Assume that we are given a set of hypotheses h_{ij} for $i, j \in [k]$, with generalization errors ϵ_{ij}. In the following sections, we analyze the generalization error of the constructions which use pair-wise binary classifiers. To help understanding the bounds, we also consider the uniform error case where every binary hypothesis has generalization error of $\bar{\epsilon}$.

In what follows, for each algorithm, we mainly analyze the error δ_i when the query point is from class i; the generalization error is thus the error of the worst i. Frequently, we consider the other classes in order of their generalization errors. We shall use the following notation. For each $i \in Y$ and $1 \leq j < k$, let $r_i(j)$ denote the class having the j-th largest generalization error when compared with class i, i.e., $\epsilon_{i,r_i(1)} \geq \epsilon_{i,r_i(2)} \geq \cdots \epsilon_{i,r_i(k-1)}$. We break ties arbitary.

3.1 Max-Win

Using the black-box approach, the worst-case bound on the generalization error of Max-Win can be proved using union bound.

Theorem 1. *If the query point is from class i, Max-Win gives wrong prediction with probability at most $\sum_{j \neq i} \epsilon_{i,j}$. Thus, the generalization error of Max-Win is at most $\max_{i \in [k]} \sum_{j \neq i} \epsilon_{i,j}$. In uniform case, this is $(k-1)\bar{\epsilon}$.*

Proof. Suppose that the input x is from class i. If all classifiers $h_{ij}(x)$, for all possible j, answer i, Max-Win would definitely return i as a prediction, because i would get $k-1$ votes while all other classes would get at most $k-2$ votes. The probability that $h_{ij}(x)$ for a fixed j returns a wrong prediction is at most ϵ_{ij}. Therefore, using union bound, the probability that some h_{ij} makes a wrong prediction is at most $\sum_{j:j \neq i} \epsilon_{ij}$. The theorem follows because we consider the worst case i.

We note that this bound is very loose. However, we believe that given only black-box information of the generalization error of the base classifiers, this bound is best possible. In practice, Max-Win performs very well. The reason might be from the gap between the score of the winning class and the second runner up. We note that if one looks into each binary classifier, thus ignoring the black-box approach, the margin analysis of the voting method in Schapire, Freund, Barlett, and Lee [15] directly applies to Max-Win. Also, following the same research direction, the result of Paugam-Moisy, Elisseeff, and Guermeur [16] uses the margin analysis to show the generalization error of the One-vs-All approach.

3.2 DDAG

For DDAG, we have $\binom{k}{2}$ binary classifiers. If they all give correct predictions, the final prediction must be correct. This gives a bound of $\sum_{j \neq i} \epsilon_{i,j}$ for the generalization error. However, we can do a lot better by noting that if the query is of class i, relevant classifiers are those concerning class i. Furthermore, these classifiers will not be called unless i is the candidate with minimum class id or

the maximum id, i.e., when all the smaller-id classes or the larger-id classes get eliminated. In the worst case, this happens when either only larger-id classes are eliminated, or only smaller ones. Therefore, we get the following theorem.

Theorem 2. *If the query point is from class i, DDAG gives wrong prediction with probability at most $\delta_i \stackrel{def}{=} \max\{\sum_{j<i} \epsilon_{i,j}, \sum_{j>i} \epsilon_{i,j}\}$. Therefore, the generalization error of DDAG is at most $\max_{i\in[k]} \delta_i$. In uniform case, this is $(k-1)\bar{\epsilon}$.*

Comparison to previous bounds. Platt, Cristianini, and Shawe-Taylor [6] consider the Proceptron DDAG, i.e., DDAG with a perceptron at each node. Let γ_i denote the margin observed at node i. Using the technique of Bennett, Cristianini, Shawe-Taylor, and Wu [17], which is also based on that of [18], they show that with probability at least $1-\delta$ the generalization error of DDAG is at most

$$\frac{130R^2}{m}\left(D'\log(4em)\log(4m) + \log\frac{2(2m)^K}{\delta}\right),$$

where K is the number of nodes in DDAG, $D' = \sum_{i=1}^{K}\frac{1}{\gamma_i^2}$, m is the number of samples used for training, and R is radius of the ball containing the distribution's support, for DDAG that correctly classifies m examples. They also consider the generalization error for class j. Let j-nodes denote the set of perceptron nodes involving class j. They get the bound of

$$\frac{130R^2}{m}\left(D'\log(4em)\log(4m) + \log\frac{2(2m)^{k-1}}{\delta}\right),$$

where $D' = \sum_{i\in j-\text{nodes}}\frac{1}{\gamma_i^2}$.

With previous analysis, we also get the same bound. For a perceptron which correctly classifies m examples with margin γ on the distribution whose support contains in a ball in $\mathbb{R}^n$ centered at the origin of radius R, Shawe-Taylor, Barlett, Williamson, and Anthony [18] gives a bound of

$$\frac{2}{m}\left(l\log\left(\frac{8em}{l}\right)\log(32m) + \log\frac{8m}{\delta}\right),$$

where $l = \lfloor 577R^2/\gamma^2\rfloor$. We plug this into Theorem 2. Recall that the bound in Theorem 2 uses only subsets of j-nodes; thus, it is at most

$$\sum_{i\in j-\text{nodes}}\left[\frac{2}{m}\left(\left(\frac{577R^2}{\gamma_i^2}\right)\log\left(\frac{8em}{(557R^2)/\gamma_i^2}\right)\log(32m) + \log\frac{8m}{\delta}\right)\right]$$

$$\leq \left(\frac{1154R^2}{m}\right)\left(\sum_{i\in j-\text{nodes}}\left(\frac{1}{\gamma_i^2}\right)\right)\log(8em)\log(32m) + \frac{2}{m}\left(\sum_{i\in j-\text{nodes}}\log\frac{8m}{\delta}\right),$$

$$\leq \left(\frac{1154R^2}{m}\right)[D'\log(8em)\log(32m)] + \left(\frac{2}{m}\right)(k-1)\log\frac{8m}{\delta},$$

$$= \left(\frac{1154R^2}{m}\right)[D' \log(8em)\log(32m)] + \left(\frac{2}{m}\right)\log\frac{(8m)^{k-1}}{\delta},$$

if $1154R^2 \geq 2$, and $R \geq \gamma_i$ for all i. This bound is similar to the bound in [6] up to a small constant factor.

