Scalable Semi-Supervised Aggregation of Classifiers

Authored by:

Yoav Freund
Akshay Balsubramani

Abstract

We present and empirically evaluate an efficient algorithm that learns to aggregate the predictions of an ensemble of binary classifiers. The algorithm uses the structure of the ensemble predictions on unlabeled data to yield significant performance improvements. It does this without making assumptions on the structure or origin of the ensemble, without parameters, and as scalably as linear learning. We empirically demonstrate these performance gains with random forests.

1 Paper Body

Ensemble-based learning is a very successful approach to learning classifiers, including well-known methods like boosting [1], bagging [2], and random forests [3]. The power of these methods has been clearly demonstrated in open large-scale learning competitions such as the Netflix Prize [4] and the ImageNet Challenge [5]. In general, these methods train a large number of ?base? classifiers and then combine them using a (possibly weighted) majority vote. By aggregating over classifiers, ensemble methods reduce the variance of the predictions, and sometimes also reduce the bias [6]. The ensemble methods above rely solely on a labeled training set of data. In this paper we propose an ensemble method that uses a large unlabeled data set in addition to the labeled set. Our work is therefore at the intersection of semi-supervised learning [7, 8] and ensemble learning. This paper is based on recent theoretical results of the authors [9]. Our main contributions here are to extend and apply those results with a new algorithm in the context of random forests [3] and to perform experiments in which we show that, when the number of labeled examples is small, our algorithm?s performance is at least that of random forests, and often significantly better. For the sake of completeness, we provide an intuitive introduction to the analysis given in [9]. How can unlabeled data help in the context of ensemble learning? Consider a simple example with six equiprobable data points. The ensemble consists of six classifiers, partitioned into three ?A? rules and

three ?B? rules. Suppose that the ?A? rules each have error 1/3 and the ?B? rules each have error 1/6. 1 If given only this information, we might take the majority vote over the six rules, possibly giving lower weights to the ?A? rules because they have higher errors. Suppose, however, that we are given the unlabeled information in Table 1. The columns of this table correspond to the six classifiers and the rows to the six unlabeled examples. Each entry corresponds to the prediction of the given classifier on the given example. As we see, the main difference between the ?A? rules and the ?B? rules is that any two ?A? rules disagree with probability 1/3, whereas the ?B? rules always agree. For this example, it can be seen (e.g. proved by contradiction) that the only possible true labeling of the unlabeled data that is consistent with Table 1 and with the errors of the classifiers is that all the examples are labeled ?+?. Consequently, we conclude that the majority vote over the ?A? rules has zero error, performing significantly better than any of the base rules. In contrast, giving the ?B? rules equal weight would 1 We assume that (bounds on) the errors are, with high probability, true on the actual distribution. Such bounds can be derived using large deviation bounds or bootstrap-type methods.

1

result in an a rule with error 1/6. Crucially, our reasoning to this point has solely used the structure of the unlabeled examples along with the error rates in Table 1 to constrain our search for the true labeling.

x1 x2 x3 x4 x5 x6 error

A classifiers + + + + + + + + + + + + 1/3 1/3 1/3

B classifiers + + + + + + + + + + + + + + + + 1/6 1/6 1/6

Table 1: An example of the utility of unlabeled examples in ensemble learning By such reasoning alone, we have correctly predicted according to a weighted majority vote. This example provides some insight into the ways in which unlabeled data can be useful: ? When combining classifiers, diversity is important. It can be better to combine less accurate rules that disagree with each other than to combine more accurate rules that tend to agree. ? The bounds on the errors of the rules can be seen as a set of constraints on the true labeling. A complementary set of constraints is provided by the unlabeled examples. These sets of constraints can be combined to improve the accuracy of the ensemble classifier. The above setup was recently introduced and analyzed in [9]. That paper characterizes the problem as a zero-sum game between a predictor and an adversary. It then describes the minimax solution of the game, which corresponds to an efficient algorithm for transductive learning. In this paper, we build on the worst-case framework of [9] to devise an efficient and practical semisupervised aggregation algorithm for random forests. To achieve this, we extend the framework to handle specialists ? classifiers which only venture to predict on a subset of the data, and abstain from predicting on the rest. Specialists can be very useful in targeting regions of the data on which to precisely suggest a prediction. The high-level idea of our algorithm is to artificially generate new specialists from the ensemble. We incorporate these, and the targeted information they carry, into the worst-case framework of [9]. The resulting aggregated predictor inherits the advantages of the original framework: (A) Efficient: Learning

