Distributionally Robust Group Backwards Compatibility

Martin Bertran†, Natalia Martinez†, Alex Oesterling, and Guillermo Sapiro
Duke University

Abstract

Machine learning models are updated as new data is acquired or new architectures are developed. These updates usually increase model performance, but may introduce backward compatibility errors, where individual users or groups of users see their performance on the updated model adversely affected. This problem can also be present when training datasets do not accurately reflect overall population demographics, with some groups having overall lower participation in the data collection process, posing a significant fairness concern. We analyze how ideas from distributional robustness and minimax fairness can aid backward compatibility in this scenario, and propose two methods to directly address this issue. Our theoretical analysis is backed by experimental results on CIFAR-10, CelebA, and Waterbirds, three standard image classification datasets. Code available at github.com/natalialmg/GroupBC

1 Introduction

Machine learning (ML) leverages increasing amounts of data to improve model performance. Consequently, continual data collection and model improvement is an integral part of the ML life-cycle [Chen and Liu 2016, De Lange et al. 2019, Parisi et al. 2019]. Although model updates are often designed to improve average performance, some segments of the population may actually see their performance degraded by the model update. Backward Compatibility (BC) was discussed in Bansal et al. [2019], Srivastava et al. [2020] to monitor this undesirable phenomenon where, in a classification setting, samples that were accurately classified by a previous model are now incorrectly classified under the updated model, reducing the perceived reliability of the system. The current notion of BC may have shortcomings in measuring how compatibility errors arise in different populations, this is especially relevant since some population segments may be misrepresented in the data collecting process. Even when this is not the case, datasets and populations evolve over time, and potential compatibility mismatches that were previously tolerable may be exacerbated.

Here we provide a general analysis of BC were we formulate the BC definitions in Srivastava et al. [2020], Yan et al. [2021] as a statistical property of the data distribution and the models under comparison. We first show how BC measures are dependent on the error rates of the models. We then analyze how variance of the learning algorithm affects compatibility of independently-trained models, providing a consistent explanation on why ensembles have empirically proven to improve model compatibility at no accuracy cost [Yan et al. 2021]. Moreover, we show that classifier confidence is directly related to per-sample BC.

We analyze group distributional robustness of BC, were we consider model updates under data shifts. We assume data samples come from a mixture of distributions corresponding to different groups (e.g., domains, demographics, classes); as new samples are acquired, the overall group

*† Equal contribution

Workshop on Distribution Shifts, 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.
composition of the training set may drift over time. We wish to update our model over the newly augmented dataset in a way that guarantees a desired BC measure across all groups. We evaluate how existing ideas from group robust approaches [Hashimoto et al. 2018, Sagawa et al. 2019, Martinez et al. 2020, Diana et al. 2021, Martínez et al. 2021] can help in this scenario. We additionally propose two methods, Sample-Referenced Model (SRM) and Group-Referenced Model (GRM) to specifically tackle backward-compatible model updating. We use three common image classification datasets, CelebA, Waterbird, and CIFAR10 [Liu et al. 2018, Sagawa et al. 2019, Wah et al. 2011, Krizhevsky et al. 2021] for empirical validation.

2 Preliminaries

We consider the supervised classification scenario where we have an input variable $X \in \mathcal{X}$ and target variable $Y \in \mathcal{Y}$ related by the joint distribution $P_{X,Y}$. Given a dataset $D \sim P_{X,Y}^N$, a learning kernel $P_{h|D}$ defines a conditional distribution over functions $h : \mathcal{X} \rightarrow \Delta^{\mathcal{Y}}$ in an hypothesis class $\mathcal{H}$. The function $h$ outputs a score over the possible values of $Y$ given $X$. We distinguish between $h$ and $\bar{h}$ where $\bar{h} : \mathcal{X} \rightarrow \mathcal{Y}$ outputs a member of the target variable based on the score provided by model $h$ (e.g., $\bar{h}(x) = \arg \max_{y \in \mathcal{Y}} h(y|x)$ or $\bar{h}(x) \sim \text{Multinomial}[h(x)]$). We note the accuracy and error of a model $h$ on an arbitrary distribution $X,Y \sim Q_{X,Y}$ as $\text{Acc}(h|Q) = \mathbb{E}_Q[1(\bar{h}(X) = Y)]$ and $\text{Err}(h|Q) = \mathbb{E}_Q[1(h(X) \neq Y)] = 1 - \text{Acc}(h|Q)$.

2.1 Model compatibility

Backward compatibility (BC) was introduced in ML to evaluate the prediction consistency between two models trained to perform the same classification task.

