Integer Programming-based Error-Correcting Output Code Design for Robust Classification: Supplementary material

Samarth Gupta[1] **Saurabh Amin**[2]

[1]Center for Computational Science and Engineering, Massachusetts Institute of Technology, USA
[2]Laboratory for Information & Decision Systems, Massachusetts Institute of Technology, USA

{samarthg, amins}@mit.edu

A PROOFS

Lemma 1. *The feasible space enclosed by the constraints constituting the edges of any clique $\mathcal{C}$ in $\mathcal{G}_p^{inf}$ is same as that enclosed by the constraint:*

$$\sum_{i \in \mathcal{C}} x_i \leq 1. \tag{16}$$

Proof: We use mathematical induction to show that the result holds for any clique $\mathcal{C}_n$, of size $n \geq 3$. Assume that the theorem holds for a clique of size $n-1$. We know that a clique $\mathcal{C}_n$ (size n) contains n distinct cliques $\mathcal{C}^1_{n-1}, \ldots, \mathcal{C}^n_{n-1}$ of size $n-1$ such that $\mathcal{C}^1_{n-1} \cup \mathcal{C}^2_{n-1} \ldots \mathcal{C}^{n-1}_{n-2} \cup \mathcal{C}^n_{n-1} = \mathcal{C}_n$. Under induction hypothesis, we can write the following set of n equations:

$$\begin{array}{cccccccccc}
x_1 & + & x_2 & + & \cdots & + & x_{n-1} & & & \leq 1 \\
x_1 & + & x_2 & + & \cdots & + & & + & x_n & \leq 1 \\
\vdots & & & & \vdots & & & & \vdots & \vdots \\
x_1 & + & & & \cdots & + & x_{n-1} & + & x_n & \leq 1 \\
 & & x_2 & + & \cdots & + & x_{n-1} & + & x_n & \leq 1
\end{array}$$

Adding these equations, we obtain:

$$x_1 + x_2 + \cdots + x_{n-1} + x_n \leq \frac{n}{n-1} \tag{17}$$

For $n \geq 3$, we know the following trivial bound:

$$\frac{n}{n-1} < 2 \tag{18}$$

Using (17) and (18):

$$x_1 + x_2 + \cdots + x_{n-1} + x_n < 2 \tag{19}$$

Since $x_i \in \{0,1\} \ \forall i \in \{1, \ldots, n\}$:

$$x_1 + x_2 + \cdots + x_{n-1} + x_n \in \mathbb{Z}^+ \cup \{0\} \tag{20}$$

Using (19) and (20):

$$x_1 + x_2 + \cdots + x_{n-1} + x_n \leq 1 \tag{21}$$

Thus, (16) holds for a clique of size n. To complete the induction argument, we need to show that the result holds for $n = 3$.

For $n = 3$, we have:

$$\begin{aligned} x_1 + x_2 &\leq 1 \\ x_1 + x_3 &\leq 1 \\ x_2 + x_3 &\leq 1 \end{aligned}$$

Summing the above equations, we get:

$$x_1 + x_2 + x_3 \leq 1.5$$

Again, since $x_i \in \{0, 1\} \; \forall i \in \{1, 2, 3\}$:

$$x_1 + x_2 + x_3 \in \mathbb{Z}^+ \cup \{0\},$$

and we conclude that:

$$x_1 + x_2 + x_3 \leq 1.$$

Therefore, the result also holds for $n = 3$. ∎

Lemma 2. *The feasible space enclosed by the constraint set $\mathcal{S}_p^{inf}$ (or its graphical equivalent $\mathcal{G}_p^{inf}$) in $\mathcal{IP}2$ is same as that enclosed by a much smaller constraint set formed by $\mathcal{ECC}(\mathcal{G}_p^{inf})$.*

Proof: Since the graph $\mathcal{G}_p^{inf}$ does not contain any isolated nodes and loops, and every edge in $\mathcal{G}_p^{inf}$ is covered in atleast one clique, therefore we can write:

$$\bigcup_{i=1}^{k} \mathcal{C}_i = \mathcal{G}_p^{inf}.$$

Also, $\mathcal{G}_p^{inf} \equiv \mathcal{S}_p^{inf}$, therefore $\{\mathcal{C}_1, \ldots, \mathcal{C}_k\} \equiv \mathcal{S}_p^{inf}$. ∎

