Appendix

A ACKNOWLEDGEMENTS.

We thank Ilaria Liccardi, Danny Weitzner, Taylor Reynolds, and Anonymous reviewers for feedback on this work. We are grateful to the MIT Quest for Intelligence initiative for providing cloud computing credits for this work. Julius Adebayo is supported by the Open Philanthropy Fellowship.

B EXTENDED RELATED WORK.

Overview of Recent work The recent work of Adebayo et al. (2020) presents debugging tests for assessing feature attribution methods. The spurious correlation setting that we consider here fits under their framework. However, they only consider feature attribution methods. Here we extend this analysis to concept and training point ranking methods. Critically, Adebayo et al. (2020) show that feature attributions are able to identify spurious training signals. We make a similar finding in this work; however, we further demonstrate this finding for concept and training point ranking methods. Our work takes important departures from theirs: 1) we explain the source of this phenomenon, and 2) we demonstrate that naive application of these methods might be unable to detect spurious correlation in practice. Adebayo et al. (2020) assume the spurious correlation training bug is known, a priori; however, here we demonstrate that the more challenging task is identifying the spurious signal in the first place.

More recently, Han et al. (2020) demonstrate that training point ranking via influence functions is able to identify the dependence of an NLP model on dataset artifacts. In addition, they show correspondence between the insights observed with the input-gradient feature attribution and the training point ranking. Along similar lines, Guo et al. (2020) present fast approximations for computing the training point ranking for a test point. In addition, they show how to identify and correct model errors in a natural language task. Similar to the distinctions that we note with the work by Adebayo et al. (2020) above, here, they also assume that the spurious signal being identified is known a priori.

Post hoc explanations, more generally, have been shown to be able to identify a model's reliance on spurious training signals (Ribeiro et al., 2016; Meng et al., 2018; Lapuschkin et al., 2019; DeGrave et al., 2020; Ross et al., 2017). Recent work by Rieger et al. (2020) showed that regularizing model attributions during training can help lead to models that avoid spurious correlation and enable improved debugging by experts. Similarly, Erion et al. (2019) show that regularizing the expected gradient attribution during training confers similar benefits. Koh and Liang (2017) used influence functions to identify domain shift. Kim et al. (2018) also perform a user study to understand if attribution methods can be used for catch spurious correlation.

However, similar methods have also been shown to struggle in the hands of end-users for diagnosing model errors (Alqaraawi et al., 2020; Adebayo et al., 2020). This contradiction reflects the challenge that we explore in this work. Often, post hoc explanations have been shown to be effective for identifying spurious signals that were suspected or known a priori; however, these methods seem to struggle when confronted with the task of identifying an unexpected spurious signal.

Increasingly, insights into why overparametrized DNNs rely on spurious training set signals is starting to be theoretically and empirically analyzed (Sagawa et al., 2019; 2020; Khani and Liang, 2020; Nagarajan et al., 2020), yet it is still unclear how to reliably detect that a model is relying on such signals prior to model deployment.

Assessing whether a post hoc explanation approach is faithful to the underlying model being explained has been addressed in recent works, yet this challenge remains elusive (Hooker et al., 2019; Tomsett et al., 2020). Generally, the class of approaches that modify backpropagation with positive aggregation have been shown to be invariant to the higher layer parameters of a DNN (Mahendran and Vedaldi, 2016; Nie et al., 2018; Adebayo et al., 2018; Sixt et al., 2019). In an intriguing demonstration, Srinivas and Fleuret (2021) show that the input-gradient, a key feature attribution primitive, might not capture

discriminative signals about input sensitivity. Instead they show that input-gradient likely captures the ability of the model to be able to generate class-conditional inputs.

User studies are typically the classic approach for evaluating the effective of an explanation (Doshi-Velez and Kim, 2017). Poursabzi-Sangdeh et al. (2018) manipulate the features of a predictive model trained to predict housing prices to assess how well end-users can identify model mistakes. Their results indicate that users found it challenging to debug these linear models with the model coefficients. Recent work by Chu et al. (2020) and Shen and Huang (2020) has shown similar results in the DNN setting as well. Alqaraawi et al. (2020) find that the LRP explanation method improves participant understanding of model behavior for an image classification task, but provides limited utility to end-users when predicting the model's output on new inputs.

Post hoc explanations have been shown to be fragile and very easily manipulated Ghorbani et al. (2019a); Heo et al. (2019); Dombrowski et al. (2019); Anders et al. (2020); Slack et al. (2020); Lakkaraju and Bastani (2020). Our work tackles a difference concern: whether they are suitable for detecting unexpected spurious training set signals.

Yeh et al. (2020) and Ghorbani et al. (2019b) both present approaches to automatically discover concepts and quantify a model's dependence on these concepts. These approaches are promising directions for addressing the challenge we identify in this work. Ghorbani et al. (2019b)'s approach, ACE, segments input images and clusters to discover inherent clusters. We present some analysis on this approach in the appendix. Critically, this approach would identify spurious signals that the underlying segmentation algorithm can discover such the image tag, but not the blur or other visually imperceptible features. Koh et al. (2020) and Chen et al. (2020) present approaches that learn DNN models whose features inherently map onto concepts of interest. In this work, we assume the model is given, focusing on post hoc explanations for models that are not inherently interpretable. Recent work at the intersection of causal inference and explanations might also open up avenues to help reveal unexpected confounding, some of which could be unknown spurious signals. Along this line, (Bahadori and Heckerman, 2021) present an instrumental variable approach for debiasing concept based explanations that might be confounded. Kazhdan et al. (2020) present CME, an approach identify the important concepts that can help improve a model's performance.

We rely on Guo et al. (2020) for fast approximations for computing the training point ranking for a test point. In addition, they show how to identify and correct model errors in a natural language task.However, Basu et al. (2020) show influence functions for DNNs are fragile and perhaps inaccurate for deeper networks.

