Debiasing surgeon: fantastic weights and how to find them – Appendix

Rémi Nahon, Ivan Luiz De Moura Matos, Van-Tam Nguyen, and Enzo Tartaglione

LTCI, Télécom Paris, Institut Polytechnique de Paris, France
{name.surname}@telecom-paris.fr

A Visualizations

A.1 Is FFW helping the network focus on the right features ?

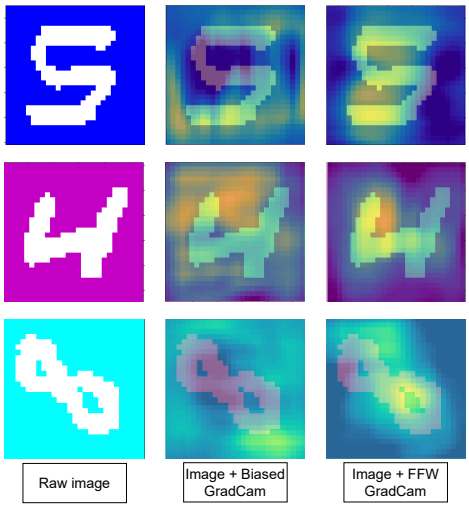

Fig. 4: Grad-Cam visualization of the effects of FFW on Biased-MNIST with $\rho = 0.997$

For Fig.4, we applied the Grad-Cam visualization method from [5] to FFW on Biased-MNIST. The raw sample (on the left) leads to high activation around the digit for the biased model (in the middle) but after pruning with FFW, the higher activations are placed on the digit.

A.2 Pruning distribution across the networks.

Fig. 5 shows the proportions of pruning in multiple networks for multiple datasets and two networks. It should be noted that while the network's pruning is spread