Let us consider the general case where we are given models $h_1 \sim P_{h_1|D^1}$ and $h_2 \sim P_{h_2|D^2}$ where $D^1 \sim (P_{X,Y}^1)^{n_1}$, $D^2 \sim (P_{X,Y}^2)^{n_2}$ are datasets over distributions that share support. For a given test distribution $Q_{X,Y}$ on the same support, we can define the total compatibility (TC) and the (directed) negative flip rate (NFR) between models $h_1$ and $h_2$ on distribution $Q_{X,Y}$ as

$$\begin{align*}
TC(h_2, h_1|Q) &= \mathbb{E}_Q[1(\bar{h}_1(X) = Y \land \bar{h}_2(X) = Y) + 1(\bar{h}_1(X) \neq Y \land \bar{h}_2(X) \neq Y)], \\
NFR(h_1 \rightarrow h_2|Q) &= \mathbb{E}_{X,Y \sim Q}[1(\bar{h}_1(X) = Y \land \bar{h}_2(X) \neq Y)].
\end{align*}$$

Note that TC can be decomposed as the sum of two disjoint events, one being positive compatibility (PC) or the probability of both models getting the correct answer, and negative compatibility (NC), which measures the probability of a double error. NFR measures the likelihood of errors occurring on samples that the reference model $h_1$ had previously classified correctly. These quantities are the probabilistic version of metrics that were first defined empirically in [Srivastava et al. 2020, Yan et al. 2021]. It is straightforward to show that they can be bounded using the error rate and accuracies of the respective models.

**Preposition 2.1.** Given two models $h_1, h_2$ defined over the same input and output support $\mathcal{X}, \mathcal{Y}$ and a joint distribution $Q_{X,Y} : \text{Supp}(Q) \subseteq \mathcal{X} \times \mathcal{Y}$ we have the following:

$$\begin{align*}
TC(h_2, h_1|Q) &\geq 1 - \text{min}(\text{Acc}(h_1|Q), \text{Err}(h_2|Q)) - \text{min}(\text{Err}(h_1|Q), \text{Acc}(h_2|Q)), \\
NFR(h_1 \rightarrow h_2|Q) &\leq \text{min}(\text{Acc}(h_1|Q), \text{Err}(h_2|Q)).
\end{align*}$$

Since $NFR(h_1 \rightarrow h_2|Q)$ is upper bounded by the error rate of model $h_2$ on dataset $Q$, it is partially addressed by reducing the empirical error on samples from $Q$. Proof is provided in Appendix A.1. We explore how properties of the learning kernel can affect these metrics in the Appendix A.2 which also formally addresses the impact of ensemble learning on BC.

3 Group backward compatibility under data shifts

Consider a model $h_1 \sim P_{h_1|D^1}$ trained on a dataset containing samples from a variety of groups $G \in \mathcal{G}$, that is $D^1 = \{x_i, y_i, g_i\}_{i=1}^{n_1} \sim (P_{G|X,Y=G}^{n_1}) \otimes P_{X,Y}$. This model achieves a given per-group performance, empirically measured using a held out database $D^{test} = \{D^{test}_g \sim P^{n_2}_{X,Y|G=g} \}_{g \in \mathcal{G}}$. We further assume we expand our training dataset to $D^2$ such that $D^1 \subset D^2$ by potentially changing
We perform experiments on the CIFAR-10, CelebA [Liu et al. 2018], and Waterbird [Sagawa et al. 2019] datasets. CIFAR-10 is a standard benchmark for multi-label classification with 10 different classes that we considered as both target and group variables. CelebA contains over 200 thousand facial images with various attribute annotations; similarly to Sagawa et al. [2019], we use CelebA as a binary classification dataset by predicting the “blond” label. We used the Cartesian product of

the fraction of samples of each group (e.g., $|D_1^g|/|D_1| \neq |D_2^g|/|D_2|$, for some $g \in \mathcal{G}$), without modifying the group-conditional data distributions $P_{X,Y|G}$; this models non-uniform data acquisition.

We now analyze how BC is affected at the group level when a model is updated due to new data being acquired. We evaluate how per-group accuracy and NFR are affected when updated models are obtained by empirical risk minimization (ERM) or by worst group risk minimization (e.g., minimax group fairness [Martinez et al. 2020], Diana et al. [2021] or group distributional robustness [Sagawa et al. 2019]). Lastly, we propose a method to account for the worst NFR at group or at sample level, and show how this can improve the updated model in terms of group BC.

### 3.1 Base models

We identify two objectives to learn a model $h$, expected risk minimization (ERM) and group minimax fairness (GMMF), a relaxed version of Martinez et al. [2020], Diana et al. [2021], which is already robust to group shifts. We present their statistical formulations, and refer the reader to Appendix A.3 for implementation details,

$$\text{(ERM):} \min_{h \in \mathcal{H}} \mathbb{E}_P[\ell(h(X), Y)], \quad \text{(GMMF):} \min_{h \in \mathcal{H}} \max_{Q \geq \epsilon} \mathbb{E}_P[Q(h(X), Y)],$$

where $\ell: \Delta^{[X]} \times Y \rightarrow \mathbb{R}_+$ is a trainable loss function (e.g., cross entropy or Brier score), and $\epsilon \geq 0$ is a baseline group probability.