Other recent work has cast doubt on the utility of trying to explain 'traditionally' trained deep network models. For example, Srinivas and Fleuret (2021) show that the input-gradient might not reflect the discriminative capabilities of a DNN, but instead encode for an implicit density model. More recently, Shah et al. (2021) show that the input and loss gradients of traditionally trained models *do not* indicate the importance features that a DNN model relies on for its output—a phenomenon they term feature inversion. Further they show that adversarially trained models do not exhibit feature inversion. Taken together these results might explain some of the counter intuitive findings that we observe even when the spurious signal is known since we only consider non-adversarially trained models in this work. Along a different direction, post hoc explanations have been shown to be fragile and very easily manipulated Ghorbani et al. (2019a); Heo et al. (2019); Dombrowski et al. (2019); Anders et al. (2020); Slack et al. (2020); Lakkaraju and Bastani (2020).

C Detailed Overview of Explanation Methods

In this section, we provide additional implementation details for the explanation methods that we consider in this work. To start with the model setup: let's say we are given input-output pairs, $\{x_i, y_i\}_i^n$, where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$; and a classifier's goal is to learn a function, $f_\theta : \mathcal{X} \rightarrow \mathcal{Y}$ that generalizes to new inputs via empirical risk minimization (ERM). In this work, we assume that f_θ is an over-parametrized deep neural network (DNN) trained on image data for classification (C classes).

We plan to release our larger codebase with the paper; however, in lieu of this we include a zip folder that includes representative implementations. For each of the explanation methods described here, we implement them from scratch and then compare to open-source libraries that also implement these methods. We found that our results for both settings is comparable.

Feature Attributions. An attribution functional, $E : \mathcal{F} \times \mathbb{R}^d \times \mathbb{R} \rightarrow \mathbb{R}^d$, maps the input, $x_i \in \mathbb{R}^d$, the model, f_θ, output, $f_k(x)$, to an attribution map, $M_{x_i} \in \mathbb{R}^d$. The class of feature attribution methods is large, so in this work we pick: Input Gradient, SmoothGrad, Integrated Gradients, and Guided Backprop. We choose these approaches since they were the top-ranked methods tested under the spurious correlation setting of Adebayo et al. (2020).

1. **The *Input-Gradient (Gradient)*** Simonyan et al. (2014); Baehrens et al. (2010) map, $|\nabla_{x_i} F_i(x_i)|$, is a key primitive upon which several other methods are based.
2. ***SmoothGrad* Smilkov et al. (2017)** corresponds to the average of noisy input gradients: $M_{\text{sg}}(x) = \frac{1}{N}\sum_{i=1}^{N} \nabla_{x_i} F_i(x_i + n_i)$ where n_i is sampled according to a random Gaussian noise. We considered 50 noisy inputs, selected the standard deviation of the noise to be $0.15 *$ input range. Here input range refers to the difference between the maximum and minimum value in the input.
3. ***Integrated Gradients* (Sundararajan et al. (2017))** sums input gradients along an interpolation path from the "baseline input", $\bar{x}$, to x_i: $M_{\text{IntGrad}}(x_i) = (x_i - \bar{x}) \times \int_0^1 \frac{\partial S(\bar{x}+\alpha(x_i-\bar{x}))}{\partial x_i} d\alpha$. For integrated gradients we set the baseline input to be a vector containing the minimum possible values across all input dimensions. This often corresponds an all-black image. The choice of a baseline for IntGrad is not without controversy; however, we follow this setup since it is one of the more widely used baselines for image data.
4. ***Guided Backpropagation (GBP)* Springenberg et al. (2014)** modifies the backpropagation process at ReLU units in DNNs. Let, $a = \max(0, b)$, then for a backward pass, $\frac{\partial l}{\partial s} = 1_{s>0} \frac{\partial l}{\partial b}$, where l is a function of s. For GBP, $\frac{\partial l}{\partial s} = 1_{s>0} 1_{\frac{\partial l}{\partial s}>0} \frac{\partial l}{\partial b}$.

Feature Attributions: Implementation. We implement all of these methods from scratch in the PyTorch framework and also compare our implementations to the output of the Captum (PyTorch).

Concept-Based Approaches. We now discuss additional implementation details of our concept based approach. We select the TCAV approach to quantify the sensitivity of a DNN model's class score to user provided inputs represent a particular class. Given hidden representations, h_l, from a particular layer of a DNN for for images belonging to concept class C. We can derive the sensitivity score as: $\nabla h_{l,k}(f_l(x)).\theta_c^l$. The previous expression indicates the sensitivity of the class score (logit) for class k to inputs indicating concept, C, given hidden representations from layer l from the DNN f. The concept vector, θ_c^l, typically corresponds to a the weights of a linear classifier trained to separate the images for a particular concept class from or images.

For completeness, we show in Figure 20 an overview of the clinical concepts that we consider in this work. These are the representative clinical attributes that a radiologist would inspect to ascertain the bone age of a particular input. These concepts are: DIP, PIP, MCP, Radius, Ulna, and Wrist.

Concept Implementation. To compute the TCAV score for each concept, we collect representations from all hidden 'layers' of the model and train linear models to obtains the concept vector for the corresponding attribute. We then compute the class sensitivity score for each concept attribute. We train the linear model 100 times and perform statistical significance testing in order to mitigate the case where a spurious concept is selected. For each concept class, we use 325 images that part of the training, validation, or test sets. These new set of images were annotated by a board certified radiology with the clinical bone age regions (MCP, PIP, DIP etc) that we chose.

Influence Functions for Training Point Ranking. The final kind of interpretation that we consider is training point ranking via influence functions. In the case of training point ranking via influence functions, we rank the training samples, in terms of 'influence', on the loss of a test example. Specifically, if we up-weighted a training point and retrained the model, then by how much would the loss on a given test example change? Koh and Liang (2017) analytically derive the analytically formulas for computing this quantity. Given a test point, x_t, the influence of a training point, x_i, on the test loss is: $I(x_t, x_i) = -\nabla_\theta \ell(x_t, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \ell(x_i, \hat{\theta})$, where H is the empirical Hessian of the loss.

Estimating the influence requires computing hessian-vector products, so it can be difficult to scale to model with large number of parameters, and recent work has shown that influence estimate for test

points for deep networks can be inaccurate due to non-convexity (Basu et al., 2020). Consequently, we estimate influence on a linear model student network trained to mimic the original DNN. We empirically verify that the predictions of the student network seem to mimic the original DNN.

