EXPLAINABLE AI FOR NATURAL ADVERSARIAL IMAGES

Tomas Folke, ZhaoBin Li†, Ravi B. Sojitra, Scott Cheng-Hsin Yang & Patrick Shafto
Department of of Mathematics and Computer Science, Rutgers University, Newark, NJ 07102, USA
{tomas.folke, ravisoji, scott.cheng.hsin.yang, patrick.shafto}@gmail.com
†liz2@carleton.edu

ABSTRACT

Adversarial images highlight how vulnerable modern image classifiers are to perturbations outside of their training set. Human oversight might mitigate this weakness, but depends on humans understanding the AI well enough to predict when it is likely to make a mistake. In previous work we have found that humans tend to assume that the AI's decision process mirrors their own. Here we evaluate if methods from explainable AI can disrupt this assumption to help participants predict AI classifications for adversarial and standard images. We find that both saliency maps and examples facilitate catching AI errors, but their effects are not additive, and saliency maps are more effective than examples.

1 INTRODUCTION

Adversarial images are images that cause the AI to be confidently wrong (Szegedy et al., 2013; Nguyen et al., 2015), despite being easily classified by humans. Large regions of the possible input space might lead to such misclassifications (Goodfellow et al., 2014). Sensitivity to adversarial images can leave an AI vulnerable to purposeful attacks (Eykholt et al., 2018), but even in naturalistic settings some images behave "adversarially" in the sense that algorithms confidently misclassify them even though a human would not (*natural adversarial images*, see Hendrycks et al. (2019)). It has proven challenging to build systems that are robust to adversarial images, or give low confidence to adversarial mistakes (Hendrycks et al., 2019; Goodfellow et al., 2014; Papernot et al., 2016). Therefore it would be helpful if humans could catch and veto such cases. However, the default human assumption seems to be that AI classifiers share their perceptions and beliefs (Yang et al., 2021). This assumption makes it harder for humans to identify adversarial cases because they themselves are not fooled by such cases (Papernot et al., 2016; Harding et al., 2018). Here we test whether explanations help people to predict misclassifications of natural adversarial images, which is an essential prerequisite for effective human oversight of AI systems.

A popular class of methods to explain AI is *explanation-by-examples*. Explanation-by-examples takes an AI model and its training data as inputs and selects a small subset of cases that exert high impact on the inference of the explainee. Humans have the ability to induce principles from a few examples (Mill, 1884; Lake & Piantadosi, 2020), which is why examples are extensively used in formal education (Chi et al., 1989; Aleven, 1997; Bills et al., 2006). The explanation-by-examples approach has many desirable properties: It is fully model-agnostic and applicable to all types of machine learning (Chen et al., 2018); it is domain- and modality-general (Kanehira & Harada, 2019); and it can be used to generate both global (Kim et al., 2014; Vong et al., 2018) and local explanations (Papernot & McDaniel, 2018; Goyal et al., 2019).

We have developed a computational framework for explanation-by-examples called *Bayesian Teaching* (Yang & Shafto, 2017; Vong et al., 2018). Based in the cognitive science of human learning (Shafto & Goodman, 2008; Shafto et al., 2014), and drawing upon deep connections to probabilistic machine learning (Murphy, 2012; Eaves & Shafto, 2016), Bayesian Teaching integrates models of human and machine learning in a single system. Bayesian Teaching casts the problem of XAI as a problem of teaching—selecting optimal examples to teach the human user what the AI system has inferred. The explanatory examples selected in this teaching framework have been shown to match what humans find representative of the underlying generative process (Tenenbaum et al., 2001).

We select the optimal teaching examples based on a model of the learner (Shafto et al., 2014):

$$P_{teacher}(\mathbb{D} \mid Y = c, x) \propto P_{learner}(Y = c \mid \mathbb{D}, x) = \int P(Y = c \mid x, \boldsymbol{W})\, p(\boldsymbol{W} \mid \mathbb{D})\, d\boldsymbol{W}. \quad (1)$$

A quality teaching set is one that correctly helps the learner to update their prediction Y of a new image x belonging to the target category c after learning from observed teaching set $\mathbb{D}$. The target category is the category predicted by the target model, which is a ResNet-50 with pre-trained ImageNet weights.[1] We used Bayesian teaching to generate explanations at two levels of granularity: case-level examples and saliency maps (see Appendix for details). Then we evaluated how these two explanation features impacted human understanding of AI with adversarial examples. We evaluated human understanding by testing how well participants could predict the AI classifications for adversarial images, and compared their predictive performance relative to AI errors and correct classifications for standard images.

2 METHODS

EXPERIMENTAL DESIGN

The Natural Adversarial ImageNet dataset (Hendrycks et al., 2019) contains 200 categories that belong to a subset of the 1000 categories in ImageNet (Russakovsky et al., 2015). From these 200 categories, we selected 30 categories that span the spectrum of model accuracy based on ResNet-50's predictions on the standard ImageNet's validation set. For each of these 30 categories, we made three types of trials characterized by the model's prediction on *target images*: (1) model hit on an image sampled from standard ImageNet, (2) model error on an image sampled from standard ImageNet, and (3) model error on an image sampled from Natural Adversarial ImageNet. All target images were randomly sampled from the chosen categories and dataset. We used these target images in a two alternative forced choice task, where participants were asked to predict the model classifications. The two options are referred to as the *target category* and the *alternative category*.

