nature machine intelligence

ARTICLES

https://doi.org/10.1038/s42256-021-00343-w

Improving performance of deep learning models with axiomatic attribution priors and expected gradients

Gabriel Erion[1,2,4], Joseph D. Janizek[1,2,4], Pascal Sturmfels[1,4], Scott M. Lundberg[1,3] and Su-In Lee[1] ✉

Recent research has demonstrated that feature attribution methods for deep networks can themselves be incorporated into training; these attribution priors optimize for a model whose attributions have certain desirable properties—most frequently, that particular features are important or unimportant. These attribution priors are often based on attribution methods that are not guaranteed to satisfy desirable interpretability axioms, such as completeness and implementation invariance. Here we introduce attribution priors to optimize for higher-level properties of explanations, such as smoothness and sparsity, enabled by a fast new attribution method formulation called expected gradients that satisfies many important interpretability axioms. This improves model performance on many real-world tasks where previous attribution priors fail. Our experiments show that the gains from combining higher-level attribution priors with expected gradients attributions are consistent across image, gene expression and healthcare datasets. We believe that this work motivates and provides the necessary tools to support the widespread adoption of axiomatic attribution priors in many areas of applied machine learning. The implementations and our results have been made freely available to academic communities.

Recent work on interpreting machine learning (ML) models has focused on feature attribution methods. Given an input datum, a model and a prediction, such methods assign a number to each input feature that represents how important the feature was for making the prediction. Current research also investigates the axioms that attribution methods should satisfy[1–4] and how they provide insight into model behaviour[5–8]. Feature attribution methods often reveal problems in a model or dataset. For example, a model may place too much importance on undesirable features, rely on many features when sparsity is desired or be sensitive to high-frequency noise. In such cases, humans often have a prior belief about how a model should treat input features but find it difficult to mathematically encode this prior for neural networks in terms of the model parameters.

One method to address such problems is what we call an attribution prior: if it is possible for explanations to reveal problems in a model, then constraining the model's explanations during training can help the model avoid such problems. It is worth noting that the vast majority of feature attribution methods focus exclusively on explaining why a given prediction was made. Only a very small number of papers have investigated incorporating attributions themselves into model training. The first such paper, by Ross et al.[9], used a binary indicator of whether each feature should or should not be important for making predictions on each sample in the dataset and penalized the gradients of unimportant features. A very recent publication successfully used the gradient-based prior of Ross et al. as part of a human-in-the-loop strategy to improve model generalization performance and user trust, as well as contributing their own model-agnostic method for penalizing feature importances[10]. Such results create a clear synergy with our study, which improves the quality of calculated feature importances and develops new forms of attribution priors. This has the potential to greatly expand both the number of ways that a human-in-the-loop can influence deep models and the precision with which they can do so. However, two drawbacks limit this method's applicability to real-world problems. First, gradients do not satisfy the same theoretical guarantees as modern feature attribution methods. This leads to well-known problems such as saturation: operations, such as rectified linear units (ReLUs) and sigmoids, which have large flat 'saturated' regions, can lead to zero gradient attribution even for important features[2]. Second, it can be difficult to specify which features should be important in a binary manner.

Additional recent work discusses the need for priors that incorporate human intuition to develop robust and interpretable models[11]. Still, it remains challenging to encode priors such as 'have smoother attributions across an image' or 'treat this group of features similarly' by penalizing a model's input gradients or parameters. Some recent attribution priors have proposed regularizing integrated gradients (IG) attributions[12,13]. While promising, this work suffers from three major weaknesses: it does not clearly demonstrate improvements over gradient-based attribution priors, it penalizes attribution deviation from a target value rather than encoding sophisticated priors such as those we mention above, and it imposes a large computational cost by training with tens to hundreds of reference samples per batch. A contemporary method called contextual decomposition explanation penalization (CDEP) uses a framework similar to attribution priors and penalizes explanations generated by the contextual decomposition method[14]. Unlike all other interpretability methods discussed in this paper, CDEP penalizes explanations for pre-specified groups of features, meaning it is best suited for a different set of problems than we consider. More discussion of CDEP can be found in 'Attribution priors are a flexible framework for encoding domain knowledge' and Supplementary Sections A and B.

[1]Paul G. Allen School of Computer Science and Engineering, University of Washington, Seattle, WA, USA. [2]Medical Scientist Training Program, University of Washington, Seattle, WA, USA. [3]Microsoft Research, Seattle, WA, USA. [4]These authors contributed equally: Gabriel Erion, Joseph D. Janizek, Pascal Sturmfels. ✉e-mail: suinlee@cs.washington.edu

The main contribution of this work is a broadened interpretation of attribution priors that includes any case in which the training objective incorporates differentiable functions of a model's feature attributions. This can be seen as a generalization of gradient-based regularization[9,15–18] and it can be used to encode meaningful domain knowledge more effectively than existing methods. Whereas previous attribution priors generally took the form of 'encourage feature *i*'s attribution to be near a pre-determined target value', the priors we present here consider relative importance among multiple features and do not require pre-determined target values for any feature's attribution. Specifically, we introduce an image prior enforcing that neighbouring pixels have similar attributions, a graph prior for biological data enforcing that related genes have similar attributions, and a sparsity prior enforcing that a few features have large attributions while all others have near-zero attributions.

We also introduce a new general-purpose feature attribution method to enforce these priors, expected gradients (EG). As mentioned above, virtually all attribution methods are designed to explain a model's prediction to humans, not to be penalized during training. This means many such methods may be computationally difficult to incorporate into the training process. EG is an attribution method explicitly designed for regularization as an attribution prior (Fig. 1a); it can be efficiently regularized during training due to its formulation as an expectation, which naturally lends itself to batched estimates of the attribution. It also eliminates a hyperparameter choice required by IG[2]. Since these attributions are used not only to interpret trained models but also as part of the training objective itself, it is essential to guarantee that the attributions will be of high quality. We therefore show that our attribution method satisfies important interpretability axioms.

Across three different prediction tasks, we show that training with EG outperforms training with previous, more limited versions of attribution priors. On images, our image prior produces a model that is more interpretable and generalizes better to noisy data. On gene expression data, our graph prior reduces prediction error and better captures biological signal. Finally, on a patient mortality prediction task, our sparsity prior yields a sparser model and improves performance when learning from limited training data.

Results

Attribution priors are a flexible framework for encoding domain knowledge. Let $X \in \mathbb{R}^{n\times p}$ denote a dataset with labels $y \in \mathbb{R}^{n\times o}$, where n is the number of samples, p is the number of features and o is the number of outputs. In standard deep learning, we find optimal parameters θ by minimizing loss $\mathcal{L}$, with a regularization term $\Omega'(\theta)$ weighted by λ' on the parameters:

$$\theta = \mathrm{argmin}_{\theta}\mathcal{L}(\theta;X,y) + \lambda'\Omega'(\theta).$$

Attribution priors involve a model's attributions, represented by the matrix $\Phi(\theta,X)$, where each entry ϕ_i^{ℓ} is the importance of feature i in the model's output for sample ℓ. The attribution prior is a scalar-valued penalty function of the feature attributions $\Omega(\Phi(\theta,X))$, which represents a log-transformed prior probability distribution over possible attributions (λ is the regularization strength). The attribution prior is modular and agnostic to the particular attribution method. This results in the optimization:

$$\theta = \mathrm{argmin}_{\theta}\mathcal{L}(\theta;X,y) + \lambda\Omega(\Phi(\theta,X)),$$

where the standard regularization term has simply been replaced with an arbitrary, differentiable penalty function on the feature attributions.

While feature attributions have previously been used in training (more details in 'Previous attribution priors' in Methods)[9,12], our approach offers two novel components. First, we demonstrate that calculating Φ with attribution methods that satisfy previously established interpretability axioms improves performance (see 'EG outperforms other attribution methods' and 'Expected gradients' in Methods for further discussion of interpretability axioms). Second, rather than simply encouraging each feature's attribution to be near a target value as in previous work, we enforce high-level priors over the relationships between features.

In image data, we use a Laplace zero-mean prior on the difference between attributions of adjacent pixels, which encourages a low total variation (high smoothness) of attributions:

$$\Omega_{\mathrm{pixel}}(\Phi(\theta,X)) = \sum_{\ell}\sum_{i,j}|\phi^{\ell}_{i+1,j} - \phi^{\ell}_{i,j}| + |\phi^{\ell}_{i,j+1} - \phi^{\ell}_{i,j}|,$$

where i,j indexes the pixels of an image by rows and columns, respectively and ℓ indexes each image.

In gene expression data, we use a Gaussian zero-mean prior on the difference between mean absolute attributions $\bar{\phi}_i$ of functionally related genes, which encourages such similar genes to have similar attributions:

$$\Omega_{\mathrm{graph}}(\Phi(\theta,X)) = \sum_{i,j} W_{i,j}(\bar{\phi}_i - \bar{\phi}_j)^2 = \bar{\phi}^T L_{\mathrm{G}}\bar{\phi},$$

where T represents a vector transpose, $W_{i,j}$ is the weight of connection between two genes in a biological graph, and L_{G} is the graph Laplacian.

Finally, in health data where sparsity is desired, we use a prior on the Gini coefficient of the mean absolute attributions $\bar{\phi}_i$, which encourages a small number of features to have a large percentage of the total attribution while others are near zero:

$$\Omega_{\mathrm{sparse}}(\Phi(\theta,X)) = -\frac{\sum_{i=1}^{p}\sum_{j=1}^{p}|\bar{\phi}_i - \bar{\phi}_j|}{n\sum_{i=1}^{p}\bar{\phi}_i} = -2G(\bar{\phi}),$$

where G is the Gini coefficient.

None of these priors requires specifying target values for features, and all improve performance over simpler baselines. For more details on our priors, see 'Specific priors' in Methods, and for more details on previous attribution priors, see 'Previous attribution priors' in Methods. We also note that these priors involve the relationships between the attributions for all features in the dataset. Gradients, IG and our method (EG) discussed below are all designed for calculating such attributions. The CDEP method discussed above differs in that it penalizes the attributions of a single pre-specified group of features[14]; while CDEP has reported better performance with certain types of prior than EG and gradients, we believe this is due to the fact that the methods are inherently best suited to different types of prior. Using CDEP with the specific priors proposed in this work would require several orders of magnitude more backward passes of the model during training than our approach. CDEP also uses additional preprocessing steps that are not necessary in our approach, which further distinguishes the scenarios in which each method is most applicable. For further discussion of related work, including a discussion of specific cases for which our method and CDEP are best suited, see Supplementary Sections A and B.