Implementation Details. There are two other training point ranking methods that we also consider in this work (Yeh et al., 2018; Pruthi et al., 2020). For 150 inputs in the test set, we compare the Spearman rank correlation of the training point due to Influence Functions to these other two approaches. We obtain mean values of 0.88, and 0.76 respectively, which suggests high similarity amongst these approaches. Ultimately, we chose to present the main results in the draft for the training point ranking due to influence functions approach.

We rely on the fast influence heuristic of (Guo et al., 2020) to speed up the influence ranking computations. We were able to obtain a 5-10X speed in doing so. In addition, we trained a student multi-class logistic regression model to mimic the original model for each model we want to compute influence for. Here for all training points, we collect embeddings across all layers and pass these embedding through a random projection to obtain a 1000-dimensional approximation. We then train linear models to mimic the original deep network using these features. The correlation between the output of the student models and the original teacher models was found to be 0.87. Ultimately, we went with the fast influence implementation of (Guo et al., 2020).

C.1 Additional Details on Metrics

D How are the approaches tested used in practice?

Feature Attributions To understand or debug a model, a practitioner would have to inspect, one sample at a time, the attribution of a collection of inputs. In inspecting these attributions, the practitioner can then form an hypothesis about the behavior of the model on certain inputs. Along these lines, to detect a model's reliance on spurious signals with feature attributions, one of the following must occur: i) The practitioner should inspect attributions for inputs that contain the spurious signal and notice that the spurious signal constitute the key feature for a model's output decision or ii) The practitioner should inspect attributions for inputs that do not contain the spurious signal and notice 'an issue' with these attributions. In addition, the output decision to be explained has to also be the class for which the spurious signal encodes for. For example, in our experiments the hospital tag encodes for the pre-puberty class. These two settings are exactly what we measure with the SSIM metric in our analysis for feature attributions.

Concept Activation these classes of approaches measure the dependence of a particular class to a user provided concept. For example, one could measure the pre-puberty class' dependence on the hospital tag concept. As part of this approach, the hospital tag is user defined, and it is up to the practitioner to decide whether to test this concept. Other concepts include low frequency signals, patches of the image, color dimensions, or even any conceivable high-level concept the practitioner is interested in. For example, in the bone-age model, the concepts DIP, PIP, MCP, Wrist etc, are user defined clinical concepts that we chose and which are well specified for this setting.

For each concept of interest, the practitioner collects input examples that have this concept. For example, to test a model's dependence on a hospital tag, we collect images that all have hospital tags in them. Given this collection of examples, the TCAV approach can then be used to calculate a score, TCAV score, that corresponds to the sensitivity of a particular output class to the input concept. The key insight that makes the TCAV approach susceptible to the limitations we point out are 1) it is up to the practitioner to decide which concept to test, and 2) our findings suggests that unless the spurious signal is explicitly tested, the TCAV scores for other concepts do not indicate that a model is reliant on a spurious signal. We note that even though we show all the concept scores in a single bar chart, these concepts are actually tested independently.

Training Point Ranking These class of approaches ranks all training points by influence on the test loss of an input. Qualitatively, if a training point has a high influence on the test loss of an input, it means that if the model were re-trained without that training point, the test loss of that test point would change significantly. Intuitively, the most highly ranked training points for a given test input should also be points for which the model relies on semantically similar features. For example,

one would expect the most highly ranked training points for a pre-puberty test-input to be other pre-puberty inputs as well (assuming a model has learned the right semantic features)

To use this approach to identify spurious signals, one would have to inspect an input that contains the spurious signal and further notice that the top ranked training inputs also include the training signal. Specifically, to test that a model is relying on the hospital tag to identify pre-puberty inputs one would have to use a test-input that has a hospital tag and then notice that all the top ranked inputs also have hospital tags. The crux of our argument is that the choice and decision to use a test-input with an hospital tag is the critical choice underlying whether the training point ranking methods can be used to effectively detect a model's reliance on spurious signals. One measure of reliability for an influence function approach is the ICM metric that we compute in the paper.

Taken together, in the scenario where a practitioner is handed a model and asked to assess whether it relies on any spurious signal, detecting a model's reliance spurious signals requires that the practitioner know which specific signals to test ahead of time.

E Model Training & Dataset Processing

Bone Age Dataset We consider the high stakes task of predicting the bone age category from a radiograph to one of five classes based on age: Infancy/Toddler, Pre-Puberty, Early/MiD Puberty, Late Puberty, and Post Puberty. This task is one that is routinely performed by radiologists and as been previously studied with a variety of DNN. The dataset we use is derived from the Pediatric Bone Age Machine learning challenge conducted by the radiological society of North America in 2017 Halabi et al. (2019). The dataset consists of 12282 training, 1425 validation, and 200 test samples. We resize all images to (299 by 299) grayscale images for model training. We note here that the training, validation, and test set splits correspond to similar splits used for the competition, so we retain this split.

Knee Dataset We also consider the high stakes task of predicting the Kellgren and Lawrence (KL) grade of osteoarthritis based on Knee Xrays. The KL grade ranges from the integers 0 to 4; 5 classes, so we treat it as a classification task. The Knee Radiographs are obtained from the open source release by Chen et al. (2019) and consists of 5778 training samples, 826 validation samples, and 1656 test samples. Again here, we follow Chen et al. (2019) data splits. The images were resized to be 299 by 299 pixels.

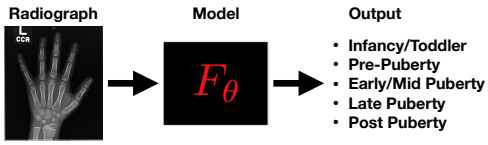

Figure 6: Illustration of the Bone Age Task.

Dog Dataset The third dataset that we use in this work is the dog breed classification dataset that is a combination of Stanford dogs dataset Khosla et al. (2011) and the Oxford Cats and Dogs datasets Parkhi et al. (2012b) following Adebayo et al. (2020). Similarly, we restrict our attention to 10 dog classes: Beagle, Boxer, Chihuahua, Newfoundland, Saint Bernard, Pugs, Pomeranian, Great Pyrenees, Yorkshire Terrier, Wheaten Terrier. Instead of the background spurious signal ofAdebayo et al. (2020), we focus instead on the three spurious signals tested in this work.