Our analysis can also be extended to include the case where there are training errors for perceptrons as well.

3.3 Improvements to DDAG: ADAG, Randomized-DDAG

ADAG. The solution of ADAG depends crucially on the order of elimination. However, since the path is short, we need only $l = \lceil \log k \rceil$ applications of the binary classfiers. For query point in class i, we can assume that these l classifications are done with the worst l classes.

Theorem 3. *If the query point is from class i, ADAG gives wrong prediction with probability at most $\delta_i \stackrel{def}{=} \sum_{j=1}^{l} \epsilon_{i,\, r_i(j)}$. Therefore, the generalization error of ADAG is at most $\max_i \delta_i$. In uniform case, this is $O(\log k)\bar{\epsilon}$.*

We note that the generalization error of ADAG seems to depend on the order of elimination. However, we are not sure if the worst-case behavior can be improved with randomization.

Randomized-DDAG. For DDAG, the following theorem shows that randomization does help.

Theorem 4. *If the correct class is class i, Randomized-DDAG gives wrong prediction with probability at most*

$$\delta_i \stackrel{def}{=} \sum_{l=1}^{k-1} \left(\frac{2\epsilon_{i,\, r_i(k-l)}}{k-l+1}\right).$$

Thus, the generalization error of Randomized-DDAG is at most $\max_i \delta_i$. In uniform case, this is $O(\log k)\bar{\epsilon}$.

Proof. We consider the process of the randomized DDAG in rounds; there are $n-1$ rounds. In each round, two candidate classes are chosen and a binary classifier that distinguishes between such classes is called. The winner stays on to the next round; the loser are eliminated. The first round is round 1. We analyze the probability that the correct class, say class i, gets eliminated during the process. Suppose that i survives after l rounds. On the one hand, if i is not chosen, it remains to round $l+1$ automatically; this happens with probability $1-\frac{2}{k-l}$. On the other hand, if it is chosen, it remains in the next round only if the classifier predicts correctly. Let D_l denote the event that class i gets eliminated in round l, given that it survives up to round $l-1$. Randomized-DDAG predicts incorrectly if one of these events occurs; thus, the probability that it gives incorrect prediction is $\Pr[\bigcup_{l=1}^{k-1} D_l]$.

For the analysis, we consider a different procedure. Suppose that i is known. In round l, we describe an equlivalent procedure for choosing a pair of classes. If i has been eliminated in previous rounds, we pick a pair of classes randomly. Now if i remains one of the candidates, we decide whether class i will be evaluated by fliping a coin with $2/(k-l+1)$ head probability. If the coin comes up tail, we pick a pair of classes beside i, and in this case i remains to the next round with probability 1, and some class j get eliminated. If the coin comes up head, we choose another class randomly. Suppose that j is picked, the probability that the classifier gives wrong answer is thus ϵ_{ij}. For both cases, we call class j the *pairing class*. The notion of pairing class is important to our analysis, and note that, by definition, i cannot be a pairing class.

To finish the analysis, we consider the choices for the pairing class to be the worst case. Suppose that the sequence $\mathcal{S}$ of pairing classes over the execution of the algorithm is $c_1, c_2, \ldots, c_{k-1}$. Thus, the probability that i gets eliminated in round l is $\Pr[D_l|\mathcal{S}] = \frac{2\epsilon_{i,\,c_l}}{k-l+1}$. Using the union bound, we have that the probability that randomized DDAG fails, given $\mathcal{S}$, is at most

$$\sum_{l=1}^{k-1} \frac{2\epsilon_{i,\,c_l}}{k-l+1} = \left(\frac{2\epsilon_{i,\,c_1}}{k} + \frac{2\epsilon_{i,\,c_2}}{k-1} + \cdots + \frac{2\epsilon_{i,\,c_{k-2}}}{2} + \frac{2\epsilon_{i,\,c_{k-1}}}{1} \right)$$

This value is maximum when classes j in $\mathcal{S}$ are in increasing order of ϵ_{ij}. Thus, the randomized DDAG gives wrong prediction with probability at most

$$\delta_i = \sum_{l=1}^{k-1} \frac{2\epsilon_{i,\,r_i(k-l)}}{k-l+1}.$$

The theorem follows.

4 Generalization errors for the output coding approaches

We use the black-box approach to analyze the generalization error of the output coding method. The proof is very similar to that of [10], but instead of training error, we use generalization of the base classifier. In Subsection 4.1, we consider the simpler case where the coding matrix $\mathbf{M}$ contains only $+1$ and -1. In the next subsection, we prove the theorem for the general matrix as in Allwein et al.

4.1 Coding matrix without don't care bits

We note that the argument of Guruswami and Sahai [10], that bounds the training error, applies to the generalization error as well if we replace the averaging argument with Markov's inequality. We state and reprove their result here for completeness.