reduces to solving a scalable p-dimensional convex optimization, and test-time prediction is as efficient and parallelizable as p-dimensional linear prediction. (B) Versatile/robust: No assumptions about the structure or origin of the predictions or labels. (C) No introduced parameters: The aggregation method is completely data-dependent. (D) Safe: Accuracy guaranteed to be at least that of the best classifier in the ensemble. We develop these ideas in the rest of this paper, reviewing the core worst-case setting of [9] in Section 2, and specifying how to incorporate specialists and the resulting learning algorithm in Section 3. Then we perform an exploratory evaluation of the framework on data in Section 4. Though the framework of [9] and our extensions can be applied to any ensemble of arbitrary origin, in this paper we focus on random forests, which have been repeatedly demonstrated to have state-of-theart, robust classification performance in a wide variety of situations [10]. We use a random forest as a base ensemble whose predictions we aggregate. But unlike conventional random forests, we do not simply take a majority vote over tree predictions, instead using a unlabeled-data-dependent aggregation strategy dictated by the worst-case framework we employ.

2

Preliminaries

A few definitions are required to discuss these issues concretely, following [9]. Write [a]+ = max(0, a) and [n] = {1, 2, . . . , n}. All vector inequalities are componentwise. 2

We first consider an ensemble H = {h1 , . . . , hp } and unlabeled data x1 , . . . , xn on which we wish to predict. As in [9], the predictions and labels are allowed to be randomized, represented by values in [?1, 1] instead of just the two values {?1, 1}. The ensemble?s predictions on the unlabeled data are denoted by F: ? ? h1 (x1) h1 (x2) ? ? ? h1 (xn) ? .. .. ? ? [?1, 1]p?n .. (1) F = ? ... . . . ? hp (x1) hp (x2) ? ? ? hp (xn) We use vector notation for the rows and columns of F: hi = (hi (x1), ? ? ? , hi (xn))¿ and xj = (h1 (xj), ? ? ? , hp (xj))¿ . The true labels on the test data T are represented by z = (z1 ; . . . ; zn) ? [?1, 1]n . The labels z are hidden from the predictor, but we assume the predictor has knowledge of P a correlation vector b ? (0, 1]p such that n1 j hi (xj)zj ? bi , i.e. n1 Fz ? b. These p constraints on z exactly represent upper bounds on individual classifier error rates, which can be estimated from the training set w.h.p. when all the data are drawn i.i.d., in a standard way also used by empirical risk minimization (ERM) methods that simply predict with the minimum-error classifier [9]. 2.1

The Transductive Binary Classification Game

The idea of [9] is to formulate the ensemble aggregation problem as a two-player zero-sum game between a predictor and an adversary. In this game, the predictor is the first player, who plays g = (g1 ; g2 ; . . . ; gn), a randomized label gi ? [?1, 1] for each example {xi }ni=1 . The adversary then sets the labels z ? [?1, 1]n under the ensemble classifier error constraints defined by b. 2 The predictor?s goal is to minimize the worst-case expected classification error on the test data (w.r.t.

the randomized labelings z and g), which is just 21 1 ? n1 z¿ g . This is

equivalently viewed as maximizing worst-case correlation n1 z¿ g. To summarize concretely, we study the following game: V :=

min

max

g?[?1,1]n

n

z?[?1,1] , 1 n Fz?b

1 ¿ z g n

(2)