EG outperforms other attribution methods. Attribution priors involve using feature attributions not just as a post-hoc analysis method but also as a key part of the training objective. Thus, it is essential to guarantee as much as possible that the attribution method used will produce high-quality attributions and run fast enough to be calculated for each training batch. We propose an

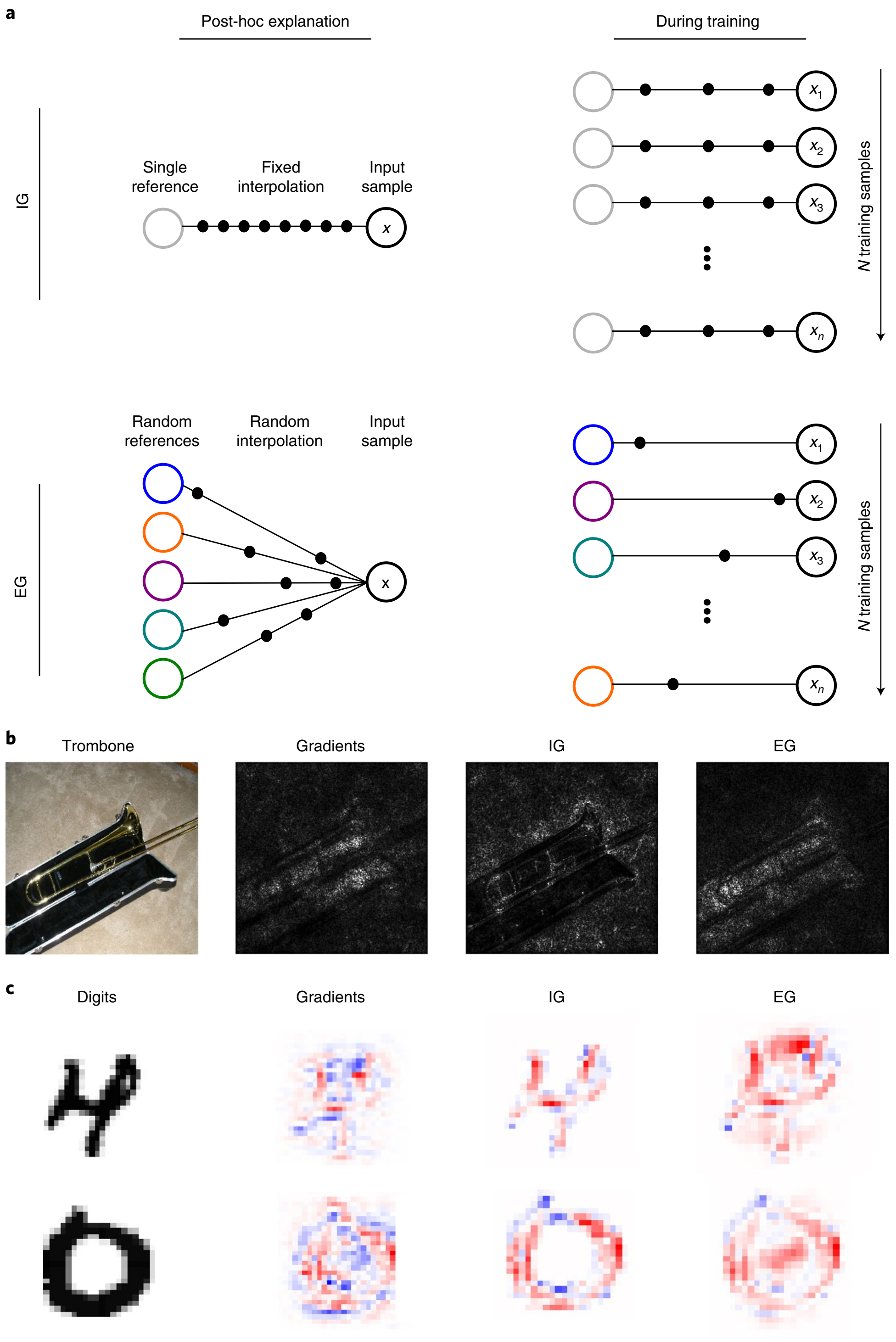

Fig. 1 | EG is a feature attribution method designed to be regularized during training. a, A comparison of our method, EG, to IG as both a post-hoc explanation method (left), and as a differentiable feature attribution to be penalized during training to enforce attribution priors (right). **b**, Comparison of saliency maps generated by three different attribution methods on an image from the ImageNet dataset. The saliency maps demonstrate how the IG attribution method fails to highlight black pixels as important when black is used as a baseline input, while EG is capable of highlighting the black pixels in these images as important. **c**, Comparison of saliency maps for the same three attribution methods for two MNIST digits. Again, IG fails to highlight potentially relevant image regions (like the empty middle of the 0 or the empty region at the top of the 4, which might make the digit resemble a 9 if it were filled in).

axiomatic feature attribution method called expected gradients (EG), which avoids problems with existing methods and is naturally suited to being incorporated into training. EG extends the IG method[2], and like IG, satisfies a variety of desirable interpretability axioms such as completeness (the feature attributions sum to the output for a given sample) and implementation invariance (the attributions are identical for any of the infinite possible implementations of the same function). Because these methods satisfy completeness, they are not subject to the problems with input saturation that affect gradient attributions. Because these methods satisfy

Table 1 | Synthetic data benchmark results for attribution methods

Method	Remove positive	Remove negative	Remove absolute	Convergence time
EG	**3.612**	**3.759**	**0.897**	**0.150**
IG	3.539	3.687	0.872	0.989
Gradients	0.035	0.110	0.729	0.250
Random	−0.053	0.034	0.400	-

Larger numbers mean a better feature attribution method for all metrics other than convergence time, for which a smaller number indicates faster convergence. The first three metrics measure the quality of the method for correctly identifying important features, whereas convergence time indicates how effectively the method is regularized during training as an attribution prior. The 'remove positive' metric measures the average magnitude change in model output when the features identified as having the largest positive impact by each method are masked by the feature mean, whereas 'remove negative' measures the average magnitude change in model output when the features identified as having the largest negative impact by each method are masked by the feature mean. The 'remove absolute' metric measures the average increase in model loss when the features identified as having the largest magnitude impact on the model are masked by the feature mean. Each model is trained on 900 samples and tested using 100 samples. EG attains the best benchmark scores of all of the tested attribution methods ($P=7.2\times10^{-5}$, one-tailed binomial test, tested across all 18 attribution performance metrics, see Supplementary Section D for details on exact calculation of these metrics and exhaustive list of metrics considered).

implementation invariance, they are straightforward to practically apply to any differentiable model, regardless of specific network architectures (see 'Expected gradients' in Methods for an extended discussion of the interpretability axioms satisfied by EG).

IG generates feature attributions by integrating the gradients of the model's output f between the sample of interest and a reference sample x' (Fig. 1a, left).

$$\mathrm{IG}_i(x) := \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} \mathrm{d}\alpha$$

where ∂ represents a partial derivative and α represents progress along the integration path. If the attribution function Φ in our attribution prior $\Omega(\Phi(\theta,X))$ is IG, regularizing Φ would require hundreds of extra gradient calls every training step (the original IG paper[2] recommends 20 to 300 gradient calls to compute attributions). This makes training with IG prohibitively slow—in fact, ref. [12] finds that using IG can take up to 30 times longer than standard training even when only back-propagating gradients through part of the network. However, most deep learning models today are trained using some variant of batch gradient descent, where the gradient of a loss function is approximated over many training steps using mini-batches of data. We can dramatically improve speed over an IG attribution prior by using a similar idea and formulating the IG integral as an expectation over integration path steps α drawn from a uniform distribution U (see Table 1 and Supplementary Section D.1 for more details on convergence time benchmark). This Monte Carlo estimate of the integral is the core of our EG method, defined below for a single reference x':

$$\mathrm{SingleRefEG}_i(x) = \mathbb{E}_{\alpha\sim U(0,1)}\left[\,(x_i - x'_i) \times \frac{\partial f(x' + \alpha \times (x - x'))}{\partial x_i}\,\right]$$

Just like the gradient of the loss, EG attributions can be calculated in a batched manner during training (Fig. 1a, right). We let k be the number of samples we draw for this Monte Carlo integral at each mini-batch. Remarkably, because the variance in each batched EG attribution will be smoothed over thousands of batches during training, we find that as small as $k=1$ suffices to regularize the explanations.

This expectation formulation also enables us to solve a long-standing problem with IG as an attribution method—the choice of the required background reference x'. For example, in image tasks, the image of all zeros is often chosen as a baseline, but doing so implies that black pixels will not be highlighted as important (Fig. 1b,c). This problem can be solved by integrating gradients over multiple references. However, calculating multiple Riemann integrals is expensive in terms of time and memory, probably prohibitively so if calculated during every batch of training (Fig. 1a, right). EG naturally accommodates multiple references by performing the Monte Carlo integral with samples from multiple references and interpolation points (here, x is the sample, x' is a reference and D is the reference distribution):

$$\mathrm{EG}_i(x) = \mathbb{E}_{x'\sim D,\alpha\sim U(0,1)}\left[\,(x_i - x'_i) \times \frac{\partial f(x' + \alpha \times (x - x'))}{\partial x_i}\,\right]$$

In principle, any distribution D over reference samples could be used to calculate EG attributions; choosing which distribution to use depends on the nature of the attribution problem. For example, setting D to be a single sample recovers single-reference EG: the same reference setup as IG but with the Monte Carlo speedup of EG (Supplementary Section D.1). By default, we do not choose D to be a single sample but rather a uniform distribution over the entire training set. This tells us which features cause x's output to be different from the output at all other points in the dataset, on average. In certain cases, we may want to use a different distribution D. For example, we might want to distinguish between subgroups and understand why a digit is classified as a 'seven' rather than a 'one' by choosing references only from the 'one'-labelled training samples. We could also account for baseline subgroup characteristics by explaining, for example, an 80-year-old patient's mortality risk relative to other 80 year olds; this could prevent age and age-correlated features from being trivially listed as the most important. While our formulation and implementation of EG support any choice of distribution D, the examples in this paper do not focus on subgroup analysis, so we set D to be a uniform distribution over the training set (see 'Expected gradients' in Methods and Supplementary Section C for implementation details and pseudocode).