Theorem 5 ([10]). *Suppose that the coding matrix $\mathbf{M} \in \{+1, -1\}^{k \times l}$ with l columns has minimum Hamming distance Δ. Also, let ϵ_s be the generalization error of the binary classifier for column s. Then, the generalization error of the multiclass classifier is at most $\frac{2}{\Delta}(\sum_{s=1}^{l} \epsilon_s)$.*

Proof. For each column s, let a random variable S_s be 1 if the hypothesis for that column makes a wrong prediction, and 0 otherwise. Clearly $\mathbf{E}[S_s] = \epsilon_s$. Let random variable S be the number of columns having wrong predictions. By linearity of expectation, $\mathbf{E}[S] = \mathbf{E}[\sum_{s=1}^{l} S_s] = \sum_{s=1}^{l} \mathbf{E}[S_s] = \sum_{s=1}^{l} \epsilon_s$. Since the coding matrix with Hamming distance Δ can correct up to $d \stackrel{def}{=} \lfloor (\Delta - 1)/2 \rfloor$ errors, if $S \leq d$, the algorithm would return the correct class. The case where the algorithm makes a wrong mistake is when $S > d$, which occurs with probability $\Pr[S > d] \leq Pr[S \geq \Delta/2] \leq \mathbf{E}[S]/(\Delta/2) = \frac{2}{\Delta} \sum_{s=1}^{l} \epsilon_s$, by Markov's Inequality, as claimed.

4.2 General matrix

We now consider the general matrix. As in Allwein et al., we assign each class $r \in Y$ a row in the coding matrix $\mathbf{M} \in \{-1, 0, +1\}^{k \times l}$. However, the proof from the previous section does not follow unless we change the distance function $\Delta(u, v)$ between two rows u, v of $\mathbf{M}$. Previously, the distance between any bit with "don't care" bit is $1/2$. This is too optimistic, when dealing with generalization error. Therefore, we ignore that "distance," i.e., we define $\Delta(u, v)$ to be

$$\Delta(u, v) = \sum_{s=1}^{l} \begin{cases} 1 \text{ if } u_s \neq v_s, u_s \neq 0, \text{ and } v_s \neq 0. \\ 0 \text{ otherwise} \end{cases}$$

We let $\Delta \stackrel{def}{=} \Delta(M)$ be $\min_{u,v \in M} \Delta(u, v)$, the minimum distance between any pair of rows in M. Also, let I_i denote the set of indices s such that $M(i, s) \neq 0$. The following theorem states the generalization error bound for the combined classifier.

Theorem 6. *Given the coding matrix M, if the generalization error of the classifier for the s-th column is ϵ_s, the probability that given an instance x of class i, the multiclass classifier using Hamming decoding predicts x's class incorrectly is at most $2(\sum_{s \in I_i} \epsilon_s)/\Delta$. Therefore, the generalization error is at most $(\frac{2}{\Delta}) \max_i \sum_{s \in I_i} \epsilon_s$.*

Proof. For each $s \in I$, let an indicator random variable E_s be one if $C_s(x) \neq M(i, s)$, and zero otherwise. Therefore, $\Pr[E_s = 1] = \epsilon_s$, by the definition of ϵ_s. Let E be the number of error bits, i.e., $E = \sum_{s \in I} E_s$. By linearity of expectation, we have $\mathbf{E}[E] = \sum_{s \in I} \epsilon_s$. Since the minimum hamming distance between any two rows is Δ, if $E < \Delta/2$, the classifier would always predict class i, i.e., it makes no mistake. Using Markov's Inequality, we have

$$\Pr[\text{predict incorrectly}] \leq \Pr[E \geq \Delta/2] \leq \mathbf{E}[E]/(\Delta/2) = \frac{2}{\Delta} \sum_{s \in I} \epsilon_s;$$

thus, the theorem is proved.

We note that the generalization error of Max-Win (Section 3.1) also follows from this theorem, since in that case $\Delta = 1$.

5 Learnability

In this section, we discuss the equivalence of the learnability, and efficient learnability of the muliclass concepts and the induced all-pair concepts. One direction of the equivalence, from multiclass to pair-wise, is simple. We focus on the other direction.

The equivalence of the learnability is implicit in the work of Ben-David, Cesa-Bianchi, Haussler, and Long [19], which proves various relation between many notions of dimensions, including the Natarajan dimension [20]. Consider a particular pairwise concept $\mathcal{C}_{ij}$ induced by any classes i and j. If this concept is learnable, the VC-dimension [21] of $\mathcal{C}_{ij}$ is finite [22]. Since this is true for any i and j, this means that the uniform Natarajan dimension of the concept class is finite; hence the Natarajan dimension is also finite, by Theorem 5 in [19]. This shows that the multiclass concept is learnable.

For efficient learnability, we note that if an all-pair concept is efficiently learnable, one can use Max-Win (or any other constructions discussed in this paper) to construct a polynomial-time learner for the multiclass concept. Specifically, if one needs a multiclass classifier with error probability ϵ. We first find a binary classifier for each pair of classes with generalization error ϵ/k^2. Theorem 1 ensures that the combined classifier has the generalization within the required bound. The running time for constructing hypothesis only increases by a polynomial factor of k. Thus, we get the desired reduction.

6 Discussions and open problems

By considering the base binary classifiers as black-boxes, we are able to analyze various algorithms for the multiclass classification problem. The analysis focuses mainly on the combinatorial structures of the constructions. The result in this paper contradicts previous beliefs that the order of the class evaluation in DDAG has no significant effect, and also it gives supportive argument for a new algorithm such as ADAG.

We list a few interesting open problems here.