For misclassified images the target category is the model prediction, and the alternative category is the ground truth. For correctly classified images the target category is the model's predicted category, and the alternative category is the category most confusable with the target category, according to the confusion matrix constructed on ResNet-50's predictions on the standard ImageNet's validation set. The standard images and adversarial images were matched with regards to the ground truth category of the target image, but not with regards to the target category. For example, when the ground truth of the target image was an accordion and the AI was wrong, the target category (the AI prediction) was "vacuum" in the standard case, but "breastplate" in the adversarial case.

The above procedure generated 90 trials that specify the target image, the target category, and the alternative category. These specifications were fed into the Bayesian Teaching framework, which produced a teaching set of four explanatory examples—two from the target category and two from the alternative category—for each trial. The four examples were selected so that the learner model, once exposed to them, would infer the target image to be of the target category with probability > 0.8. Out of the 90 trials selected, Bayesian Teaching could not find four examples that met this criterion for one of the standard incorrect trials. Thus, the experiment had a total of 89 trials.

PARTICIPANTS

The study protocol was approved by Rutgers University IRB. Informed consent was obtained from all participants. Participants were randomly allocated to four levels of explanation: (1) no explanation, (2) saliency maps only, (3) examples only, or (4) saliency maps and examples. We tested 40 participants per condition, resulting in a total sample of 160 participants.

LEARNER MODEL

Because the ResNet-50 model encodes statistical patterns reflecting human labels, we adapted the ResNet-50 architecture for the learner model, which is a useful, albeit simplifying, assumption. Under the Bayesian Teaching framework, the learner model learns probabilistically. Converting the

[1] `https://pytorch.org/docs/stable/torchvision/models.html`

whole ResNet-50 into a probabilistic model would be computationally intractable. Hence, we simplified the probabilistic approach by modifying only the classification layer of the learner model to make probabilistic decisions, while keeping the convolutional base deterministic. Since our aim is to teach humans a binary classification, we scaled down and converted the deterministic softmax layer, originally designed for 1000 categories, to a Bayesian logistic regression layer with two classes. We set a normal prior over the weights of the Bayesian classification layer for the learner model. The normal prior is obtained by performing a Kronecker factored Laplace approximation (Ritter et al., 2018) over the classification layer of the original ResNet-50 model, trained on ImageNet dataset over 100 epochs with data augmentation. Then, we used Laplace approximation to obtain a normal posterior over the weights of the learner, the $p(\boldsymbol{W} \mid \mathbb{D})$ in Equation 1. This was used in conjunction with the sigmoid likelihood $P(Y = c \mid x, \boldsymbol{W})$ to produce the posterior predictive $P_{learner}(Y = c \mid \mathbb{D}, x)$ in Equation 1. See Appendix A for the details.

STIMULI GENERATION

Given the specified target and alternative categories, we sampled the target images randomly from the dataset. To generate the teaching examples for each target image, we sampled 200 teaching sets as possible candidates. Let $\{x, c\}$ be the pair of target image and label, $\mathbb{D}$ be the teaching set, and $P_{learner}(Y = c \mid \mathbb{D}, x)$ be the probability of the targeted prediction of the learner model. For each teaching set $\mathbb{D}$, we re-initialized the prior from the two rows of the normal prior corresponding to the target and alternative categories. Then we trained the learner model using data augmentation over $\mathbb{D}$ for 128 epochs. Next we used Monte Carlo sampling to estimate $P_{learner}(Y = c \mid \mathbb{D}, x)$. The first teaching set for which $P_{learner}(Y = c \mid \mathbb{D}, x) > 0.8$ was selected as a quality teaching set for the experiment. The saliency map for each image (including both target and explanatory examples) is generated following the same procedure as described in (Yang et al., 2021) (see Appendix B).

3 RESULTS

We aim to determine how well humans can predict AI classifications as a function of whether the target image is adversarial, and what explanation features they have access to. To test this we first compared the performance of three nested logistic hierarchical regressions. The simplest model represents the null hypothesis that predictive accuracy differed between participants, target categories, and trial types, but that the explanations did not impact predictive performance. This *null model* was formalized such that human predictive accuracy at the trial level was based on two random intercepts based on participant and target category, respectively, and a fixed effect of trial type (standard correct, standard incorrect, and adversarial incorrect; treating adversarial incorrect as the reference condition). The second model represents the hypothesis that the explanations impacted the predictive performance of the participants, but that explanation effectiveness was constant across trial types. This *explanation model* expanded on the null model by adding main effects for whether participants were exposed to saliency map explanations and example explanations. The final model represented the hypothesis that the impact of the two explanation features (examples and saliency maps) were not additive, and that they varied between trial types. This *interaction model* built on the explanation model by adding interaction terms for the two explanation features and the trial types. The explanation model captured prediction accuracy better than the null model according to a likelihood ratio test ($\chi^2(2) = 45.37$, $p < .0001$), and the interaction model outperformed the explanation model ($\chi^2(7) = 209.62$, $p < .0001$). These results are consistent with the hypothesis that explanations did impact performance differently for different trial types. To explore these effects more thoroughly we studied the coefficients of the interaction model, see Table 1, and Figure 1.