In a simple experiment using synthetic data to assess the impact of k on the convergence time of model training (rather than the convergence of a single explanation), we found that regularizing EG with $k=1$ was more effective at removing a model's dependency on one of two correlated features than gradients or even IG with more than k samples (Table 1, Supplementary Section D and Supplementary Fig. 4). The $k=1$ setting also appeared optimal for EG; setting $k>1$ required more total gradient calls for convergence (Supplementary Section D.1 and Supplementary Fig. 3). We also compare EG to other feature attribution methods using synthetic data benchmarks introduced in ref. [5] (Table 1), which are available as part of the SHAP software package. These benchmark metrics evaluate whether each attribution method finds the most important features for a given dataset and model. EG significantly outperforms the next best feature attribution method ($P=7.2\times10^{-5}$, one-tailed binomial test). We believe this demonstrates another benefit of EG; by averaging attributions over multiple reference samples, it becomes more robust to the wide array of patterns of missingness and re-imputation tested in the benchmark. We provide more details and additional benchmarks in Supplementary Section D.

A pixel attribution prior improves robustness to image noise. Previous work on interpreting image models has focused on creating pixel attribution maps, which assign a value to each pixel indicating how important that pixel was for a model's prediction[2,19]. Attribution maps can be noisy and difficult to understand due to their tendency to highlight seemingly unimportant background pixels, indicating the model may be vulnerable to adversarial attacks[20]. Although we may prefer a model with smoother attributions, existing methods

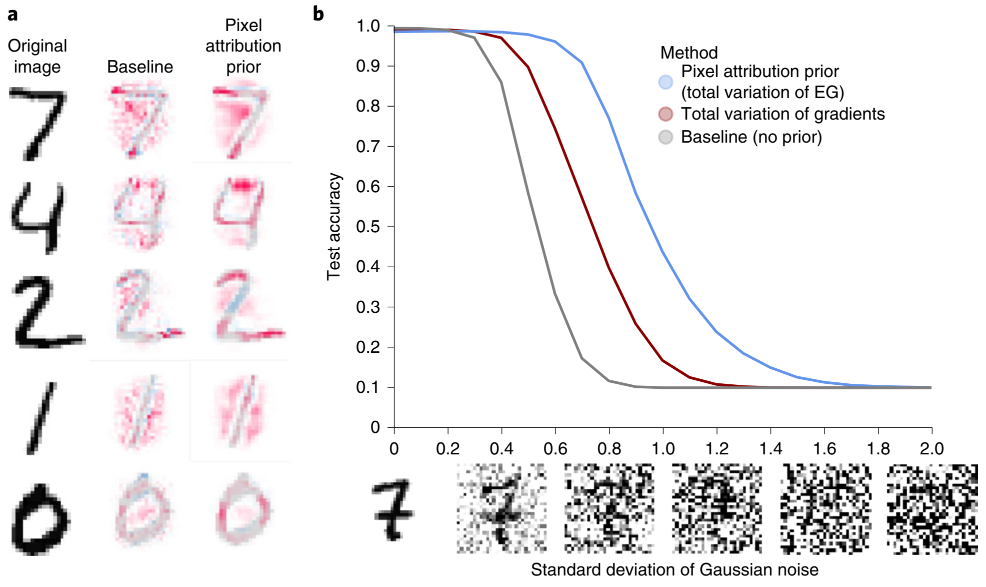

Fig. 2 | Pixel attribution prior improves saliency map smoothness and increases robustness of MNIST classifier to noise. a, EG attributions (from 100 samples) on MNIST for both an unregularized model and a model trained with an attribution prior regularized using EG. The latter achieves visually smoother attributions, and it better highlights how the network classifies digits (for example, the top part of the 4 being very important). Unlike previous methods that take additional steps to smooth saliency maps after training[21,22], these are unmodified saliency maps directly from the learned model. **b**, Training with an attribution prior on total variance of EG attributions induces robustness to Gaussian noise without specifically training for robustness. This robustness greatly exceeds that provided by an attribution prior on the total variance of model gradients. Shaded bars around each line indicate standard deviation of the accuracy results; however, the bars are small enough to be indistinguishable in this plot.

only post-process attribution maps but do not change model behaviour[19,21,22]. Such techniques may not be faithful to the original model[11]. In this section, we describe how we applied our framework to train image models with naturally smoother attributions.

To regularize pixel-level attributions, we used the following intuition: neighbouring pixels should have a similar impact on an image model's output. To encode this intuition, we chose a total variation loss on pixel-level attributions (see 'Specific priors' in Methods for more detail). We applied this pixel smoothness attribution prior to the Modified National Institute of Standards and Technology (MNIST) database, containing handwritten digits classified from 0 to 9, and the Canadian Institute for Advanced Research (CIFAR)-10 dataset, containing colour images classified into 10 categories such as cats, dogs and cars[15,23]. On MNIST we trained a two-layer convolutional neural network; for CIFAR-10 we trained a VGG16 network from scratch (see 'Image model experimental settings' in Methods for more details)[24]. In both cases, we optimized hyperparameters for the baseline model without an attribution prior. To choose λ, we searched over values in $[10^{-20}, 10^{-1}]$ and chose the λ that minimized the attribution prior penalty and achieved a test accuracy within 1% of the baseline model for MNIST and 10% for CIFAR-10. Figures 2 and 3 show EG attribution maps for both the baseline and the model regularized with an attribution prior on five randomly selected test images on MNIST and CIFAR-10, respectively. In all examples, the attribution prior yields a model with visually smoother attributions. Remarkably, in many instances, smoother attributions better highlight the target object's structure.

Recent work has suggested that image classifiers are brittle to small domain shifts: small changes in the underlying distribution of the training and test set can lead to large reductions in test accuracy[25]. To simulate a domain shift, we applied Gaussian noise to images in the test set and re-evaluated the performance of the regularized and baseline models. As an adaptation of ref. [9], we also compared the attribution prior model with regularizing the total variation of gradients with the same criteria for choosing λ. For each method, we trained five models with different random initializations. In Figs. 2 and 3, we plot the mean and standard deviation of test accuracy on MNIST and CIFAR-10, respectively, as a function of standard deviation of added Gaussian noise. The figures show that our regularized model is more robust to noise than both the baseline and gradient-based models.

Both the robustness and more intuitive saliency maps our method provides come at the cost of reduced test set accuracy (0.93 ± 0.002 for the baseline versus 0.85 ± 0.003 for pixel attribution prior model on CIFAR-10). Mathematically, adding a penalty term to the optimization objective should only ever reduce training set performance; it is reasonable that in many cases this can lead to a reduction in test-set performance as well. However, test accuracy is not the only metric of interest for image classifiers. The trade-off between robustness and accuracy that we observe is consistent with previous work that suggests image classifiers trained solely to maximize test accuracy rely on features that are brittle and difficult to interpret[11,26,27]. Despite this trade-off, we find that at a stricter hyperparameter cutoff for λ on CIFAR-10—within 1% test accuracy of the baseline, rather than 10%—our methods still achieve modest but significant robustness relative to the baseline. We also evaluated our method against several other attribution priors including IG and, for ablation purposes, single-reference EG (Supplementary Figs. 10 and 11). We found that the pixel attribution prior outperformed standard IG and that most of this additional performance was due to our random interpolation. Both the pixel attribution prior and single-reference EG were much more robust than all other methods; however, only the pixel attribution prior, which used multiple references, could highlight important foreground and background regions in addition to providing robustness and smoothness. For details of the EG versus IG comparison, results at different hyperparameter thresholds, more details on our training procedure and additional experiments on MNIST, CIFAR-10 and ImageNet, see Supplementary Sections E–H.

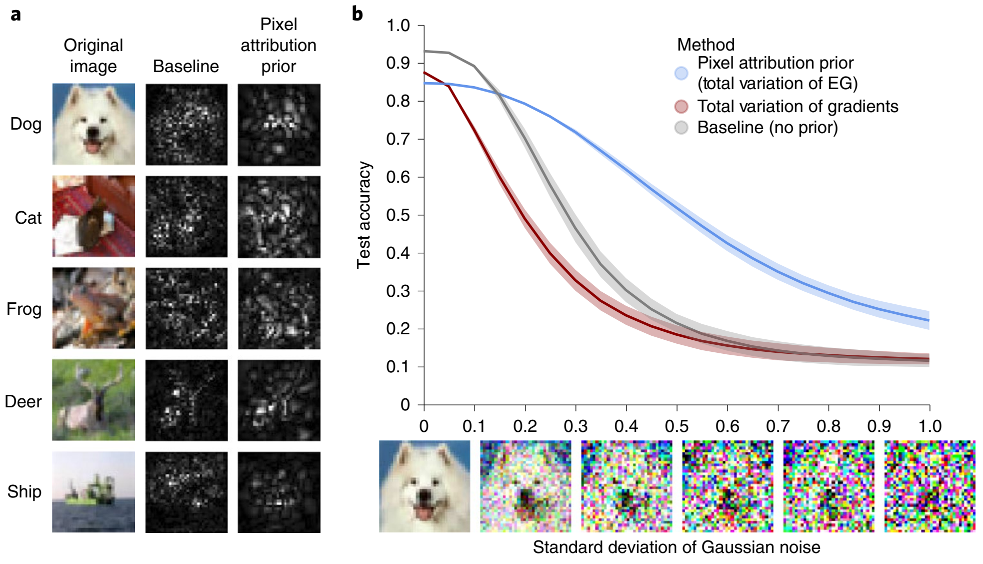

Fig. 3 | Pixel attribution prior improves saliency map smoothness and increases robustness of CIFAR-10 classifier to noise. a, EG attributions (from 100 samples) on CIFAR-10 for both the baseline model and the model trained with an attribution prior for five randomly selected images classified correctly by both models. Training with an attribution prior generates visually smoother attribution maps in all cases. Notably, these smoothed attributions also appear more localized towards the object of interest. **b**, Training with an attribution prior on total variance of EG attributions induces robustness to Gaussian noise, achieving more than double the accuracy of the baseline at high noise levels. This robustness is not achievable by choosing total variation of gradients as the attribution function. Shaded bars around each line indicate standard deviation of the accuracy results.

