Name: Dong Cao

Date: August 21, 2021

CS 388

Annotated Bibliography

Learning from consensus in the test set

Basic idea: For each conflict data point (i.e. where the classifiers in the ensemble disagree), combine the probability of a class being the ground truth based on the classifiers' votes with the probability of that class being the ground truth based on the labels of the nearby non-conflict points (using a method similar to weighted KNN but also incorporating the number of conflict, uncertain points within the circle).

- A weighted voting framework for classifiers ensembles

Table 1 Scopes of optimality (denoted by a black square) and the number of tunable parameters of the 4 combiners for a problem with c classes and an ensemble of L classifiers

Combiner	1	2	3	4	Number of parameters
Majority vote	■	–	–	–	none
Weighted majority vote	■	■	–	–	$L + c$
Recall	■	■	■	–	$L * (c + 1)$
Naive Bayes	■	■	■	■	$L * c^2 + c$

Column headings: *1* Equal p, *2* Classifier-specific p_i, *3* Classifier- and class-specific p_i, *4* Full confusion matrix

This paper is a theoretical, statistical analysis that uses conditional probabilities (of a label being accurate given that it's produced by a particular classifier, that it belongs to a particular class, that it can be confused with other classes) to determine optimal conditions under which a combiner (i.e. voting method) is expected to perform the best.

The tradeoff is that with the relaxation of the assumption (i.e. the scope of optimality is larger), the number of prior parameters to estimate also increases. However, these estimations may not be as trustworthy if there is no sufficient data. "In practice, however, the success of a particular combiner will depend

partly on the assumptions and partly on the availability of sufficient data to make reliable estimates of the parameters."
The authors then evaluate their analysis using both simulated and real data. In a simulated environment, the analysis is proved correct. But with real datasets, the performance difference between the classifiers is blurred due to the underlying assumption of independence among classes being violated and the estimation of prior parameters being less trustworthy.
Even though I don't think the scope of my project will reach such a deep level of theoretical analysis (therefore, it won't involve using simulated data), I do learn a lot about how to do the evaluation and present the result. Most importantly, the paper gives me an idea about what to look for when comparing different voting methods: number of classes, number of classifiers, whether the distribution of the classes is balanced, etc.

- On combining classifiers
  This paper lays the theoretical foundation for combining classifiers, each of which produces the posteriori probability of a sample belonging to a particular class. These posteriori probabilities are then combined using different algebraic operations to arrive at the final prediction. This new perspective of taking into account how confident a classifier is when predicting a sample may be an approach I can take if I end up needing to add more complication to my algorithm to improve its performance. One thing I know for sure is that my algorithm will definitely need to combine probabilities and the algebraic operations proposed will be a good starting point. Besides, this paper also mentions the correlation of the classifiers and how this may affect the performance of the ensemble, another thing I will need to pay attention to.
  Despite all the good things, the fact that this paper was written in 1998 with some different terminology and it uses a number of notations without consistent explanation makes this paper extremely hard to follow. Luckily, I found another paper that examines the problem discussed in a much more narrow space (On combining classifiers using sum and product rules) and that defines the problem

and the solutions much more clearly. The understanding of the "narrow problem" can be easily extended and applied to the other methods proposed in the "big" paper.

- Rotation Forest: A New Classifier Ensemble Method

  This paper proposes a new technique of generating classifier ensembles that use feature extraction (PCA) to increase diversity but at the same time preserve all the principal components to ensure information variability and individual accuracy of each base classifier. Even though the nature of this approach is different from mine, I do learn a lot from this paper, especially about how to present a novel method:
  - How to describe the algorithm
  - How to defend and advocate a particular implementation (despite some criticism against it)
  - How to specify the conditions under which the experiments were carried out and how the outcome was measured (with a note about statistical testing). I especially like how they present the result in a variety of ways using several graphs and tables so the outcome can be interpreted from different angles.

  As mentioned, this paper studies in detail the importance of diversity of an ensemble, a factor that can be crucial to the performance of my proposed method. Even if I don't have to resort to the Rotation Forest algorithm, the more basic techniques of introducing diversity that have been discussed in the paper will surely be helpful to me.