Absent intervention, human predictive accuracy is similar for standard incorrect trials and adversarial incorrect trials, but much higher for standard correct images. This may imply that absent explanations, humans tend to assume that the AI makes correct classifications, in line with previous work showing that the default human assumption is that AI classifications will match their own (Yang et al., 2021). While both saliency maps and examples significantly improve predictive performance on adversarial images, this improvement is about four times larger for the saliency maps. Additionally, the effect of the two explanation features are not additive. Comparing the relative benefit of explanations on standard incorrect trials versus adversarial trials, we note that the improvement from saliency maps is significantly smaller for standard trials relative to adversarial trials. The improve-

Table 1: Coefficients of the interaction model

	Coefficient	SE
1. (Intercept)	-1.06^{***}	(0.14)
2. Trial type: Standard incorrect	0.09	(0.10)
3. Trial type: Standard correct	4.64^{***}	(0.18)
4. Saliency maps present	1.01^{***}	(0.12)
5. Examples present	0.46^{***}	(0.12)
6. Trial type: Standard incorrect × Saliency maps present	-0.29^{*}	(0.13)
7. Trial type: Standard correct × Saliency maps present	-1.71^{***}	(0.22)
8. Trial type: Standard incorrect × Examples present	-0.24	(0.13)
9. Trial type: Standard correct × Examples present	-2.01^{***}	(0.21)
10. Saliency maps present × Examples present	-0.53^{**}	(0.17)
11. Trial type: Standard incorrect × Saliency maps present × Examples present	0.11	(0.18)
12. Trial type: Standard correct × Saliency maps present × Examples present	1.18^{***}	(0.27)

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

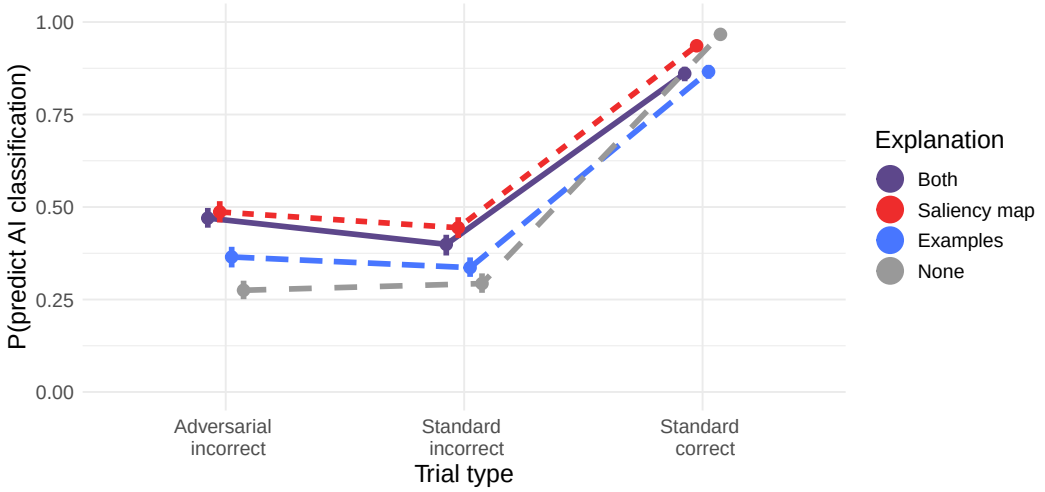

Figure 1: Explanations improve predictive performance for AI mistakes. Saliency maps are more beneficial than examples. The effects of the two explanation types are not additive, as combined explanations are associated with slightly worse performance than the saliency maps alone. Error bars signify 95% bootstrapped confidence intervals.

ment from examples is also smaller for standard incorrect trials, but not significantly so. Finally, for the standard correct trials all interventions are associated with a decrease in performance.

4 Discussion

In this paper we tested whether explanations generated by Bayesian teaching help humans predict AI classifications for standard and adversarial images. We found that explanations, and saliency maps in particular, improved participants' predictive accuracy for AI mistakes. We also learned that saliency maps were better at alerting participants to adversarial (as opposed to standard) misclassifications, presumably because saliency maps show when the classifier attends to strange features, as it tends to do in adversarial cases. In this study we focused on natural adversarial cases for two reasons: 1) We expect they pose a harder prediction problem for humans, relative to artificial adversarial images that are often distorted in ways that humans can detect, 2) We expect that natural adversarial images help users become aware of failure modes of the AI model outside of the training set. As such, adversarial images, together with explanations, may help users develop more sophisticated mental models of an AI's decision rules.

REFERENCES

Vincent AWMM Aleven. *Teaching case-based argumentation through a model and examples*. Citeseer, 1997.

Liz Bills, Tommy Dreyfus, John Mason, Pessia Tsamir, Anne Watson, and Orit Zaslavsky. Exemplification in mathematics education. In *Proceedings of the 30th Conference of the International Group for the Psychology of Mathematics Education*, volume 1, pp. 126–154. ERIC, 2006.

Jianbo Chen, Le Song, Martin Wainwright, and Michael Jordan. Learning to explain: An information-theoretic perspective on model interpretation. In *International Conference on Machine Learning*, pp. 882–891, 2018.

Michelene TH Chi, Miriam Bassok, Matthew W Lewis, Peter Reimann, and Robert Glaser. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive science*, 13(2):145–182, 1989.