1. Our analysis of the generalization error of Max-Win is very loose. We believe that the observed gap between the winning class and the second runner-up (as in [15]) can be used to give better bounds. However, the margin seems to be a property of the construction; therefore, we do not know how to put it into our framework. This might give an evidence of the limitation of the black-box analysis.
2. There are a few other algorithms which perform well in practice, but have no proof of the generalization performance, notably, an improvement to ADAG, called Reordering-ADAG by Phetkaew, Kijsirikul, and Rivepiboon [23], that uses the error information to minimize the overall prediction error.
3. For the coding approach, we consider only Hamming decoding. It might be possible to extend our approach for the loss-based decoding.

4. It is interesting to see if one can combine an improvement to the proof of Allwein et al. by Klautau, Jevtić, and Orlitsky [24] with our technique.
5. One of the biggest open problems in multiclass learning is the question that asks if these new techniques devised to improve the simplest One-vs-All do really help [25]. The technique in this paper, however, offers no insight into this problem.

7 Acknowledgement

We would like to thank Aharon Bar-Hillel and Danupon Nanongkai for thoughtful discussions. We also thank the anonymous referees for helpful comments.

References

1. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. J. Comput. Syst. Sci. **55** (1997) 119–139
2. Vapnik, V.: Statistical Learning Theory. Wiley (1998)
3. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20** (1995) 273–297
4. Friedman, J.H.: Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University (1996)
5. Hastie, T., Tibshirani, R.: Classification by pairwise coupling. In: NIPS '97: Proceedings of the 1997 conference on Advances in neural information processing systems 10, Cambridge, MA, USA, MIT Press (1998) 507–513
6. Platt, J., Cristianini, N., Shawe-Taylor, J.: Large margin DAGs for multiclass classification. In: Advance in Neural Information Processing System. Volume 12., MIT Press (2000)
7. Kreßel, U.H.G.: Pairwise classification and support vector machines. In: Advances in kernel methods: support vector learning. MIT Press, Cambridge, MA, USA (1999) 255–268
8. Kijsirikul, B., Ussivakul, N., Meknavin, S.: Adaptive directed acyclic graphs for multiclass classification. In: PRICAI 2002. (2002) 158–168
9. Dietterich, T.G., Bakiri, G.: Error-correcting output codes: a general method for improving multiclass inductive learning programs. In Dean, T.L., McKeown, K., eds.: Proceedings of the Ninth AAAI National Conference on Artificial Intelligence, Menlo Park, CA, AAAI Press (1991) 572–577
10. Guruswami, V., Sahai, A.: Multiclass learning, boosting, and error-correcting codes. In: Computational Learning Theory. (1999) 145–155
11. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers. J. Mach. Learn. Res. **1** (2001) 113–141
12. Har-Peled, S., Roth, D., Zimak, D.: Constraint classification: A new approach to multiclass classification and ranking. In: NIPS. (2003)
13. Bar-Hillel, A., Weinshall, D.: Learning with equivalence constraints, and the relation to multiclass learning. In: COLT. (2003)
14. Fakcharoenphol, J.: A note on random DDAG. Manuscript (2003)
15. Schapire, R.E., Freund, Y., Bartlett, P.L., Lee, W.S.: Boosting the margin: a new explanation for the effectiveness of voting methods. Annals of Statistics **26** (1998) 1651–1686

16. Paugam-Moisy, H., Elisseeff, A., Guermeur, Y.: Generalization performance of multiclass discriminant models. In: Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, July 24-27, 2000, Volume 4, IEEE (2000)
17. Bennett, K.P., Cristianini, N., Shawe-Taylor, J., Wu, D.: Enlarging the margins in perceptron decision trees. Mach. Learn. **41** (2000) 295–313
18. Shawe-Taylor, J., Bartlett, P.L., Williamson, R.C., Anthony, M.: A framework for structural risk minimisation. In: COLT '96: Proceedings of the ninth annual conference on Computational learning theory, ACM Press (1996) 68–76
19. Ben-David, S., Cesa-Bianchi, N., Haussler, D., Long, P.M.: Characterizations of learnability for classes of {0, :., n}-valued functions. J. Comput. Syst. Sci. **50** (1995) 74–86
20. Natarajan, B.K.: On learning sets and functions. Mach. Learn. **4** (1989) 67–97
21. Vapnik, V.N., Chervonenkis, A.Y.: On the uniform convergence of relative frequencies of events to their probabilities. Theoret. Probi. and its Appl. **16** (1971) 264–280
22. Blumer, A., Ehrenfeucht, A., Haussler, D., Warmuth, M.K.: Learnability and the vapnik-chervonenkis dimension. J. ACM **36** (1989) 929–965
23. Phetkaew, T., Kijsirikul, B., Rivepiboon, W.: Reordering adaptive directed acyclic graphs for multiclass support vector machines. In: Proceedings of the Third International Conference on Intelligent Technologies (InTech 2002). (2002)
24. Klautau, A., Jevtić;, N., Orlitsky, A.: On nearest-neighbor error-correcting output codes with application to all-pairs multiclass support vector machines. J. Mach. Learn. Res. **4** (2003) 1–15
25. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. J. Mach. Learn. Res. **5** (2004) 101–141

The value of information on base classifier performance in multiclass-binary reduction algorithms

(Manuscript in preparation)

Jittat Fakcharoenphol* Boonserm Kijsirikul†

September 11, 2006

Abstract

In this paper, we study algorithms for reducing multiclass categorization problems to multiple binary problems which then are solved by base binary classifiers. Recently there are techniques that use the error information of the base classifiers to improve the performance of the reduction. The question regarding the value of this information, i.e., how much it helps improving the generalization performance of the resulting multiclass classifier, arises naturally. Using a game-theoretic framework, we give a lowerbound on the worst-case generalization error for any reductions. We also present a natural randomized multiclass-binary reduction algorithm that uses *no error information* and analyze its generalization error bound. The gap between the lowerbound and the error bound of this algorithm is a room for possible improvements. We describe examples of the error distributions among the base classifiers where the gap is small and large.