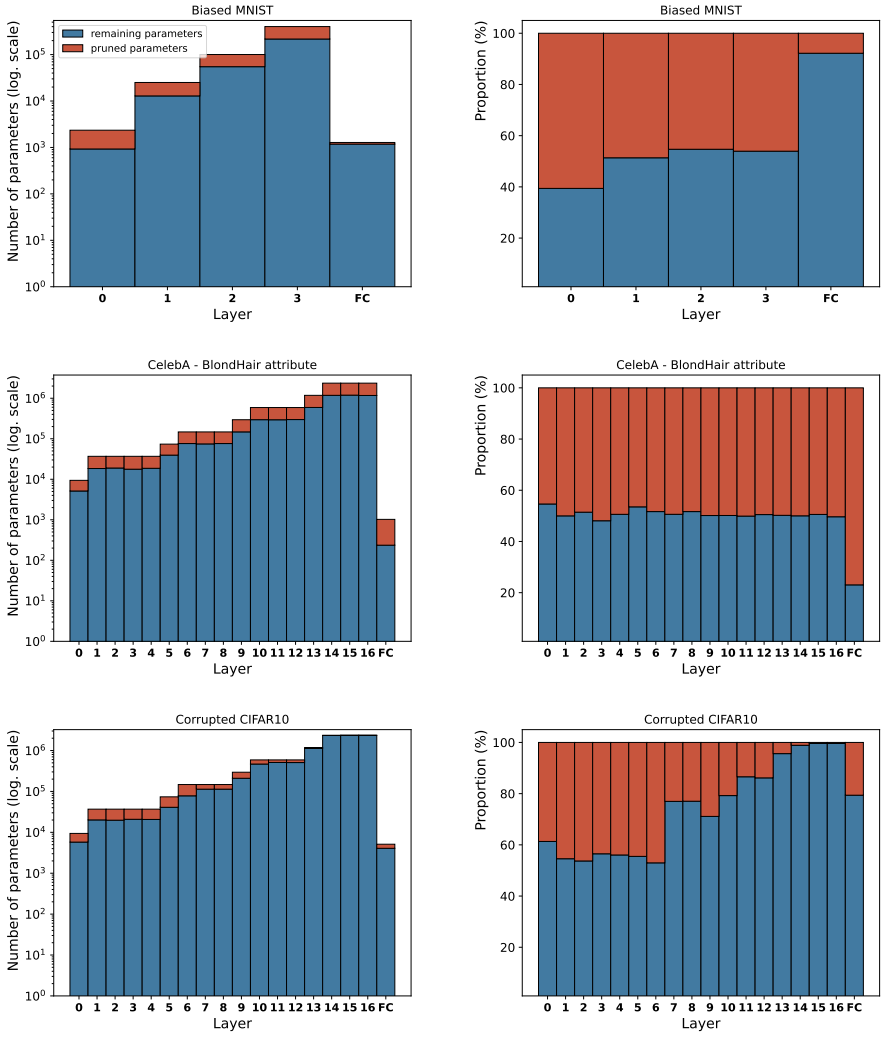

Fig. 5: Absolute number (top row) and proportions (bottom row) of pruned parameters after applying FFW to Biased MNIST, CelebA, and Corrupted CIFAR10.

over all layers for CelebA, it is focused on the first few layers for Corrupted Cifar10, potentially indicating the higher simplicity of the bias (a simple filter applied to the images) for that dataset.

A.3 Variations on the Mutual Information Minimization

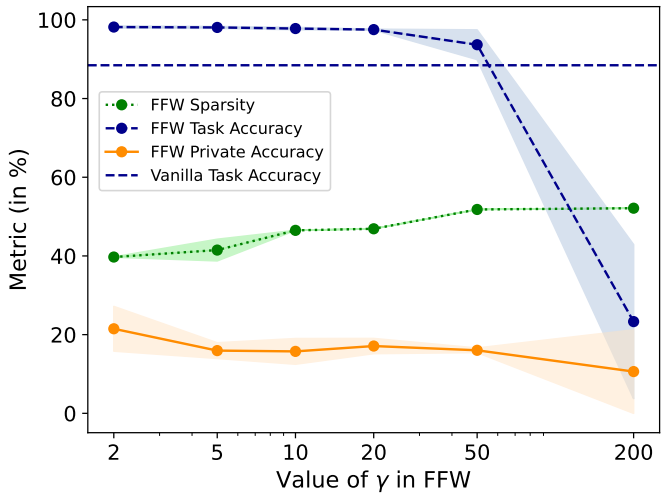

Fig. 6: Results for FFW applied on Biased-MNIST ($\rho = 0.99$) with different γ on the validation set

In Fig. 6 we can visualize the results of Tab. 1, showing that the Task Accuracy can be maintained high while minimizing the Private Accuracy for γ ranging from 5 to 50 on that specific dataset.

A.4 Comparison of Pruning Strategies

In Fig. 7, we can compare FFW to a vanilla magnitude pruning approach. While the magnitude pruning approach destroys both the Task and the Bias information while pruning, which is clear in the fact the both curves drop as sparsity increase, FFW does the opposite. At its 46% of sparsity, the pruning leads to a Private Accuracy close to random guess while increasing the Task Accuracy.

A.5 Effect of the extraction point

Fig. 8 helps us visualize the results of Tab.5. Indeed it shows that until the completion of at least a few epochs, the model fits only the bias and therefore, it

Table 7: Results for Biased-MNIST ($\rho = 0.99$) with different pruning strategies.

Strategy	Sparsity	Accuracy	
		Task	Bias
Vanilla Model	0	88.46 ± 0.63	98.45 ± 0.20
Magnitude Pruning	0.5	88.06 ± 0.8	99.04 ± 0.17
	0.75	78.16 ± 4.59	99.58 ± 0.14
	0.9	31.14 ± 9.96	98.30 ± 0.23
	0.95	10.50 ± 1.16	77.45 ± 11.29
	0.975	9.52 ± 0.53	40.98 ± 7.18
FFW Structured	0.46	97.63 ± 1.02	16.89 ± 3.15
FFW	0.47	97.79 ± 0.30	15.64 ± 3.26