Lemma 3. *Suppose $\mathcal{G}_1, \ldots \mathcal{G}_m$ are edge-disjoint subgraphs of $\mathcal{G}_p^{inf}$, such that:*

1. $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset \;\; \forall \; i, j \in \{1, \ldots, m\}^2 | i < j$
2. $\bigcup_{i=1}^{m} \mathcal{G}_i = \mathcal{G}_p^{inf}$

The union of the edge clique covers of individual subgraphs $\mathcal{G}_1, \ldots \mathcal{G}_m$ is a valid edge clique cover of $\mathcal{G}_p^{inf}$:

$$\bigcup^{m} \mathcal{ECC}(\mathcal{G}_i) = \mathcal{ECC}(\mathcal{G}_p^{inf}).$$

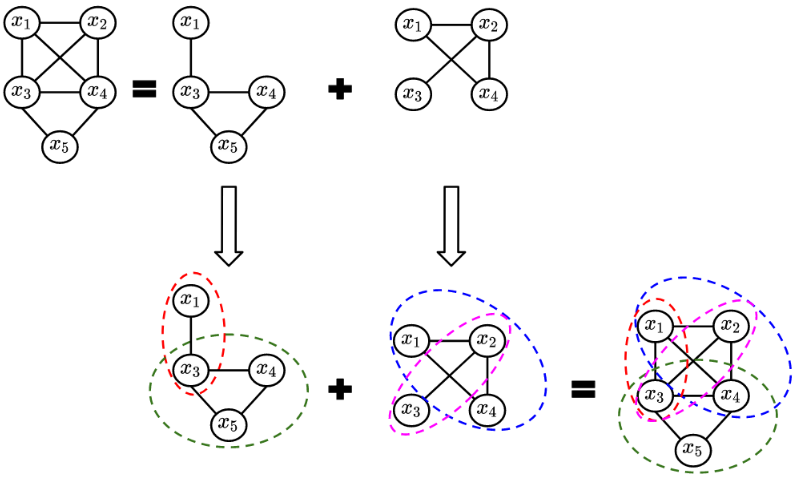


Figure 5: Edge Clique Cover generated by combining the edge clique covers of the individual subgraphs.

Proof: Recall form the definition of Edge Clique Cover, a set of cliques is a valid edge-clique-cover of a given graph, if the following two requirements are satisfied by the clique set:

I. Every edge of the graph is covered in atleast one clique.

II. No clique is completely contained in another clique.

Consider the following arguments:

1. For any $i \in \{1, \dots m\}$, $\mathcal{ECC}(\mathcal{G}_i)$ is a valid edge-clique-cover for subgraph $\mathcal{G}_i$.
2. Every edge in $\mathcal{G}_p^{inf}$ is covered in atleast one subgraph as $\bigcup_{i=1}^{m} \mathcal{G}_i = \mathcal{G}_p^{inf}$, therefore every edge in $\mathcal{G}_p^{inf}$ is contained in atleast one of the cliques in the set: $\bigcup_{i=1}^{m} \mathcal{ECC}(\mathcal{G}_i)$. Thus requirement I is satisfied.
3. For a clique to be completely contained in another clique, there should be atleast one common edge between any two distinct subgraphs $\mathcal{G}_i$ and $\mathcal{G}_j$. Since, the subgraphs are edge-disjoint, i.e. $\mathcal{G}_i \cap \mathcal{G}_j = \emptyset$, therefore no clique can be completely contained in another clique. Thus requirement II is also satisfied. ■

B A SAMPLE EXHAUSTIVE CODE

Table 11: Exhaustive code (all possible valid columns) for $k = 5$

Classes	Codewords														
	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	-1	-1	-1	-1	-1	-1	-1	-1	1	1	1	1	1	1	1
3	-1	-1	-1	-1	1	1	1	1	-1	-1	-1	-1	1	1	1
4	-1	-1	1	1	-1	-1	1	1	-1	-1	1	1	-1	-1	1
5	-1	1	-1	1	-1	1	-1	1	-1	1	-1	1	-1	1	-1

C CODEBOOK DESIGN CRITERIA

We provide more details about the balanced column and the data distribution criteria:

C.1 BALANCED COLUMNS

The balanced columns criteria ensures that out of $2^{k-1} - 1$ columns in the exhaustive code $\mathcal{M}$, those columns are preferred (in the final solution) for which the resulting binary classification problem has similar number of data-points in each of the two resulting classes. For a k class classification problem let $N_1, \dots N_k$ be the number of training data points in each class.