Baxter S. Eaves and Patrick Shafto. Toward a general, scalable framework for Bayesian teaching with applications to topic models. In *IJCAI 2016 workshop on Interactive Machine Learning*, 2016. URL `http://arxiv.org/abs/1605.07999`.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, 2018.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. *arXiv preprint arXiv:1904.07451*, 2019.

Samuel Harding, Prashanth Rajivan, Bennett I Bertenthal, and Cleotilde Gonzalez. Human decisions on targeted and non-targeted adversarial sample. In *CogSci*, 2018.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

Atsushi Kanehira and Tatsuya Harada. Learning to explain with complemental examples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8603–8611, 2019.

Been Kim, Cynthia Rudin, and Julie A Shah. The bayesian case model: A generative approach for case-based reasoning and prototype classification. In *Advances in Neural Information Processing Systems*, pp. 1952–1960, 2014.

Brenden M Lake and Steven T Piantadosi. People infer recursive visual concepts from just a few examples. *Computational Brain & Behavior*, 3(1):54–65, 2020.

John Stuart Mill. *A system of logic ratiocinative and inductive: Being a connected view of the principles of evidence and the methods of scientific investigation*. Harper, 1884.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 427–436, 2015.

Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z Berkay Celik, and Ananthram Swami. The limitations of deep learning in adversarial settings. In *2016 IEEE European symposium on security and privacy (EuroS&P)*, pp. 372–387. IEEE, 2016.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Patrick Shafto and Noah D. Goodman. Teaching games: Statistical sampling assumptions for learning in pedagogical situations. In *Proceedings of the 30th annual conference of the Cognitive Science Society*, Austin, TX, 2008. Cognitive Science Society.

Patrick Shafto, Noah D. Goodman, and Thomas L. Griffiths. A rational account of pedagogical reasoning: Teaching by, and learning from, examples. *Cognitive Psychology*, 71:55–89, 2014. ISSN 00100285. doi: 10.1016/j.cogpsych.2013.12.004.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Joshua B Tenenbaum, Thomas L Griffiths, et al. The rational basis of representativeness. In *Proceedings of the 23rd annual conference of the Cognitive Science Society*, pp. 103641. Citeseer, 2001.

Wai Keen Vong, Ravi B. Sojitra, Anderson Reyes, Scott Cheng-Hsin Yang, and Patrick Shafto. Bayesian teaching of image categories. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, 2018.

Scott Cheng-Hsin Yang and Patrick Shafto. Explainable artificial intelligence via bayesian teaching. *NIPS 2017 workshop on Teaching Machines, Robots, and Humans.*, 2017.

Scott Cheng-Hsin Yang, Wai Keen Vong, Ravi B Sojitra, Tomas Folke, and Patrick Shafto. Mitigating belief projection in explainable artificial intelligence via bayesian teaching. *arXiv preprint arXiv:2102.03919*, 2021.

A Appendix

To give an overview, Section A.1 describes the computation of the posterior predictive given the softmax likelihood and the posterior on the weights of the classification layer; see Equation 2. Section A.2 describes the approximation used to obtain the posterior on the weights, which is obtained from the softmax likelihood and a normal prior on the weights; see Equation 3. Section A.3 describes the construction of the normal prior, which uses the weights obtained by training the classification layer of the RestNet-50 on ImageNet as the mean, and the hessian of the log-likelihood loss function as the precision matrix. These components complete the specifications of the learner model.

A.1 Posterior Predictive

Following (Murphy, 2012), we used Monte Carlo integration to estimate the learner model's posterior predictive, $P(Y = c \mid x, \mathbb{D})$. First, we trained the model on a dataset $\mathbb{D} = \{d_1, \ldots, d_n\} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ to learn the weights $\boldsymbol{W}$ for the classification layer while retaining the pretrained weights for the convolutional base. Given a new datapoint x and label c, the predictive probability on $Y = c$ is:

$$\begin{aligned} P(Y = c \mid x, \mathbb{D}) &= \int P(Y = c \mid x, \mathbb{D}, \boldsymbol{W})\, p(\boldsymbol{W} \mid x, \mathbb{D})\, d\boldsymbol{W} \\ &= \int P(Y = c \mid x, \boldsymbol{W})\, p(\boldsymbol{W} \mid \mathbb{D})\, d\boldsymbol{W} \\ &\approx \frac{1}{s} \sum_{m=1}^{s} P(Y = c \mid x, \boldsymbol{W}_m), \end{aligned} \tag{2}$$

where $\boldsymbol{W}_m$ are samples from the normal posterior obtained using Laplace approximation. For both models, we set $s = 100$, i.e. we sampled a set of 100 weights per image.

