MARGIN: Uncovering Deep Neural Networks Using Graph Signal Analysis

Rushil Anirudh[1]*, Jayaraman J. Thiagarajan[1]*, Rahul Sridhar[2] and Peer-Timo Bremer[1]

[1]Center for Applied Scientific Computing (CASC), Lawrence Livermore National Laboratory, Livermore, CA, United States, [2]Walmart Labs, California, CA, United States
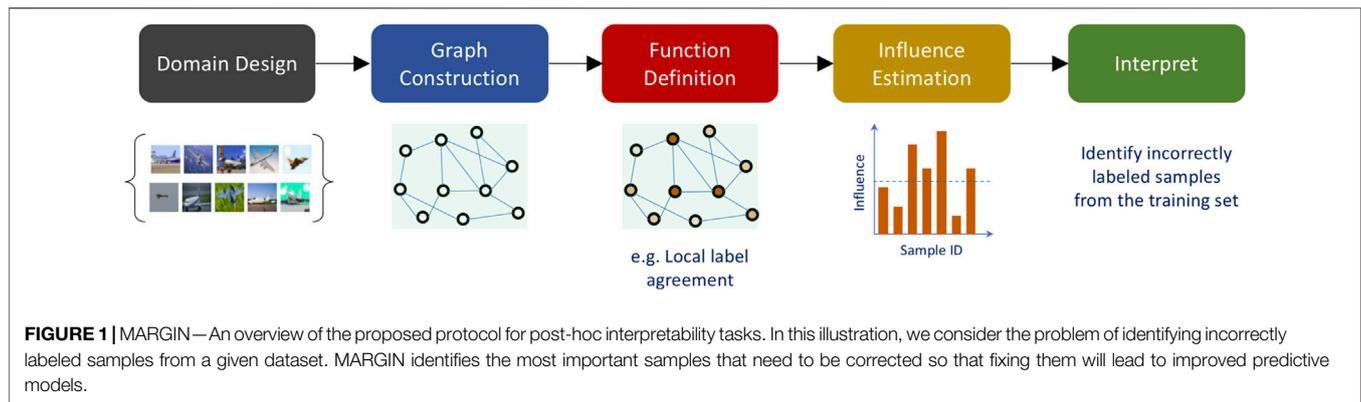
Interpretability has emerged as a crucial aspect of building trust in machine learning systems, aimed at providing insights into the working of complex neural networks that are otherwise opaque to a user. There are a plethora of existing solutions addressing various aspects of interpretability ranging from identifying prototypical samples in a dataset to explaining image predictions or explaining mis-classifications. While all of these diverse techniques address seemingly different aspects of interpretability, we hypothesize that a large family of interepretability tasks are variants of the same central problem which is identifying *relative* change in a model's prediction. This paper introduces MARGIN, a simple yet general approach to address a large set of interpretability tasks MARGIN exploits ideas rooted in graph signal analysis to determine influential nodes in a graph, which are defined as those nodes that maximally describe a function defined on the graph. By carefully defining task-specific graphs and functions, we demonstrate that MARGIN outperforms existing approaches in a number of disparate interpretability challenges.

Keywords: graph signal processing, interpretability, influence sampling, adversarial attacks, machine learning

INTRODUCTION

With widespread adoption of deep learning solutions in science and engineering, obtaining post-hoc interpretations of the learned models has emerged as a crucial research direction. This is driven by a community-wide effort to develop a new set of meta-techniques able to provide insights into complex neural network systems, and explain their training or predictions. Despite being identified as a key research direction, there exists no well-accepted definition for interpretability. Instead, in different contexts, it may refer to a variety of tasks ranging from debugging models (Ribeiro et al., 2016), to determining anomalies in the training data (Koh and Liang, 2017). While some recent efforts (Lipton, 2016; Doshi-Velez and Kim, 2017) provide a more formal definition for interpretability as generating *interpretable rules*, these focus on instance-level explanations, i.e. understanding how a network arrived at a particular decision for a single instance. In practice, interpretability covers a wider range of challenges, such as characterizing data distributions and separating hyperplanes of classifiers, identifying noisy labels during training, detecting adversarial attacks, or generating saliency maps for image classification. As discussed below, solutions to all such problems have been proposed each using custom tailored, task-specific approaches. For example, a variety of tools aim to explain which parts of an image are the most responsible for a prediction. However, these cannot be easily re-purposed to identify which samples in a dataset were most helpful or harmful to train a classifier.