Models We consider two different kinds of models: i) a small vanilla DNN based on Raghu et al. (2019), and a Resnet-50 model. The small DNN consists of: conv-relu-batchnorm-maxpooling operation successively, and two fully connected layers at the end. All convolutional kernels have stride 1, and kernel size 5. We train this model with SGD with momentum (set to 0.9) and an initial learning rate of 0.01. We use a learning rate scheduler that decays the learning rate every 10 epochs by $\gamma = 0.1$.

Hyper-Parameter Tuning for Model Training We used the Ray Tune library for hyper-parameter tuning of all the models used in this work. For the ResNet-50, we tuned with Ray, but the best performing models retained the default parameter settings. In the case of the Small DNN model, we tune the batch size, and learning rate with the validation set.

Compute We perform all of our experiments on a VM with 60GB of RAM and a k80 GPU on Google Cloud.

A Note about random seeds and Model Runs. We train 5 models each (different random seeds) for each category and in the following tables the average performance metrics for these models. We found that the standard error of the mean for typically between $0.01 - 0.05$, so we omit these in the tables to improve readability.

F Additional Results: Feature Attributions

In a series of subsequent figures with this appendix. We show additional saliency visualizations for settings considered in the main text. Here show a single visualization for the bone age setting since we found that the saliency attributions for such setting were quite faint. Overall we obtain similar results between the knee and bone age datasets, so we opt to show the bone age visualizations here instead.

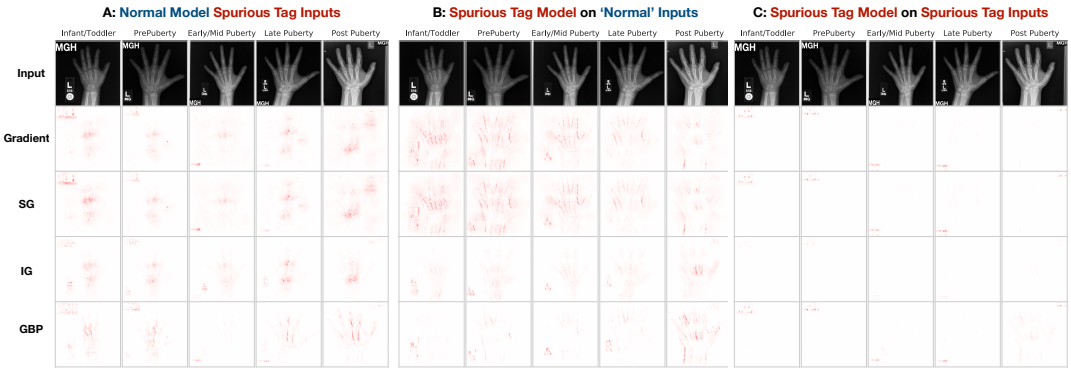

Figure 7: **Detecting Spurious *Tag*.** Here we show in A) Feature attributions for 5 different inputs across the four feature attribution methods with a normal model but with spurious Tag inputs; B) Feature attributions on the same 5 inputs as in (A), but **without** spurious Tag inputs with a model that has learned a spurious alignment between Pre-Puberty and Tag; C) Feature attributions on the same 5 inputs as in (A), but **with** the spurious Tag inputs with a model that has learned a spurious alignment between Pre-Puberty and Tag.

G Additional Results: Concepts Approaches

Normal Model compared to radiologist Concept Ranking For each of the 100 re-runs, we computed the rank correlation between the concept ranking for the normal model and rankings provided by a board certified radiologist.

Concepts for Dogs and Bone Age. We consider TCAV for the bone age and dog breeds task. For bone age, we choose as concepts the partitions of a hand, which correspond to the parts of the hand a radiologist would inspect to ascertain the age from a radiograph. We show, in Figure 20, a representation of these concepts. In the dog breeds classification task, consider concepts related floppiness of the ear, erectness of the ear, the dog head type, and color.

We manually collect data on the dog breed concept. Here we showed Amazon mechanical Turkers an image from each class (10 classes) and then asked them to indicate whether each specific dog class had floppy ears or not, erect ears or not, single colored or not etc. We normalize each attribute to be binary and apply each concept broadly across each class.

Discussion on ACE . The ACE concept approach segments an image and uses the image segments as concepts as part of a TCAV pipeline. In an experimental analysis, we segment the spurious images

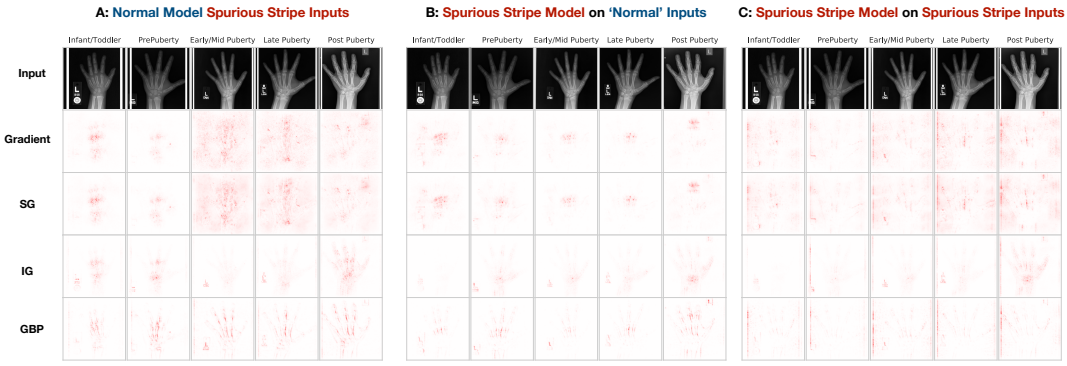

Figure 8: **Detecting Spurious *Stripe*.** Here we show in A) Feature attributions for 5 different inputs across the four feature attribution methods with a normal model but with spurious Stripes inputs; B) Feature attributions on the same 5 inputs as in (A), but **without** the spurious Stripe with a model that has learned a spurious alignment between Pre-Puberty and Stripe; C) Feature attributions on the same 5 inputs as in (A), but **with** the spurious Stripe with a model that has learned a spurious alignment between Pre-Puberty and Stripe.