A graph attribution prior improves anticancer drug response prediction. In the image domain, our attribution prior took the form of a penalty encouraging smoothness over adjacent pixels. In other domains, there may be prior information about specific relationships between features that can be encoded as a graph (such as social networks, knowledge graphs or protein–protein interactions). For example, previous work in bioinformatics has shown that protein–protein interaction networks contain valuable information for improving performance on biological prediction tasks[28]. Therefore, in this domain, we regularized attributions to be smooth over the protein–protein feature graph analogously to the regular graph of pixels in the image.

Incorporating the Ω_{graph} attribution prior not only led to a model with more reasonable attributions but also improved predictive performance by letting us incorporate prior biological knowledge into the training process. We downloaded publicly available gene expression and drug response data for patients with acute myeloid leukaemia (AML, a type of blood cancer) and tried to predict patients' drug response from their gene expression[29]. For this regression task, an input sample was a patient's gene expression profile plus a one-hot encoded vector indicating which drug was tested in that patient, while the label we tried to predict was drug response (measured by IC50, a continuous value representing the concentration of the drug required to kill half of the patient's tumour cells). To define the graph used by our prior, we downloaded the tissue-specific gene-interaction graph for the tissue most closely related to AML in the HumanBase database[30].

A two-layer neural network trained with our graph attribution prior (Ω_{graph}) significantly outperforms all other methods in terms of test set performance as measured by R^2, which indicates the fraction of the variance in the output explained by the model (Fig. 4, see 'Biological experiments' in Methods for significance testing). Unsurprisingly, when we replace the biological graph from HumanBase with a randomized graph, we find that the test performance is no better than the performance of a neural network trained without any attribution prior. Extending the method proposed in ref. [9] by applying our new graph prior as a penalty on the model's gradients, rather than a penalty on the axiomatically correct expected gradient feature attribution, does not perform significantly better than a baseline neural network. We also observe substantially improved test performance when using the prior graph information to regularize a linear LASSO model. Finally, we note that our graph attribution prior neural network significantly outperforms graph convolutional neural networks, a recent method for utilizing graph information in deep neural networks[31].

To find out whether our model's attributions match biological domain knowledge, we first compared the list of top genes generated by our network trained with a graph attribution prior (ranked by mean absolute feature attribution) to a 'ground truth' list of AML-relevant genes found by querying the GeneCards database (Fig. 4b). When we count the number of AML-relevant genes at each position in our network's top gene list and compare this to the number of AML-relevant genes at each position in a standard neural network's top gene list, we see that the graph attribution prior network captures significantly more biologically relevant genes.

In addition, to check for biological pathway-level enrichments, we conducted gene set enrichment analysis (a modified Kolmogorov–Smirnov test). We measured whether our top genes, ranked by mean absolute feature attribution, were enriched for membership in any pathways (see 'Biological experiments' in Methods and Supplementary Section I for more detail, including the top pathways for each model)[32]. We find that the neural network with the tissue-specific graph attribution prior captures far more biologically relevant pathways (increased number of significant pathways after false discovery rate correction) than a neural network without attribution priors[33]. Furthermore, the pathways our model uses more closely match biological expert knowledge, that is, they included prognostically useful AML gene expression profiles as well as important AML-related transcription factors (Supplementary Section I)[34,35]. These results are expected, given that neural networks trained without priors can learn a relatively sparse basis of genes that will not enrich for specific pathways (for example, a single gene from each correlated pathway), while those trained with our graph prior will spread credit among functionally related

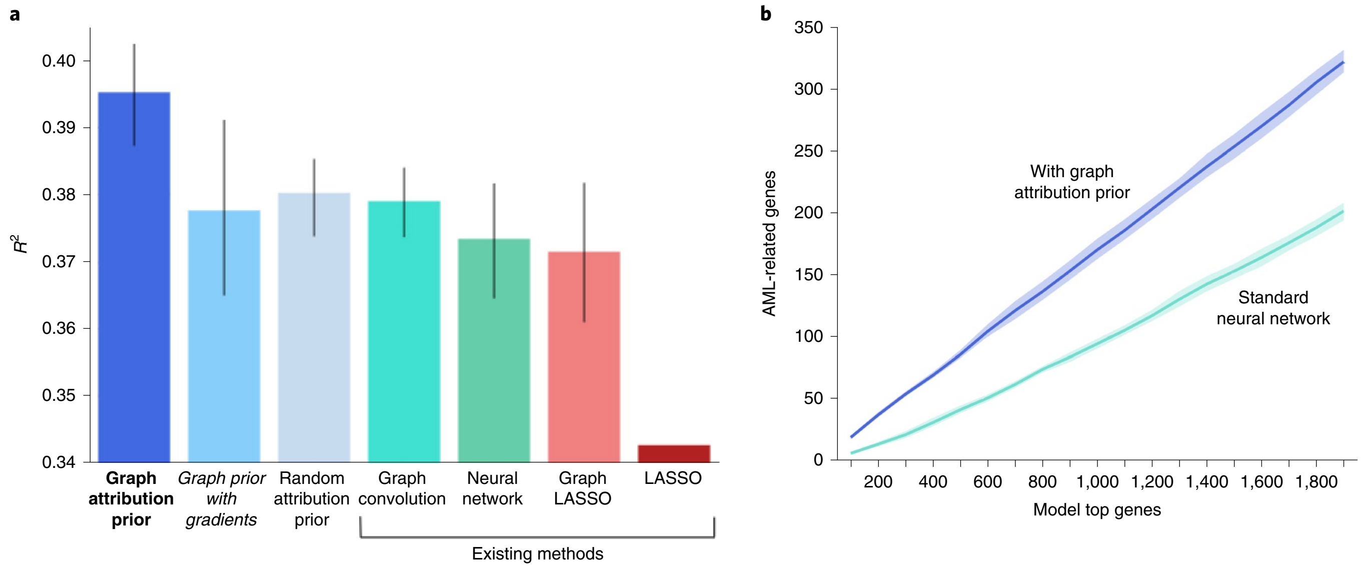

Fig. 4 | Graph attribution prior improves test accuracy and biological relevance of anticancer drug response prediction model. a, A neural network trained with our graph attribution prior (bold) attains the best test performance, while one trained with the same graph penalty on the gradients (italics, adapted from ref. [9]) does not perform significantly better than a standard neural network (error bars indicate the extent of the bootstrapped 95% confidence interval of the mean test set R^2 value, over ten retrainings of the model on random re-splits of the data). **b**, A neural network trained with our graph attribution prior gives more weight to AML-relevant genes than a standard neural network trained without the graph attribution prior (solid line indicates average over ten random re-splits of the data and retrainings of the model, error bands indicate the extent of the bootstrapped 95% confidence interval).

genes. This demonstrates the graph prior's value as an accurate and efficient way to encourage neural networks to treat functionally related genes similarly.

A sparsity prior improves performance with limited training data. Feature selection and sparsity are popular ways to alleviate the curse of dimensionality, facilitate interpretability and improve generalization by building models that use a small number of input features. A straightforward way to build a sparse deep model is to apply an L1 penalty to the first layer (and possibly subsequent layers) of the network. Similarly, the sparse group lasso (SGL) method penalizes all weights connected to a given feature[36,37], while a simple existing attribution prior approach[38] penalizes the gradients of each feature in the model.

These approaches suffer from two problems. First, a feature with small gradients or first-layer weights may still strongly affect the model's output[39]. A feature whose attribution value (for example, IG or EG) is zero is much less likely to have any effect on predictions. Second, successfully minimizing penalties such as L1—regardless of attribution type—is not necessarily the best way to create a sparse model. A model that puts weight w on 1 feature is penalized more than one that puts weight $w/2p$ on each of p features. Previous work on sparse linear regression has shown that the Gini coefficient G of the weights, proportional to 0.5 minus the area under the cumulative distribution function of sorted values, avoids such problems and corresponds more directly to a sparse model[40,41]. We extend this analysis to deep models by noting that the Gini coefficient can be written differentiably and used as an attribution prior.

Here we show that the Ω_{sparse} attribution prior can build sparser models that perform better in settings with limited training data. We use a publicly available healthcare mortality prediction dataset of 13,000 patients[42], whose 35 features (118 after one-hot encoding) represent medical data such as a patient's age, vital signs and laboratory measurements. The binary outcome is survival after ten years. Sparse models in this setting may enable accurate models to be trained with very few labelled patient samples or reduce cost by accurately risk-stratifying patients using few lab tests. We randomly sampled training and validation sets of only 100 patients each, placing all other patients in the test set, and ran each experiment 200 times with a new random sample to average out variance. We built three-layer binary classifier neural networks regularized using L1, SGL and sparse attribution prior penalties to predict patient survival, as well as an L1 penalty on gradients adapted for global sparsity from refs. [9,38]. The regularization strength was tuned from 10^{-7} to 10^5 using the validation set for all methods (see 'Sparsity experiments' in Methods and Supplementary Section J.2).

The sparse attribution prior enables more accurate test predictions (Fig. 5a) and sparser models (Fig. 5c) when limited training data is available, with $P < 10^{-4}$ and $t \geq 4.314$ by paired-samples t-test for all comparisons. We also plot the average cumulative importance of sorted features and find that the sparse attribution prior more effectively concentrates importance in the top few features (Fig. 5d). In particular, we observe that L1 penalizing the model's gradients as in ref. [38] rather than its EG attributions performs poorly in terms of both sparsity and performance. A Gini penalty on gradients improves sparsity but does not outperform other baselines such as SGL and L1 in area under a receiver operating characteristic curve (ROC AUC). Finally, we plot the average sparsity of the models (Gini coefficient) against their validation ROC AUC across the full range of regularization strengths. The sparse attribution prior exhibits higher sparsity than other models and a smooth trade-off between sparsity and ROC AUC (Fig. 5b). Details and results for other penalties, including L2, dropout and other attribution priors, are in Supplementary Section J.