Instead, we argue that many existing interpretability techniques solve a variant of essentially the same task–understanding *relative* changes in the model's prediction, where the changes are either *global* in nature, i.e., across an entire distribution or *local*, i.e., within a single sample. In this paper, we

FIGURE 1 | MARGIN—An overview of the proposed protocol for post-hoc interpretability tasks. In this illustration, we consider the problem of identifying incorrectly labeled samples from a given dataset. MARGIN identifies the most important samples that need to be corrected so that fixing them will lead to improved predictive models.

propose the MARGIN (Model Analysis and Reasoning using Graph-based Interpretability) framework, which directly applies to a wide variety of interpretability tasks. MARGIN poses each task as an *hypothesis* test and derives a measure of *influence* that indicates which parts of the data/model maximally support (or contradict) the hypothesis. More specifically, for each task we construct a graph whose nodes represent entities of interest, and define a function on this graph that encodes a hypothesis. For example, if the task is to determine which samples need to be reviewed in a dataset containing noisy labels, the domain is the set of samples, while the function can be local label agreement that measures how misaligned are the neighborhoods of the input samples (or their features) and their corresponding labels. Using graph signal processing (Sandryhaila and Moura, 2013; Shuman et al., 2013) one can then identify which nodes are essential to reconstructing the chosen function (hypothesis), which most likely will correspond to those with flipped labels. In order words, through a careful selection of graph construction strategies and hypothesis functions, this general procedure can be used to solve a wide-range of post-hoc intereperetability tasks.

This generic formulation, while extremely simple in its implementation, provides a powerful protocol to realize several meta-learning techniques, by allowing the user to incorporate rich semantic information, in a straightforward manner. In a nutshell, the proposed protocol is comprised of the following steps: 1) identifying the domain for interpretability (for e.g. intra-sample vs inter sample), 2) constructing a neighborhood graph to model the domain (for e.g. pixel space vs. latent space), 3) defining an *explanation function* at the nodes of the graph, 4) performing graph signal analysis to estimate the *influence* structure in the domain, and 5) creating interpretations based on the estimated influence structure. **Figure 1** illustrates the steps involved in MARGIN for *a posteriori* interpretability.

Overview

Using different choices for graph construction and the explanation function design, we present five case studies to demonstrate the broad applicability of MARGIN for *a posteriori* interpretability. First, in *Case Study I—Prototypes and Criticisms* we study a unsupervised problem of identifying samples which well characterize the underlying data distribution, referred to as prototypes and criticisms respectively (Kim et al.,

2016). We show that the MARGIN is highly effective at characterizing data distributions and can shed light into the regimes where classifier performance can suffer. In *Case Study II—Explanations for Image Classification*, we obtain pixel-level explanations from an image classifier using MARGIN, without the need to access the model internals, i.e., black-box and show that the inferred feature importance estimates are meaningful. In *Case Study III—Detecting Incorrectly Labeled Samples*, we employ MARGIN to identify label corruptions in the training data and demonstrate significant improvements over popular approaches such as influence functions. In *Case Study IV—Interpreting Decision Boundaries*, we illustrate the application of MARGIN in analyzing pre-trained classifiers and identifying the most influential samples in describing the decision surfaces, akin to memorable examples in continual learning (Pan et al., 2020). Finally, in *Case Study V—Characterizing Statistics of Adversarial Examples* we extend two recently proposed statistical techniques to detect adversarial examples from harmless examples, and demonstrate that incorporating them inside MARGIN improves their discriminative power significantly.