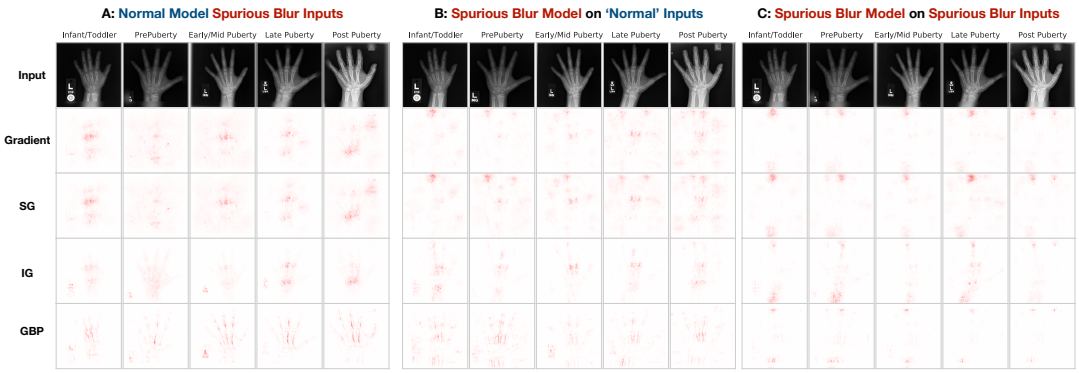

Figure 9: **Detecting Spurious *Blur*.** Here we show in A) Feature attributions for 5 different inputs across the four feature attribution methods with a normal model but with spurious Blur inputs; B) Feature attributions on the same 5 inputs as in (A), but **without** the spurious Blur with a model that has learned a spurious alignment between Pre-Puberty and Blur; C) Feature attributions on the same 5 inputs as in (A), but **with** the spurious blur with a model that has learned a spurious alignment between Pre-Puberty and Blur.

using the various segmentation algorithms in algorithm to identify how often the spurious signal is selected as a distinct segment. We found segmentation to be ineffective in the stripe and blur settings. For the spurious tag, we found segmentation to be effective in only 25 percent of the examples tested, which suggests that the underlying segmentation algorithm as part of ACE might be ineffective for detecting hidden spurious signals.

H Additional Results: Influence Functions

In Figure 22, we show the influence metric that we computed in the main text for the small DNN model for the Knee Arthritis prediction. In this task, the spurious signal is always aligned with the Grade 2 class. Here again, we see that in the spurious settings each metric actually improves.

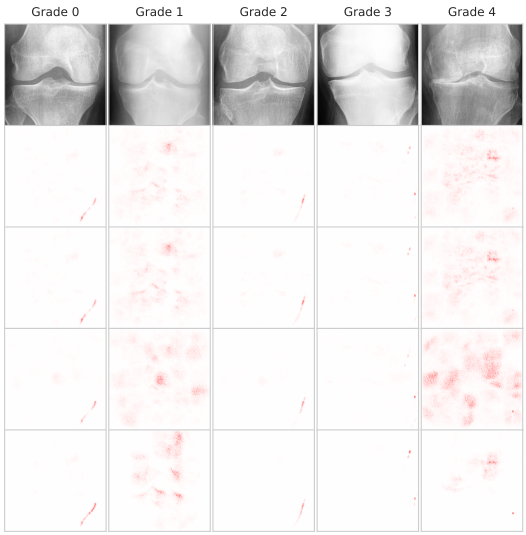

Figure 10: Spurious Images for Spurious Knee Model Examples

I User Study Discussion

In this section we present additional results of the user study along with break down of some demographic information about the participants.

Gender	Counts
Male	130
Female	62
N/A	8

Table 5: **Gender Breakdown across participants**.

ML Background	Counts
ML Researcher	30
ML Practitioner	21
Major Familiarity with ML	50
Limited Familiarity	47
No Familiarity	52

Table 6: **Machine Learning Experience Breakdown across participants**.

We now provide additional tables for the the feature attribution metrics from the main text.

Metric	Gradient	SmoothGrad	Integrated Gradients	Guided BackProp
SISM-GT	0.68	0.74	0.62	0.92
SEM	0.0091	0.017	0.019	0.0038
NISM-SISM	0.0013	8.8e-4	0.07	0.37
SEM	0.0012	0.0082	0.076	0.059

Table 7: **Metrics for Bone Age Spurious Tag Model.**

Metric	Gradient	SmoothGrad	Integrated Gradients	Guided BackProp
SISM-GT	0.78	0.68	0.59	0.89
SEM	0.0011	0.012	0.019	0.0088
NISM-SISM	0.0013	7.2e-5	0.17	0.46
SEM	0.0016	0.0032	0.026	0.029

Table 8: **Metrics for Bone Age Spurious Stripe Model.**

Metric	Gradient	SmoothGrad	Integrated Gradients	Guided BackProp
SISM-GT	0.7	0.84	0.49	0.92
SEM	0.001	0.02	0.06	0.008
NISM-SISM	0.0013	2.4e-4	0.21	0.86
SEM	0.006	0.002	0.06	0.09

Table 9: **Metrics for Bone Age Spurious Blur Model.**

Table 10: Performance metrics for each attribution method across tasks for the Tag Setting.

Method	Bone Age				Knee				Dog Breed			
	Grad	SG	IG	GBP	Grad	SG	IG	GBP	Grad	SG	IG	GBP
K-SSD	0.65	0.66	0.67	0.81	0.51	0.49	0.47	0.76	0.71	0.76	0.79	0.88
CCM	0.37	0.39	0.35	0.75	0.32	0.33	0.35	0.66	0.42	0.41	0.39	0.64
FAM	0.51	0.55	0.53	0.68	0.46	0.47	0.45	0.69	0.59	0.64	0.68	0.73

Table 11: Performance metrics for each attribution method across tasks for the Blur Setting.

