



Classifier Combining

APR Course, Delft, The Netherlands
 Marco Loog



Multiple Multiples

- Multiple Classifiers
- Multiple Representations
- Multiple Sensor Sets

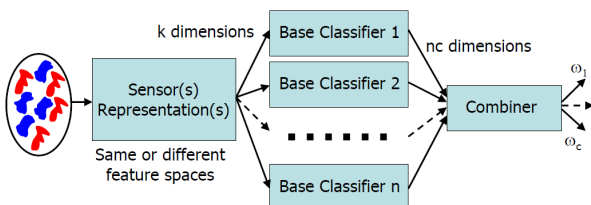


The Basic Questions

- How to reach a committee decision?
- How to design a combiner?
- How to constitute a committee?
- How to generate base classifiers?



Combining Architecture



Part I

The Combiner



Combiner Types

- Fixed rules based on crisp labels or confidences [estimated posterior probabilities]
- Special trained rules based on classifier confidences
- General trained rules interpreting base-classifier outputs as features

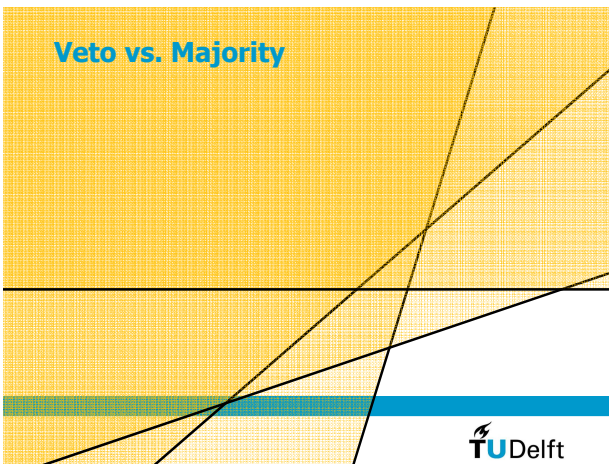


Fixed Combining Rules

- Object is assigned to class ω_i if combination of outcomes y_{ij} for class ω_i over all classifications $y_{ij} = S_j(x)$ is maximum
- Example combiners
 - Using labels : Voting, veto, majority
 - Using posteriors : Product, minimum, sum, mean, median, maximum, percentiles, etc.
- E.g. decision forests



Veto vs. Majority



Non-probabilistic Posteriors

- E.g. How do we get posterior estimates out of a support vector machine?
- General classification rule S may just output $S(x) > 0$ for class A and $S(x) < 0$ for class B
- Fit a logistics function / sigmoid
- `classc`



Combining of Confidences

- Product rule [`prod`]
 - Similar to logical AND
 - Experts should agree
- Minimum rule [`min`]
 - Assign according to least objecting expert
 - Often similar behavior as product rule
- Mean rule [`mean`], median rule [`median`]
 - Improvements by averaging out noise in experts
- What about sum rule?
- And majority voting?



Possible "Derivation" of Product Rule

- Assume independence of feature spaces... given class label
- $P(f, \phi | \omega) = P(f | \omega)P(\phi | \omega)$



[Further] Rules of Thumb

- Product, minimum
 - Independent feature spaces
 - Different expertise areas
 - Posteriors should be well estimated
- Sum, mean, median, majority
 - Equal posterior estimation in same feature space
 - Differently trained classifiers; based on same distribution
 - Bad behavior if some classifiers very good or very bad
- Maximum
 - Relies on most confident classifier ["shouts the loudest"]
 - Bad behavior if classifiers are [for instance] overtrained



Posteriors?

- How to turn output of combiner into posteriors?

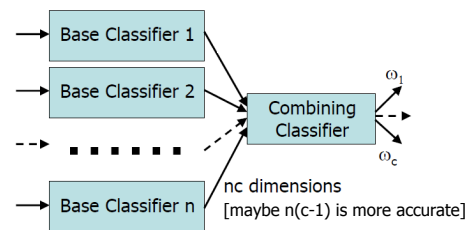


Suboptimality of Fixed Rules

- But surely the assumptions do not hold...



Trained Combiners

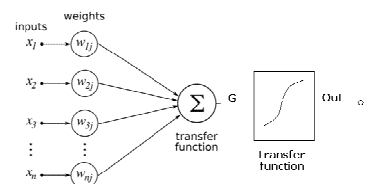


Common Trained Combiners

- Special trained combiners
 - Decision templates [parallels with NMC]
 - Behavior-knowledge space
 - Dynamic classifier selection
 - Error correcting output coding
- General classifiers
 - Nearest mean classifier
 - Fisher
 - Decision trees
 - Etc.



Something on ANNs?