RELATED WORK

We outline recent works that are closely related to the central framework, and themes around MARGIN. Papers pertinent to individual case studies are identified in their respective sections. Our goal in this paper is to design a core framework that is capable of being repurposed to interpretability tasks, ranging from explaining decisions of a predictive model, detecting outliers to identifying label corruptions in the training data. While post-hoc explanation methods are the *modus-operandi* in interpreting the decisions of a black box model, their scope has widened significantly in the recent years. For example, popular sensitivity analysis such as LIME (Ribeiro et al., 2016) and SHAP (Lundberg and Lee, 2017) or gradient-based methods such as Saliency Maps (Simonyan et al., 2013), Integrated Gradients (Sundararajan et al., 2017), Grad-CAM (Selvaraju et al., 2017), DeepLIFT (Shrikumar et al., 2017) and DeepSHAP (Lundberg and Lee, 2017) are routinely used to produce sample-wise, local explanations by measuring the sensitivity of the black-box to perturbations in the input features (Fong and Vedaldi, 2017). Despite their

wide-spread use, they cannot be readily utilized to obtain dataset-level explanations, e.g., which are the most influential examples in a dataset for a given test sample, or to detect distribution shifts (Thiagarajan et al., 2020). On the other hand, in (Koh and Liang, 2017), the authors proposed a strategy to select influential samples by extending ideas from robust statistics, which was shown to be applicable to a variety of scenarios. However, such methods cannot be used for obtaining feature importance estimates. Another important challenge with most existing post-hoc explanation techniques is their computational complexity. In contrast, MARGIN leverages the generality of graph structures to scalably generate explanations, and through of use of appropriate hypothesis functions can support a large-class of interpretations.

In a nutshell, MARGIN reposes the problem of generating explanations as an influential node selection problem, wherein the node can correspond to a sample-level or feature-level explanations and the influence is measured based on a hypothesis function. Defining suitable objectives for detecting influential features in an image or influential samples in a dataset has been an important topic of research in explainable AI. For example, CXPlain (Schwab and Karlen, 2019) and Attentive Mixture of Experts (Schwab et al., 2019) utilize a Granger-causality based objective to quantify feature importances. In addition, prediction uncertainties Chakraborty et al. (2017) or even loss estimates Thiagarajan et al. (2020) have been widely adopted to characterize vulnerabilities of a trained model. Note that, MARGIN can directly use any of these objectives to choose the most relevant explanations. In this paper, we consider a variety of interpretability tasks and recommend suitable hypothesis functions for each of the tasks.

Since MARGIN relies on ideas from graph signal processing (GSP) to select the most relevant explanations, we briefly review existing work in this area. Broadly, there are two classes of approaches in GSP–one that builds on spectral graph theory using the graph Laplacian matrix (Shuman et al., 2013), and the other based on algebraic signal processing that builds upon graph shift operator (Sandryhaila and Moura, 2013). While both are applicable to our framework, we adopt the latter formulation. Our approach relies on defining a measure of influence at each node, which is related to sampling of graph signals. This is an active research area, with several works generalizing ideas of sampling and interpolation to the domain of graphs, such as (Pesenson, 2008; Gadde et al., 2014; Chen et al., 2015).

A GENERIC PROTOCOL FOR INTERPRETABILITY

In this section, we provide an overview of the different steps of MARGIN and describe the proposed influence estimation technique in the next section.

Domain Design and Graph Construction

The domain definition step is crucial for the generalization of MARGIN across different scenarios. In order to enable instance-level interpretations (e.g. creating saliency maps), a single instance of data, possibly along with its perturbed variants, will form the domain; whereas a more holistic understanding of the model can be obtained (e.g. extracting prototypes/criticisms) by defining the entire dataset as the domain. Regardless of the choice of domain, we propose to model it using nearest neighbor graphs, as it enables a concise representation of the relationships between the domain elements.