Method	Bone Age				Knee				Dog Breed			
	Grad	SG	IG	GBP	Grad	SG	IG	GBP	Grad	SG	IG	GBP
K-SSD	0.21	0.20	0.19	0.13	0.13	0.18	0.17	0.31	0.29	0.30	0.31	0.35
CCM	0.28	0.29	0.24	0.64	0.23	0.22	0.27	0.67	0.38	0.33	0.35	0.71
FAM	0.48	0.49	0.47	0.51	0.36	0.38	0.33	0.58	0.55	0.56	0.47	0.73

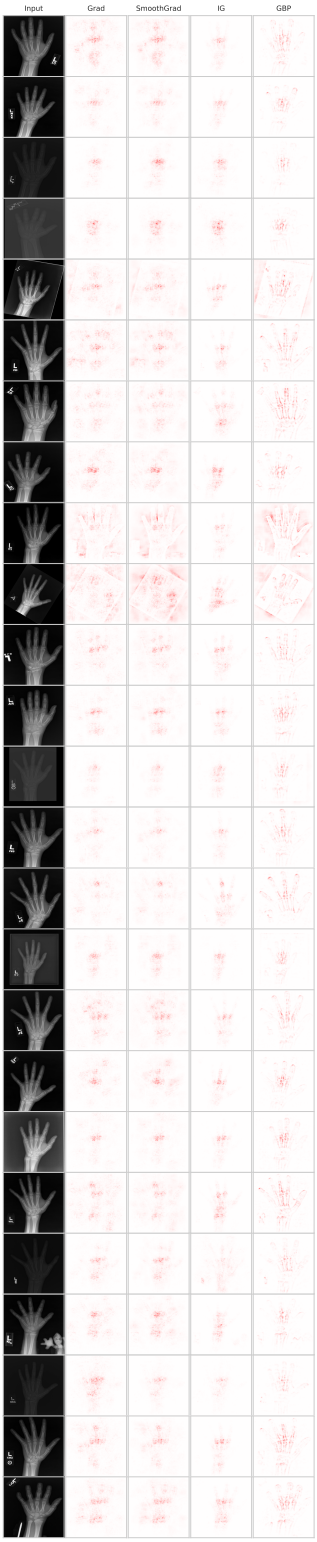

Figure 11: Normal Model on Normal Images

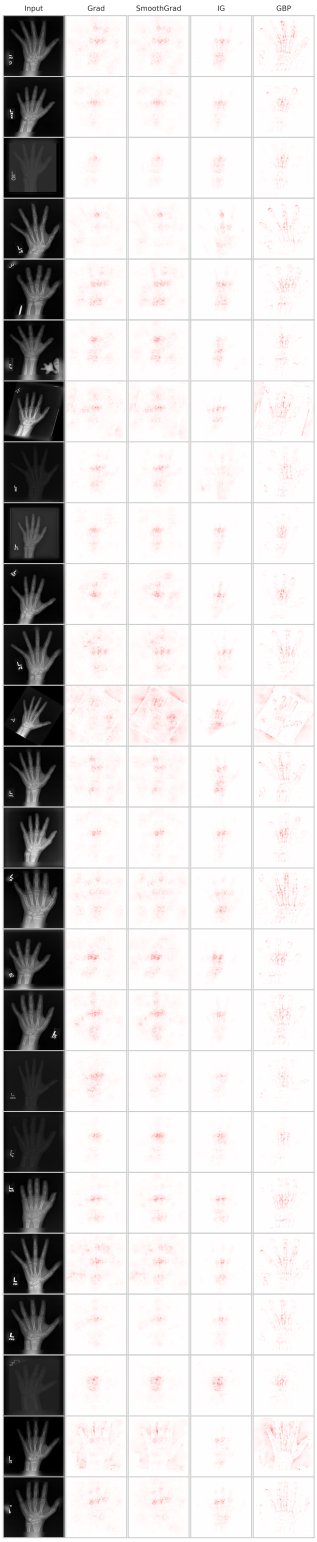

Figure 12: Normal Model on Spurious Blur Images

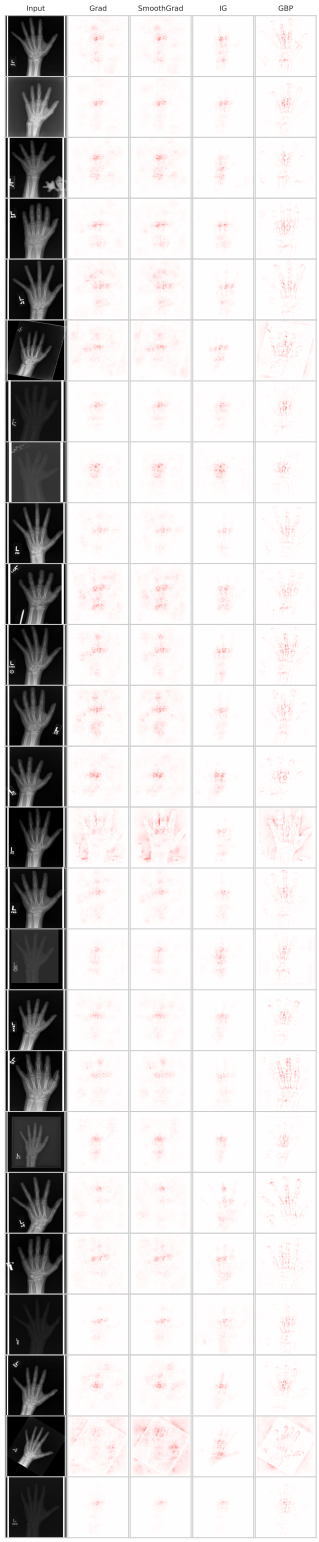

Figure 13: Normal Model on Spurious Stripe Images

Figure 14: Normal Model on Spurious Tag Images

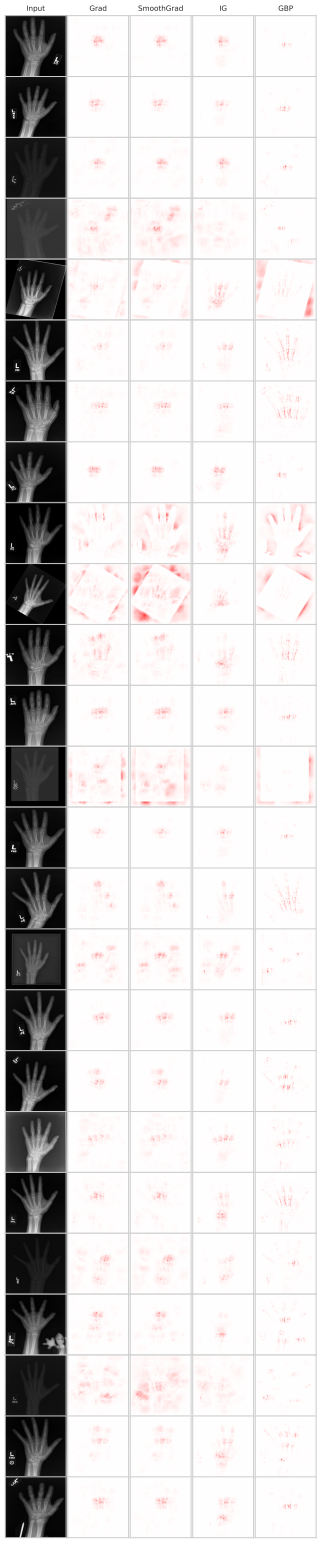