For a column l, let I denote the set of classes for which $\mathcal{M}(\cdot, l) = 1$ and J denote the set of classes for which $\mathcal{M}(\cdot, l) = -1$. The binary classification problem resulting from column l will have the following number of data points in each class:

$$N_l^+ = \sum_{i:\, \mathcal{M}(i,l)=1} N_i \qquad \text{(no. of training examples in the positive class)}$$

$$N_l^- = \sum_{i:\, \mathcal{M}(i,l)=-1} N_i \qquad \text{(no. of training examples in the negative class)}$$

We would like to select columns for which $N_l^+ \approx N_l^-$, i.e. the each resulting class has similar number of training points. Ideally, we would like to only select columns with minimum possible value of $|N_l^+ - N_l^-|$, however for smaller k, this may be too restrictive, therefore we set a threshold BC_{max}, i.e. columns for which $|N_l^+ - N_l^-| \leq BC_{max}$ are considered and the remaining columns are removed from $\mathcal{M}$.

C.2 DATA DISTRIBUTION

In the main text we outlined that the requirement of data-distribution can be incorporated by changing the objective function in $\mathcal{IP}3$ to following:

$$\min_{x_i} \sum_{(p,q)\in\{1,\dots,k\}^2 | p<q} |d_H^{p,q}(x_i) - \hat{d}^{p,q}|$$

where $\hat{d}^{p,q}$ represents the desired class-pairwise hamming distance between classes p and q computed with a similarity measure.

Let $\mathcal{X}_p = \{X_1^p, \ldots, X_{|\mathcal{X}_p|}^p\}$ and $\mathcal{X}_q = \{X_1^q, \ldots, X_{|\mathcal{X}_q|}^q\}$ denote the training samples for classes p and q respectively. The similarity measure is calculated as:

$$S_{pq} = \frac{1}{|\mathcal{X}_p|} \frac{1}{|\mathcal{X}_q|} \sum_{i=1}^{|\mathcal{X}_p|} \sum_{j=1}^{|\mathcal{X}_q|} \mathcal{K}(X_i^p, X_j^q)$$

where $\mathcal{K}(\cdot, \cdot)$ is a Mercer's kernel.
In our experiments using the above similarity measure (with rbf kernel) we did not see a significant improvement on MNIST dataset. This is mainly because the above similarity measure does not correctly identify class-pairs which are hard to separate from the ones which are easily separable. As future work, we aim to use more refined similarity measures such as the margin of a trained SVM classifier and more recent similarity measure proposed by Wan et al. [2020]. The measure proposed by Wan et al. [2020] is computationally efficient, hence suited for large datasets. Further, this similarity measure has provided significant gains in performance in [Wang et al., 2020] on a related problem of verification of robust classifiers.

D EDGE-CLIQUE-COVER HEURISTICS

To find the edge-clique-cover of the conflict graph $\mathcal{G}_p^{inf}$ we use the heuristics proposed by Conte et al. [2016]. For a network with n vertices and m edges, this heuristic requires $\mathcal{O}(m+n)$ space and the time cost is upper bounded by $\mathcal{O}(m\Delta)$, where Δ is the maximum degree of the network. The actual runtime is linear in m. Other state-of-the-art heuristics in literature such as Gramm et al. [2009] requires $\mathcal{O}(n^2)$ memory space and $\mathcal{O}(mn)$ time.