Discussion

The immense popularity of deep learning has driven its application in many areas with diverse, complicated domain knowledge. While it is in principle possible to hand-design network architectures to encode this knowledge, a more practical approach involves the use of attribution priors, which penalize the importance a model

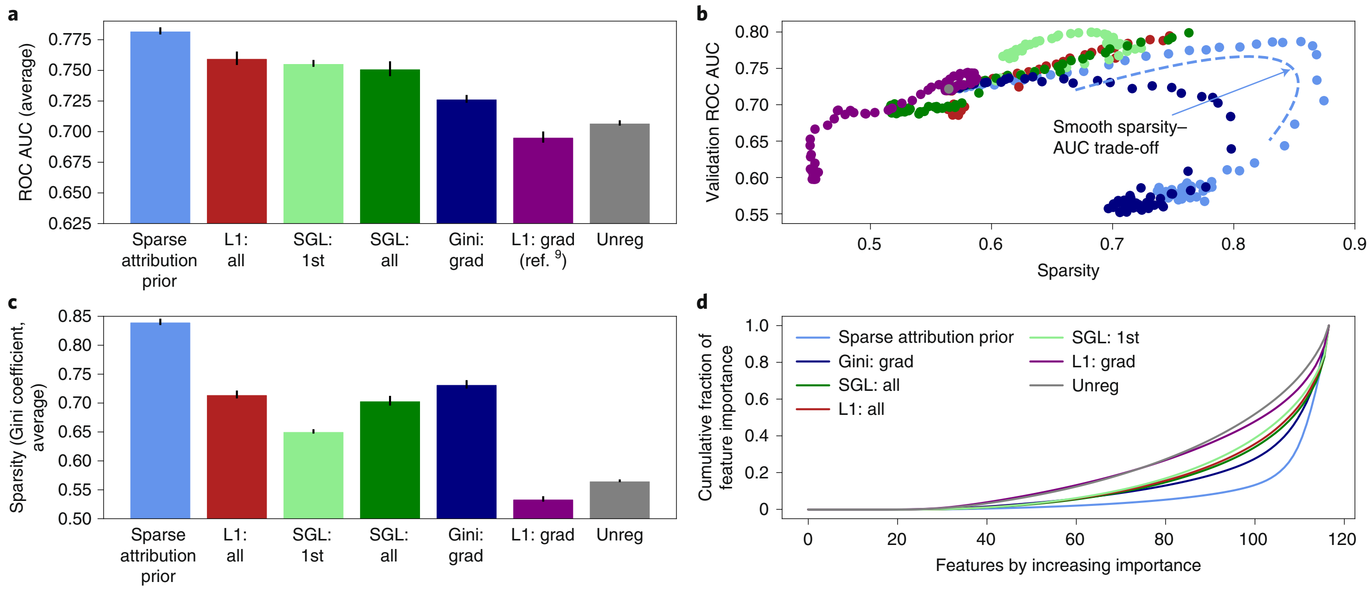

Fig. 5 | Sparse attribution prior builds sparser and more accurate healthcare mortality models. a,**c**, A sparse attribution prior enables more accurate test predictions (**a**) and sparser models (**c**) across 200 small subsampled datasets (100 training and 100 validation samples, all other samples used for test set) than other penalties, including gradients. **b**, Across the full range of tuned parameters, the sparse attribution prior achieves the greatest sparsity and a smooth sparsity–validation performance trade-off. **d**, A sparse attribution prior concentrates a larger fraction of global feature importance in the top few features. 'Gini', 'L1' and 'SGL' indicate the Gini, L1 and SGL penalties, respectively, 'grad' indicates a penalty on the gradients, 'all' indicates a penalty on all weights in the model and '1st' indicates a penalty on only the first weight layer. 'Unreg' indicates an unregularized model.

places on each of its input features when making predictions. Unfortunately, previous attribution priors have been limited, both theoretically and computationally. Binary penalties only specify whether features should or should not be important and fail to capture relationships among features. Approaches that focus only on a model's input gradients change the local decision boundary but often fail to impact a model's underlying decision-making. Attribution priors on more complicated attributions, such as IG, have proven computationally difficult.

Our work advances previous work both by introducing novel, flexible attribution priors for multiple domains and by enabling the training of such priors with a newly defined feature attribution method. Our priors lead to smoother and more interpretable image models, biological predictive models that incorporate graph-based prior knowledge and sparser healthcare models that perform better in data-scarce scenarios. Our attribution method not only enables the training of said priors but also outperforms its predecessor—IG—in terms of reliably identifying the features models use to make predictions.

There remain many avenues for future work in this area. We chose to base our prior on an improved version of IG because it is the most prominent differentiable feature attribution method we are aware of, but a wide array of other attribution methods exist. Our framework makes it straightforward to substitute any other attribution method as long as it is differentiable, and studying the effectiveness of other attribution methods as priors would be valuable. In addition, while we develop new, more sophisticated attribution priors and show their value, there is ample room to improve on our priors and evaluate entirely new ones for other tasks. Determining the best attribution priors for particular tasks opens a further avenue of research. We believe that surveys of domain experts to establish model desiderata for particular applications will help to develop the best priors for any given situation while offering a valuable opportunity to put humans in the loop. Overall, the dual advances of sophisticated attribution priors and EG enable a broader view of attribution priors: as tools to achieve domain-specific goals without sacrificing efficiency.

Methods

Previous attribution priors. The first instance of what we now call an attribution prior was proposed in ref. [9], where the regularization term was modified to place a constant penalty on the gradients of undesirable features:

$$\theta = \mathrm{argmin}_{\theta}\mathcal{L}(\theta;X,y) + \lambda''||A \odot \frac{\partial \mathcal{L}}{\partial X}||_F^2.$$

Here the attribution method is the gradients of the model, represented by the matrix $\frac{\partial \mathcal{L}}{\partial X}$ whose ℓ, ith entry is the gradient of the loss at the ℓth sample with respect to the ith feature. A is a binary matrix indicating that features should be penalized in which samples, and F is the Frobenius norm.

A more general interpretation of attribution priors is that any function of any feature attribution method could be used to penalize a loss function, thus encoding prior knowledge about what properties the attributions of a model should have. For some model parameters θ, let $\Phi(\theta, X)$ be a feature attribution method, which is a function of θ and the data X. Let ϕ_i^ℓ be the feature importance of feature i in sample ℓ. We formally define an attribution prior as a scalar-valued penalty function of the feature attributions $\Omega(\Phi(\theta,X))$, which represents a log-transformed prior probability distribution over possible attributions:

$$\theta = \mathrm{argmin}_{\theta}\mathcal{L}(\theta;X,y) + \lambda\Omega(\Phi(\theta,X)),$$

where λ is the regularization strength. Note that the attribution prior function Ω is agnostic to the attribution method Φ.

Previous attribution priors[9,12] required specifying an exact target value for the model's attributions, but often we do not know in advance which features are important in advance. In general, there is no requirement that $\Phi(\theta, X)$ constrain attributions to particular values. The 'Results' section presented three newly developed attribution priors for different tasks that improve performance without requiring pre-specified attribution targets for any particular feature.

Expected gradients. EG is an extension of IG[2] with fewer hyperparameter choices. Like several other attribution methods, IG aims to explain the difference between a model's current prediction and the prediction that the model would make when given a baseline input. This baseline input is meant to represent some uninformative reference input that represents not knowing the value of the input features. Although choosing such an input is necessary for several feature attribution methods[2,39,43], the choice is often made arbitrarily. For example, for image tasks, the image of all zeros is often chosen as a baseline, but doing so implies that black pixels will not be highlighted as important by existing feature attribution methods. In many domains, it is not clear how to choose a baseline that correctly represents a lack of information.

Our method avoids an arbitrary choice of baseline; it models not knowing the value of a feature by integrating over a dataset. For a model f, the IG value for feature i is defined as:

$$\mathrm{IG}_i(x, x') := (x_i - x') \times \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} \mathrm{d}\alpha,$$

where x is the target input and x' is baseline input. To avoid specifying x', we define the EG value for feature i as:

$$\mathrm{EG}_i(x) := \int_{x'} \left((x_i - x') \times \int_{\alpha=0}^{1} \frac{\partial f(x' + \alpha(x - x'))}{\partial x_i} \mathrm{d}\alpha \right) p_D(x')\mathrm{d}x',$$

where D is the underlying data distribution. Since EG is also a diagonal path method, it satisfies the same axioms as IG[44]. Directly integrating over the training distribution is intractable; therefore, we instead reformulate the integrals as expectations:

$$\mathrm{EG}_i(x) := \mathbb{E}_{x' \sim D, \alpha \sim U(0,1)} \left[(x_i - x') \times \frac{\partial f(x' + \alpha \times (x - x'))}{\partial x_i} \right].$$

This expectation-based formulation lends itself to a natural, sampling based approximation method: (1) draw samples of x' from the training dataset and α from $U(0, 1)$, (2) compute the value inside the expectation for each sample and (3) average over samples. For a pseudocode description of EG, see Supplementary Section C.

EG also satisfies a set of important interpretability axioms: implementation invariance, sensitivity, completeness, linearity and symmetry preserving.