### 3.2 Accounting for negative flip rates

On scenarios where maintaining or improving NFR is of interest, we note that the updated model $h_2$ can also be made to explicitly depend on the previous model $h_1$. We first note that the loss of a sample $\ell(h(x_i), y_i)$ is a strong predictor of NFR. In particular, for binary classification, any model $h_2$ satisfying $\ell(h_2(x), y) \leq \ell(h_1(x), y)$ is automatically NFR-compatible in that sample. With this in mind, we can define the sample-referenced model (SRM) as

$$\text{(SRM):} \min_{h \in \mathcal{H}} \max_{Q \geq \epsilon} \mathbb{E}_{P_X,Y}[Q(h(X), Y) - \ell(h_1(X), Y)], \text{ s.t. } Q \geq \epsilon \forall X, Y. \quad (4)$$

Here $\epsilon$ controls the minimum importance weight of each sample $X, Y$ in the support. Note that, from the model’s perspective, the SRM objective reduces to minimizing a positively-weighted per-sample loss. However, the distribution $Q$ is updated based on the per-sample excess loss the current model has over the previous model. Similarly to Martinez et al. [2021], we use an empirical estimate of this objective and maximize it via projected gradient ascent (PGA) on the empirical distribution, with SGD for model optimization, Algorithm A.2 in Appendix A.3 describes the optimization procedure.

While SRM directly tackles NFR, it may suffer from generalization issues in practice, where NFR compatibility on train samples may be a poor predictor of NFR compatibility on new samples from a similar data distribution. To address this, we also propose a group-referenced model (GRM), where all per-group losses of the updated model are expected to improve w.r.t the baseline; a compromise objective between SRM and ERM,

$$\text{(GRM):} \min_{h \in \mathcal{H}} \max_{Q \geq \epsilon} \mathbb{E}_{P_X,Y}[Q(h(X), Y) - L_G(h_1)], \quad L_G(h_1) = \mathbb{E}_{P_{X,Y|G}}[\ell(h_1(X), Y)]. \quad (5)$$

Here we minimize the group with worst average loss improvement with respect to the previous model $h_1$. Algorithm A.1 shows the procedure.

### 4 Experiments and results

We perform experiments on the CIFAR-10, CelebA [Liu et al. 2018], and Waterbird [Sagawa et al. 2019] datasets. CIFAR-10 is a standard benchmark for multi-label classification with 10 different classes that we considered as both target and group variables. CelebA contains over 200 thousand facial images with various attribute annotations; similarly to Sagawa et al. [2019], we use CelebA as a binary classification dataset by predicting the “blond” label. We used the Cartesian product of

---

2We note that they share the objective with group distributional robustness [Sagawa et al. 2019].
“blond” and “binary gender” as the group label, giving 4 groups total. The Waterbird dataset is a subset from the CUB image dataset Wah et al. [2011], where the goal is to predict the type of bird (“waterbird” or “landbird”). As in Sagawa et al. [2019], we used the Cartesian product of the bird class and the image background (“land” or “water”), leading to a total of 4 groups. We use ResNet-18 and ResNet-34 architectures He et al. [2016] as our base classifiers in all cases, and use SGD with cosine annealing learning rate Loshchilov and Hutter [2016]. Details are provided in Appendix A.4.

We evaluate the effects that different training and model updating techniques have on NFR in the demographic shift scenario outlined in Section 3. We train a baseline model $h_1$ using either ERM or GMMF on a dataset $D^1$ comprising 40% of the original training samples from either Cifar10, CelebA, or Waterbird. To model extreme demographic shift and its effect on NFR, we train a successor model $h_2$ on an expanded dataset $D^2$ that comprises all of the samples in $D^1$, plus all samples from a given group that were on the original training dataset. Since $h_2$ has access to both an expanded dataset $D^{H_2}$ and a reference model $h_1$, we evaluate all options for model updating (ERM, GMMF, SRM, GRM).

Experiments include all possible combinations of base training methods (ERM, GMMF), model update algorithms (ERM, GMMF, SRM, GRM), and extension groups (i.e., the experiments are repeated for every possible choice of group extension). Table 1 summarizes the average performance over these extensions for each dataset, detailed results are available in Appendix A.4. We note that GRM has the smallest negative impact on accuracy, and correspondingly better worst case NFR performance, as expected by design. When transitioning from ERM to GMMF, we see large accuracy improvements on the worst class, but we take a significant accuracy penalty on at least one of the other classes as a result. We note that SRM generally obtains similar results to ERM as an update method, showing that per-sample generalization on test set is a hard-to-achieve objective, which suggests GRM may be a more suitable objective in practice. More work is warranted to tackle this important scenario.