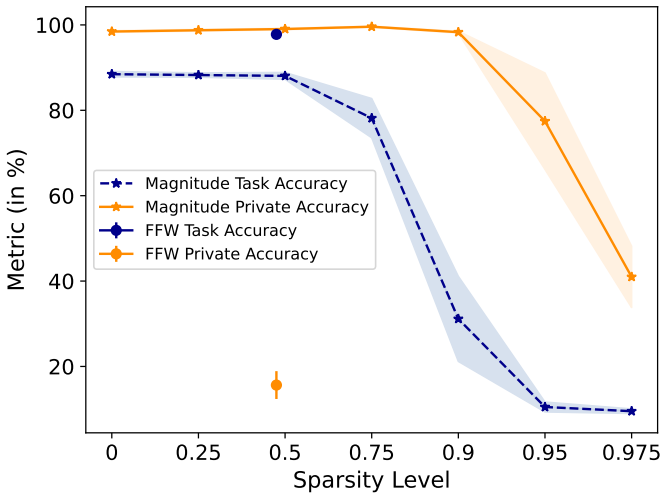

Fig. 7: Results of Magnitude Pruning on Biased MNIST ($\rho = 0.99$) at different levels of sparsity compared to FFW.

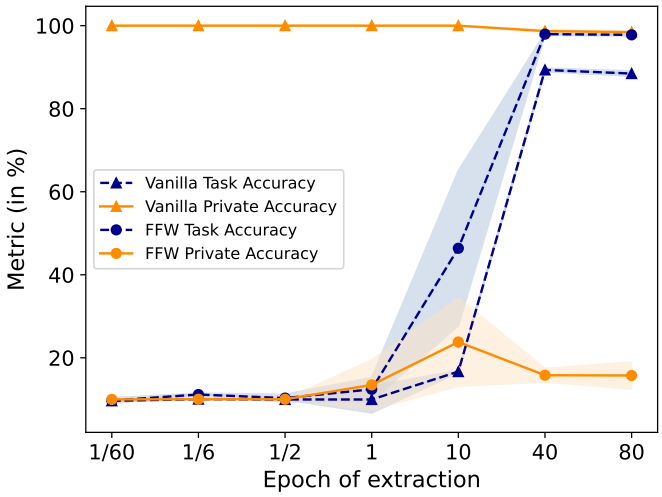

Fig. 8: Results of FFW on Biased MNIST ($\rho = 0.99$) applied after different biased training durations.

Table 8: Training time for our main experiments performed on a NVIDIA A100-PCIE-40GB GPU.

Dataset	Model used	Training time	Unbiased training set
Biased-MNIST [1]	SimpleConvNet [1]	3h20	36000
CelebA [4]	ResNet18	1h15	437
Corrupted CIFAR10 [2]	ResNet18	0h52	6000
Multi-Color MNIST [3]	3-layers MLP	0h22	6000

is impossible to extract a debiased subnetwork by our method. However, as the network learns unbiased features, our method keeps minimizing the propagation of biased information but starts progressively extracting subnetworks that are efficient on the target task.

A.6 Training times

We provide the average training times of our method (after training the vanilla model) in Tab. 8. This duration varies with the sizes of the unbiased training set, of the input images as well as the model type.

B Details for Section 3

In this section, we will provide all the derivations for Sec. 3 in the main paper. more specifically, Sec. B.1 will propose the derivation for (3), Sec. B.2 will discuss the joint probability that will be employed to derive (4) in Sec. B.3 and (6) in Sec. B.4.

B.1 Derivation for (3)

Given the joint probability as in (2), we can easily express the mutual information between $\hat{B}$ and $\hat{Y}$ as

$$\begin{aligned}\mathcal{I}(\hat{B},\hat{Y}) =& \sum_{i,j} p(\hat{b}_j,\hat{y}_i)\log_2\frac{p(\hat{b}_j,\hat{y}_i)}{p(\hat{b}_j)p(\hat{y}_i)}\\ =&\frac{N_B}{N_C}\rho\log_2\left[\frac{\rho N_C N_B}{N_C}\right]+\frac{N_B(N_B-1)}{N_C(N_B-1)}(1-\rho)\log_2\left[\frac{(1-\rho)N_C N_B}{N_B-1}\right]\\ =&\frac{N_B}{N_C}\left\{\rho\log_2(N_B)+\rho\log_2(\rho)+(1-\rho)\log_2(N_B)+(1-\rho)\log_2\left(\frac{1-\rho}{N_B-1}\right)\right\}\\ =&\frac{N_B}{N_C}\left\{\log_2(N_B)+\rho\log_2(\rho)+(1-\rho)\log_2\left(\frac{1-\rho}{N_B-1}\right)\right\} \qquad (11)\end{aligned}$$