E DENSE/SPARSE CODES

Random codes is another way of generating codebooks as proposed in Allwein et al. [2000]. Here, authors propose generating 10000 matrices, whose entries are randomly selected. If the elements are chosen uniformly at random from $\{+1, -1\}$, then the resulting codebooks are called **dense codes** and if the elements are taken from $\{-1, 0, +1\}$ then the resulting codebooks are called **sparse codes**. In sparse codes, 0 is chosen with probability $1/2$ and ± 1 are each chosen with probability $1/4$. Out of the $10,000$ random matrices generated, after discarding matrices which do-not constitute a valid codebook, *the one with the largest minimum Hamming distance among rows is selected.* Note that since out of the $10,000$ matrices the one with the *largest minimum Hamming distance* is selected, therefore despite the matrices being generated randomly, the final codebook *can have high row-separation.*

F DETAILS ABOUT REAL-WORLD DATASETS (SMALL/MEDIUM)

In Table 12 we provide details about *Glass*, *Ecoli* and *Yeast* datasets taken from the UCI repository Dua and Graff [2017].

Table 12: Real-world Dataset Characteristics

	# of samples	# of features	# of classes (k)
Glass	214	9	6
Ecoli	336	7	8
Yeast	1484	8	10

G CLASS PROBABILITY ESTIMATES

In section 5.2 under Adversarial Robustness, we discussed how class probability estimates enable us to estimate the adversarial robustness of ECOC based classifiers using white-box attacks. For binary codebooks we obtain class probability estimates using the procedure from Zadrozny [2002], Hastie and Tibshirani [1998]. After evaluating an input $\tilde{x}$ on each binary classifier, we obtain a probability estimate, denoted $r_l(\tilde{x})$, for each column l (i.e., binary classifier) in $\mathcal{M}$. Let I

denote the set of classes for which $\mathcal{M}(\cdot, l) = 1$ and J denote the set of classes for which $\mathcal{M}(\cdot, l) = -1$. Then the class probability estimate for $i \in \{1, \dots k\}$ on an input $\tilde{x}$ is given as follows:

$$\hat{p}_i(\tilde{x}) = \sum_{l:\ \mathcal{M}(i,l)=1} r_l(\tilde{x}) + \sum_{l:\ \mathcal{M}(i,l)=-1} (1 - r_l(\tilde{x}))\,, \tag{22}$$

where differentiability with respect to $\tilde{x}$ is maintained.

The above estimates work well for binary codes, however we need to be careful for ternary (or sparse) codes. For ternary codes, hypotheses which have zero (for a particular class) do-not contribute to the above sum in (22). Therefore due to zero entries, estimates for different classes can significantly vary in relative magnitude. This can be easily fixed by simple normalization. Raw estimates in (22) can be normalized as follows:

$$\hat{p}^*_i(\tilde{x}) = \frac{1}{\sum_{l=1}^{L} \mathbb{1}_{\{\ \mathcal{M}(i,l)=1 \vee \mathcal{M}(i,l)=-1\ \}}} \Bigg(\sum_{l:\ \mathcal{M}(i,l)=1} r_l(\tilde{x}) + \sum_{l:\ \mathcal{M}(i,l)=-1} (1 - r_l(\tilde{x}))\ \Bigg), \tag{23}$$

where $\mathbb{1}_{\{\pi\}}$ is the indicator function which evaluates to 1 when the predicate π is true and 0 otherwise.

In Figure 6, we show how these estimates can be computed for 1-vs-1 codebook, when working with binary deep neural networks. Since, each row in 1-vs-1 has the same number of zeros therefore normalization is not necessary.