1 Introduction

Multiclass categorization problems are usually solved by a reduction to multiple binary problems. For problems with n classes, one class of algorithms trains $\binom{n}{2}$ binary classifiers, each distingushing between a pair of classes. Then given a query point $\boldsymbol{x}$, an algorithm in this class chooses a pair of classes, say i and j, then uses a trained binary classifier to determine which of the classes are more probable, and eliminates the losing class. There are, however, many ways of choosing a pair of classes, resulting in many proposed algorithms, e.g., DDAG [10], Randomized-DDAG [3], ADAG [7], and Reordering-ADAG [9]. In recent results of Phetkaew, Kijsirikul, and Rivepiboon [9], algorithms that explicitly use generalization error bounds for binary classifiers to pair up classes are presented. The question regarding how much this information helps arises.

In this paper, we introduce a game-theoretic framework for analyzing this class of algorithms that try to minimize the generalization errors using information on generalization error bounds of the binary classifiers. In particular, we give a lowerbound on the worst-case generalization error for any algorithms, and also present a randomized algorithm based on ADAG with a generalization error bound close to the lowerbound.

Section 2 introduces the class-pairing game and give the connection to the multiclass learning problem. We prove the lowerbound in Section 3, and present a randomized algorithm with its performance analysis in Section 4. Finally, in Section 5, we give families of examples where the gaps are small and large.

1.1 Related work

Analysis of algorithms under game-theoretic settings, especially for on-line algorithms, is well-known, e.g., see the book by Borodin and El-Yaniv [1]. In the area of learning theory, see results by Freund and Schapire [5], Breiman [2], and Grove and Schuurmans [6] on Boosting.

Generalization performance of multiclass classifiers that uses binary classifiers as subroutines has been analyzed in various settings, see, e.g., [4, 8, 10, 11].

*Kasetsart University, Bangkok, Thailand.
†Chulalongkorn University, Bangkok, Thailand.

2 Class-pairing game

We first define a more general two-player zero-sum game called *class-elimination game*. In this game, two players, Alice and Bob, are given a complete weighted graph $G = (V, E)$ with weight w on each edge is given. Let n denote $|V|$. For simplicity, we assume $V = \{1, \ldots, n\}$. The game proceeds in steps, in which one node is removed. from the graph. Let G_k, for $k = 0, \ldots, n-1$, be the graph after step k. Initially, $G_0 = G$. In step k, Alice picks an edge (i, j) in G_{k-1}. Then Bob chooses which of the two nodes, i or j, to be removed, i.e., G_k is G_{k-1} with one node, depending on Bob's choice, removed. The weight $w(i, j)$ is added to the score of the remaining node, which is zero initially. After step $n-1$, there are only one node left; it's score is the value of the game, which is also Bob's payoff. Since this is a zero-sum game, Alice's payoff is the negative of that value, i.e., Alice wants to minimize the score of the last node.

We illustrate the connections to the multiclass learning problem. In the multiclass learning problem, we want to classify data from set $\mathcal{X}$ into n classes. The learning algorithm for this problem outputs a hypothesis $h : \mathcal{X} \to [n]$, or usually classed a classifier, where $[n]$ denote $\{1, 2, \ldots, n\}$. We want h to predicts the correct label for a given data point. Formally, suppose that the data points and labels are sampled from distribution $\mathcal{D}$ over $\mathcal{X} \times [n]$, we call

$$\Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y]$$

the *generalization error* of h. A given multiclass learning problem induces $\binom{n}{2}$ binary problems, i.e., for each pair of classes i and j, a binary classification problem wants to distinguished data points from class i from data points from class j. A learning algorithm for a pair of classes (i, j) outputs a hypothesis $A_{i,j} : \mathcal{X} \to \{i, j\}$. The (binary) generalization error of $A_{i,j}$ is

$$\Pr_{(x,y)\sim\mathcal{D}}[A_{i,j}(x) \neq y | y \in \{i, j\}].$$

This learning environment can be modeled in the class elimination game as follows. We associate node i in G with class $i \in [n]$. The weight $w(i, j)$ in the graph represents the generalization error of $A_{i,j}$. Consider a classification algorithm A that uses classifiers $A_{i,j}$'s as subroutines. A would proceed in steps like this: given a data point x, A picks a pair of candidate classes i and j, and invokes $A_{i,j}$; A then eliminates either i or j from the candidate classes of x, depending on the output of $A_{i,j}$.

We first explain the idea on how one can analyze the generalization error using a restricted form of the class elimination game. We later prove the equivalence formally in Lemma 1.

We are given a data point $x \in X$, assume that the correct class is n. Note that for any classifier $A_{i,j}$ such that $n \neq \{i, j\}$, any result of $A_{i,j}(x)$ will not contradict its known generalization performance, since x is in one of its "don't care" classes. This gives Bob full power of choosing its result. On the other hand, for any classfiers $A_{p,n}$, we only allows Bob to return n, ensuring that the final class in the game is n; this reason for this will be clear shortly. In the analysis, we assume that all results of $A_{i,j}(x)$ for all $i, j \neq n$ are known, but have not revealed to Alice.

