Combining Classifiers

Outline

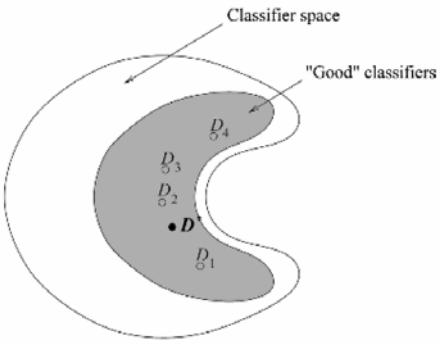
- Motivations
- Taxonomies
- Fusion of Label Outputs
- Fusion of Continuous-Values Outputs
- Classifier selection
- Bagging and boosting

Warnings and Motivations

- T.K. Ho (Multiple classifier combination: Lessons and the next steps, 2002)
  - Instead of looking for the best set of features and the best classifier, now we look for the best set of classifiers and then the best combination method. One can imagine that very soon we will be looking for the best set of combination methods and then the best way to use them all. If we do not take the chance to review the fundamental problems arising from this challenge, we are bound to be driven into such an infinite recurrence, dragging along more and more and more complicated combination schemes and theories and gradually losing sight of the original problem.
- 'No Panacea Theorem' for classifier combination (R. Hu, R.I. Damper – Pattern Recognition, 2008)
- Motivations: T.G. Dietterich (Ensemble methods in machine learning, 2000)
  - Statistical
  - Computational
  - Representational

Motivations

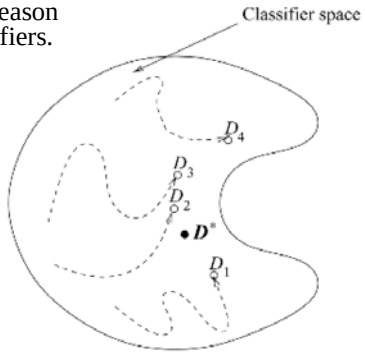
- The statistical reason for combining classifiers.



- D^* is the best classifier for the problem, the outer curve shows the space of all classifiers; the shaded area is the space of classifiers with good performances on the data set.

Motivations

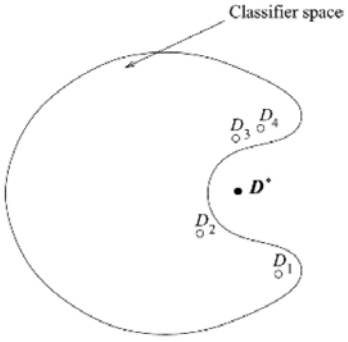
- The computational reason for combining classifiers.



- *D** is the best classifier for the problem, the closed space shows the space of all classifiers, the dashed lines are the hypothetical trajectories for the classifiers during training.

Motivations

- The representational reason for combining classifiers.

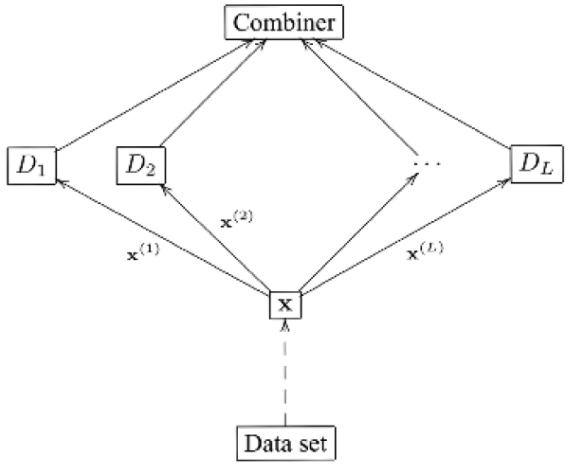


- *D** is the best classifier for the problem; the closed shape shows the chosen space of classifiers.

Early and related works

- In the field of pattern recognition, the idea of combining multiple classifiers appeared during the last decade under many names: hybrid methods, multiple experts, mixture of experts, cooperative agents, classifier ensembles, multiple classifier systems, etc.
- Earliest works are the ones of Dasarathy and Sheela (1978) and Rastrigin and Erenstein (1981)
- A seminal paper is the one by Xu et al. (1992)

Approaches to building classifier ensembles



A. *Combination level*: Design different combiners.

B. *Classifier level*: Use different base classifiers.

C. *Feature level*: Use different feature subsets.

D. *Data level*: Use different data subsets.

Taxonomies

- Fusion and Selection
  - cascaded classifiers (serial combination)
- Decision optimization and Coverage optimization
- Trainable and Nontrainable Ensembles
  - data dependent ensemble
    - implicitly
    - explicitly (selection also…)

To train or not to train?

- **Large data sets**
  - train a single (possibly complex) classifier
  - train the base classifiers on nonoverlapping training sets
  - use the pasting-small-votes idea
  - evaluate the ensemble and the single classifiers very precisely (using a large testing set) so as to be able to decide what to use in practice.

To train or not to train?

- **Small data sets: Duin's tips**
  - If a single training set is used with a *nontrainable* combiner, then make sure that the base classifiers are not overtrained.
  - If a single training set is used with a *trainable* combiner, then leave the base classifiers undertrained and subsequently complete the training of the combiner on the training set.
  - Use *separate* training sets for the base classifiers and for the combiners. Then the base classifiers can be overtrained on their training set. The bias will be corrected by training the combiner on the separate training set.
    - the second training set, on which the ensemble should be trained, may be partly overlapping with the first training set used for the individual classifiers.

To train or not to train?

- **Idea of Stacked Generalization**
  - Split the dataset in 4 folds ($A \cup B \cup C \cup D = \mathbf{Z}$)
  - Train each classifier (say D_1, D_2, D_3) according to a standard four-fold cross-validation process
  - The combiner is trained on a data set of size N obtained in the following way. For any data point in subset A, we take the outputs for that point from the versions of D_1, D_2, and D_3 built on (BCD). The three outputs together with the label of the point form a data point in the training set for the combiner.
  - After the combiner has been trained, D_1, D_2, and D_3 are retrained, this time on the whole of $\mathbf{Z}$.
  - The new classifiers and the combiner are then ready for operation.