5 Concluding remarks

In this work we analyzed BC in the context of machine learning and how it can be controlled. We analyzed group BC in the context of iterative model updating when our training dataset is expanded without respecting prior demographic or group proportions. We empirically show how the performance on the groups can be affected when updating ERM or GMMF models. Moreover, we proposed and analyzed two approaches to update the previous model with the goal of minimizing the NFR and the decrease in accuracy of each group in held-out data. Our experimental results on common image classification datasets serve as empirical validation for the ideas advanced in this work. Future work should focus on analyzing the generalization properties of these approaches.

References

G. Bansal, B. Nushi, E. Kamar, D. S. Weld, W. S. Lasecki, and E. Horvitz. Updates in human-ai teams: Understanding and addressing the performance/compatibility tradeoff. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 2429–2437, 2019.

Z. Chen and B. Liu. Lifelong machine learning: Synthesis lectures on artificial intelligence and machine learning. San Rafael, CA, USA: Morgan and Claypool Publishers, pages 1–127, 2016.

---

Algorithm 1: Group-Referenced Learning

Input: Dataset $D^{tr} = \{x_i, y_i, g_i\}_{i=1}^n$, reference model $h_1$, parametric model $h_g$, $\gamma$: model learning rate, $\gamma$: adversary learning rate, batch size $n_B$, aggregation size $N$.

1: Init parameters and group weights $\theta^0, \lambda = \{\frac{n_B}{n}\}_{g=1}^{|G|}$
2: Compute reference group loss $L_h(h_1)$ using Eq. 5
3: Expand dataset $D^{tr} = \{x_i, y_i, g_i\}_{i=1}^n$
4: while not converged do
5: Sample aggregated batch $AB \sim D^{tr} \otimes (N \times n_B)$
6: for $n = 1$ to $N$ do
7: Sample batch w/o replacement $B \sim AB \otimes n_B$
8: Compute group losses in batch $n_g(B) = \sum_{i \in B} \mathbb{1}(g_i = g)$
9: $L_g(h_g; B) = \frac{1}{n_g(B)} \sum_{i \in B} \mathbb{1}(g_i = g) \ell(h(x_i), y_i)$
10: Update model parameters $\theta \leftarrow \theta + \gamma \nabla \theta \left( \sum_{g \in G} \lambda_g (L_g(h_g) - L_g(h_1)) \right)$
11: Update group weights $\lambda \leftarrow \prod_{\lambda \geq 0} \left( \lambda + \gamma L_g(h_g; AB) \right)$
12: end for
13: end while
14: return $h_2$
| Dataset     | Update Method | ERM | GMMF |
|-------------|---------------|-----|------|
|             |               | ΔAcc | ΔNFR | ΔNFR | ΔAcc | ΔNFR | ΔNFR |
|             |               | min | max | min | max | min | max | min | max | min | max | min | max |
| CelebA      | ERM           | -4.7% | 7.9% | 0.932 | 0.996 | 0.003 | 0.315 | -48.3% | 5.5% | 0.516 | 1.000 | 0.006 | 0.100 |
|             | GRM           | -1.5% | 24.2% | 0.953 | 0.987 | 0.004 | 0.285 | -1.7% | 1.3% | 0.958 | 0.987 | 0.037 | 0.074 |
|             | GMMF          | -7.9% | 53.1% | 0.921 | 0.994 | 0.006 | 0.090 | -3.4% | 1.6% | 0.953 | 0.983 | 0.041 | 0.063 |
|             | SRM           | -2.9% | 6.3% | 0.917 | 0.997 | 0.003 | 0.284 | -41.1% | 5.2% | 0.589 | 1.000 | 0.008 | 0.102 |
| Waterbird   | ERM           | -3.7% | 7.7% | 0.940 | 0.996 | 0.004 | 0.327 | -15.1% | 5.5% | 0.833 | 0.996 | 0.009 | 0.204 |
|             | GRM           | -3.6% | 10.3% | 0.953 | 0.991 | 0.006 | 0.305 | -3.8% | 1.9% | 0.944 | 0.986 | 0.039 | 0.189 |
|             | GMMF          | -5.8% | 18.3% | 0.931 | 0.996 | 0.007 | 0.238 | -2.7% | 1.5% | 0.946 | 0.985 | 0.038 | 0.188 |
|             | SRM           | -5.3% | 5.4% | 0.932 | 0.995 | 0.005 | 0.353 | 17.2% | 4.0% | 0.817 | 0.996 | 0.015 | 0.209 |
| CIFAR-10    | ERM           | -7.7% | 5.0% | 0.865 | 0.981 | 0.040 | 0.189 | -11.1% | 5.9% | 0.842 | 0.981 | 0.030 | 0.164 |
|             | GRM           | -5.0% | 6.9% | 0.902 | 0.986 | 0.035 | 0.178 | -5.8% | 5.2% | 0.888 | 0.986 | 0.030 | 0.159 |
|             | GMMF          | -6.5% | 6.0% | 0.896 | 0.982 | 0.033 | 0.166 | -6.1% | 5.6% | 0.893 | 0.979 | 0.031 | 0.152 |
|             | SRM           | -4.3% | 6.9% | 0.898 | 0.988 | 0.031 | 0.175 | -24.7% | 7.4% | 0.714 | 0.979 | 0.024 | 0.139 |