- Implementation invariance states that two networks with outputs that are equal over all inputs should have equivalent attributions. Any attribution method based on the gradients of a network will satisfy this axiom[2], meaning that IG, EG and gradients will all be implementation invariant.
- Sensitivity (sometimes called dummy) states that when a model does not depend on a feature at all, it receives zero importance. IG, EG and gradients all satisfy sensitivity because the gradient with respect to an irrelevant feature will be zero everywhere.
- Completeness states that the attributions should sum to the difference between the output of a function at the input to be explained and the output of that function at a baseline. Gradients do not satisfy completeness due to saturation at the inputs; elements such as ReLUs may cause gradients to be zero, making completeness impossible[2]. IG and EG both satisfy completeness due to the gradient theorem (fundamental theorem of calculus for line integrals)[2]. For EG, the function being integrated is the expectation of the model's output, so completeness means that the attributions sum to the difference between the model's output for the input and the model's output averaged over all possible baselines.
- Linearity states that for a model that is a linear combination of two submodels $f(x) = af_1(x) + bf_2(x)$, where a and b are arbitrary scalars, the attributions are a linear combination of the submodels' attributions $\phi(x) = a\phi_1(x) + b\phi_2(x)$. This will hold for IG, EG and gradients because gradients are linear.
- Symmetry preserving states that symmetric variables with identical values should achieve identical attributions. IG is symmetry preserving since it is a straight line path method, and EG will also be symmetry preserving, as a symmetric function of symmetric functions will itself be symmetrical[2].

Unlike previous attribution methods, EG is explicitly designed for natural batched training. This enables an order of magnitude increase in computational efficiency relative to previous approaches for training with attribution priors. We further improve performance by reducing the need for additional data reading. Specifically, for each input in a batch of inputs, we need k additional inputs to calculate EG attributions for that input batch. As long as k is smaller than the batch size, we can avoid any additional data reading by re-using the same batch of input data as a reference batch, as in ref. [45]. We accomplish this by shifting the batch of input k times, such that each input in the batch uses k other inputs from the batch as its reference values.

Specific priors. Here we elaborate on the explicit form of the attribution priors we used in this paper. In general, minimizing the error of a model corresponds to maximizing the likelihood of the data under a generative model consisting of the learned model plus parametric noise. For example, minimizing mean squared error in a regression task corresponds to maximizing the likelihood of the data under the learned model, assuming Gaussian-distributed errors:

$$\mathrm{argmin}_\theta ||f_\theta(X) - y||_2^2 = \mathrm{argmax}_\theta \exp\left(-||f_\theta(X) - y||_2^2\right) = \theta_{\mathrm{MLE}},$$

where θ_{MLE} is the maximum-likelihood estimate (MLE) of θ under the model $Y = f_\theta(X) + \mathcal{N}(0, \sigma)$.

An additive regularization term is equivalent to adding a multiplicative (independent) prior to yield a maximum a posteriori (MAP) estimate:

$$\begin{aligned} &\mathrm{argmin}_\theta ||f_\theta(X) - y||_2^2 + \lambda||\theta||_2^2 \\ &= \mathrm{argmax}_\theta \exp\left(-||f_\theta(X) - y||_2^2\right) \exp\left(-\lambda||\theta||_2^2\right) = \theta_{\mathrm{MAP}}, \end{aligned}$$

Here, adding an L2 penalty is equivalent to MAP for $Y = f_\theta(X) + \mathcal{N}(0, \sigma)$ with a $\mathcal{N}(0, \frac{1}{\lambda})$ prior. We next discuss the functional form of the attribution priors enforced by our penalties.

Pixel attribution prior. Our pixel attribution prior is based on the anisotropic total variation loss and is given as follows:

$$\Omega_{\mathrm{pixel}}(\Phi(\theta, X)) = \sum_\ell \sum_{i,j} |\phi_{i+1,j}^\ell - \phi_{i,j}^\ell| + |\phi_{i,j+1}^\ell - \phi_{i,j}^\ell|,$$

where $\phi_{i,j}^\ell$ is the attribution for the i, jth pixel in the ℓ-th training image. Research shows[46] that this penalty is equivalent to placing zero-mean, i.i.d., Laplace-distributed priors on the differences between adjacent pixel values, that is, $\phi_{i+1,j}^\ell - \phi_{i,j}^\ell \approx$ Laplace $(0, \lambda^{-1})$ and $\phi_{i,j+1}^\ell - \phi_{i,j}^\ell \approx$ Laplace $(0, \lambda^{-1})$. Reference [46] does not call our penalty 'total variation', but it is in fact the widely used anisotropic version of total variation and is directly implemented in Tensorflow[47–49].

Graph attribution prior. For our graph attribution prior, we used a protein–protein or gene–gene interaction network and represented these networks as a weighted, undirected graph. Formally, assume we have a weighted adjacency matrix $W \in \mathbb{R}_+^{p\times p}$ for an undirected graph, where the entries encode our prior belief about the pairwise similarity of the importances between two features. For a biological network, $W_{i,j}$ encodes either the probability or strength of interaction between the ith and jth genes (or proteins). We encouraged similarity along graph edges by penalizing the squared Euclidean distance between each pair of feature attributions in proportion to how similar we believe them to be. Using the graph Laplacian ($L_G = D - W$), where D is the diagonal degree matrix of the weighted graph, this becomes:

$$\Omega_{\mathrm{graph}}(\Phi(\theta, X)) = \sum_{i,j} W_{i,j}(\bar{\phi}_i - \bar{\phi}_j)^2 = \bar{\phi}^T L_G \bar{\phi}.$$

In this case, we choose to penalize global rather than local feature attributions. We define $\bar{\phi}_i$ to be the importance of feature i across all samples in our dataset, where this global attribution is calculated as the average magnitude of the feature attribution across all samples: $\bar{\phi}_i = \frac{1}{n}\sum_{\ell=1}^n |\phi_i^\ell|$. Just as the image penalty is equivalent to placing a Laplace prior on adjacent pixels in a regular graph, the graph penalty Ω_{graph} is equivalent to placing a Gaussian prior on adjacent features in an arbitrary graph with Laplacian L_G (ref. [46]).

Sparse attribution prior. Our sparsity prior uses the Gini coefficient G as a penalty, which is written:

$$\Omega_{\mathrm{sparse}}(\Phi(\theta, X)) = -\frac{\sum_{i=1}^p \sum_{j=1}^p |\bar{\phi}_i - \bar{\phi}_j|}{n\sum_{i=1}^p \bar{\phi}_i} = -2G(\bar{\phi}),$$

By taking exponentials of this function, we find that minimizing the sparsity regularizer is equivalent to maximizing likelihood under a prior proportional to the following:

$$\prod_{i=1}^p \prod_{j=1}^p \exp\left(\frac{1}{\sum_{i=1}^p \bar{\phi}_i} |\bar{\phi}_i - \bar{\phi}_j|\right),$$

To our knowledge, this prior does not directly correspond to a named distribution. However, we observe that its maximum value occurs when one $\bar{\phi}_i$ is 1 and all others are 0, and that its minimum occurs when all $\bar{\phi}_i$ are equal. This is similar to the total variation penalty Ω_{image}, but it is normalized and has a flipped sign to encourage differences. The corresponding attribution prior is maximized when global attributions are zero for all but one feature and minimized when attributions are uniform across features.

Image model experimental settings. We trained a VGG16 model from scratch modified for the CIFAR-10 dataset, containing 60,000 coloured 32×32-pixel images divided into 10 categories, as in ref. [50]. To train this network, we used stochastic gradient descent with an initial learning rate of 0.1 and an exponential decay of 0.5 applied every 20 epochs. Additionally, we used a momentum level of 0.9. For augmentation, we shifted each image horizontally and vertically by a pixel shift uniformly drawn from the range [−3, 3], and we randomly rotated each image by an angle uniformly drawn from the range [−15, 15]. We used a batch size of 128.

Before training, we normalized the training dataset to have zero mean and unit variance, and standardized the test set with the mean and variance of the training set. We used $k=1$ background reference samples for our attribution prior while training. When training with attributions over images, we first normalized the per-pixel attribution maps by dividing by the standard deviation before computing the total variation; otherwise, the total variation can be made arbitrarily small without changing model predictions by scaling down the pixel attributions close to zero. See Supplementary Section F for more details.

We repeated the same experiment as above on MNIST, which contains 60,000 black-and-white 28×28-pixel images of handwritten digits. We trained a convolutional neural network with two convolutional layers and a single hidden layer. The convolutional layers each had 5×5 filters, a stride length of 1, and 32 and 64 filters total. Each convolutional layer was followed by a max pooling layer of size 2 with stride length 2. The hidden layer had 1,024 units and a dropout rate of 0.5 during training[51]. Dropout was turned off when calculating the gradients with respect to the attributions. We trained with the Adam optimizer with the default parameters (learning rate $\alpha=0.001$, gradient average decay rate $\beta_1=0.9$, squared gradient average decay rate $\beta_2=0.999$, and numerical stability constant $\epsilon=10^{-8}$)[52]. We trained with an initial learning rate of 0.0001, with an exponential decay of 0.95 for every epoch, for a total of 60 epochs. For all models, we trained with a batch size of 50 images and used $k=1$ background reference sample per attribution while training. See Supplementary Section G for more details.

Biological experiments. *Significance testing of results.* To test the difference in R^2 attained by each method, we used a t-test for the means of two independent samples of scores (as implemented in SciPy)[53]. This is a two-sided test and can be applied to R^2 as R^2 is a linear transformation of mean squared error, which satisfies normality assumptions by the central limit theorem. When we compare the R^2 attained from ten independent retrainings of the neural network to the R^2 attained from ten independent retrainings of the attribution prior model, we find that predictive performance is significantly higher for the model with the graph attribution prior (t statistic $=3.59$, $P=2.06\times10^{-3}$).

To ensure that the increased performance in the attribution prior model was due to real biological information, we replaced the gene-interaction graph with a randomized graph (symmetric matrix with identical number of non-zero entries to the real graph, but entries placed in random positions). We then compared the R^2 attained from ten independent retrainings of a neural network with no graph attribution prior to ten independent retrainings of an neural network regularized with the random graph and found that test error was not significantly different between these two models (t statistic $=1.25$, $P=0.23$). We also compared to graph convolutional neural networks, and found that our network with a graph attribution prior outperformed the graph convolutional neural network (t statistic $=3.30$, $P=4.0\times10^{-3}$). Finally, we compared to an L2 penalty applied uniformly across all attributions, and found that this attribution prior did not significantly increase performance from baseline (t statistic $=1.7$, $P=0.12$, see Supplementary Fig. 15).