MODEL IMPLEMENTATION

For multinomial logistic regression on the learner model, $\boldsymbol{w}_c$ represents the weights for class c in all classes $\mathbb{C}$, and the predictive probability is:

$$P(Y = c \mid x, \mathbb{D}) \approx \frac{1}{s} \sum_{m=1}^{s} \frac{\exp(\boldsymbol{w}_{m,c}^T x)}{\sum_{c' \in \mathbb{C}} \exp(\boldsymbol{w}_{m,c'}^T x)}$$

A.2 APPROXIMATING THE POSTERIOR

Following (Murphy, 2012), we used Laplace approximation to obtain the posterior of the weights $p(\boldsymbol{W} \mid \mathbb{D})$ for the classification layer. The posterior for a model trained on dataset $\mathbb{D}$ is given by:

$$\begin{aligned} p(\boldsymbol{W} \mid \mathbb{D}) &= \frac{P(\mathbb{D} \mid \boldsymbol{W})\, p(\boldsymbol{W})}{P(\mathbb{D})} \\ &= \frac{1}{P(\mathbb{D})} e^{\log(P(\mathbb{D}|\boldsymbol{W})\, p(\boldsymbol{W}))}. \end{aligned} \tag{3}$$

To obtain the mode of the posterior, we define a loss function $\mathbf{L}(\mathbf{W}) \triangleq -\log(P(\mathbb{D} \mid \boldsymbol{W}) p(\boldsymbol{W}))$ and solve for:

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} L(\boldsymbol{W}).$$

The mode $\boldsymbol{W}^*$ will be used as the mean of the approximate posterior, which is set to be a normal distribution. To obtain the covariance matrix for the posterior normal, we perform a second-order Taylor expansion around $\boldsymbol{W}^*$ whereby:

$$L(\boldsymbol{W}) \approx L(\boldsymbol{W}^*) - (\boldsymbol{W} - \boldsymbol{W}^*) \left.\frac{\partial L(\boldsymbol{W})}{\partial \boldsymbol{W}}\right|_{\boldsymbol{W}=\boldsymbol{W}^*} + \frac{1}{2}(\boldsymbol{W} - \boldsymbol{W}^*)^T \left.\frac{\partial^2 L(\boldsymbol{W})}{\partial \boldsymbol{W}^2}\right|_{\boldsymbol{W}=\boldsymbol{W}^*} (\boldsymbol{W} - \boldsymbol{W}^*).$$

Because $\boldsymbol{W}^*$ is the mode (i.e., the maxima), the gradient $\frac{\partial L(\boldsymbol{W})}{\partial \boldsymbol{W}}|_{\boldsymbol{W}=\boldsymbol{W}^*}$ is zero. Define the hessian of the loss $\boldsymbol{H} \triangleq \frac{\partial^2 L(\boldsymbol{W})}{\partial \boldsymbol{W}^2}|_{\boldsymbol{W}=\boldsymbol{W}^*}$. The posterior then becomes:

$$\begin{aligned} p(\boldsymbol{W} \mid \mathbb{D}) &= \frac{1}{P(\mathbb{D})} \exp[-L(\boldsymbol{W}^*) - \frac{1}{2}(\boldsymbol{W} - \boldsymbol{W}^*)^T \boldsymbol{H} (\boldsymbol{W} - \boldsymbol{W}^*)] \\ &= \frac{e^{-L(\boldsymbol{W}^*)}}{P(\mathbb{D})} \exp[-\frac{1}{2}(\boldsymbol{W} - \boldsymbol{W}^*)^T \boldsymbol{H} (\boldsymbol{W} - \boldsymbol{W}^*)]. \end{aligned}$$

By presuming $P(\mathbb{D}) = e^{-L(\boldsymbol{W}^*)} * \sqrt{(2\pi)^k |\boldsymbol{H}^{-1}|}$, where k is the size of the input for the classification layer, (i.e., the length of the feature vector after getting transformed by the convolutional base), $p(\boldsymbol{W} \mid \mathbb{D})$ is equivalent to a multivariate normal distribution whereby:

$$\boldsymbol{W} \sim \mathcal{N}(\boldsymbol{W}^*, \boldsymbol{H}^{-1}).$$

MODEL IMPLEMENTATION

The negative log likelihood is given by:

$$
\begin{aligned}
-\log P(\mathbb{D} \mid \boldsymbol{W}) &= -\log \prod_i \prod_c P(Y_i = c|x_i, \boldsymbol{W})^{\mathbf{1}_{\mathrm{Y_i=c}}} \\
&= -\sum_i \sum_c \mathbf{1}_{\mathrm{Y_i=c}} \log \left[\frac{\exp(\boldsymbol{w}_c^T x_i)}{\sum_{c' \in C} \exp(\boldsymbol{w}_{c'}^T x_i)} \right] \\
&= -\sum_i \left[\sum_c \mathbf{1}_{\mathrm{Y_i=c}} \boldsymbol{w}_c^T x_i - \log \left(\sum_{c' \in \mathbb{C}} \exp(\boldsymbol{w}_{c'}^T x_i) \right) \right].
\end{aligned}
$$

Therefore, given prior $\boldsymbol{W}_0 \sim \mathcal{N}(\boldsymbol{M}_0, \boldsymbol{\Sigma}_0)$, $L(\boldsymbol{W})$ is:

$$
\begin{aligned}
L(\boldsymbol{W}) &= -\log(P(\mathbb{D} \mid \boldsymbol{W})\, p(\boldsymbol{W}_0)) \\
&= -\log P(\mathbb{D} \mid \boldsymbol{W}) - \log p(\boldsymbol{W}_0) \\
&= \sum_i \left[\sum_c \mathbf{1}_{\mathrm{y_i=c}} \boldsymbol{w}_c^T x_i - \log(\sum_{c' \in C} \exp \boldsymbol{w}_{c'}^T x_i) \right] \\
&+ \log(\sqrt{(2\pi)^k |\boldsymbol{\Sigma}_0|}) + (\boldsymbol{W} - \boldsymbol{M}_0)^T \boldsymbol{\Sigma}_0^{-1} (\boldsymbol{W} - \boldsymbol{M}_0). \qquad (4)
\end{aligned}
$$

For the learner model, we solved for $\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} L(\boldsymbol{W})$ using PyTorch L-BFGS optimizer because the size of the training set is small for the learner model. Also, we computed the hessian $\boldsymbol{H}$ using a hessian solver built upon PyTorch.[2] This learner model is then used to obtain the teaching set $\mathbb{D}$ as explanatory examples as described in the main text.