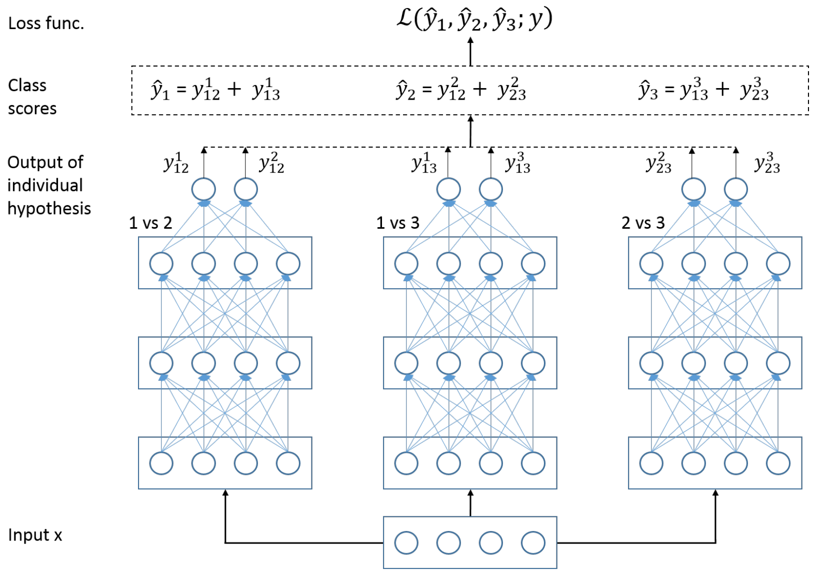

Loss func.
$\mathcal{L}(\hat{y}_1, \hat{y}_2, \hat{y}_3; y)$
Class scores
$\hat{y}_1 = y^1_{12} + y^1_{13}$
$\hat{y}_2 = y^2_{12} + y^2_{23}$
$\hat{y}_3 = y^3_{13} + y^3_{23}$
Output of individual hypothesis
y^1_{12}
y^2_{12}
y^1_{13}
y^3_{13}
y^2_{23}
y^3_{23}
1 vs 2
1 vs 3
2 vs 3
Input x

Figure 6: Combining output of individual hypotheses of 1-vs-1 to generate class scores while maintaining differentiability.

H ADVERSARIAL ACCURACY AND DIFFERENT TYPES OF ATTACKS

For ECOC based classifiers, evaluation of natural or clean accuracy over an example (generally from test-set) is straightforward, and can be easily done either by using a decoding scheme such as Hamming decoding or by calculating class probability estimates and choosing the class with the highest probability.

We now mathematically define the problem of evaluating the adversarial accuracy using class probability estimates. Suppose c be the true class associated with a given input x' and let $i \in \{1, \dots, k\}/\{c\}$ be the target class for which the attacker is trying to generate an adversarial perturbation. Attacker aims to solve the following non-convex problem:

$$f^*(x') = \max_{\delta:\ x'+\delta\ \in\ \mathcal{Q}(x')} \hat{p}_i(x' + \delta) - \hat{p}_c(x' + \delta) \tag{24}$$

In (24), set $\mathcal{Q}(x')$ for l_∞-norm based perturbations is given as follows:

$$\mathcal{Q}(x') = \{x \in R^d \bigm| \ ||x - x'||_\infty \leq \epsilon \ ; \ x'_l \leq x \leq x'_u\}.$$

For a valid[1] adversarial perturbation δ, the objective function value of (24) would be strictly positive for some target class i. Different attacks such as black-box and white-box attacks attempt to solve the above outlined problem (24) under different settings (or threat model).

In *black-box* setting, only the output of the classifier i.e. the class probabilities or score of each class is known to the attacker. No model information is available to the attacker, i.e. the network architecture and the weights of the network. In this setting, since only class probability estimates are available, therefore analytical computation of gradients is not possible. The problem is generally solved using off-the-shelf black-box optimizers comprising of heuristics based algorithms such as Particle Swarm Optimization (PSO), Genetic Algorithms (GAs) etc. However, given the efficacy of gradient based attacks, one can also try to compute an estimate of the gradient and then use this estimate to run gradient-based attacks, for details see Ilyas et al. [2018]. SPSA proposed in Spall [1992] is another black-box optimization method which is based on gradient estimation.

In *white-box* setting, the class probability estimates along with the model architecture and weights are known to the attacker. White-box setting can also be referred to as complete information setting. In white-box setting, the projected gradient descent or the PGD-attack proposed in Madry et al. [2018] has emerged as one of the strongest known attack. Another popular gradient based attack known as Fast-Gradient-Sign method (FGSM) was proposed in Goodfellow et al. [2015]. FGSM can be viewed as simply a single step PGD attack and therefore is a much weaker attack in comparison to PGD-attack.