B.2 Joint probability between $\hat{B}$, $\hat{Y}$, Y

A clear dependency between ρ and (3), as already showcased in Sec. B.1, exists. This measure is applied to ground-truth labels, investigating the common information between them (and for instance, the information that is possible to disentangle). Nonetheless, in the more general case, the trained model (whose output is modelizable as the random variable Y) is not a perfect learner, having $H(Y|\hat{Y}) \neq 0$. The model, in this case, does not correctly classify the target for two reasons.

1. It gets confused by the bias features, and it tends to learn to classify samples based on them. We model this tendency of learning biased features with ϕ, which we call *biasedness*. The higher the biasedness is, the more the model relies on features that we desire to suppress, inducing bias in the model and for instance error in the model.
2. Some extra error ε, non-directly related to the bias features, which can be caused, for example, by stochastic unbiased effects, to underfit, or to other high-order dependencies between data. This contribution is already visible in (1).

We can write the discrete joint probability for $\hat{B},\hat{Y},Y$, composed of the following terms.

– When target, bias, and prediction are aligned, the bias is aligned with the target class and correctly classified. Considering that we are not perfect learners, we introduce the error term ε.

- When the target and bias are misaligned and the prediction is correct, it means that the model has learned the correct feature and the bias is being contrasted. This effect is due to the dual effect of both the model's biasedness ϕ and the inherent ground-truth dependence between the use of the biased feature and the target label K_{bia}.
- When target and bias are not aligned, but the prediction is incorrect and bias and output are aligned, it means that the model has learned the bias, introducing the error we target to minimize in this work.
- In all the other cases, the error of the model is due to higher-order dependencies, not directly related to the biasedness ϕ.

Under the assumption $N_B = N_C = N$, we can write the joint distribution

$$\begin{aligned} p(\hat{B}, \hat{Y}, Y) = & \frac{1}{N} \cdot \Bigg[\delta_{\hat{b}\hat{y}y} \rho(1-\varepsilon) + \delta_{\hat{y}y} \bar{\delta}_{\hat{b}y} \bar{\delta}_{\hat{b}\hat{y}} \frac{(1-\phi)(1-\rho)}{N-1} (1 - K_{\text{bia}}) + \\ & + \bar{\delta}_{\hat{y}y} \delta_{\hat{b}y} \bar{\delta}_{\hat{b}\hat{y}} \frac{\phi(1-\rho)}{N-1} K_{\text{bia}} + \bar{\delta}_{\hat{y}y} \bar{\delta}_{\hat{b}y} \delta_{\hat{b}\hat{y}} \frac{\varepsilon\rho^2}{N-2+\rho} + \\ & + \bar{\delta}_{\hat{y}y} \bar{\delta}_{\hat{b}y} \bar{\delta}_{\hat{b}\hat{y}} \frac{\varepsilon\rho(1-\rho)}{(N-1)(N-2+\rho)} \Bigg]. \end{aligned} \tag{12}$$

B.3 Derivation for (4)

Let the joint probability as in (12). We can marginalize on $\hat{Y}$ by summing all the N_C (in our simplified case, N) biases per given target class and prediction of the model:

$$p(\hat{B}, Y) = \frac{1}{N} \left\{ \delta_{\hat{b}y} \left[\rho(1-\varepsilon) + \phi(1-\rho) \right] + \bar{\delta}_{\hat{b}y} \left[\frac{(1-\phi)(1-\rho)}{N-1} + \frac{\rho\varepsilon}{N-2+\rho} \right] \right\}. \tag{13}$$