Consider playing the game along with the execution of the classification algorithm A. At step k, assume that n has not been eliminated. Alice start by choosing an edge e. If n is not one of e's end nodes, we reveal Bob's choice for it and continue on the next step. However, if n is adjacent to e, i.e., $e = (n, p_k)$ for some p, there is a chance that n gets eliminated. Let F_k be the event that n remains in the k-th round and gets eliminated in this round, and let E_p be the event that $A_{p_k,n}$ gives wrong prediction. We note that event F_k implies E_{p_k}, i.e., $F_k \subseteq E_{p_k}$. Note that class p, can be determined by assuming that in previous rounds all classifiers $A_{p',n}$ invoked make correct predictions. This is the reason why we assume that $A_{p',n}(x) = n$. Let K be the set of steps k such that classifiers $A_{p_k,n}$ is invoked.

To make a wrong prediction, event F_k, for some $k \in K$, must occur. Thus,

$$\begin{aligned}
\Pr[A(x) \neq n] &= \Pr[\bigcup_{k\in K} F_k] \\
&\leq \Pr[\bigcup_{k\in K} E_p] \\
&\leq \sum_{k\in K} \Pr[E_{p_k}] = \sum_{k\in K} w(p_k, n),
\end{aligned}$$

which is exactly Bob's payoff in the game. This implies that the generalization error is at most Bob's payoff, proving the first part of the following lemma.

Lemma 1 *For any probability space $\mathcal{P}$ consistent with the information on the generalization error of the binary classifiers, i.e.,*

$$\Pr_{(x,y)\sim\mathcal{P}}[A_{i,j}(x) \neq y | y \in \{i,j\}] = w(i,j),$$

if there is a strategy for Alice that guarantees a payoff at least $-\epsilon$ when Bob's strategy is non-adaptive, there is an algorithm A with generalization error at most ϵ. On the other hand, for any instance of the game, there is a probability space $\mathcal{P}$ and a set of binary classifiers satisfying the above performance constraint, such that the generalization error of A is at exactly Bob's payoff.

The proof of the first part is in the previous discussion. The proof for the second part will be provided in the final version.

Therefore, to analyze the generalization error of a multiclass-binary construction, we need only to analyze the game where the algorithm plays as Alice.

To analyze ADAG-like algorithm, we introduce another game, the *class-pairing game*. As in the class-elimination game, Alice and Bob are given a weighted complete graph with n nodes. With out loss of generality, we assume that n is a power of two, i.e., there is some integer ℓ such that $n = 2^\ell$. If n is not a power of two, we add dummy classes with zero adjacent edge weights. The game proceeds in ℓ rounds. For each round, Alice picks a perfect matching, and for each edge (i,j) in the matching, Bob removes one of the node. Therefore, for each round, half of the nodes are removed. This game is a class-elimination game where Alice picks many edges (to form a perfect matching) before Bob reveals his choices.

3 The lowerbound

In this section we prove the lowerbound for Bob's payoff. We give the strategy for Bob that would leave one node s in the graph. We first sort the nodes according the weights of the adjacent edges with s, i.e., we reindex the nodes so that $s = n$ and $w(n,i) \leq w(n,i+1)$ for $1 \leq i < n-1$. For brevity, we write $w(i)$ for $w(n,i)$. We call this sequence of non-decreasing weights $w(1), w(2), \ldots, w(n-1)$ a *weight sequence* of node n of length $n-1$. A weight sequence of round i is a weight sequence of length $2^{\ell-i+1}$.

We consider the *greedy strategy* for Bob: when Bob has to choose between class i and j, Bob always chooses the class with higher weight, i.e., Bob chooses $\max\{i,j\}$.

To give some intuition, we do a thought experiment with Alice strategy. Suppose that Alice knows the final node n. What can Alice do to minimize her loss? Clearly, Alice wants to pair up nodes with small weights with n. For other nodes, if she matches i with j, for $i > j$, she can be sure that j would be eliminated, removing weight $w(j)$ from the game. One strategy for her is to pair up 1 with n, and 2 with 3, 4 with 5, and so on. Finally, $n-2$ is paired up with $n-2$. Therefore, in this round, her loss is $w(1)$. If she continues with this, her total loss will be $w(1) + w(3) + w(7) + \cdots + w(n-1) = \sum_{i=1}^{\ell} w(2^i - 1)$.

The following lemma shows that this is the best Alice can do.

Theorem 1 *If Bob follows the greedy strategy, Bob can always get at least $\sum_{i=1}^{\ell} w(2^i - 1)$.*

Proof: We prove by induction on ℓ. The base case where $\ell = 1$ is clear. We shall prove the inductive step.

Let $n = 2^\ell$, for $\ell > 1$. Let $C = \{1, 2, \ldots, n\}$. Consider any perfect matching M that Alice choose. Given this choice of Alice, the set of classes C' after this round is $\{\max\{a,b\} : (a,b) \in M\}$. Note that the goal class n is in C' since it has the highest index. We treat C' as an increasing sequence. Therefore, the weight sequence u of the next round is $u(i) = u(C'(i))$ for $i = 1, \ldots, n/2 - 1$.

Let p denote the class matched with n in M. Consider another weight sequence $\tilde{u}$ of length $n/2 - 1$, obtained from another matching M', defined as $\tilde{u}(i) = w(2i)$ for $1 \leq i < p/2$ and $\tilde{u}(i) = w(2i+1)$ for $p/2 \leq i < n/2$. We claim that u dominates $\tilde{u}$, i.e., $u(i) \geq \tilde{u}(i)$ for all $1 \leq i < n/2 - 1$. To see this, consider any index i. Note that $C'(i)$ is larger than $i-1$ larger indices in the set of disjoint pairs from $C - \{p\}$. Therefore, $|\{j \in C - \{p\} : j < C'(i)\}| \geq 2(i-1) + 1$, i.e., $C'(i) \geq 2i$ if $C'(i) < p$, and $C'(i) \geq 2i + 1$ if $C'(i) > p$. The claim follows from the definition of $\tilde{u}$. Figure 1 illustrates this claim.