More specifically, given the set of samples $\{x_i\}$, we construct a k-nearest neighbor domain graph that captures local geometry of the data samples. The metric for graph construction (that determines neighborhoods/edges) can arise from prior knowledge about the domain or designed based on latent representations from pre-trained models. For example, if we use the latent features from AlexNet (Krizhevsky et al., 2012), the resulting graph respects the distance metric inferred by AlexNet for image classification. Though the difficulty in choosing an appropriate k for designing robust graphs is well known, designing better graphs is beyond the scope of this paper. In our experiments, we find that our results are not very sensitive to the choice of k.

Formally, an undirected weighted graph is represented by the triplet $\mathcal{G} = (\nu, E, W)$, where ν denotes the set of nodes, E denotes the set of edges and W is an adjacency matrix that specifies the weights on the edges, where $W_{n,m}$ corresponds to the edge weight between nodes v_n and v_m. Let $N_n = \{m | W_{n,m}^1 0\}$ define the neighborhood of node v_n, i.e. the set of nodes connected to it. The normalized graph Laplacian, L, is then constructed as $I - D^{-1/2} W D^{-1/2}$, where $D_{nn} = \sum_m W_{n,m}$ is the degree matrix and I denotes the identity matrix.

Explanation Function Definition

A key component of MARGIN is to construct an explanation function that measures how well each node in the graph supports the presented hypothesis. The function acts on each vertex of the graph as: $f(n) : v_n a R$ for all n vertices in the graph $\mathcal{G}$. This function is also referred to as the graph signal defined on the graph domain. We expect this function to capture properties of the explaination that are deemed important. Let us illustrate this process with an example–in order to create saliency maps for image classification, one can build a graph where each node corresponds to a potential explanation (i.e. a subset of pixels), while the edges can measure how likely can two explanations produce similar predictions. In such a scenario, one can hypothesize that an *ideal* explanation will be *sparse*, in terms of the number of pixels, since that is more interpretable. Consequently, the size of an explanation can be used as the function. *Case Studies* will present a more detailed discussion.

Influence Estimation

This is the central analysis step in MARGIN for obtaining influence estimates at the nodes of $\mathcal{G}$, that can reveal which nodes can maximally describe the variations in the chosen explanation function. Implicitly, this step can be viewed as a *soft*-sample selection strategy with respect to the structure induced by the domain graph. We propose to perform this estimation using tools from graph signal analysis. *Proposed*

Influence Estimation describes the proposed algorithm for influence estimation.

From Influence to Interpretation

Depending on the hypothesis chosen for *a posteriori* analysis, this step requires the design of an appropriate strategy for transferring the estimated influences into an interpretable explanation.

PROPOSED INFLUENCE ESTIMATION

Given a nearest neighbor graph $\mathcal{G}$ along with an explanation function f, we propose to employ graph signal analysis to estimate node influence scores. Before we describe the algorithm, we will present a brief overview of the preliminaries.

Definitions. We use the notation and terminology from (Sandryhaila and Moura, 2013) in defining an operator analogous to the *time-shift* or *delay* operator in classical signal processing. The function f assigns a scalar value to each vertex as defined earlier, as a result the entire function is written as $f : \nu \mapsto R^N$, where $|\nu| = N$, i.e., f is a collection of scalar values at each vertex, ordered according to the same order of vertices in the graph. When the graph does not have any special structure (i.e., it is Euclidean), f is nothing but a vector valued function. We consider the simplest scenario here where the function only takes a scalar value at each node, however more general cases maybe considered where the value at each node is multi-dimensional. During a graph shift operation, the function $f(n)$ at node ν_n is replaced by a weighted linear combination of its neighbors: $\hat{f} = Af$, where A is the graph shift operator, which is the simplest, non-trivial graph filter. Commonly used choices for A include the adjacency matrix W, transition matrix $D^{-1}W$ and the graph Laplacian L.

The set of eigenvectors of the graph shift operator is referred to as the graph Fourier basis, $A = U\Lambda U^T$, where $U \in \mathbb{R}^{N \times N}$, and the Fourier transform of a signal $f \in \mathbb{R}^N$ is defined as $U^T f$. The ordered eigenvalues corresponding to these eigenvectors represent frequencies of the signal, with λ_1 to λ_N representing the smallest to largest frequencies. The notion of frequency on the graph corresponds to the rate of change of the function across nodes in a neighborhood. A higher change corresponds to a high frequency, while a smooth variation corresponds to a low frequency. In this context, the graph filtering using a graph shift operator corresponds to a *low-pass* filter that dispenses high frequency components in the function. Similarly, a simple *high-pass* filter can be easily designed as $\hat{f}_h = f - \hat{f}$.