Train/validation/test-set allocation. To increase the number of samples in our dataset, we used as a feature the identity of the drug being tested, rather than one of a number of possible output tasks in a multi-task prediction. This follows from previous literature on training neural networks to predict drug response[54]. This yielded 30,816 samples (covering 218 patients and 145 anticancer drugs). Defining a sample as a drug and a patient, however, meant we had to choose carefully how to stratify samples into our train, validation and test sets. While it is perfectly legitimate in general to randomly stratify samples into these sets, we wanted to specifically focus on how well our model could learn trends from gene expression data that would generalize to new patients. Therefore, we stratified samples at a patient level rather than at the level of individual samples (for example, no samples from any patient in the test set ever appeared in the training set). We split 20% of the total patients into a test set (6,155 samples) and then split 20% of the training data into a validation set for hyperparameter selection (4,709 samples).

Model class implementations and hyperparameters tested. LASSO. We used the scikit-learn implementation of the LASSO[55,56]. We tested a range of α parameters from 10^{-9} to 1, and we found that the optimal value for α was 10^{-2} by mean squared error on the validation set.

Graph LASSO. For our graph LASSO, we used the Adam optimizer in TensorFlow[47], with a learning rate of 10^{-5} to optimize the following loss function:

$$\mathcal{L}(w;X,y)=\| Xw-y\|_2^2+\lambda'\ \| w\|_1+\nu' w^T L_G w, \quad (1)$$

where $w\in\mathbb{R}^d$ is the weights vector of our linear model and L_G is the graph Laplacian of our HumanBase network[30]. In particular, we downloaded the 'Top Edges' version of the haematopoietic stem cell network, which was thresholded to only have non-zero values for pairwise interactions that had a posterior probability greater than 0.1. We used the value of λ' selected as optimal in the regular LASSO model (10^{-2}, which corresponds to the α parameter in scikit-learn) and then tuned over ν' values ranging from 10^{-3} to 100. We found that a value of 10 was optimal according to MSE on the validation set.

Neural networks. We tested a variety of hyperparameter settings and network architectures via validation set performance to choose our best neural networks, including the following feed-forward network architectures (where each element in a list denotes the size of a hidden layer): [512, 256], [256, 128], [256, 256] and [1,000, 100]. We tested a range of L1 penalties on all of the weights of the network, from 10^{-7} to 10^{-2}. All models attempted to optimize a least squares loss using the Adam optimizer, with learning rates again selected by hyperparameter tuning ranging from 10^{-5} to 10^{-3}. Finally, we implemented an early stopping parameter of 20 rounds to select the number of epochs of training (training was stopped after no improvement on validation error for 20 epochs, and the number of epochs was chosen based on optimal validation set error). We found that the optimal architecture (chosen by lowest validation set error) had two hidden layers of size 512 and 256, an L1 penalty on the weights of 10^{-3} and a learning rate of 10^{-5}. We additionally found that 120 was the optimal number of training epochs.

Attribution prior neural networks. We next applied our attribution prior to the neural networks. First, we tuned networks to the optimal conditions described above. We then added extra epochs of fine-tuning where we ran an alternating minimization of the following objectives:

$$\mathcal{L}(\theta;X,y)=\| f_\theta(X)-y\|_2^2+\lambda\ \| \theta\|_1 \quad (2)$$

$$\mathcal{L}(\theta;X)=\Omega_{\text{graph}}(\Phi(\theta,X))=\nu\bar{\phi}^T L_G\bar{\phi} \quad (3)$$

Following ref. [9], we selected ν to be 100 so that the Ω_{graph} term would initially be equal in magnitude to the least squares and L1 loss terms. We found that five extra epochs of tuning were optimal by validation set error. We drew $k=10$ background samples for our attributions. To test our attribution prior using gradients as the feature attribution method (rather than expected gradients), we followed the exact same procedure, only we replaced $\bar{\phi}$ with the average magnitude of the gradients rather than the EG.

Graph convolutional networks. We followed the implementation of graph convolution described in ref. [31]. The architectures were searched as follows: in every network, we first had a single graph convolutional layer (we were limited to one graph convolution layer due to memory constraints on each Nvidia GTX 1080 Ti GPU that we used), followed by two fully connected layers of sizes (512, 256), (512, 128) or (256, 128). We tuned over a wide range of hyperparameters, including L2 penalties on the weights ranging from 10^{-5} to 10^{-2}, L1 penalties on the weights ranging from 10^{-5} to 10^{-2}, learning rates of 10^{-5} to 10^{-3} and dropout rates ranging from 0.2 to 0.8. We found the optimal hyperparameters based on validation set error were two hidden layers of size 512 and size 256, an L2 penalty on the weights of 10^{-5}, a learning rate of 10^{-5} and a dropout rate of 0.6. We again used an early stopping parameter and found that 47 epochs was the optimal number.

Sparsity experiments. *Data description and processing.* Our sparsity experiments used data from the National Health and Nutrition Examination I Survey (NHANES I)[42] and contained 35 variables (expanded to 118 features by one-hot encoding of categorical variables) gathered from 13,000 patients. The measurements included demographic information such as age, sex and BMI as well as physiological measurements such as blood, urine and vital sign measurements. The prediction task was a binary classification of whether the patient was still alive (1) or not (0) ten years after data were gathered.

Data were mean-imputed and standardized so that each feature had zero mean and unit variance. For each of the 200 experimental replicates, 100 train and 100 validation points were sampled uniformly at random; all other points were allocated to the test set.

Model. We trained a range of neural networks to predict survival in the NHANES data. The architecture, nonlinearities and training rounds were all held constant at values that performed well on an unregularized network, and the type and degree of regularization were varied. All models used ReLU activations and a single output with binary cross-entropy loss; in addition, all models ran for 100 epochs with a stochastic gradient descent optimizer with learning rate 0.001 on the size-100 training data. The entire 100-sample training set fit in one batch. Because the training set was so small, all of its 100 samples were used for EG attributions during training and evaluation, yielding $k=100$. Each model was trained on a single GPU on a desktop workstation with 4 Nvidia 1080 Ti GPUs.

Architecture. We considered a range of architectures, including single-hidden-layer 32-node, 128-node and 512-node networks, two-layer [128, 32]-node and [512, 128]-node networks, and a three-layer [512, 128, 32]-node network; we fixed the [512, 128, 32] architecture for future experiments.

Regularizers. We tested a large array of regularizers in addition to those considered in the main text. For details, see Supplementary Section J.1.

Hyperparameter tuning. We selected the hyperparameters for our models based on validation performance. We searched all L1, L2, SGL and attribution prior penalties with 121 points sampled on a log scale over $[10^{-7}, 10^5]$ (Supplementary Fig. 18).

Other penalties, not displayed in the main text experiments, are discussed in Supplementary Section J.2.

Main text methods. **Performance and sparsity bar plots**. The performance bar graph (Fig. 5a) was generated by plotting mean test ROC AUC of the best model of each type (chosen by validation ROC AUC) averaged over each of the 200 subsampled datasets, with confidence intervals given by 2 times the standard error over the 200 replicates. The sparsity bar graph (Fig. 5c) was constructed using the same process, but with Gini coefficients rather than ROC AUCs.

Feature importance distribution plot. The distribution of feature importances was plotted in the main text as a Lorenz curve (Fig. 5, bottom right): for each model, the features were sorted by global attribution value $\bar{\phi}_i$, and the cumulative normalized value of the lowest q features was plotted, from 0 at $q=0$ to 1 at $q=p$. A lower area under the curve indicates more features had relatively small attribution values, indicating that the model was sparser. Because 200 replicates were run on small subsampled datasets, the Lorenz curve for each model was plotted using the averaged mean absolute sorted feature importances over all replicates. Thus, for a given model type, the $q=1$ point represented the mean absolute feature importance of the least important feature averaged over each replicate, $q=2$ added the mean importance for the second least important feature averaged over each replicate, and so on.

Performance versus sparsity plot. Validation ROC AUC and model sparsity were calculated for each of the 121 regularization strengths and averaged over each of the 200 replicates. These were plotted on a scatterplot to show the possible range of model sparsities and ROC AUC performances (Fig. 5, top right) as well as the trade-off between sparsity and performance.

Statistical significance. Statistical significance of the sparse attribution prior performance was assessed by comparing the test ROC AUCs of the sparse attribution prior models on each of the 200 subsampled datasets to those of the other models (L1 gradients, L1 weights, SGL and unregularized). Significance was assessed by two-sided paired-samples t-test, paired by subsampled dataset. The same process was used to calculate the significance of model sparsity as measured by the Gini coefficient. Detailed tables of the resulting P values and test statistics t are shown in Supplementary Section J.3.

Data availability

The data for all experiments and figures in the paper are publicly available. A downloadable version of the dataset used for the sparsity experiment, as well as links to download the datasets used in the image and graph prior experiments, is available at https://github.com/suinleelab/attributionpriors. Data for the benchmarks were published as part of ref. [57] and can be accessed at https://github.com/suinleelab/treeexplainer-study/tree/master/benchmark.

Code availability

Implementations of attribution priors for Tensorflow and PyTorch are available at https://github.com/suinleelab/attributionpriors. This repository also contains code reproducing main results from the paper. The specific version of code used in this paper is archived at ref. [58].