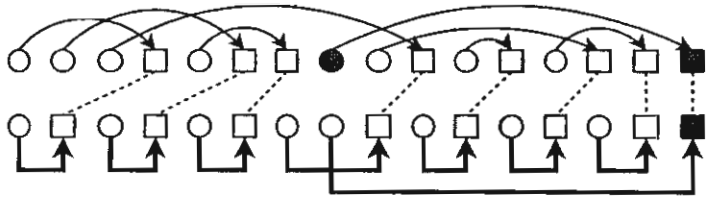

Figure 1: Matchings M (top) and M' (bottom). Note that the weight sequence u of M dominates $\tilde{u}$.

We now prove the theorem for the case that the weight sequence of the next round is $\tilde{u}$. The theorem follows from the fact that u dominates $\tilde{u}$.

Since $\tilde{u}$ is a weight sequence of length $2^{\ell-1}-1$, by the induction hypothesis, Bob can get the payoff at least $\sum_{j=1}^{\ell-1} \tilde{u}(2^j-1)$. Together with the payoff Bob receives in this round, Bob gets

$$w(p) + \sum_{j=1}^{\ell-1} \tilde{u}(2^j-1).$$

Let $q = \lfloor \log(p+1) \rfloor$. We show that this is at least $\sum_{i=1}^{\ell} w(2^i-1)$ by breaking the sum into three cases around q as follows.

$$\begin{aligned}
\sum_{i=1}^{\ell} w(2^i-1) &= \sum_{i=1}^{q-1} w(2^i-1) + w(2^q-1) + \sum_{i=q+1}^{\ell} w(2^i-1) \\
&\leq \sum_{i=1}^{q-1} \tilde{u}(2^i-1) + w(p) + \sum_{i=q+1}^{\ell} w(2^i-1) \\
&= \sum_{i=1}^{q-1} \tilde{u}(2^i-1) + w(p) + \sum_{i=q}^{\ell-1} w(2^{i+1}-1) \\
&= \sum_{i=1}^{q-1} \tilde{u}(2^i-1) + w(p) + \sum_{i=q}^{\ell-1} \tilde{u}(2^i-1) \\
&= w(p) + \sum_{j=1}^{\ell-1} \tilde{u}(2^j-1).
\end{aligned}$$

The theorem follows. ∎

4 The upperbound

In this section, we introduced a randomized strategy for Alice for the class-matching game: for each round Alice picks a random perfect matching. A simple exchange argument shows that for this randomized strategy, the greedy strategy is the best non-adaptive one Bob can play. More precisely, suppose that finally, Bob chooses class i as the last remaining class. The best way he should have chosen is to play greedy with i as the solution class from the beginning.

Without loss of generality, suppose that Bob choose n as the solution class. We first analyze the probability that class i is not eliminated after k rounds. We can view the game as a component merging process. Initially, each node is a singleton component. For each round, components are paired up, and they are merged. The classes left in each round are ones which have the highest indices in their components. After round k, $n/2^k$ components are left, each of size 2^k. Therefore, class i is not eliminated if i is the node with the highest index in the component containing i. Since the merging is random, any components containing i is equally likely. There are $\binom{n-1}{2^k-1}$ possible components containing i, while

there are, however, $\binom{i-1}{2^k-1}$ components where i is the highest index. Let $A_{i,k}$ be the event that i is not eliminated after round k. Hence,

$$\Pr[A_{i,k}] = \frac{\binom{i-1}{2^k-1}}{\binom{n-1}{2^k-1}} = \frac{(i-1)}{(n-1)} \cdot \frac{(i-2)}{(n-2)} \cdots \frac{(i-2^k+1)}{(n-2^k+1)}.$$

Note that if $i < 2^k$, $q_{i,k} = 0$.

Let even $B_{i,k}$ denote the probability that class i contribute to Bob's payoff in round k, i.e., in round k, edge (i, n) is chosen. We have that

$$\begin{aligned}
\Pr[B_{i,k}] &= \Pr[B_{i,k}|A_{i,k-1}] \cdot \Pr[A_{i,k-1}] \\
&= \frac{1}{n/2^{k-1}-1} \cdot Pr[A_{i,k-1}] \\
&= \frac{1}{\frac{n-2^{k-1}}{2^{k-1}}} \cdot \frac{\binom{i-1}{2^{k-1}-1}}{\binom{n-1}{2^{k-1}-1}} \\
&= \frac{\binom{i-1}{2^{k-1}-1}}{\binom{n-1}{2^{k-1}}}
\end{aligned}$$

Events $B_{i,k}$ for $1 \le k \le \ell$ are disjoint. Denote by p_i the probability that class i contributes to Bob's payoff in any round. We have

$$\begin{aligned}
p_i &= \Pr[B_{i,1} \cup B_{i,2} \cup \cdots \cup B_{i,\ell}] \\
&= \sum_{k=1}^{\ell} \Pr[B_{i,k}] \\
&= \sum_{k=1}^{\ell} \frac{\binom{i-1}{2^{k-1}-1}}{\binom{n-1}{2^{k-1}}}
\end{aligned}$$

Hence, the expected payoff for Bob is $\sum_{i=1}^{n-1} p_i w(i)$. The following thorem summarizes the above discussion.

Theorem 2 *If Alice uses the random matching strategy, the expected payoff for Bob is*

$$\sum_{i=1}^{n-1} w(i) \left(\sum_{k=1}^{\ell} \frac{\binom{i-1}{2^{k-1}-1}}{\binom{n-1}{2^{k-1}}} \right)$$

5 The gaps

In the later discussion, we let LB denote the lowerbound for Bob's payoff from Section 3, and UB denote the upperbound on the expected payoff for Bob when Alice plays randomized strategy from Section 4.