The minimax theorem ([1], p.144) applies to the game (2), and there is an optimal strategy g? such 1 ¿ ? that min n z g ? V , guaranteeing worst-case prediction error 12 (1 ? V) on the n unlabeled z?[?1,1] , n 1 n Fz?b

data. This optimal strategy g? is a simple function of a particular weighting over the p hypotheses ? a nonnegative p-vector. Definition 1 (Slack Function). Let ? ? 0p be a weight vector over H (not necessarily a distribution). ¿ The vector of ensemble predictions is F¿ ? = (x¿ 1 ?, . . . , xn ?), whose elements? magnitudes are the margins. The prediction slack function is ?(?, b) := ?(?) := ?b¿ ? +

n

1 X ¿ xj ? ? 1 + n j=1

(3)

and this is convex in ?. The optimal weight vector ? ? is any minimizer ? ? ? arg min [?(?)]. ??0p

The main result of [9] uses these to describe the minimax equilibrium of the game (2). Theorem 2 ([9]). The minimax value of the game (2) is V = ??(? ?). The minimax optimal predictions are defined as follows: for all j ? [n], ¿ ? ¿ ? x ? ¡ 1 xj ? j gj? := gj (? ?) = ¿ ? sgn(xj ?) otherwise 2

Since b is calculated from the training set and deviation bounds, we assume the problem feasible w.h.p.

3

2.2

Interpretation

Theorem 2 suggests a statistical learning algorithm for aggregating the p ensemble classifiers? predictions: estimate b from the training (labeled) set, optimize the convex slack function ?(?) to find ? ? , and finally predict with gj (? ?) on each example j in the test set. The resulting predictions are guaranteed to have low error, as measured by V . In particular, it is easy to prove [9] that V is at least maxi bi , the performance of the best classifier. The slack function (3) merits further scrutiny. Its term depends

only on the labeled data and

Pfirst n

not the unlabeled set, while the second term n1 j=1 x¿ j ? ? 1 + incorporates only unlabeled information. These two terms trade off smoothly ? as the problem setting becomes fully supervised and unlabeled information is absent, the first term dominates, and ? ? tends to put all its weight on the best single classifier like ERM. Indeed, this viewpoint suggests a (loose) interpretation of

the second term as an unsupervised regularizer for the otherwise fully supervised optimization of the ?average? error b¿ ?. It turns out that a change in the regularization factor corresponds to different constraints on the true labels z: 1 ¿ Theorem 3 ([9]). Let V? := max min z g for any ? ¿ 0. Then V? = g?[?1,1]n z?[??,?]n , n 1 n Fz?b h i Pn ¿ ? ¿ min??0p ?b ? + n j=1 xj ? ? 1 + . So the regularized optimization assumes each zi ? [??, ?]. For ? ¡ 1, this is equivalent to assuming the usual binary labels (? = 1), and then adding uniform random label noise: flipping the label w.p. 21 (1??) on each of the n examples independently. This encourages ?clipping? of the ensemble ? ? ? predictions x¿ j ? to the ? -weighted majority vote predictions, as specified by g . 2.3

Advantages and Disadvantages

This formulation has several significant merits that would seem to recommend its use in practical situations. It is very efficient ? once b is estimated (a scalable task, given the labeled set), the slack function ? is effectively an average over convex functions of i.i.d. unlabeled examples, and consequently is amenable to standard convex optimization techniques [9] like stochastic gradient descent (SGD) and variants. These only operate in p dimensions, independent of n (which is p). The slack function is Lipschitz and well-behaved, resulting in stable approximate learning. Moreover, test-time prediction is extremely efficient, because it only requires the p-dimensional weighting ? ? and can be computed example-by-example on the test set using only a dot product in Rp . The form of g? and its dependence on ? ? facilitates interpretation as well, as it resembles familiar objects: sigmoid link functions for linear classifiers. Other advantages of this method also bear mention: it makes no assumptions on the structure of H or F, is provably robust against the worst case, and adds no input parameters that need tuning. These benefits are notable because they will be inherited by our extension of the framework in this paper. However, this algorithm?s practical performance can still be mediocre on real data, which is often easier to predict than an adversarial setup would have us believe. As a result, we seek to add more information in the form of constraints on the adversary, to narrow the gap between it and reality.