From this, following the definition of mutual information, we can write

$$\begin{aligned} \mathcal{I}(\hat{B}, Y) = & \sum_{i,j} p(\hat{b}_j, y_i) \log_2 \frac{p(\hat{b}_j, y_i)}{p(\hat{b}_j) p(y_i)} \\ = & \frac{1}{N} \Big\{ N \cdot \left[\rho(1-\varepsilon) + \phi(1-\rho) \right] \cdot \log_2 (N \cdot (\rho(1-\varepsilon) + \phi(1-\rho))) + \\ & + (N^2 - N) \cdot \left[\frac{(1-\phi)(1-\rho)}{N-1} + \frac{\rho\varepsilon}{N-2+\rho} \right] \cdot \\ & \cdot \log_2 \left[N \cdot \left(\frac{(1-\phi)(1-\rho)}{N-1} + \frac{\rho\varepsilon}{N-2+\rho} \right) \right] \Big\} \end{aligned}$$

$$\begin{aligned} = & \left[\rho(1-\varepsilon) + \phi(1-\rho) \right] \cdot \left[\log_2 N + \log_2 (\rho(1-\varepsilon) + \phi(1-\rho) \right] + \\ & + \left[(1-\phi)(1-\rho) + \frac{(N-1)\rho\varepsilon}{N-2+\rho} \right] \cdot \\ & \cdot \left[\log_2 N + \log_2 \left(\frac{(1-\phi)(1-\rho)}{N-1} + \frac{\rho\varepsilon}{N-2+\rho} \right) \right] \end{aligned}$$

$$
\begin{aligned}
&= \log_2 \left[\rho(1-\varepsilon) + \phi(1-\rho)\right]^{\rho(1-\varepsilon)+\phi(1-\rho)} \\
&\quad + \log_2 \left(\frac{(1-\phi)(1-\rho)}{N-1} + \frac{\rho\varepsilon}{N-2+\rho} \right)^{(1-\phi)(1-\rho)+\frac{(N-1)\rho\varepsilon}{N-2+\rho}} \\
&\quad + \log_2 N \left[\rho(1-\varepsilon) + \phi(1-\rho) + \right. \\
&\quad \left. + \ (1-\phi)(1-\rho) + \frac{(N-1)\rho\varepsilon}{N-2+\rho} \right] \\
&= \log_2 \left[\rho(1-\varepsilon) + \phi(1-\rho)\right]^{\rho(1-\varepsilon)+\phi(1-\rho)} \\
&\quad + \log_2 \left(\frac{(1-\phi)(1-\rho)}{N-1} + \frac{\rho\varepsilon}{N-2+\rho} \right)^{(1-\phi)(1-\rho)+\frac{(N-1)\rho\varepsilon}{N-2+\rho}} \\
&\quad + \log_2 N \left[1 - \rho\varepsilon + \frac{(N-1)\rho\varepsilon}{N-2+\rho} \right]. \qquad (14)
\end{aligned}
$$

From here, we can easily obtain the normalized mutual information by scaling down the results of a factor $\log_2(N)$. Under the assumption that $\varepsilon = 0$, we obtain

$$
\begin{aligned}
\mathcal{I}(\hat{B}, Y) =& \frac{1}{\log_2(N)} \left\{ (\rho + \phi(1-\rho)) \log_2 \left[\rho + \phi(1-\rho)\right] + \right. \\
& \left. + [(1-\phi)(1-\rho)] \log_2 \left[\frac{(1-\phi)(1-\rho)}{N-1} \right] + 1 \right\} \\
=& \frac{\rho + \phi(1-\rho)}{\log_2(N)} \log_2 \left[\rho + \phi(1-\rho)\right] + \frac{(1-\phi)(1-\rho)}{\log_2(N)} \log_2 \left[\frac{(1-\phi)(1-\rho)}{N-1} \right] + 1. \qquad (15)
\end{aligned}
$$

We apologize for the typos in (4), which will be corrected as in (15) in the final version of the paper.