- A Weighted Majority Voting Ensemble Approach for Classification

  This paper is probably the closest thing to what I intend for this topic when it proposes a novel voting method that takes into account the performance of the classifiers by assigning a weight to each of them. Because they claim that this is a new method, the first part of the paper is a literature review of the related works

and how the proposed method is different from the existing ones. This is a good practice that I should follow.

In addition, I like the way they describe their algorithm by demonstrating how it works with a specific example. This is also the second time I see the result of a paper presented in a variety of ways (in this paper: both ranking and pairwise comparison, tables and graphs).

However, it's disappointing that the outcome wasn't evaluated with hypothesis testing, which makes the conclusion "weaker".

- Combination of multiple classifiers using local accuracy estimates

  My intention when reading this paper was to see if its proposed method shares any similarity with mine. It turns out that the method suggested in this paper follows the Dynamic Classifier Selection approach, in which only a single best classifier is picked for each sample and "the basic idea is to estimate each classifier's accuracy in local regions of feature space surrounding an unknown test sample, and then use the decision of the most locally accurate classifier."

  Even though my technique is different from the one introduced in this paper, the paper does confirm a common structure of presenting a novel ensemble method: discussion of existing related methods and explanation of the proposed method's variation, introduction of the base classifiers, experiment, and performance comparison.

  There are also new things that I notice. The paper introduces a very creative concept of an "oracle", which chooses the correct class if any of the individual classifiers did so. This serves as a theoretical upper bound and along with the lower bound defined by the best individual classifier, it gives the authors an idea about the "room" for improvement. Some other interesting observations are how altering the classifier mix or varying the individual classifiers' sensitivity may affect the combining algorithm's performance (the latter can be applied to the study of this topic).

- Evaluating the effect of voting methods on ensemble-based classification

This paper focused particularly on assessing and comparing the existing voting techniques. It points out a weakness of the Majority Voting method due to its reduced effectiveness in multi-class problems (as opposed to binary-class problems). Other factors that can impact the performance of a voting system are noisy data and the size of the dataset.
Because this is an experimental study, the authors conduct a detailed literature review of the results reported in previous works. However, there are also good suggestions on top of that. For example, the authors use a different evaluation framework based on case studies with an increasing level of difficulty instead of running the methods on a variety of benchmark datasets. They also advocate the use of fewer base classifiers or voters to "reveal the influence of the voting methods [themselves]. With more voters, one can expect that the results would be better anyway, regardless of the voting method." They also underscore that "the goal of our study is to assess the influence of different voting methods, not to obtain an error rate as small as possible on the datasets", thereby directing the viewers' attention to the relevant aspect of the results. All of this information will become useful when I need to compare my technique against the existing ones.
Some other things that I learn:

  - "Using ensembles of different classifiers has the advantage that not all of them will make the same mistake"
  - How to describe the constraints in place and their influence upon some implementation and experiment decisions.
  - There are several distance metrics for KNN besides the basic Euclidean length (this can be something for me to experiment with my voting system and how it can learn from consensus in the test set)

- Tri-training: exploiting unlabeled data using three classifiers
  This paper proposes a variation of the Co-Training algorithm that has sparked widespread interest in response to some of its restrictions. One of the hardest requirements by the Co-Training algorithm is that the dataset can be divided into two sufficient, redundant, and independent views. The new Tri-Training algorithm

uses three classifiers and the agreed labels of two of them to train the other one, thereby overcoming the two-independent-view requirement for training two classifiers simultaneously.

This paper evaluates the validity of the proposed algorithms in a variety of ways:

- Mathematical proof of why the algorithm is expected to work.
- Empirical experiments on 12 UCI benchmark datasets
- A case study about web page classification.

While the idea behind the Tri-Training algorithm shares some similarities with mine, there are some fundamental differences. However, it does help me realize that my method is a form of semi-supervised learning, a field that I should learn more about besides the ensemble voting techniques.

