRA).

Sun, M., Bradski, G., Xu, B.-X., Savarese, S., 2010. Depth-encoded hough voting for joint object detection and shape recovery. In: Proceedings of the 11th European conference on Computer vision: Part V. ECCV'10. Springer-Verlag, Berlin, Heidelberg, pp. 658–671.

Ting, K. M., Witten, I. H., 1997. Stacked generalization: when does it work? In: in Procs. International Joint Conference on Artificial Intelligence. Morgan Kaufmann, pp. 866–871.

Wolpert, D. H., 1992. Stacked generalization. Neural Networks 5, 241–259.