B.4 Derivation for (6)

Similarly to the approach taken in Sec. B.3, from the joint probability as in (12), we marginalize, but this time on $\hat{B}$, by summing all the N_B (in our simplified case, N) biases per given target class and prediction of the model. Under the assumption that $\varepsilon = \varepsilon_{\text{bia}}$, we have:

$$
\begin{aligned}
p(\hat{Y}, Y) =& \frac{1}{N} \left\{ \delta_{\hat{y}y} \left[\rho(1-\varepsilon_{\text{bia}}) + (1-\phi)(1-\rho)(1-K_{\text{bia}})\right] + \right. \\
& \left. + \bar{\delta}_{\hat{y}y} \left[\frac{\phi(1-\rho)}{N-1} K_{\text{bia}} + \frac{\varepsilon_{\text{bia}}\rho^2}{N-2+\rho} + \frac{N-2}{N-2+\rho} \frac{\varepsilon_{\text{bia}}\rho(1-\rho)}{N-1} \right] \right\} \qquad (16)
\end{aligned}
$$

Also in this case, following the definition of mutual information, we can write

$$\begin{aligned}
\mathcal{I}(\hat{Y},Y) =& \sum_{i,j} p(\hat{y}_j,y_i)\log_2\frac{p(\hat{y}_j,y_i)}{p(\hat{y}_j)p(y_i)} \\
=& \frac{1}{N}\left\{N\cdot[\rho(1-\varepsilon)+\phi(1-\rho)]\cdot\log_2(N\cdot(\rho(1-\varepsilon)+\phi(1-\rho)))+\right. \\
&+(N^2-N)\cdot\left[\frac{(1-\phi)(1-\rho)}{N-1}+\frac{\rho\varepsilon}{N-2+\rho}\right]\cdot \\
&\left.\log_2\left[N\cdot\left(\frac{(1-\phi)(1-\rho)}{N-1}+\frac{\rho\varepsilon}{N-2+\rho}\right)\right]\right\} \\
=& [\rho(1-\varepsilon)+\phi(1-\rho)]\cdot[\log_2 N+\log_2(\rho(1-\varepsilon)+\phi(1-\rho)]+ \\
&+\left[(1-\phi)(1-\rho)+\frac{(N-1)\rho\varepsilon}{N-2+\rho}\right]\cdot \\
&\cdot\left[\log_2 N+\log_2\left(\frac{(1-\phi)(1-\rho)}{N-1}+\frac{\rho\varepsilon}{N-2+\rho}\right)\right].
\end{aligned}$$

Here, defining

$$f(x,y,z)=xy\log_2(xz), \tag{17}$$

we can write

$$\begin{aligned}
&\mathcal{I}(\hat{Y},Y)=f\left\{\frac{1}{N}\left[\rho(1-\varepsilon_{\text{bia}})+(1-\phi)(1-\rho)(1-K_{\text{bia}})\right],N,N^2\right\}+ \\
&f\left\{\frac{1}{N}\left[\frac{\phi(1-\rho)}{N-1}K_{\text{bia}}+\left(\frac{\rho^2(N-2)+\rho(1-\rho)(N-2)}{(N-2+\rho)}\right)\varepsilon_{\text{bia}}\right],N(N-1),N^2\right\},
\end{aligned}$$

finding back (6).

References

1. Bahng, H., Chun, S., Yun, S., Choo, J., Oh, S.J.: Learning de-biased representations with biased representations. In: International Conference on Machine Learning. pp. 528–539. PMLR (2020)
2. Hendrycks, D., Dietterich, T.: Benchmarking neural network robustness to common corruptions and perturbations. In: International Conference on Learning Representations (2018)
3. Li, Z., Hoogs, A., Xu, C.: Discover and mitigate unknown biases with debiasing alternate networks. In: European Conference on Computer Vision. pp. 270–288. Springer (2022)
4. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE international conference on computer vision. pp. 3730–3738 (2015)
5. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision **128**(2), 336–359 (Oct 2019). https://doi.org/10.1007/s11263-019-01228-7, http://dx.doi.org/10.1007/s11263-019-01228-7