Algorithm 1 MARGIN's simple influence estimation.

```
1: procedure MARGIN-INFLUENCE(X,G, f)        ▷ Domain, Graph and explanation function
2:     Construct graph shift operator A from X
3:     f̂ = f − Af                             ▷ High pass spectral filter
4:     for i ∈ V do                           ▷ Iterate over all the nodes of the graph G
5:         Compute I(i) = ||f̂(i)||²₂  ∀i ∈ V
6:     return I(i)∀i ∈ V                      ▷ Influence score for each node
```

Algorithm: The overall procedure to obtain influence scores at the nodes of $\mathcal{G}$ can be found in **Algorithm 1**. Intuitively, we design a high-pass filter that eliminates the low frequency content and retains the signal energy only at those nodes that characterize the

extreme variations of the function. Following the high-pass filtering step, the influence score at a node is estimated as the magnitude of the filtered function value at that node:

$$I(i) = \left\| \widehat{f_h}(i) \right\|_2^2 \forall i \in \nu, \tag{1}$$

where $\widehat{f_h}$ corresponds to the high-pass filtered version of f. Interestingly, we find that analyzing the high frequency components of the explanation function often leads to a sparse influence structure, indicating the presence of multiple local optima that corroborate the hypothesis. Conversely, the influence structure obtained from low frequency components is typically dense and hence requires additional processing to qualify regions of disagreement.

Sensitivity to Graph Construction

A critical step in MARGIN is the graph construction process for datasets that do not naturally have a graph structure. In this work, we rely on a simple nearest neighbor graph for construction which can vary depending on the size of the neighborhood. This is a hyper parameter that must be set with validating examples, and in all our case studies we found a neighborhood size of 20-40 to be quite good in terms of computational efficiency in constructing the graph. This directly influences the quality of low pass filtering of a graph signal similar to the case in Euclidean signal processing in choosing a size of the window. As the neighborhood size increases, the filtering at each node becomes more aggressive since it averages the across several neighboring nodes, while for a small neighborhood the smoothing may not have any effect at all. MARGIN is agnostic to the type of graph construction used, since it ultimately only relies on the graph filtering process, and as a result it is applicable to more other graph constructions such as Reeb graphs (Pascucci et al., 2007) or β−skeletons.

CASE STUDIES

Considering MARGIN is very generic in nature, it is easy applicable to a wide variety of interpretability tasks. In this section we illustrate this felxibility on several example tasks. **Table 1** shows the domain design, graph construction, and function definition choices made for different use cases. Note in each case study, we construct a k-nearest-neighbor graph followed by the application of MARGIN with the main difference is in how the nodes of the graph are defined, followed by the type of function that is defined at each node.

Case Study I—Prototypes and Criticisms

A commonly encountered problem in interpretability is to identify samples that are prototypical of a dataset, and those that are statistically different from the prototypes (called criticisms). Together, they can provide a holistic understanding about the underlying data distribution. Even in cases where we do not have access to the label information, we seek a hypothesis that can pick samples which are representatives of their local neighborhood, while emphasizing statistically

Task	Domain	Nodes in G	Function	Explanation Modality
Prototypes/ Criticisms	Complete dataset	Samples	MMD (Global,Local)	Sample sub-selection
Explain prediction	Single image	Explanations	Sparsity	Saliency maps
Detect noisy-labels	Complete dataset	Samples	Local label-agreement	Samples to fix
Detect adversarial-attacks	Attacks/Noisy samples	Perturbed samples	MMD (Global)	Attack statistics
Study discrimination	Complete dataset	Samples	Local label-agreement	Confusing samples

anomalous samples. One such function was recently utilized in (Kim et al., 2016) to define prototypes and criticisms, and it was based on Maximum Mean Discrepancy (MMD).