Given the non-concave nature of the problem (24), the above attacks do not provide any guarantee in terms of finding the optimal solution, and mainly aims at finding a feasible solution to (24) with positive objective function value. If these attacks fail in generating an adversarial perturbation (especially if the attack is weak), we conclude that the model is robust against that particular attack. Therefore, to estimate the adversarial robustness accurately it is important to evaluate against strongest possible attack.

I COMPARISON WITH MULTICLASS CNN

In section 5.2 we compared the adversarial robustness of our IP generated codebook $\mathcal{IP}3$ with other standard codebooks such as 1-vs-1,1-vs-All, Sparse and Dense codes. We reported our results on MNIST and CIFAR10 datasets in Tables 9 and 10 respectively. We note that our IP generated codebook achieves non-trivial robustness *without any adversarial training*. On CIFAR10, our codebook outperforms all other standard codebooks, achieving an adversarial accuracy of $\sim 16\%$ with $\epsilon = 8/255$. However, given that we are combining the output of 20 binary classifiers, each of which is a ResNet-18, a natural question arises:

Is network capacity (of the overall classifier) the main reason for this robustness?

Recall that to evaluate the robustness we combine the outputs of each of the hypotheses (individually trained before) using our IP generated codebook and form a multi-class classifier. Figure 6 shows this for 1-vs-1 codebook for 3 classes. We then do a PGD based evaluation of the resulting multiclass classifier. To investigate the role of network capacity, we now in the same manner, combine 20 *untrained* hypotheses (ResNet-18) to form a multi-class classifier (say $\mathcal{F}(\tilde{x})$). We now nominally train this 10-class classifier $\mathcal{F}(\tilde{x})$ *end-to-end* using the entire training set. $\mathcal{F}(\tilde{x})$ has exactly the same network architecture and capacity as our multiclass ECOC based classifier resulting from our IP generated codebook.

We now evaluate the adversarial accuracy of $\mathcal{F}(\tilde{x})$ using the same PGD attack which we used for different codebooks including our IP generated codebook. We report our results in the last row of Tables 9 and 10 with type as *Multiclass*. The lack of robustness of $\mathcal{F}(\tilde{x})$ or *Multiclass* shows that network capacity alone in itself is not the reason for robustness of IP generated ECOC based classifier.

Finally, we note that since the individual untrained hypotheses are combined using a codebook in the final layer, therefore $\mathcal{F}(\tilde{x})$ is similar to the approach taken in Verma and Swami [2019].

[1] An adversarial perturbation δ does not necessarily need to be the $\arg\max$ of (24)

J ESTIMATING ERROR-CORRELATION BETWEEN INDIVIDUAL HYPOTHESES OF A CODEBOOK

In our discussion in section 3, we highlighted that in communicating over a noisy channel, Error-Correcting Codes are powerful only when the errors made due to noise are random. For classification setup like ours, this implies that any two hypotheses (or classifiers) should not make errors on the same inputs. To avoid this, we ensured large column separation in our IP formulation. However, we may still end up with hypotheses whose final predictions (or errors) are correlated. Therefore, measurement of such pairwise correlations between hypotheses can provide us with insights to better understand the final performance of a particular codebook. Moreover, it also will provide us with corroborative evidence to the fact that correlation between hypotheses should be avoided.

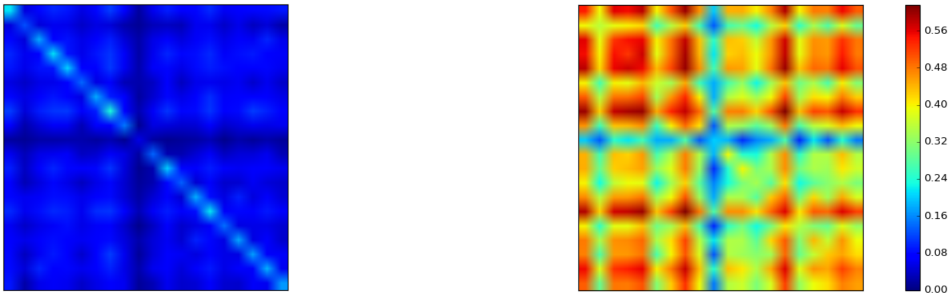
Assuming that we have already trained each of our individual hypotheses for a given codebook. Also, let N_{test} denote the number of images in our test-set. For every binary classifier (corresponding to a column) in the codebook, we can compute the 0-1 loss for all images in the test set so that we have a vector $h_l \in \{0,1\}^{N_{test}} \ \forall \ l \in \{1, \dots, L\}$. We can now compute the error-correlations between these binary vectors h_i & $h_j \ \forall \ (i,j) \ \in \ \{1, \dots, L\}^2$. This can be represented in a $L \times L$ matrix, which we will refer to as the correlation matrix (denoted as $\mathcal{P}$) in our subsequent discussion. We propose the following measure:

$$\mathcal{P}_{i,j} = \frac{\sum_{n=1}^{N_{test}} \mathbb{1}\{h_i[n] = 1 \wedge h_j[n] = 1\}}{\sum_{n=1}^{N_{test}} \mathbb{1}\{h_i[n] = 1 \wedge h_j[n] = 1\} + \sum_{n=1}^{N_{test}} \mathbb{1}\{h_i[n] = 0 \wedge h_j[n] = 0\}}, \tag{25}$$

where $\mathbb{1}\{\pi\}$ is the indicator function which evaluates to 1 when the predicate π is true and 0 otherwise.

The above measure (25) accounts for both the correct and incorrect predictions made by individual hypotheses. The magnitude of this error-correlation measure (or the values in the error-correlation matrix $\mathcal{P}$) will help us in understanding the accuracy of the overall classifier or codebook.

We estimate the error-correlation matrix using the natural images from CIFAR10 dataset for the nominally trained hypotheses of our IP generated codebook. For the same hypotheses, we also estimate the error-correlation matrix using the adversarial images obtained from the PGD-attack with $\epsilon = 8/255$ on the overall classifier. We plot both the matrices in Figure 7.



(a) Natural (Accuracy: 76.25 %)

(b) Adversarial (Accuracy:16.48%)

Figure 7: Error-Correlation matrices estimated using the hypotheses of the IP-generated codebook on natural and adversarial images of CIFAR10 dataset.

From Figure 7, we note that the error-correlation values on the natural and adversarial dataset differ by almost an order of magnitude. On natural images, much higher accuracy is achieved as the error-correlation is low, while on adversarial images higher error-correlation values result in lower accuracy. Therefore for higher accuracy, error-correlation between hypotheses should be avoided.

References

Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

Alessio Conte, Roberto Grossi, and Andrea Marino. Clique Covering of Large Real-World Networks. In *31st Annual ACM Symposium on Applied Computing (SAC 2016)*, pages 1134–1139, Pisa, Italy, April 2016. ACM.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Jens Gramm, Jiong Guo, Falk Hüffner, and Rolf Niedermeier. Data reduction and exact algorithms for clique cover. *ACM J. Exp. Algorithmics*, 13, February 2009. ISSN 1084-6654.

Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. In *Advances in Neural Information Processing Systems 10*, NIPS '97, pages 507–513, Cambridge, MA, USA, 1998. MIT Press. ISBN 0-262-10076-2.

Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2137–2146, 2018.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083, 2018.

James C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

Gunjan Verma and Ananthram Swami. Error correcting output codes improve probability estimation and adversarial robustness of deep neural networks. In *Advances in Neural Information Processing Systems 32*, pages 8646–8656. 2019.

Alvin Wan, Lisa Dunlap, Daniel Ho, Jihan Yin, Scott Lee, Henry Jin, Suzanne Petryk, Sarah Adel Bargal, and Joseph E. Gonzalez. Nbdt: Neural-backed decision trees. 2020.

Shiqi Wang, Kevin Eykholt, Taesung Lee, Jiyong Jang, and Ian M. Molloy. Adaptive verifiable training using pairwise class similarity. *CoRR*, abs/2012.07887, 2020. URL `https://arxiv.org/abs/2012.07887`.

Bianca Zadrozny. Reducing multiclass to binary by coupling probability estimates. In *Advances in Neural Information Processing Systems 14*, pages 1041–1048. MIT Press, 2002.