3

Learning with Specialists

To address this issue, we examine a generalized scenario in which each classifier in the ensemble can abstain on any subset of the examples instead of predicting ?1. It is a specialist that predicts only over a subset of the input, and we think of its abstain/participate decision being randomized in the same way as the randomized label on each example. In this section, we extend the framework of Section 2.1 to arbitrary specialists, and discuss the semi-supervised learning algorithm that results. In our formulation, suppose that for a classifier i ? [p] and an example x, the classifier decides to abstain with probability 1 ? vi (x). But if the decision is to participate, the classifier predicts 4

hi (x) P?n [?1, 1] as previously. Our only assumption on {vi (x1), . . . , vi (xn)} is the reasonable one that j=1 vi (xj) ¿ 0, so classifier i is not a worthless specialist that abstains everywhere. The constraint on classifier i is now not on its correlation with z on the entire test set, but on the average correlation with

z restricted to occasions on which it participates. So for some [bS]i ? [0, 1],

n X vi (xj) Pn hi (xj)zj ? [bS]i (4) k=1 vi (xk) j=1 v (x)

Define ?i (xj) := Pn i vji (xk) (a distribution over j ? [n]) for convenience. Now redefine our k=1 unlabeled data matrix as follows: ? ? ?1 (x1)h1 (x1) ?1 (x2)h1 (x2) ? ? ? ?1 (xn)h1 (xn) ? ? .. .. (5) S = n? ? . . . . ?p (x1)hp (x1) ?p (x2)hp (x2) ? ? ? ?p (xn)hp (xn) Then the constraints (4) can be written as n1 Sz ? bS , analogous to the initial prediction game (2). To summarize, our specialist ensemble aggregation game is stated as VS :=

max

min

z?[?1,1]n , g?[?1,1]n 1 n Sz?bS

1 ¿ z g n

(6)

We can immediately solve this game from Thm. 2, with (S, bS) simply taking the place of (F, b). Theorem 4 (Solution of the Specialist Aggregation Game). The awake ensemble prediction w.r.t. p X

weighting ? ? 0p on example xi is S¿ ? i = n ?j (xi)hj (xi)?j . The slack function is now j=1

?S (?) :=

1 n

n h X

i ¿ S ? j ? 1 ? b¿ S? +

j=1

(7)

The minimax value of this game is VS = max??0p [??S (?)] = ??S (?S?). The minimax optimal predictions are defined as follows: for all i ? [n], ¿ ? ¿ ? S ? ¡1 S ?S i . S i ¿ ? [gS?]i = gS (?S?) = sgn(S ?S i) otherwise In the no-specialists case, the vector ?i is the uniform distribution (n1 , . . . , n1) for any i ? [p], and the problem reduces to the prediction game (2). As in the original prediction game, the minimax equilibrium depends on the data only through the ensemble predictions, but these are now of a different form. Each example is now weighted proportionally to ?j (xi). So on any given example xi , only hypotheses which participate on it will be counted; and those that specialize the most narrowly, and participate on the fewest other examples, will have more influence on the eventual prediction gi , ceteris paribus. 3.1

Creating Specialists for an Algorithm

We can now present the main ensemble aggregation method of this paper, which creates specialists from the ensemble, adding them as additional constraints (rows of S). The algorithm, H EDGE C LIPPER, is given in Fig. 1, and instantiates our specialist learning framework with a random forest [3]. As an initial exploration of the framework here, random forests are an appropriate base ensemble because they are known to exhibit state-of-the-art performance [10]. Their wellknown advantages also include scalability, robustness (to corrupt data and parameter choices), and interpretability; each of these benefits is shared by our aggregation algorithm, which consequently inherits them all. Furthermore, decision trees are a natural fit as the ensemble classifiers because

they are inherently hierarchical. Intuitively (and indeed formally too [11]), they act like nearest-neighbor (NN) predictors w.r.t. a distance that is ?adaptive? to the data. So each tree in a random forest represents a 5

somewhat different, nonparametric partition of the data space into regions in which one of the labels ?1 dominates. Each such region corresponds exactly to a leaf of the tree. The idea of H EDGE C LIPPER is simply to consider each leaf in the forest as a specialist, which predicts only on the data falling into it. By the NN intuition above, these specialists can be viewed as predicting on data that is near them, where the supervised training of the tree attempts to define the purest possible partitioning of the space. A pure partitioning results in many specialists with [bS]i ? 1, each of which contributes to the awake ensemble prediction w.r.t. ? ? over its domain, to influence it towards the correct label (inasmuch as [bS]i is high). Though the idea is complex in concept for a large forest with many arbitrarily overlapping leaves from different trees, it fits the worst-case specialist framework of the previous sections. So the algorithm is still essentially linear learning with convex optimization, as we have described.