Formulation

Following the general protocol in **Figure 1**, the domain is defined as the complete dataset, along with labels if available. Since this analysis does not rely on pre-trained models, we construct the neighborhood graph based on the Euclidean distance using $k = 25$ nearest neighbors. Inspired by (Kim et al., 2016), we define the following explanation function: For each sample x_i, we remove the chosen sample and all its connected neighbors from the graph to construct the set $\overline{X}_i = \{x_j, j \notin (i \cup N_i)\}$, and estimate the function at the ith node as $f(i) = MMD(\overline{X}_i, \overline{X}_i \cup x_i)$. MMD gives us a way to measure the difference between two distributions, and since we artficially construct the two distributions by removing a single sample, we are able to determine the importance of an individual sample (and its neighbors) within the dataset using MARGIN. Let $k : X \times X \to R$ be a kernel such as the radial basis function (RBF) kernel, and $X = \overline{X}_i \cup x_i$, then we can use the approximation for MMD given in (c.f. Eq. 5 in Kim et al. (2016)) as:

$$MMD(x_i) = \frac{1}{|\overline{\mathcal{X}}_i|} \sum_{x_m \in \overline{X}_i} k(x_i, x_m) + \frac{1}{|\mathcal{X}|} \sum_{x_j \in \mathcal{X}} k(x_i, x_j). \quad (2)$$

In cases of labeled datasets, the kernel density estimates for the MMD computation are obtained using only samples belonging to the same class. We refer to these two cases as *global* (unlabeled case) and *local* (labeled case) respectively. The hypothesis is that the regions of criticisms will tend to produce highly varying MMD scores, thereby producing high frequency content, and hence will be associated with high MARGIN scores. Conversely, we find that the samples with low MARGIN scores correspond to prototypes since they lie in regions of strong agreement of MMD scores. More specifically, we consider all samples with low MARGIN scores (within a threshold) as prototypes, and rank them by their actual function values. In contrast to the greedy inference approach in (Kim et al., 2016) that estimates prototypes and criticisms separately, they are inferred jointly in our case.

Experiment Setup and Results

We evaluate the effectiveness of the chosen samples through predictive modeling experiments with the idea that the most helpful samples should result in a good classifier, whereas a the most unhelpful/confusing samples should result in a poor classifier. We use the USPS handwritten digits data for this experiment, which consists of 9,298 images belonging to 10 classes. We use a standard train/test split for this dataset, with 7,291 training samples and the rest for testing. For fair comparisons with (Kim et al., 2016), we use a simple 1-nearest neighbor classifier. As described earlier, we consider both unsupervised (*global*) and supervised (*local*) variants of our explanation function for sample selection.
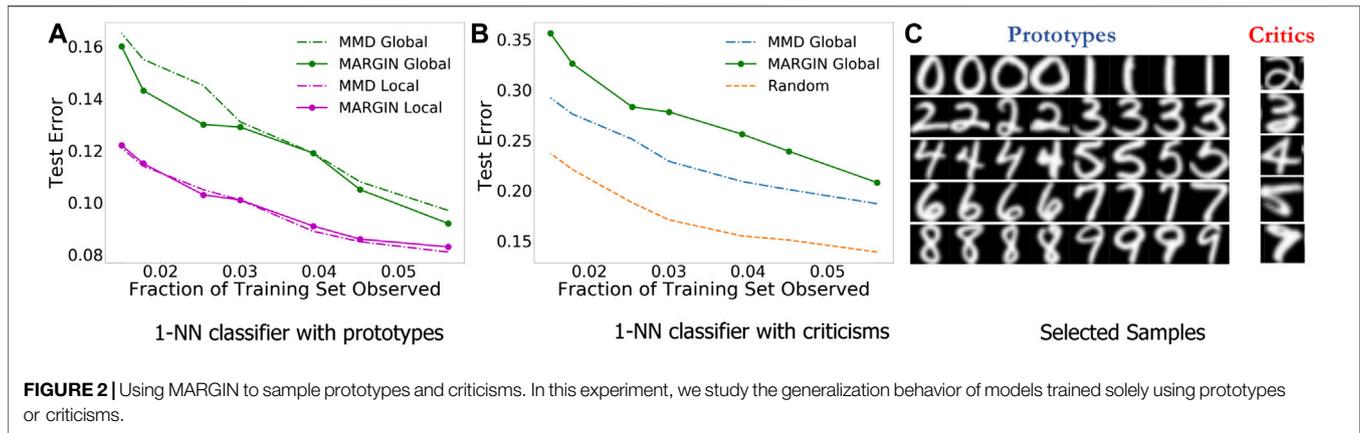
We expect the prototypical samples to be the most helpful in predictive modeling, i.e., good generalization. In **Figure 2A**, we observe that the prototypes from MARGIN perform competitively in comparison to the baseline technique. More importantly, MARGIN is particularly superior in the global case, with no access to label information. On the other hand, criticisms are expected to be the least helpful for generalization, since they often comprise boundary cases, outliers and under-sampled regions in space. Hence, we evaluate the test error using the criticisms as training data. Interestingly, as shown in **Figure 2B**, the criticisms from MARGIN achieve significantly higher test errors in comparison to samples identified using *MMD-critic* based optimization in (Kim et al., 2016). Furthermore, examples of the selected prototypes and criticisms from MARGIN are included in **Figure 2C**.