In this section, we first prove a few useful properties of LB and UB. Then, we analyze the gaps when the weights are independently sampled from uniform distributions and normal distributions. We consider both the additive gap ($UB - LB$), and the multiplicative ratio (UB/LB).

5.1 Basic properties

Fact 1 *The following are true.*

1. $\frac{UB}{LB} \le \frac{\log n \cdot w(n-1)}{w(n-1)+(\log n-1)\cdot w(1)} \le \frac{w(n-1)}{w(1)}$.

2. $\frac{UB}{LB} \leq \ell = \log n$.

3. *For any* k, $\mathbf{E}[UB] \geq \frac{n-k-1}{n-1} \log n(\mathbf{E}[w(k)])$, *where expectation is taken over the weights.*

4. $\sum_{i=1}^{j} \Pr[B_{i,k}] = \sum_{i=1}^{j} \frac{\binom{i-1}{2^{k-1}-1}}{\binom{n-1}{2^k-1}} = \frac{\binom{j}{2^k-1}}{\binom{n-1}{2^k-1}}$.

5. *Let* $q_{j,k}$ *be the probability that some node* i, *for* $i \leq j$, *is paired up with* n *in the randomized strategy for Alice. If* $j \leq n/2^{1/k}$, $q_{j,k} \leq 1/2$.

We omit the proof in this extended abstract.

5.2 Uniform distribution

Suppose that the weights are independently sampled uniformly from $(0, 1)$. It is known that for the expectation of the k-smallest of $n-1$ such random variables is k/n, i.e., $\mathbf{E}[w(k)] = k/n$.

Applying Theorem 1, we have

$$\begin{aligned}
\mathbf{E}[LB] &= \mathbf{E}[\sum_{i=1}^{\ell} w(2^i - 1)] \\
&= \sum_{i=1}^{\ell} \mathbf{E}[w(2^i - 1)] \\
&= \sum_{i=1}^{\ell} (2^i - 1)/2^\ell \\
&= (2 \cdot 2^\ell - 2 - \ell)/2^\ell \\
&< 2.
\end{aligned}$$

From Fact 1 (3), we know that $\mathbf{E}[UB] \geq (\log n)/2$, since $\mathbf{E}[w(n/2)] = 1/2$. Hence, $E[UB - LB] \geq (\log n)/2 - 2 = \Omega(\log n)$.

5.3 Normal distribution

We consider the case that weights are sampled uniformly from $\mathcal{N}(\mu, \sigma^2)$, the normal distribution with mean μ and varience σ^2, and bound the expected gap between $w(n-1)$ and $w(1)$. It is not difficult to show that for a normally distributed random variable $X \sim \mathcal{N}(\mu, \sigma^2)$, there is a constant c such that the probability that $X \geq \mu + c\sigma\sqrt{\ln n}$ is $1/n^2$. By union bound,

$$\Pr[w(n-1) \geq \mu + c\sigma\sqrt{\ln n}] \leq 1/n.$$

Similarly,

$$\Pr[w(1) \leq \mu - c\sigma\sqrt{\ln n}] \leq 1/n,$$

since the distribution is symmetric. Thus, with probability at least $1 - 2/n$, $\frac{w(n-1)}{w(1)} \leq \frac{\mu + c\sigma\sqrt{\ln n}}{\mu - c\sigma\sqrt{\ln n}}$. Otherwise, with probability at most $2/n$, $\frac{UB}{LB} \leq \log n$, from the worst case bound, contributing at most $\frac{2\log n}{n} = o(1)$ to the expectation. Therefore,

$$\mathbf{E}\left[\frac{UB}{LB}\right] \leq \frac{\mu + c\sigma\sqrt{\ln n}}{\mu - c\sigma\sqrt{\ln n}} + o(1).$$

References

[1] A. Borodin and R. El-Yaniv. *Online computation and competitive analysis.* Cambridge University Press, New York, NY, USA, 1998.

[2] L. Breiman. Prediction games and arcing algorithms. *Neural Computation*, 11(7):1493–1517, 1999.

[3] J. Fakcharoenphol. A note on random DDAG. Manuscript, Sep 2003.

[4] J. Fakcharoenphol and B. Kijsirikul. Constructing multiclass learners from binary learners: A simple black-box analysis of the generalization errors. In *ALT*, pages 135–147, 2005.

[5] Y. Freund and R. E. Schapire. Game theory, on-line prediction and boosting. In *COLT '96: Proceedings of the ninth annual conference on Computational learning theory*, pages 325–332, New York, NY, USA, 1996. ACM Press.

[6] A. J. Grove and D. Schuurmans. Boosting in the limit: maximizing the margin of learned ensembles. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 692–699, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.

[7] B. Kijsirikul, N. Ussivakul, and S. Meknavin. Adaptive directed acyclic graphs for multiclass classification. In *PRICAI 2002*, pages 158–168, 2002.

[8] H. Paugam-Moisy, A. Elisseeff, and Y. Guermeur. Generalization performance of multiclass discriminant models. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, July 24-27, 2000, Volume 4*. IEEE, 2000.

[9] T. Phetkaew, B. Kijsirikul, and W. Rivepiboon. Reordering adaptive directed acyclic graphs for multiclass support vector machines. In *Proceedings of the Third International Conference on Intelligent Technologies (InTech 2002)*, 2002.

[10] J. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In *Advance in Neural Information Processing System*, volume 12. MIT Press, 2000.

[11] R. E. Schapire, Y. Freund, P. L. Bartlett, and W. S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26(5):1651–1686, 1998.