Table 1: Minimum and maximum group improvement in accuracy, 1-NFR, and difference w.r.t NFR upper bound (ΔNFR) measured on the test set for each dataset. The initial model (baseline h₁) is trained with 40% of the original train partition, and for ERM achieved worst and best group accuracies of 37% and 99% on CelebA, 57% and 99% on Waterbird, and 74% and 93% on CIFAR-10. For the GMMF baseline the worst and best group accuracies were 90% and 94% on CelebA, 78% and 94% on Waterbird, and 77% and 95% on CIFAR-10. We report the average over all of the updated dataset D², where we augmented the original training dataset exclusively with the remaining training samples of one of the groups. We evaluate the effect of each update methodology (ERM, GMMF, GRM, SRM) for both baseline training methods for h₁ (ERM or GMMF).
M. Srivastava, B. Nushi, E. Kamar, S. Shah, and E. Horvitz. An empirical analysis of backward compatibility in machine learning systems. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3272–3280, 2020.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

S. Yan, Y. Xiong, K. Kundu, S. Yang, S. Deng, M. Wang, W. Xia, and S. Soatto. Positive-congruent training: Towards regression-free model updates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14299–14308, 2021.
A Appendix

A.1 Model compatibility proof

Given the definitions of total compatibility (TC) and negative flip rate (NFR) from Equation 1, TC can be further decomposed as the sum of Positive compatibility (PC) and Negative compatibility (NC) as follows

\[
TC(h_2, h_1|Q) = PC(h_2, h_1|Q) + NC(h_2, h_1|Q),
\]

\[
PC(h_2, h_1|Q) = \mathbb{E}_Q[\mathbf{1}(\hat{h}_1(X) = Y \land \hat{h}_2(X) = Y)],
\]

\[
NC(h_2, h_1|Q) = \mathbb{E}_Q[\mathbf{1}(\hat{h}_1(X) \neq Y \land \hat{h}_2(X) \neq Y)],
\]

\[
NFR(h_1 \rightarrow h_2|Q) = \mathbb{E}_{X,Y \sim Q}[\mathbf{1}(\hat{h}_1(X) = Y \land \hat{h}_2(X) \neq Y)].
\]

We next proceed to prove Preposition 2.1 which is extended to include upper bounds on PC and NC that arise naturally from the proof.

Preposition A.1. Given two models \( h_1, h_2 \) defined over the same input and output support \( \mathcal{X}, \mathcal{Y} \) and a joint distribution \( Q_{X,Y} \) \( \text{Supp}(Q) \subseteq \mathcal{X} \times \mathcal{Y} \) we have the following:

\[
PC(h_2, h_1|Q) \geq Acc(h_1|Q) - \min\{Acc(h_1|Q), Err(h_2|Q)\},
\]

\[
NC(h_2, h_1|Q) \geq Err(h_1|Q) - \min\{Err(h_1|Q), Acc(h_2|Q)\},
\]

\[
TC(h_2, h_1|Q) \geq 1 - \min\{Acc(h_1|Q), Err(h_2|Q)\} - \min\{Err(h_1|Q), Acc(h_2|Q)\},
\]

\[
NFR(h_1 \rightarrow h_2|Q) \leq \min\{Acc(h_1|Q), Err(h_2|Q)\}.
\]

Proof. From the definition of NFR, and using the fact that the likelihood of two events occurring simultaneously is smaller than the likelihood of either of the events happening on their own \( P(A \cap B) \leq \min\{P(A), P(B)\} \), it follows that

\[
NFR(h_1 \rightarrow h_2|Q) \leq \min\{\mathbb{E}_{X,Y \sim Q}[\mathbf{1}(\hat{h}_1(X) = Y)], \mathbb{E}_{X,Y \sim Q}[\mathbf{1}(\hat{h}_1(X) = Y)]\},
\]

\[
NFR(h_1 \rightarrow h_2|Q) \leq \min\{Acc(h_1|Q), Err(h_2|Q)\}.
\]

From the definition of PC and the upper bound of NFR it follows that

\[
PC(h_2, h_1|Q) = Acc(h_1|Q) - NFR(h_1 \rightarrow h_2|Q),
\]

\[
PC(h_2, h_1|Q) \geq Acc(h_1|Q) - \min\{Acc(h_1|Q), Err(h_2|Q)\}.
\]

A symmetrical analysis can be done to obtain the lower bound of NC. Combining the bounds for NC and PC we obtain the lower bound for TC.

\( \square \)

A.2 Interaction between model compatibility and learning kernel

It is common in deep learning to train models to reach near-zero training loss, meaning that models are perfectly compatible on the training dataset. However, due to the overparametrized nature of deep neural networks, the models are unlikely to make identical decisions on a different set of data points. The models may achieve the same average performance on unseen data, but will distribute the errors differently.