Case Study II—Explanations for Image Classification

Generating explanations for predictions is crucial to debugging black-box models and eventually building trust. Given a model, such as a deep neural network, that is designed to classify an image into one of r classes, a plausible *explanation* for a test prediction is to quantify the importance of different image regions to the overall prediction, i.e. produce a saliency map. We posit that perturbing the salient regions should result in maximal changes to the prediction. In addition, we expect *sparse* explanations to be more interpretable. In this section, we describe how MARGIN can be applied to achieve both these objectives.

Formulation

Since we are interested in producing explanations for instance-level predictions using MARGIN, the domain corresponds to a possible set of explanations for an image. Note that, the space of explanations can be combinatorially large, and hence we adopt the following greedy approach to construct the domain. We run the SLIC algorithm (Achanta et al., 2012) with varying number of superpixels, say $\{50, 100, 150, 200, 250, 300\}$, and define the

FIGURE 2 | Using MARGIN to sample prototypes and criticisms. In this experiment, we study the generalization behavior of models trained solely using prototypes or criticisms.

domain as the union of superpixels from all the independent runs. In our setup, each of these superpixels is a plausible explanation and they become the nodes of $\mathcal{G}$. The edge between nodes m and n of this graph is defined based on the relative importance of each super-pixel, i.e., $e_{mn} = \left\| \left| p_j(I) - p_j(I_m) \right| - \left| p_j(I) - p_j(I_n) \right| \right\|$, where I is the original image, and I_m is the image with the mth super-pixel masked out, and $p_j()$ extracts the softmax scores for the jth class in the image. This relative importance captures how two super-pixels are related in terms of the predictive model, which is related to a causal objective that is used in CXPlain Schwab and Karlen (2019).

For each of the explanations (super-pixels) m, we mask its pixels in the image and use the pre-trained model to obtain a measure of its saliency as before as $\left| p_j(I) - p_j(I_m) \right|$. Using these estimates, we obtain pixel-level saliency, S, as a weighted combination of their saliency from different superpixels (inversely weighted by the superpixel size). This dense saliency is similar to previous approaches such as (Zeiler and Fergus, 2014; Zhou et al., 2014).

Note that, this saliency estimation process did not impose the sparsity requirement. Hence, we use MARGIN to obtain influence scores based on their sparsity. The explanation function at each node is defined as the ratio of the size of the superpixel corresponding to that node and the size of the largest superpixel in the graph. Intuitively, MARGIN finds the sparsest explanation for different level sets of the saliency function. Subsequently, we compute pixel-level influence scores, I, as a weighted combination of their influences from different superpixels. The overall saliency map is obtained as $S_{\text{final}} = S \odot I$, where $\odot$ refers to the Hadamard product.