Algorithm 1 H EDGE C LIPPER Input: Labeled set L, unlabeled set U 1: Using L, grow trees T = {T1 , . . . , Tp } (regularized; see Sec. 3.2) 2: Using L, estimate bS on T and its leaves 3: Using U , (approximately) optimize (7) to estimate ?S? Output: The estimated weighting ?S? , for use at test time Figure 1: At left is algorithm H EDGE C LIPPER. At right is a schematic of how the forest structure is related to the unlabeled data matrix S, with a given example x highlighted. The two colors in the matrix represent ?1 predictions, and white cells abstentions.

3.2

Discussion

Trees in random forests have thousands of leaves or more in practice. As we are advocating adding so many extra specialists to the ensemble for the optimization, it is natural to ask whether this erodes some of the advantages we have claimed earlier. Computationally, it does not. When ?j (xi) = 0, i.e. classifier j abstains deterministically on xi , then the value of hj (xi) is irrelevant. So storing S in a sparse matrix format is natural in our setup, with the accompanying performance gain in computing S¿ ? while learning ? ? and predicting with it. This turns out to be crucial to efficiency ? each tree induces a partitioning of the data, so the set of rows corresponding to any tree contains n nonzero entries in total. This is seen in Fig. 1. Statistically, the situation is more complex. On one hand, there is no danger of overfitting in the traditional sense, regardless of how many specialists are added. Each additional specialist can only shrink the constraint set that the adversary must follow in the game (6). It only adds information about z, and therefore the performance VS must improve, if the game is solved exactly. However, for learning we are only concerned with approximately optimizing ?S (?) and solving the game. This presents several statistical challenges. Standard optimization methods do not converge as well in high ambient dimension, even given the structure of our problem. In addition, random forests practically perform best when each tree is grown to overfit. In our case, on any sizable test set, small leaves would cause some entries of

S to have large magnitude, 1. This can foil an algorithm like H EDGE C LIPPER by causing it to vary wildly during the optimization, particularly since those leaves? [bS]i values are only roughly estimated. From an optimization perspective, some of these issues can be addressed by e.g. (pseudo)-secondorder methods [12], whose effect would be interesting to explore in future work. Our implementation opts for another approach ? to grow trees constrained to have a nontrivial minimum weight per leaf. Of course, there are many other ways to handle this, including using the tree structure beyond the leaves; we just aim to conduct an exploratory evaluation here, as several of these areas remain ripe for future research. 6

4

Experimental Evaluation

We now turn to evaluating H EDGE C LIPPER on publicly available datasets. Our implementation uses minibatch SGD to optimize (6), runs in Python on top of the popular open-source learning package scikit-learn, and runs out-of-core (n-independent memory), taking advantage of the scalability of our formulation. 3 The datasets are drawn from UCI/LibSVM as well as data mining sites like Kaggle, and no further preprocessing was done on the data. We refer to ?Base RF? as the forest of constrained trees from which our implementation draws its specialists. We restrict the training data available to the algorithm, using mostly supervised datasets because these far outnumber medium/large-scale public semi-supervised datasets. Unused labeled examples are combined with the test examples (and the extra unlabeled set, if any is provided) to form the set of unlabeled data used by the algorithm. Further information and discussion on the protocol is in the appendix. Class-imbalanced and noisy sets are included to demonstrate the aforementioned practical advantages of H EDGE C LIPPER. Therefore, AUC is an appropriate measure of performance, and these results are in Table 2. Results are averaged over 10 runs, each drawing a different random subsample of labeled data. The best results according to a paired t-test are in bold. We find that the use of unlabeled data is sufficient to achieve improvements over even traditionally overfitted RFs in many cases. Notably, in most cases there is a significant benefit given by unlabeled data in our formulation, as compared to the base RF used. The boosting-type methods also perform fairly well, as we discuss in the next section.