Decision Templates

- Decision templates are average outcomes of base classifiers per class training set
- Assign new objects to class of nearest decision template in base-classifier outcome space



Error Correcting Output Coding

- ECOC uses small set of binary classifiers for large set of c classes
- n classifiers can distinguish at most 2^n classes
- If $n > \log_2(c)$ the system of classifiers is more robust
- ECOC studies mainly discuss coding scheme, not the way base classifiers are trained
- Combining is done by using crisp 0/1-labels



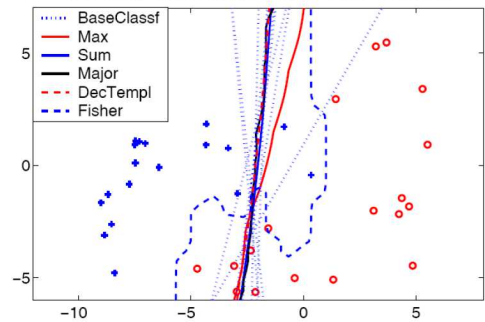
Stacked Generalization

- A procedure to combine L classifiers
 - Do N -fold cross validation to estimate L posteriors [or labels]
 - This constitute training set for combiner
- C'est tout...



E.g.1

Combining 10 Bootstrapped Nearest Mean Classifiers



E.g.2

On Combining Computer-Aided Detection Systems

Michael Noolen¹, Mathis Leij², Michael David Abraham, Gabor Mészáros, IEEE, Max A. Viergever, *IEEE, ISBI, MIPSP, MICCAI*, Mathis Prokop, and Bram van Ginneken, *IEEE*

Abstract. Computer-aided detection (CAD) is increasingly used to detect lesions and for many applications a multitude of CAD systems have been proposed. To predict CAD system performance different strategies and evaluation and it is therefore interesting to consider their combination. In this paper, we present a novel method to combine multiple CAD systems and investigate what kind of performance increase can be expected. Experimental results are presented using data from the ANKOR100 and ROC-IPW cancer CAD challenge for the detection of pulmonary nodules in ground truth CT scans and chest lesions in breast mammography. In both applications, combination results in a large and significant increase in performance when compared to the best individual CAD system.

Index Terms. Combination, computer-aided detection (CAD), lung, lung nodules, and breast nodules.

1. INTRODUCTION

COMPUTER-AIDED DETECTION (CAD) is increasingly used to detect lesions to assist physicians in the location of subtle abnormalities. A wide range of commercial systems are on the market, each specializing in a particular application area. Many CAD systems are aimed at the detection of breast, lung, or colorectal cancer with X-ray, computed tomography (CT) or magnetic resonance imaging [1]. The design of a CAD system is a difficult and challenging task. Involving many choices regarding image preprocessing, candidate detection, feature extraction, and classification strategy. Moreover, the utilized training data, often proprietary, has a large influence on CAD behavior and performance. In most applications, ANKOR100 contest [2], ISBI 2011 contest [3], MICCAI 2011 contest [4], and ISBI 2012 contest [5].

It is therefore interesting to consider the combination of multiple CAD systems. In this paper, we present a novel method to combine multiple CAD systems and investigate what kind of performance increase can be expected. Experimental results are presented using data from the ANKOR100 and ROC-IPW cancer CAD challenge for the detection of pulmonary nodules in ground truth CT scans and chest lesions in breast mammography. In both applications, combination results in a large and significant increase in performance when compared to the best individual CAD system.

In past reviews can be found in the paper's title on <http://www.scopus.com>.

E.g.3



Multiple Use of Training Sets

- Can one reuse training sets both for training base classifiers and combiner?
 - Depends on undertraining, well trained, or overtrained base classifiers



Part II

Base Classifiers Construction



Three Ways to Generate

- Random subspace approach
- Bagging
- Boosting



Random Subspace Approach

- Select dimensionality $k' \ll k$ that fits well with training set size
- Select at random n subsets of k' features
- Train n classifiers
- Combine



Bagging [Bootstrap Aggregating]

- Select a training set size $m' < m$
- Select at random n subsets of m' training objects [originally : bootstrap]
- Train a classifier [originally : decision tree]
- Combine [original: majority vote]
- Stabilize volatile classifiers



Boosting

- Initialize all objects with an equal weight
- Select a training set size $m' < m$ according to the object weights
- Train a weak classifier
- Increase the weights of the erroneously classified objects
- Repeat as long as needed
- Combine
- Improve performance of weak classifiers

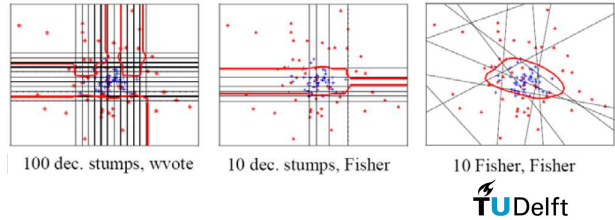


Adaboost Algorithm

1. Sample training set according to set of object weights [initially equal]
2. Use it for training simple [weak] classifier ω_i
3. Classify entire data set, using weights, to get error estimate ϵ_i
4. Store classifier weight $\alpha_i = 0.5 \log((1-\epsilon_i)/\epsilon_i)$
5. Multiply weights of erroneously classified objects with $\exp(\alpha_i)$ and correctly classified objects with $\exp(-\alpha_i)$
6. Goto 1 as long as needed
7. Final classifier : weighted voting with weights α_i



Adaboost Example



Boosting Observations

- Resampling strategy
 - Boosting principle may work for more difficult data sets
- Base classifiers
 - Use of weak base classifiers may be improved by stronger classifiers
- Combiner
 - Weighted voting performs well
 - Still, trained Fisher combiner does better than weighted voting for small sets of base classifiers



Conclusions?