Received: 23 August 2020; Accepted: 12 April 2021;
Published online: 31 May 2021

References

1. Lundberg, S. M. & Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems* Vol. 30, 4765–4774 (NeurIPS, 2017).
2. Sundararajan, M., Taly, A. & Yan, Q. Axiomatic attribution for deep networks. In *Proc. 34th International Conference on Machine Learning* Vol. 70, 3319–3328 (Journal of Machine Learning Research, 2017).
3. Štrumbelj, E. & Kononenko, I. Explaining prediction models and individual predictions with feature contributions. *Knowl. Inf. Syst.* **41**, 647–665 (2014).
4. Datta, A., Sen, S. & Zick, Y. Algorithmic transparency via quantitative input influence: theory and experiments with learning systems. In *2016 IEEE Symposium on Security and Privacy (SP)* 598–617 (IEEE, 2016).
5. Lundberg, S. M. et al. From local explanations to global understanding with explainable AI for trees. *Nat. Mach. Intell.* **2**, 56–67 (2020).
6. Lundberg, S. M. et al. Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. *Nat. Biomed. Eng.* **2**, 749–760 (2018).
7. Sayres, R. et al. Using a deep learning algorithm and integrated gradients explanation to assist grading for diabetic retinopathy. *Ophthalmology* **126**, 552–564 (2019).
8. Zech, J. R. et al. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS Med.* **15**, e1002683 (2018).
9. Ross, A. S., Hughes, M. C. & Doshi-Velez, F. Right for the right reasons: training differentiable models by constraining their explanations. In *Proc. 26th International Joint Conference on Artificial Intelligence* 2662–2670 (IJCAI, 2017).
10. Schramowski, P. et al. Making deep neural networks right for the right scientific reasons by interacting with their explanations. *Nat. Mach. Intell.* **2**, 476–486 (2020).
11. Ilyas, A. et al. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems* Vol. 32 (NeurIPS, 2019).
12. Liu, F. & Avci, B. Incorporating priors with feature attribution on text classification. In *Proc. of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)* 6274–6283 (2019).
13. Chen, J., Wu, X., Rastogi, V., Liang, Y. & Jha, S. Robust attribution regularization. In *Advances in Neural Information Processing Systems* Vol. 32 (NeurIPS, 2019).
14. Rieger, L., Singh, C., Murdoch, W. J. & Yu, B. Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. In *Proc. 37th International Conference on Machine Learning* (eds. Daumé III, H. & Singh, A.) 8116–8126 (ICML, 2020).
15. LeCun, Y., Cortes, C. & Burges, C. *MNIST Handwritten Digit Database* (AT&T Labs) http://yann.lecun.com/exdb/mnist (2010)
16. Yu, F., Xu, Z., Wang, Y., Liu, C. & Chen, X. Towards robust training of neural networks by regularizing adversarial gradients. Preprint at https://arxiv.org/abs/1805.09370 (2018).
17. Jakubovitz, D. & Giryes, R. Improving DNN robustness to adversarial attacks using Jacobian regularization. In *Proc. European Conference on Computer Vision (ECCV)* (eds. Ferrari, V., Hebert, M., Sminchisescu, C. & Weiss, Y.) 514–529 (ECCV, 2018).
18. Roth, K., Lucchi, A., Nowozin, S. & Hofmann, T. Adversarially robust training through structured gradient regularization. Preprint at https://arxiv.org/abs/1805.08736 (2018).
19. Selvaraju, R. R. et al. Grad-CAM: visual explanations from deep networks via gradient-based localization. In *Proc. IEEE International Conference on Computer Vision* 618–626 (IEEE, 2017).
20. Ross, A. S. & Doshi-Velez, F. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI Conference on Artificial Intelligence* Vol. 32 1 (AAAI, 2018).
21. Smilkov, D., Thorat, N., Kim, B., Viégas, F. & Wattenberg, M. Smoothgrad: removing noise by adding noise. Preprint at https://arxiv.org/abs/1706.03825 (2017).
22. Fong, R. C. & Vedaldi, A. Interpretable explanations of black boxes by meaningful perturbation. In *Proc. IEEE International Conference on Computer Vision* 3429–3437 (IEEE, 2017).
23. Krizhevsky, A. et al. *Learning Multiple Layers of Features from Tiny Images* Technical Report (Citeseer, 2009).
24. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations* (eds. Bengio, Y. & LeCun, Y.) (ICLR, 2015).
25. Recht, B., Roelofs, R., Schmidt, L. & Shankar, V. Do ImageNet classifiers generalize to ImageNet? *Proc. of the 36th International Conference on Machine Learning* Vol. 97, 5389–5400 (PMLR, 2019).
26. Tsipras, D., Santurkar, S., Engstrom, L., Turner, A. & Madry, A. Robustness may be at odds with accuracy. In *7th International Conference on Learning Representations* (ICLR, 2019).
27. Zhang, H. et al. Theoretically principled trade-off between robustness and accuracy. In *Proc. 36th International Conference on Machine Learning* Vol. 97, 7472–7482 (PMLR, 2019).
28. Cheng, W., Zhang, X., Guo, Z., Shi, Y. & Wang, W. Graph-regularized dual Lasso for robust eQTL mapping. *Bioinformatics* **30**, i139–i148 (2014).
29. Tyner, J. W. et al. Functional genomic landscape of acute myeloid leukaemia. *Nature* **562**, 526–531 (2018).
30. Greene, C. S. et al. Understanding multicellular function and disease with human tissue-specific networks. *Nat. Genet.* **47**, 569–576 (2015).
31. Kipf, T. N. & Welling, M. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations* (ICLR, 2017).
32. Subramanian, A. et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl Acad. Sci. USA* **102**, 15545–15550 (2005).
33. Benjamini, Y. & Hochberg, Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. B* **57**, 289–300 (1995).
34. Liu, J. et al. Meis1 is critical to the maintenance of human acute myeloid leukemia cells independent of MLL rearrangements. *Ann. Hematol.* **96**, 567–574 (2017).
35. Valk, P. J. M. et al. Prognostically useful gene-expression profiles in acute myeloid leukemia. *N. Engl. J. Med.* **350**, 1617–1628 (2004).
36. Feng, J. & Simon, N. Sparse-input neural networks for high-dimensional nonparametric regression and classification. Preprint at https://arxiv.org/abs/1711.07592 (2017).
37. Scardapane, S., Comminiello, D., Hussain, A. & Uncini, A. Group sparse regularization for deep neural networks. *Neurocomputing* **241**, 81–89 (2017).

38. Ross, A., Lage, I. & Doshi-Velez, F. The neural lasso: local linear sparsity for interpretable explanations. In *Workshop on Transparent and Interpretable Machine Learning in Safety Critical Environments, 31st Conference on Neural Information Processing Systems* (2017).
39. Shrikumar, A., Greenside, P. & Kundaje, A. Learning important features through propagating activation differences. In *Pro. 34th International Conference on Machine Learning* Vol. 70, 3145–3153 (Journal of Machine Learning Research, 2017).
40. Hurley, N. & Rickard, S. Comparing measures of sparsity. *IEEE Trans. Inf. Theory* **55**, 4723–4741 (2009).
41. Zonoobi, D., Kassim, A. A. & Venkatesh, Y. V. Gini index as sparsity measure for signal reconstruction from compressive samples. *IEEE J. Sel. Top. Signal Process.* **5**, 927–932 (2011).
42. Miller, H. W. *Plan and Operation of the Health and Nutrition Examination Survey, United States, 1971–1973* DHEW publication no. 79-55071 (PHS) (Department of Health, Education, and Welfare, 1973).
43. Binder, A., Montavon, G., Lapuschkin, S., Müller, K.-R. & Samek, W. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks* (eds. Villa, A.E.P., Masulli, P. & Rivero, A.J.P.) 63–71 (Springer, 2016).
44. Friedman, E. J. Paths and consistency in additive cost sharing. *Int. J. Game Theory* **32**, 501–518 (2004).
45. Zhang, H., Cisse, M., Dauphin, Y. N. & Lopez-Paz, D. mixup: beyond empirical risk minimization. In *6th International Conference on Learning Representations* (ICLR, 2018).
46. Bardsley, J. M. Laplace-distributed increments, the Laplace prior, and edge-preserving regularization. *J. Inverse Ill Posed Probl.* **20**, 271–285 (2012).
47. Abadi, M. et al. Tensorflow: a system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)* 265–283 (2016).
48. Lou, Y., Zeng, T., Osher, S. & Xin, J. A weighted difference of anisotropic and isotropic total variation model for image processing. *SIAM J. Imaging Sci.* **8**, 1798–1823 (2015).
49. Shi, Y. & Chang, Q. Efficient algorithm for isotropic and anisotropic total variation deblurring and denoising. *J. Appl. Math.* **2013**, 797239 (2013).
50. Liu, S. & Deng, W. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)* 730–734 (IEEE, 2015).
51. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
52. Kingma, D. P. & Ba, J. In *3rd International Conference on Learning Representations* (eds. Bengio, Y. & LeCun, Y.) (ICLR, 2015).
53. Virtanen, P. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Methods* **17**, 261–272 (2020).
54. Preuer, K. et al. DeepSynergy: predicting anti-cancer drug synergy with deep learning. *Bioinformatics* **34**, 1538–1546 (2018).
55. Tibshirani, R. Regression shrinkage and selection via the Lasso. *J. R. Stat. Soc. B* **58**, 267–288 (1996).
56. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
57. Lundberg, S. M. et al. Explainable AI for trees:from local explanations to global understanding. Preprint at https://arxiv.org/abs/1905.04610 (2019).
58. Sturmfels, P., Erion, G. & Janizek, J. D. suinleelab/attributionpriors: *Nature Machine Intelligence* code. *Zenodo* https://doi.org/10.5281/zenodo.4608599 (2021).

Acknowledgements

The results published here are partially based on data generated by the Cancer Target Discovery and Development (CTD2) Network (https://ocg.cancer.gov/programs/ctd2/data-portal) established by the National Cancer Institute's Office of Cancer Genomics. The authors received funding from the National Science Foundation (DBI-1759487 (S.-I.L.), DBI-1552309 (J.D.J., G.E., S.-I.L.), DGE-1256082 (S.M.L.)); American Cancer Society (RSG-14-257-01-TBG (J.D.J., P.S., S.-I.L)); and National Institutes of Health (R01AG061132 (J.D.J, P.S., S.-I.L), R35GM128638 (G.E., S.-I.L), F30HL151074-01 (G.E., S.-I.L), 5T32GM007266-46 (J.D.J, G.E.)).

Author contributions

G.E., J.D.J., P.S. and S.M.L. conceived the study. G.E., J.D.J. and P.S. designed algorithms and experiments. P.S. and J.D.J. implemented core libraries for the research. G.E., J.D.J. and P.S. wrote code for and ran the experiments, plotted figures and contributed to the writing. S.M.L. contributed to the writing. S.-I.L. supervised research and method development, and contributed to the writing.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42256-021-00343-w.

Correspondence and requests for materials should be addressed to S.-I.L.

Peer review information *Nature Machine Intelligence* thanks Ronny Luss, Andrew Ross and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.