- Related papers and resources:
  - ☑ ~~A weighted voting framework for classifiers ensembles~~
  - ☑ ~~On combining classifiers~~
  - ☑ ~~Rotation Forest: A New Classifier Ensemble Method~~
  - ☑ ~~A Weighted Majority Voting Ensemble Approach for Classification~~
  - ☑ ~~Tri-training: exploiting unlabeled data using three classifiers~~
  - ☑ ~~Combination of multiple classifiers using local accuracy estimates~~
  - ☑ ~~Evaluating the effect of voting methods on ensemble-based classification~~
  - ☐ A survey on semi-supervised learning
  - ☐ Multi-train: A semi-supervised heterogeneous ensemble classifier
  - ☐ Consensus-Based Combining Method for Classifier Ensembles

    ---
  - ☐ A survey of multiple classifier systems as hybrid systems
  - ☐ Ensemble-based classifiers
  - ☐ Breaking Ties of Plurality Voting in Ensembles of Distributed Neural Network Classifiers using Soft Max Accumulations

Comparing the performance of the different Hyperparameter Tuning methods

Basic idea: Hyperparameter Tuning is an important step in building a strong Machine Learning model. However, that the hyperparameter space grows exponentially and the interaction among the hyperparameters is often nonlinear limits the number of feasible methods to come up with a more optimal set of hyperparameters. I plan to examine some of the most common methods that are often used to tackle this problem and compare their performance.

- How to tune the RBF SVM hyperparameters? An empirical evaluation of 18 search algorithms

  This is a comprehensive paper that evaluates and compares 18 search algorithms for tuning 2 hyperparameters of a SVM. This is exactly what I intend to replicate for this topic but I will focus on fewer search strategies and on more classification algorithms. From this paper, I learnt how to:
  - Introduce a search strategy and how it's specifically configured and applied to the project.
  - Present the experimental protocol, measurement, how the result should be reviewed (e.g. the author points out which performance difference did go through hypothesis testing and which did not and why the decision was made), and reproducibility.
  - Discuss limitations and generalization

  I also like how the paper touches upon other aspects of hyperparameter tuning: whether the additional cost of computation is worth the slight increase in **estimated** optimality, and how to deal with multiple optimal sets of hyperparameters.

- Empirical comparison of cross-validation and internal metrics for tuning SVM hyperparameters

  One of the authors of this paper is the first author of the paper above so it's not difficult to recognize many similarities. However, this paper focuses on a very

different aspect of Hyperparameter Tuning. Instead of examining the search strategies, this paper looks at the different evaluation techniques that "power" the search for more optimal hyperparameters. In other words, it aims to answer the question: once we find a set of hyperparameters, how can we confirm that the set actually boosts the learner's performance?
The standard way of hyperparameter evaluation is using cross validation. However, this is a costly operation and several alternatives suggest using the "internal" metrics of the training set itself. In this paper, the authors have conducted an experiment to replicate and update previous works on the assessment of these techniques, but they also introduce some tweaks (using grid search in place of gradient descent search) to ensure fairness in comparing the accuracy gain and execution time of the various evaluation methods.
Some other things that I took from this paper:

- This paper again mentions a particular Friedman test (proposed by Demsar) that is often used to compare the performance of different algorithms. This test may be useful for both topics.
- It's okay to report a phenomenon even though you don't understand why it occurred.

- Effectiveness of Random Search in SVM hyper-parameter tuning

  This paper focuses on a question that I look forward to answering if I pursue this research topic: is a simple strategy like Random Search good enough for the task of hyperparameter tuning? When discussing this question, the authors bring up computational cost as an important factor, a pattern that I start to notice across papers. It's also interesting that SVM seems to attract a lot of interest. Based on the authors, this is because "SVMs are sensitive to the values of their hyper-parameters".
  The paper also confirms a standard layout of a paper whose goal is to compare different existing methods of hyperparameter tuning: literature review, experimental methodology, and result report. With respect to the evaluation and comparison techniques used, even though the authors admit the advantages of

nested cross-validation, they advocate the use of a single cross-validation step that separates data into training, validation and test partitions due to the much lower computational cost of the latter. This will be a thing for me to consider.
My other notes about this paper:

- The data sets to run the experiment are categorized into low and high complexity based on the number of features.
- I like the different types of graphs that the authors exploited to visually present the results.
- The hyper-parameters are "often dependent on each other, allowing the existence of an optimal region instead of a single global optimal solution."