A.3 NORMAL PRIOR ON LEARNER MODEL

We used a normal distribution for the prior on the weights for the learner model: $\mathcal{N}(\boldsymbol{W}_0, \Sigma_0 = \boldsymbol{H}_0^{-1})$. For the mean of the prior $\boldsymbol{W}_0$, we used the weights obtained from training the classification layer of ResNet-50 on ImageNet over 100 epochs with data augmentation.[3] For the precision matrix $\Sigma_0^{-1} = \boldsymbol{H}_0$, we aimed to use the hessian of a loss function that has the same form as Equation 4, but with different values for the likelihood and prior. However, we did not evaluate the hessian directly—the classification layer of ResNet-50 has 1024 input features and 1000 classes, resulting in a $(1024 \times 1000) \times (1024 \times 1000)$ dimensional $\boldsymbol{H}_0$ matrix, which cannot be computed or stored. Therefore, we used Kronecker factored Laplace approximation to estimate $\boldsymbol{H}_0$, as detailed in the next section, following the work in (Ritter et al., 2018).

For the rest of Appendix A, we will simplify the notation to $\boldsymbol{W}_0 \rightarrow \boldsymbol{W}$ and $\boldsymbol{H}_0 \rightarrow \boldsymbol{H}$. Thus, in the following subsections $\boldsymbol{W}$ now refers to the learner model's prior weights as opposed to the posterior weights, and $\boldsymbol{H}$ refers to the learner model's precision matrix on the normal prior as opposed to that on the normal posterior. Also, $\mathbb{D}$ now refers to the entire ImageNet dataset as opposed to the teaching set.

KRONECKER FACTORED LAPLACE APPROXIMATION

For the classification layer of ResNet-50, let the input feature vector (including the bias term) be $\boldsymbol{z} = \{z_i, \ldots, z_m\}$ and the pre-activation vector be $\boldsymbol{a} = \{a_i, \ldots, a_n\}$ (i.e., the class activations before applying the softmax function). The $m \times n$ weight matrix $\boldsymbol{W}$, whereby the ith row of $\boldsymbol{W}$ is the weight vector $\boldsymbol{w}_i$ for class i, connects the two layers with $\boldsymbol{a} = \boldsymbol{W}\boldsymbol{z}$.

[2] https://github.com/mariogeiger/hessian

[3] https://github.com/pytorch/examples/blob/d587b53f3604b029764f8c864b6831d0ab269008/imagenet/main.py

Note that we can write the hessian of the loss $\mathbf{H} \triangleq \frac{\partial^2 L(\boldsymbol{W})}{\partial \boldsymbol{W}^2}|_{\boldsymbol{W}=\boldsymbol{W}^*}$ as a sum of individual hessian of the loss per data point $\mathbb{D}_i$ (without the prior) and the negative log prior:

$$\begin{aligned}
\boldsymbol{H} &= \frac{\partial^2 L(\boldsymbol{W})}{\partial \boldsymbol{W}^2} \\
&= -\frac{\partial^2}{\partial \boldsymbol{W}^2} \log(P(\mathbb{D} \mid \boldsymbol{W})\, p(\boldsymbol{W})) \\
&= -\frac{\partial^2}{\partial \boldsymbol{W}^2} [\log P(\mathbb{D} \mid \boldsymbol{W}) + \log p(\boldsymbol{W})] \\
&= -\frac{\partial^2}{\partial \boldsymbol{W}^2} \left[\log \prod_i P(\mathbb{D}_i \mid \boldsymbol{W}) + \log p(\boldsymbol{W}) \right] \\
&= -\frac{\partial^2}{\partial \boldsymbol{W}^2} \left[\sum_i \log P(\mathbb{D}_i \mid \boldsymbol{W}) + \log p(\boldsymbol{W}) \right] \\
&= \sum_i \frac{\partial^2 L_i(\boldsymbol{W})}{\partial \boldsymbol{W}^2} - \frac{\partial^2 \log p(\boldsymbol{W})}{\partial \boldsymbol{W}^2}.
\end{aligned}$$

The first derivative of the individual loss per data point $L_i(\boldsymbol{W})$, with respect to a weight $W_{i,j}$ connecting an input feature z_j and a pre-activation node a_i is:

$$\begin{aligned}
\frac{\partial L_i(\boldsymbol{W})}{\partial W_{i,j}} &= \frac{\partial L_i(\boldsymbol{W})}{\partial a_i} \frac{\partial a_i}{\partial W_{i,j}} \\
&= z_j \frac{\partial L_i(\boldsymbol{W})}{\partial a_i}.
\end{aligned}$$