Figure 15: Spurious Stripe Model on Normal Images

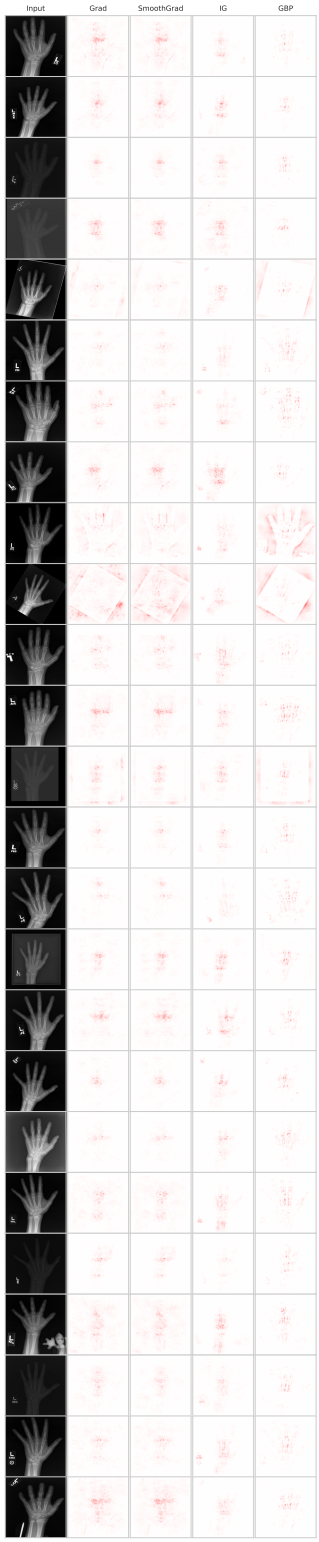

Figure 16: Spurious Tag Model on Normal Images

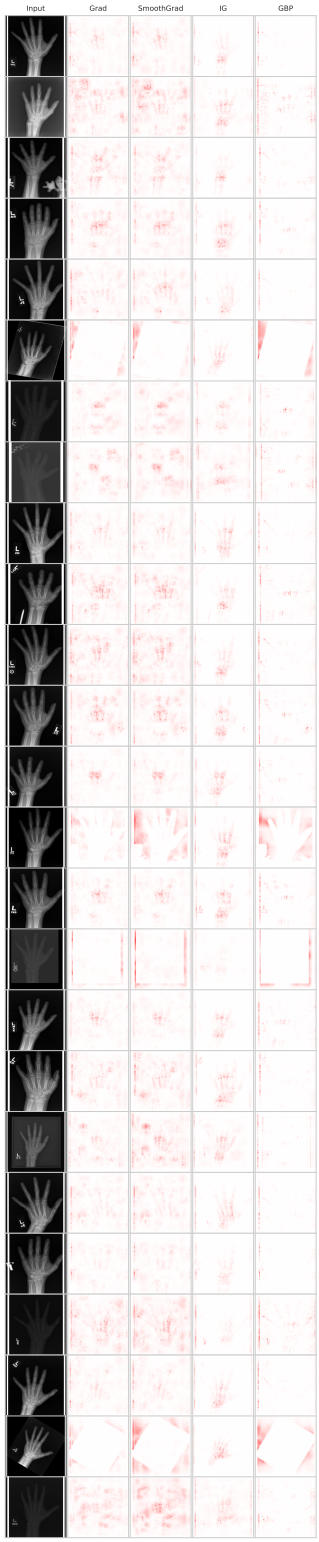

Figure 17: Spurious Stripe Model on Spurious Stripe Images

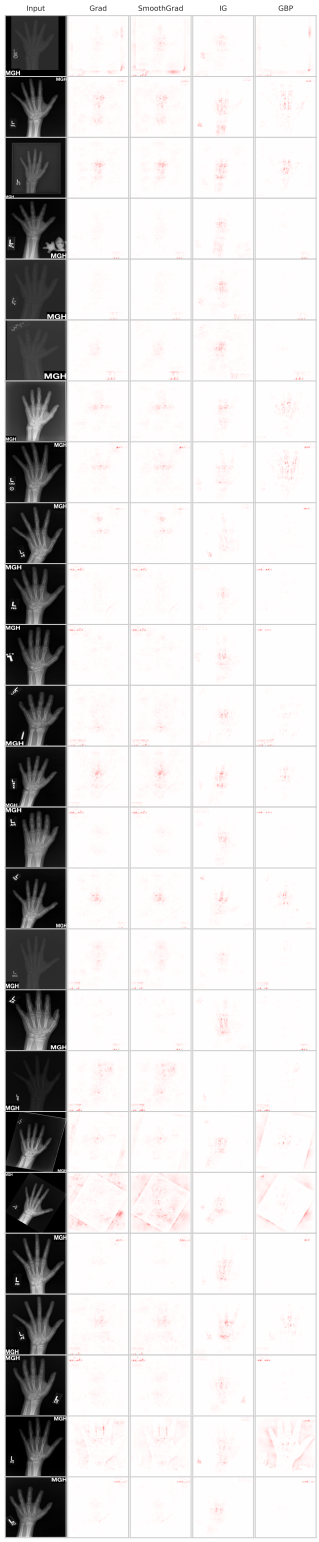

Figure 18: Spurious Tag Model on Spurious Tag Images

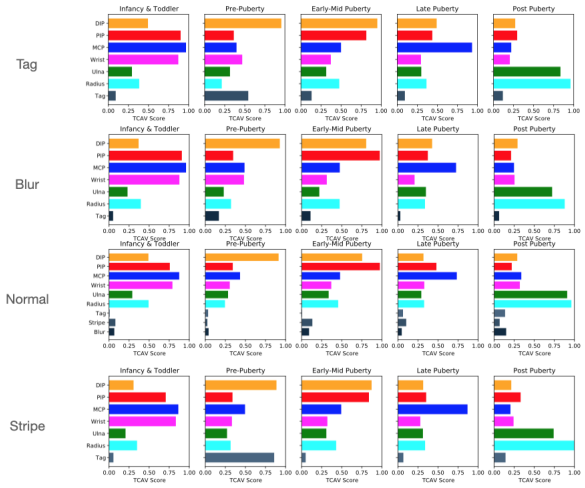

Figure 19: Results for Concept Ranking for another run of a normal and spurious bone age models.

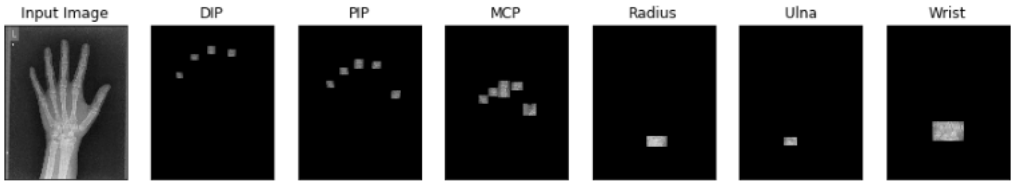

Figure 20: **Concept Partition for Bone Age Example.** Here we show how we partition a single instance into its constituent clinical concept components.