Figure 2: Class-conditional ?awake ensemble prediction? (x¿ ? ?) distributions, on SUSY. Rows (top to bottom): {1K, 10K, 100K} labels. Columns (left to right): ? = {1.0, 0.3, 3.0}, and the base RF.

The awake ensemble prediction values x¿ ? on the unlabeled set are a natural way to visualize and explore the operation of the algorithm on the data, in an analogous way to the margin distribution in boosting [6]. One representative sample is in Fig. 2, on SUSY, a dataset with many (5M) examples, roughly evenly split between ?1. These plots demonstrate that our algorithm produces much more peaked class-conditional ensemble prediction distributions than random forests, suggesting marginbased learning applications. Changing ? alters the aggressiveness of the clipping, inducing a more or less peaked distribution. The other datasets without dramatic label imbalance show very similar quali-

tative behavior in these respects, and these plots help choose ? in practice (see appendix). Toy datasets with extremely low dimension seem to exhibit little to no significant improvement from our method. We believe this is because the distinct feature splits found by the random forest are few in number, and it is the diversity in ensemble predictions that enables H EDGE C LIPPER to clip (weighted majority vote) dramatically and achieve its performance gains. On the other hand, given a large quantity of data, our algorithm is able to learn significant structure, the minimax structure appears appreciably close to reality, as evinced by the results on large datasets.

5

Related and Future Work

This paper?s framework and algorithms are superficially reminiscent of boosting, another paradigm that uses voting behavior to aggregate an ensemble and has a game-theoretic intuition [1, 15]. There is some work on semi-supervised versions of boosting [16], but it departs from this principled structure and has little in common with our approach. Classical boosting algorithms like AdaBoost [17] make no attempt to use unlabeled data. It is an interesting open problem to incorporate boosting ideas into our formulation, particularly since the two boosting-type methods acquit themselves well 3

It is possible to make this footprint independent of d as well by hashing features [13], not done here.

7

Dataset kagg-prot
training 10 100
0.567 0.714
Random Forest 0.509 0.665
H EDGE C LIPPER
0.500 0.656
AdaBoost Trees 0.520 0.681
MART [14] 0.497 0.666
Logistic Regression 0.510 0.688
Base RF
ssl-text
10 100
0.586 0.765
0.517 0.551
0.512 0.542
0.556 0.596
0.553 0.569
0.501 0.552
kagg-cred
100 1K 10K
0.558 0.602 0.603
0.518 0.534 0.563
0.501 0.510 0.535
0.528 0.585 0.586

0.542 0.565 0.566
0.502 0.505 0.510
a1a
100 1K
0.779 0.808
0.619 0.714
0.525 0.655
0.680 0.734
0.682 0.722
0.725 0.768
w1a
100 1K
0.543 0.651
0.510 0.592
0.505 0.520
0.502 0.695
0.513 0.689
0.509 0.671
covtype
100 1K 10K
0.735 0.764 0.809
0.703 0.761 0.822
0.661 0.715 0.785
0.709 0.730 0.759
0.732 0.761 0.788
0.515 0.524 0.515
ssl-secstr
10 100 1K
0.572 0.656 0.687
0.574 0.645 0.682
0.535 0.610 0.646
0.563 0.643 0.690
0.557 0.637 0.689
0.557 0.629 0.683
SUSY
1K 10K 100K
0.776 0.785 0.799
0.769 0.787 0.797
0.764 0.784 0.797
0.747 0.787 0.797
0.771 0.789 0.796
0.720 0.759 0.779
epsilon
1K
0.651
0.659

0.645
0.718
0.726
0.774
webspam-uni
1K 10K
0.936 0.967
0.928 0.970
0.920 0.957
0.923 0.945
0.928 0.953
0.840 0.901

Table 2: Area under ROC curve for H EDGE C LIPPER vs. supervised ensemble algorithms.