The second derivative with respect to another weight $W_{k,l}$ is:

$$\begin{aligned}
\frac{\partial}{\partial W_{k,l}} \frac{\partial L_i(\boldsymbol{W})}{\partial W_{i,j}} &= \frac{\partial}{\partial W_{k,l}} z_j \frac{\partial L_i(\boldsymbol{W})}{\partial a_i} \\
&= z_j \frac{\partial}{\partial a_i} \frac{\partial L_i(\boldsymbol{W})}{\partial W_{k,l}} \\
&= z_j z_l \frac{\partial^2 L_i(\boldsymbol{W})}{\partial a_i \partial a_k}.
\end{aligned}$$

We can express the hessian of $L_i(\boldsymbol{W})$ over $\boldsymbol{W}$ using a Kronecker product. Let $\mathbf{w} \triangleq vec(\boldsymbol{W})$, where the vec operator stacks the columns of $\boldsymbol{W}$ into a vector. Also, define $\mathbf{Z} \triangleq \boldsymbol{z}\boldsymbol{z}^T$ to be the outer product of $\boldsymbol{z}$, and $\mathbf{A}$ to be the hessian of $L_i(\boldsymbol{W})$ over $\boldsymbol{a}$ such that $\boldsymbol{A}_{i,j} \triangleq \frac{\partial^2 L_i(\boldsymbol{W})}{\partial a_i \partial a_j}$. Then:

$$\frac{\partial^2 L_i(\boldsymbol{W})}{\partial \boldsymbol{w}^2} = \boldsymbol{Z}_i \otimes \boldsymbol{A}_i \ .$$

We set the prior precision to be $\tau \boldsymbol{I}$, where τ is a constant controlling the precision.[4] Then the negative log prior is:

[4]This corresponds to L2 regularization. Also, this is the prior for training the classification layer of the original ResNet-50, not the prior on the learner model's weights.

$$
\begin{aligned}
-\frac{\partial^2}{\partial \boldsymbol{w}^2} \log P(\boldsymbol{w}) &= \frac{\partial^2}{\partial \boldsymbol{w}^2} \left[\log(\sqrt{(2\pi)^k |\tau^{-1}\boldsymbol{I}|}) + \frac{1}{2}(\boldsymbol{w}-\boldsymbol{w}_0)^T \tau \boldsymbol{I} (\boldsymbol{w}-\boldsymbol{w}_0) \right] \\
&= \tau \boldsymbol{I} \frac{\partial^2}{\partial \boldsymbol{w}^2} \left[\frac{1}{2} ||\boldsymbol{w}-\boldsymbol{w}_0|| \right] \\
&= \tau \boldsymbol{I} \frac{\partial}{\partial \boldsymbol{w}} (\boldsymbol{w}-\boldsymbol{w}_0) \\
&= \tau \boldsymbol{I}.
\end{aligned}
$$

Now we could express $\boldsymbol{H}$ as a Kronecker product. Using probability notation to denote $\mathbb{E}[\boldsymbol{Z}]$ and $\mathbb{E}[\boldsymbol{A}]$ as the mean of $\boldsymbol{Z}_i$ and $\boldsymbol{A}_i$ over the whole dataset of size n, and presuming that $\boldsymbol{z}$ and $\boldsymbol{a}$ are independent, $\boldsymbol{H}$ is:

$$
\begin{aligned}
\boldsymbol{H} &= \sum_i \frac{\partial^2 L_i(\boldsymbol{W})}{\partial \boldsymbol{w}^2} - \frac{\partial^2 \log P(\boldsymbol{w})}{\partial \boldsymbol{w}^2} \\
&= \sum_i \boldsymbol{Z}_i \otimes \boldsymbol{A}_i + \tau \boldsymbol{I} \\
&= n\mathbb{E}[\boldsymbol{Z}_i \otimes \boldsymbol{A}_i] + \tau \boldsymbol{I} \\
&= n[\mathbb{E}[\boldsymbol{Z}_i] \otimes \mathbb{E}[\boldsymbol{A}_i]] + \tau \boldsymbol{I} \\
&= \left(\sqrt{n}\mathbb{E}[\boldsymbol{Z}_i]\right) \otimes \left(\sqrt{n}\mathbb{E}[\boldsymbol{A}_i]\right) + \tau \boldsymbol{I}.
\end{aligned}
$$

To incorporate $\tau \boldsymbol{I}$ into the Kronecker product:

$$
\begin{aligned}
\boldsymbol{H} &= \left(\sqrt{n}\mathbb{E}[\boldsymbol{Z}_i]\right) \otimes \left(\sqrt{n}\mathbb{E}[\boldsymbol{A}_i]\right) + \tau \boldsymbol{I} \\
&\approx \left(\sqrt{n}\mathbb{E}[\boldsymbol{Z}_i] + \sqrt{\tau}\boldsymbol{I}\right) \otimes \left(\sqrt{n}\mathbb{E}[\boldsymbol{A}_i] + \sqrt{\tau}\boldsymbol{I}\right).
\end{aligned}
$$

Considering that n is large (1 million for Imagenet) and that the prior is weak, the regularization effect is negligible. Hence, we set τ to 0.