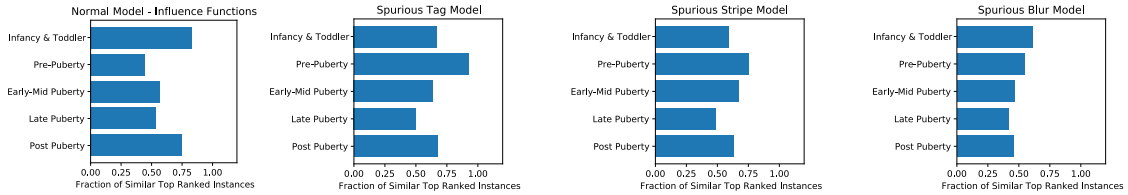

Figure 21: **Influence Function Results.** Here we show the *identical class metric (ICM)* (Hanawa et al., 2020) for a normal model and models that rely on the three spurious signals tested in this work. ICM measures what fractions of the top training inputs for a given test example belong to the same class as the true class of the test example in question.

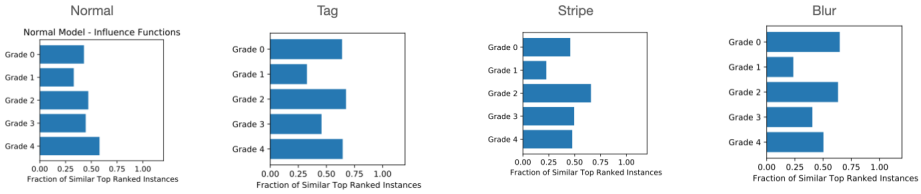

Figure 22: Influence Functions Results for Knee Settings.

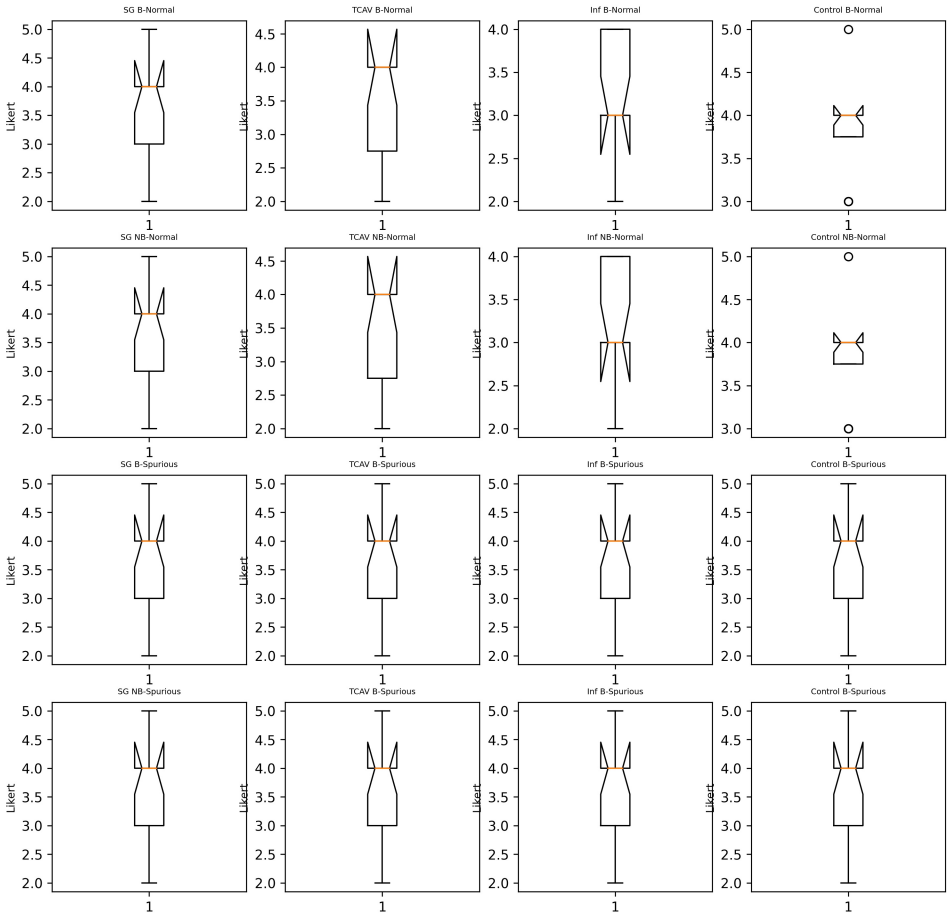

Figure 23: **Box plots of User responses.**