We analyze the compatibility between models that are independently obtained with the same learning algorithm applied to the same dataset \( \mathcal{D} \sim P^{\text{test}}_{X,Y}, h_1, h_2 \sim P^{\text{test}}_{\mathcal{D}} \). We further consider the binary classification problem where \( Y \in \{0, 1\}, h(x) \in [0, 1] \), where the decision \( \hat{h}(x) \) is obtained via thresholding, \( \hat{h}(x) = \mathbf{1}(h(x) \geq \tau) \), with a predefined threshold \( \tau \in [0, 1] \). We can express the model output as the sum of the expected model output given the dataset and learning algorithm \( \mu_h(X) = \mathbb{E}_{P_{\mathcal{D}}}[h(X)] \), plus an input-dependent error \( \epsilon(X) \),

\[
h_i(x) = \mu_h(x) + \epsilon_i(x), \quad \mathbb{E}[\epsilon_i(x)] = 0, \quad \forall x \in \mathcal{X}, \, i = 1, 2,
\]

with a corresponding decision probability \( P(\hat{h}_i(x) = 1) = P(\epsilon_i(x) \geq \tau - \mu_h(x)) \).

Given an input \( X \in \mathcal{X} \), errors \( \epsilon_1(X) \) and \( \epsilon_2(X) \) are i.i.d., in this setting, TC can be expressed as,

\[
TC(h_1, h_2|X) = P(\epsilon(X) \geq \tau - \mu_h(X)) - 2P(\epsilon(X) \geq \tau - \mu_h(X)) + 1.
\]

\( ^{\text{TC can be expressed as } TC(h_1, h_2|X) = P(h_1(X) \geq \tau)P(h_2(X) \geq \tau) + P(h_1(X) < \tau)P(h_2(X) < \tau) \text{ since we assume } h_1 \perp h_2|\mathcal{D}.} \)
For every input \( X \), TC reaches its maximum value if \( P(\epsilon(X) \geq \tau - \mu_h(X)) \) takes value 0 or 1. We can identify two ways of accomplishing this, one is choosing a lopsided threshold \( \tau \) (e.g., \( \tau \approx 0 \) or \( \tau \approx 1 \)), since \( \epsilon(X) + \mu_h(X) \in [0, 1] \), this reduces the chances of two models making different
decisions, but makes the classifiers non-informative, since the decision is only weakly affected by our
learned classifier. The other option is to reduce the variance of the noise variable \( \epsilon(X) \); we can use
Hoeffding’s inequality for this by observing that
\[
P(|\epsilon(X)| \geq |\tau - \mu_h(X)|) \leq \exp\left[-\frac{(\tau - \mu_h(X))^2}{2\text{Var}(\epsilon(X))}\right],
\]
which provides an upper bound on the probability of the error being capable of flipping the decision of
the current model w.r.t the idealized expected model \( \mu_h(X) \). One effective way of addressing noise
variance is by augmenting the learning kernel with ensembles, since the variance of the ensemble
model decays by a factor of \( n_e^{-2} \) w.r.t the model without ensembles, with \( n_e \) the number of elements
in the ensemble.

Ensembles can have a major impact on model compatibility, this has been empirically reported in
other works such as [Yan et al., 2021], and this simple analysis provides an explanation for their
effectiveness. We empirically showcase the effect ensembles have on several compatibility metrics on
CelebA [Liu et al., 2018] and CIFAR-10 [Krizhevsky et al., 2021], where we train several independent
ResNet networks [He et al., 2016] (ResNet-18 for Cifar-10 and Resnet-34 for CelebA) on the same
training dataset using SGD (effectively taking independent samples from the SGD learning kernel). We
used a simple ensemble technique where the average output of \( n_e \) independent networks constitute
our ensemble model. We show how total compatibility and positive compatibility are effectively
managed via ensembling in Table 2.

| model | CIFAR-10 | | | CelebA | | | |
|-------|---------|---------|---------|---------|---------|---------|
| base  | 0.936   | 0.952   | 0.079   | 0.912   | 0.04   | base    | 0.958   | 0.982   | 0.066   | 0.949   | 0.033   |
| 2ens  | 0.943   | 0.967   | 0.08    | 0.927   | 0.04   | 2ens    | 0.96    | 0.987   | 0.067   | 0.953   | 0.034   |
| 5ens  | 0.947   | 0.979   | 0.084   | 0.937   | 0.042  | 5ens    | 0.961   | 0.992   | 0.07    | 0.957   | 0.035   |
| 10ens | 0.951   | 0.998   | 0.086   | 0.942   | 0.043  | 10ens   | 0.96    | 0.994   | 0.073   | 0.957   | 0.037   |
| 15ens | 0.95    | 0.998   | 0.088   | 0.945   | 0.044  | 15ens   | 0.961   | 0.995   | 0.074   | 0.958   | 0.037   |
| 20ens | 0.951   | 0.991   | 0.09    | 0.946   | 0.045  | 20ens   | 0.961   | 0.996   | 0.074   | 0.959   | 0.037   |

Table 2: Evaluation of ensembles and weight regularization in accuracy (Acc), total compatibility
(TC), and positive compatibility (PC) on CelebA and CIFAR-10 datasets. The slackness of PC and
TC with respect to their lower bounds in Eq. 2 is also reported as \( \Delta TC \) and \( \Delta PC \) respectively. Results
are averaged over comparisons between 10 pairs of independent models.