- Hyperparameters and tuning strategies for random forest

  This paper aims to achieve two goals:

  1. Making a literature review of Hyperparameter Tuning for Random Forest
  2. Introducing and evaluating an R package that they implemented based on one of the search strategies discussed.

  Intended for general audiences, the paper spends a fair amount of time establishing the background knowledge and providing detailed explanation of the different factors that can influence the outcome of the hyperparameter tuning process. It also teaches me a new purpose of the Random Forest algorithm: estimating feature importance besides the more well-known purpose of predicting. The paper makes a clear distinction between these two purposes and how the hyperparameter tuning is different for each of them.
  Because one of the main goals is to introduce the package they developed, the authors put great care into explaining the implementation of the software and how it can be installed and run. This package is also compared with other existing packages in terms of the learner's performance (after being tuned) and runtime.
  Even though the focus of this paper doesn't quite suit my intention for this topic, it does give me an idea of how to explain the API of the software package that will be developed as a byproduct of my research.

- Hyperparameter tuning in Python using Optunity

  This is a fairly short paper due to the fact that its purpose is to unveil a new Python library for hyperparameter tuning. Because the paper targets general audiences, the information conveyed is concise and direct. However, basic knowledge about Hyperparameter Tuning is still presented to set up the context of the problem and why the library is necessary. More important is the explanation of certain implementation decisions (e.g. why Python was picked). A code snippet is also shown to demonstrate the use of the API. Along with the paper above, this paper serves as an example that I can follow to present my software package. It also introduces me to a handy Python library that I can consider using for this research topic (the library seems to be in active development based on the last commit on GitHub).

- Practical Bayesian Optimization of Machine Learning Algorithms

  This paper examines the limits of the current Bayesian Optimization methods by considering their high economic cost in terms of time and computation. In response to these issues, the authors have proposed three solutions:

  + A new covariance function that is believed to be more superior and "a fully Bayesian treatment of the underlying Gaussian process kernel".
  + A new concept of "expected improvement per second, which prefers to acquire points that are not only likely to be good, but that are also likely to be evaluated quickly."
  + Parallelization of the Bayesian optimization procedures by "comput[ing] Monte Carlo estimates of the acquisition function under different possible results from pending function evaluations."

  Built upon a huge number of previous works, the paper assumes that the readers have a fair amount of background knowledge. This along with the fact that it involves a lot of calculus makes it hard for me to fully appreciate the contribution of the authors. However, by looking at the three case studies of the most difficult and "expensive" problems at the time, it must be said that this work was

groundbreaking. For example, in the competitive CIFAR-10 problem, their proposed solution manages to outperform the set of hyperparameters manually tuned by a human expert.

- Bayesian Optimization for Accelerating Hyper-parameter Tuning
  This paper presents another attempt to refine the current state of Bayesian Optimization by addressing the limits of the existing methodology. It is a summary of several previous works conducted by the author to tackle a variety of issues. The solutions can be briefly listed as follows:
    - Instead of being fixed, the number of evaluations per batch is decided based on estimating the number of peaks in the underlying acquisition functions.
    - Several modifications to the Expected Improvement function to prevent unnecessary evaluations and make it more efficient.
    - The concept of a "weakly specified" search space to overcome situations in which the ranges of the hyperparameter values are not fully known.

  The second part of the paper discusses the real-world applications of the proposed solutions. As expected, one of the applications is Hyperparameter Tuning for Machine Learning models. An interesting thing about this paper is the long, detailed list of future research topics, indicating the author's active interest in this area.

- Related papers and resources:
  - ☑ ~~How to tune the RBF SVM hyperparameters? An empirical evaluation of 18 search algorithms~~
  - ☑ ~~Empirical comparison of cross-validation and internal metrics for tuning SVM hyperparameters~~
  - ☑ ~~Hyperparameters and tuning strategies for random forest~~
  - ☑ ~~Effectiveness of Random Search in SVM hyper-parameter tuning~~
  - ☑ ~~Practical Bayesian Optimization of Machine Learning Algorithms~~
  - ☑ ~~Bayesian Optimization for Accelerating Hyper-parameter Tuning~~

- [x] ~~Hyperparameter tuning in Python using Optunity~~