in our results, and can pack information parsimoniously into many fewer ensemble classifiers than random forests. There is a long-recognized connection between transductive and semi-supervised learning, and our method bridges these two settings. Popular variants on supervised learning such as the transductive SVM [18] and graph-based or nearest-neighbor algorithms, which dominate the semi-supervised literature [8], have shown promise largely in data-poor regimes because they face major scalability challenges. Our focus on ensemble aggregation instead allows us to keep a computationally inexpensive linear formulation and avoid considering the underlying feature space of the data. Largely unsupervised ensemble methods have been explored especially in applications like crowdsourcing, in which the method of [19] gave rise to a plethora of Bayesian methods under various conditional independence generative assumptions on F [20]. Using tree structure to construct new features has been applied successfully, though without guarantees [21]. Learning with specialists has been studied in an adversarial online setting as in the work of Freund et al. [22]. Though that paper?s setting and focus is different from ours, the optimal algorithms it derives also depend on each specialist?s average error on the examples on which it is awake. Finally, we re-emphasize the generality of our formulation, which leaves many interesting questions to be explored. The specialists we form are not restricted to being trees; there are other ways of dividing the data like clustering methods. Indeed, the ensemble can be heterogeneous and even incorporate other semi-supervised methods. Our method is complementary to myriad classification algorithms, and we hope to stimulate inquiry into the many research avenues this opens. Acknowledgements The authors acknowledge support from the National Science Foundation under grant IIS-1162581.

8

2 References

[1] Robert E. Schapire and Yoav Freund. Boosting: Foundations and Algorithms. The MIT Press, 2012. [2] Leo Breiman. Bagging predictors. Machine

learning, 24(2):123?140, 1996. [3] Leo Breiman. Random forests. Machine learning, 45(1):5?32, 2001. [4] Yehuda Koren. The bellkor solution to the netflix grand prize. 2009. [5] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 2015. [6] Robert E Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. Annals of statistics, pages 1651?1686, 1998. [7] Olivier Chapelle, Bernhard Sch?olkopf, and Alexander Zien. Semi-supervised learning. [8] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. Synthesis lectures on artificial intelligence and machine learning, 3(1):1?130, 2009. [9] Akshay Balsubramani and Yoav Freund. Optimally combining classifiers using unlabeled data. In Conference on Learning Theory, 2015. [10] Rich Caruana, Nikos Karampatziakis, and Ainur Yessenalina. An empirical evaluation of supervised learning in high dimensions. In Proceedings of the 25th international conference on Machine learning, pages 96?103. ACM, 2008. [11] Yi Lin and Yongho Jeon. Random forests and adaptive nearest neighbors. Journal of the American Statistical Association, 101(474):578?590, 2006. [12] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research, 12:2121?2159, 2011. [13] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In Proceedings of the 26th Annual International Conference on Machine Learning, pages 1113?1120. ACM, 2009. [14] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. Annals of statistics, pages 1189?1232, 2001. [15] Yoav Freund and Robert E Schapire. Game theory, on-line prediction and boosting. In Proceedings of the ninth annual conference on Computational learning theory, pages 325?332. ACM, 1996. [16] P Kumar Mallapragada, Rong Jin, Anil K Jain, and Yi Liu. Semiboost: Boosting for semi-supervised learning. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 31(11):2000?2014, 2009. [17] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci., 55(1):119?139, 1997. [18] Thorsten Joachims. Transductive inference for text classification using support vector machines. In Proceedings of the Sixteenth International Conference on Machine Learning, pages 200?209. Morgan Kaufmann Publishers Inc., 1999. [19] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. Applied statistics, pages 20?28, 1979. [20] Fabio Parisi, Francesco Strino, Boaz Nadler, and Yuval Kluger. Ranking and combining multiple predictors without labeled data. Proceedings of the National Academy of Sciences, 111(4):1253?1258, 2014. [21] Frank Moosmann, Bill Triggs, and Frederic Jurie. Fast discriminative visual codebooks using randomized clustering forests. In Twentieth Annual Conference on Neural Information Processing Systems (NIPS?06), pages 985?992. MIT Press, 2007. [22] Yoav Freund, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. Using and combining pre-

dictors that specialize. In Proceedings of the twenty-ninth annual ACM symposium on Theory of computing, pages 334?343. ACM, 1997. [23] Predicting a Biological Response. 2012. https://www.kaggle.com/c/bioresponse. [24] Give Me Some Credit. 2011. https://www.kaggle.com/c/GiveMeSomeCredit.

9