We observe that the average accuracy increases for larger ensemble sizes, this effect is well known,
and one of the main reasons why ensembles are a popular choice for increasing model performance
when feasible. This accuracy increasing effect has diminishing returns with the number of ensembles.
Most importantly for our analysis, we note that both total compatibility and positive compatibility
also increase significantly as a function of ensemble size, as predicted by the analysis of section
[A.2]. Interestingly, this increase outpaces the natural growth of TC and PC predicted by the simple
increase in accuracy, as shown by the \( \Delta TC \) and \( \Delta PC \) columns in the same table, further suggesting
that the excess increase is due to the reduction in output variance that ensembles provide as a learning
algorithm.

Confidence informs compatibility

Decision confidence (\( \max_{y \in Y} h^y(x) \)) indicates how close a datapoint is to the decision boundary of
the model. A large confidence value also indicates robustness to perturbations caused by differences
in learned models, since it would require larger \( \epsilon(X) \) values to observe different decisions between
two models. We verify this observation empirically in the same setting as before. Figure 1 shows
how TC and PC vary as a function of confidence level, and the improvements introduced by the use
of ensembles.

From Figure 1, we observe that confidence is indeed a strong predictor of both total and positive
compatibility, with lower confidence values associated with a smaller likelihood of samples being
compatible across other models. We note that ensembles also share this effect, but the compatibility
across models is uniformly improved for all confidence levels.
Figure 1: Total compatibility (TC) and positive compatibility (PC) as a function of label confidence on CIFAR-10 and CelebA datasets. Results are shown for varying number of ensembles. We note that confidence is a strong predictor of both TC and PC, and that ensembles uniformly improve on both of these metrics for all confidence values.

A.3 Model learning algorithms

Here we provide additional details for all algorithm implementations. Model updating in all cases is done via SGD, so here we provide the empirical objective for all update algorithms and discuss how the adversarial distribution $Q$ is modelled and updated.

To start with, ERM has the following statistical and empirical formulation:

\[
\text{ERM:} \quad \min_{h \in \mathcal{H}} \mathbb{E}_P[\ell(h(X), Y)], \\
\min_{h \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), y_i). \tag{13}
\]

GMMF and GRM can both be likewise expressed in both statistical and empirical formulation as:

\[
\text{(GMMF):} \quad \min_{h \in \mathcal{H}} \max_{Q \geq \epsilon} \mathbb{E}_P[\frac{Q_{h}}{n} \ell(h(X), Y)], \\
\min_{h \in \mathcal{H}} \max_{\lambda_{g} \in \Delta_{G}} \sum_{g \in \mathcal{G}} \lambda_{g} \sum_{i=1}^{n} \frac{1}{n_{g}} \ell(h(x_i), y_i), \tag{14}
\]
Algorithm 2

Table 3 outlines all key architectural choices and hyperparameters used on each dataset.

A.4.1 Architecture and hyperparameters

A.4.4 Experimental section extension

Finally, SRM is also presented in both formulations:

\[
\begin{align*}
\text{(GRM): } & \min_{h \in H} \max_{Q \in \mathcal{Q}_{\geq \epsilon}} \mathbb{E}_P \left[ \frac{Q_G}{P_G} (\ell(h(X), Y) - L_G(h_1)) \right], \\
& \max_{\lambda \in \Delta_{H}^{G-1}} \sum_{g \in H} \lambda_g \left( \sum_{i=1}^{n} \frac{1}{n_g} \ell(h(x_i), y_i) - \hat{L}_g(h_1) \right), \\
& L_G(h_1) = \mathbb{E}_{P_X,Y \mid G} \left[ \ell(h_1(X), Y) \right], \\
& \hat{L}_g(h_1) = \sum_{i=1}^{n} \frac{1}{n_g} \ell(h(x_i), y_i).
\end{align*}
\]  

(15)

Here we use the vector \( \lambda_g \in \Delta_{H}^{G-1} \) to denote the empirical estimate of distributions \( Q_G \) satisfying \( Q_G \geq \epsilon \), and \( n_g \) in the empirical formulation represents the number of samples of group \( g \). We note that the distribution \( Q_G \) represents a potential shift in group composition w.r.t the training distribution \( P_G \), the distributions denoted by \( P \) indicates the underlying distribution from which the current training dataset is sampled from. The training algorithm for GRM is presented in Algorithm [1] and the algorithm for GMMF is identical if we set all per-group reference values \( \hat{L}_g(h_1) = 0 \).