MATRIX NORMAL POSTERIOR

Once we have expressed $\boldsymbol{H}$ as a Kronecker product, we can express the learner model's prior of the weights (which confusingly is a posterior itself from training on the ImageNet dataset) as a matrix normal distribution, where the covariance is broken down into two manageable 1024×1024 and 1000×1000 dimension matrices.

Considering that the inverse of a Kronecker product is the Kronecker product of the inverses and defining $\mathbf{U} \triangleq \sqrt{n}\mathbb{E}[\boldsymbol{Z}_i] + \sqrt{\tau}\boldsymbol{I}$ and $\mathbf{V} \triangleq \sqrt{n}\mathbb{E}[\boldsymbol{A}_i] + \sqrt{\tau}\boldsymbol{I}$:

$$
\boldsymbol{H}^{-1} = (\boldsymbol{U} \otimes \boldsymbol{V})^{-1} = \boldsymbol{U}^{-1} \otimes \boldsymbol{V}^{-1}
$$

Then[5]:

$$
\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{w}^*, \boldsymbol{H}^{-1}) \iff \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{w}^*, \boldsymbol{U}^{-1} \otimes \boldsymbol{V}^{-1}) \iff \boldsymbol{W} \sim \mathcal{MN}(\boldsymbol{W}^*, \boldsymbol{U}^{-1}, \boldsymbol{V}^{-1})
$$

We can sample from the matrix normal distribution using Cholesky decomposition. Letting $\boldsymbol{u}$ be the lower triangular matrix of $\boldsymbol{U}^{-1} = \boldsymbol{u}\boldsymbol{u}^T$, $\boldsymbol{v}$ be the upper triangular matrix of $\boldsymbol{V}^{-1} = \boldsymbol{v}^T\boldsymbol{v}$, and $\mathbf{Q} \sim \mathcal{MN}(0, \boldsymbol{I}, \boldsymbol{I})$ be the standard normal distribution (which can be sampled from a univariate standard normal distribution and reshaped into an $m \times n$ matrix):

[5]We believed that in the original paper (Ritter et al., 2018) the authors mistakenly swapped $\boldsymbol{U}$ and $\boldsymbol{V}$.

$$\boldsymbol{W}_s = \boldsymbol{W}^* + \boldsymbol{u}\mathbf{Q}_s\boldsymbol{v}$$

The validation accuracy of the Kronecker factored probabilistic ResNet-50 on ImageNet, using Monte Carlo sampling drawn from the matrix normal posterior, is 75.9% for top-1 and 92.8% for top-5, close to the original ResNet-50 accuracy.

B APPENDIX

Following (Yang et al., 2021), we generated saliency maps by using Bayesian Teaching to select *pixels* of an image that help a learner model to the targeted prediction. Let $q_{teacher}(m \mid Y = c, x)$ be the probability that a mask m will lead the learner model to predict the image x to be in category c when the mask is applied to the image. This is expressed by Bayes' rule as

$$q_{teacher}(m \mid Y = c, x) = \frac{Q_{learner}(Y = c \mid x, m)p(m)}{\int_{\Omega_M} Q_{learner}(Y = c \mid x, m)p(m)}.$$

Here, $Q_{learner}(Y = c \mid x, m)$ is the probability that the ResNet-50 model with pre-trained ImageNet weights will predict the x masked by m to be c; $p(m)$ is the prior probability of m; and $\Omega_M = [0, 1]^{W \times H}$ is the space of all possible masks on an image with $W \times H$ pixels. We used a sigmoid-function squashed Gaussian process prior for $p(m)$.

Instead of sampling the saliency maps directly from the above equation, we find the expected saliency map for each image by Monte Carlo integration:

$$\begin{aligned} \mathrm{E}[M \mid x, c] &= \int_{\Omega_M} m \; q_{teacher}(m \mid Y = c, x) \\ &\approx \frac{\sum_{i=1}^{N} m_i \; Q_{learner}(Y = c \mid x, m_i)}{\sum_{i=1}^{N} Q_{learner}(Y = c \mid x, m_i)}, \end{aligned} \tag{5}$$

where m_i are samples from the prior distribution $p(m)$, and $N = 1000$ is the number of Monte Carlo samples used. The expected mask is used as the saliency map.

B.1 IMPLEMENTATION

To generate the saliency map for an image x, we first resized x to be 224-by-224 pixels. A set of 1000 2D functions were sampled from a 2D Gaussian process (GP) with an overall variance of 100, a constant mean of -100, and a radial-basis-function kernel with length scale 22.4 pixels in both dimensions. The sampled functions were evaluated on a 224-by-224 grid, and the function values were mostly in the range of $[-500, 300]$. A sigmoid function, $1/(1 + \exp(-a))$, was applied to the sampled functions to transform each of the function values a to be within the range $[0, 1]$. This resulted in 1000 masks. The mean of the GP controlled how many effective zeros there were in the mask, and the variance of the GP determined how fast neighboring pixel values in the mask changed from zero to one. The 1000 masks were the m_i's in Equation 5. We produced 1000 masked images by element-wise multiplying the image x with each of the masks. The term $Q_{learner}(Y = c \mid x, m_i)$ was the ResNet-50's predictive probability that the i^{th} masked image was in category c. Having obtained these predictive probabilities, we averaged the 1000 masks according to Equation 5 to produce the saliency map of image x.