Here the vector \( \lambda \in \Delta_{H}^{\mathbb{N}^n-1} \) already represents the importance weight ratio \( \frac{Q_X,Y}{P_X,Y} \). The algorithm, shown in Algorithm [3] uses PGA on the importance weights \( \lambda \).

Algorithm 2 Sample-Referenced Learning

Input: Dataset \( D^{tr} = \{x_i, y_i, g_i\}_{i=1}^{n} \), reference model \( h_1 \), parametric model \( h_\theta \), \( \eta \): model learning rate, \( \gamma \): adversary learning rate, batch size \( n_B \), aggregation size \( N \).

1: Init parameters and weights \( \theta^0, \lambda = \{ \frac{1}{n} \}_{i=1}^{n} \)
2: Expand dataset
3: \( \ell_i(h_1) \triangleq \ell(h_1(x_i), y_i) \)
4: \( D^{tr} = \{x_i, y_i, g_i, \lambda_i, \ell_i(h_1), i\}_{i=1}^{n} \)
5: while not converged do
6: Sample aggregated batch \( AB \sim D^{tr} \otimes (N \times n_B) \)
7: for \( n = 1 \) to \( N \) do
8: Sample batch w/o replacement \( B \sim AB \otimes n_B \)
9: Compute sample losses \( \ell_i(h_\theta) = \ell(h_\theta(x_i), y_i) \)
10: Update model parameters
11: \( \theta \leftarrow \theta + \eta \nabla \theta \frac{1}{n_B} \sum_{i \in B} \lambda_i \ell_i(h_\theta) \)
12: end for
13: Compute weight mass in aggregated batch
14: \( C = \sum_{i \in AB} \lambda_i \)
15: Update weights in aggregated batch
16: \( \lambda_{i \in AB} \leftarrow \prod_{j > i} \left( \lambda_{i \in AB} + \gamma \ell_{i \in AB}(h_\theta) \right) \)
17: end while
return \( h_\theta \)

A.4 Experimental section extension

A.4.1 Architecture and hyperparameters

Table [5] outlines all key architectural choices and hyperparameters used on each dataset.
Table 3: Summary of architecture and hyperparameters for each dataset. We also use momentum (0.9) and cosine annealing

| Dataset     | Network | Optimization | Learning rate | Weight decay | Batch size | N   | η  |
|-------------|---------|--------------|---------------|--------------|------------|-----|-----|
| Waterbirds  | ResNet-34 | SGD         | 10^{-3}       | 10^{-4}     | 64         | 80  | 0.1 |
| CelebA     | ResNet-34 | SGD         | 10^{-3}       | 10^{-4}     | 64         | 160 | 0.1 |
| CIFAR-10   | ResNet-18 | SGD         | 10^{-1}       | 10^{-4}     | 128        | 80  | 0.1 |

Other notable changes include reducing the kernel size of the initial convolution layer to 3 for CIFAR-10 (as standard for this dataset with smaller image sizes). Images for Waterbird and CelebA are scaled, normalized, and cropped to $128 \times 128$ for faster training. We use the default train and test partitions for all datasets.

A.4.2 Additional results

Here we provide the full experimental results. The initial model ($h_1$) is trained with 40% of the original train partition, and for ERM it achieved worst and best group accuracies of 37% and 99% on CelebA, 57% and 99% on Waterbird, and 74% and 93% on CIFAR-10. For the GMMF baseline the worst and best group accuracies were 90% and 94% on CelebA, 78% and 94% on Waterbird, and 77% and 95% on CIFAR-10. We next report results for each updated dataset $D^2$, where we augmented the original training dataset exclusively with the remaining training samples of one of the groups. Tables 4, 5, and 6 show these results for the CelebA, CIFAR-10, and Waterbirds datasets respectively.

Table 4: CelebA results. Minimum and maximum group improvement in accuracy, 1-NFR, and difference w.r.t NFR upper bound ($\Delta$NFR) in the test split for CelebA dataset. The initial model ($h_1$) is trained with 40% of the original train partition. We evaluate 4 cases for the updated dataset $D^2$, where we augmented the original training dataset exclusively with the remaining training samples of one of the groups (Augmented Group). We evaluate the effect of each update methodology (ERM, GMMF, GRM, SRM) on both of the baseline training methods for $h_1$ (ERM or GMMF).
Table 5: Waterbird results. Minimum and maximum group improvement in accuracy, 1-NFR, and difference w.r.t NFR upper bound (ΔNFR) in the test split for Waterbird dataset. The initial model (h₁) is trained with 40% of the original train partition. We evaluate 4 cases for the updated dataset D₂, where we augmented the original training dataset exclusively with the remaining training samples of one of the groups (Augmented Group). We evaluate the effect of each update methodology (ERM, GMMF, GRM, SRM) on both of the baseline training methods for h₁ (ERM or GMMF).