Experiment Setup and Results

Using images from the ImageNet database (Russakovsky et al., 2015), and the AlexNet (Krizhevsky et al., 2012) model, we demonstrate that MARGIN can effectively produce explanations for the classification. **Figure 3** illustrates the process of obtaining the final saliency map for an image from the *Tabby Cat* class. Interestingly, we see that the mouth and whiskers are highlighted as the most salient regions for its prediction. **Figure 4** shows the saliency maps from MARGIN for several other cases. For comparison, we show results from

Grad-CAM (Selvaraju et al., 2017), which is a white-box approach that accesses the gradients in the network. We find that, using only a black-box approach, MARGIN produces explanations that strongly corroborate with Grad-CAM and in some cases produces more interpretable explanations. For example, in the case of an *Ice Cream* image, MARGIN identifies the ice cream, and the spoon, as salient regions, while Grad-CAM highlights only the ice cream and quite a few background regions as salient. Similarly, in the case of a *fountain* image, MARGIN highlights the fountain, and the sky, while Grad-CAM highlights the background (trees) slightly more than the fountain itself, which is not readily interpretable.

Case Study III—Detecting Incorrectly Labeled Samples

An increasingly important problem in real-world applications is concerned with the quality of labels in supervisory tasks. Since the presence of noisy labels can impact model learning, recent approaches attempt to compensate by perturbing the labels of samples that are determined to be high-risk of being corrupted, or when possible have annotators check the labels of those high-risk samples. In this section, we propose to employ MARGIN to recover incorrectly labeled samples. In particular, we consider a binary classification task, where we assume β% of the labels are randomly flipped in each class. In order to identify samples which were incorrectly labeled, we select samples with the highest MARGIN score, followed by simulating a human user correcting the labels for the top K samples. Ideally, we would like K, the number of samples checked by the user, to be as small as possible.

Formulation

Similar to Case Study I, the entire dataset is used to define the domain. Since we expect the flips to be random, we hypothesize that they will occur in regions where the labels of corrupted samples are different from their neighbors. Instead of directly using the label at each node as the explanation function, we believe a more smoothly varying function will allow us to extract regions of high frequency changes more robustly. As a result, we propose to measure the level of *distrust* at a given node, by measuring how many of its neighbors disagree with its label:
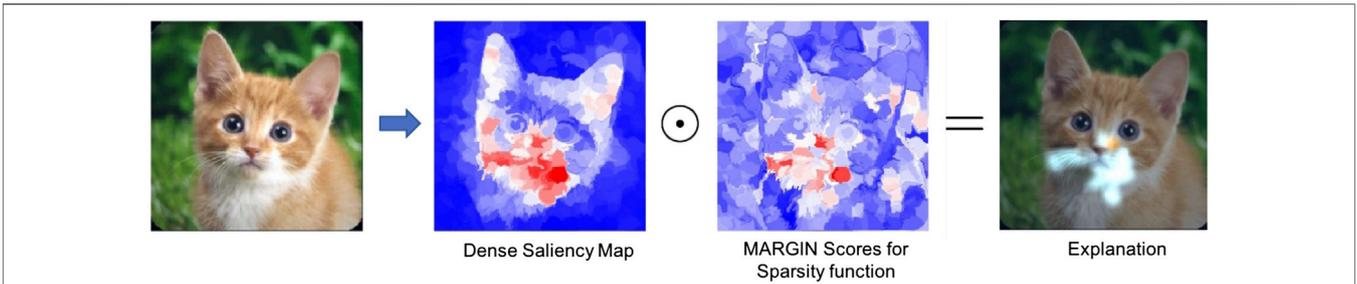
FIGURE 3 | We show the entire process of constructing the saliency map for one particular image (Tabby Cat) from ImageNet. From **left to right:** original image (dense) saliency map S, sparsity map I, and finally the explanation from MARGIN, S_{final}.
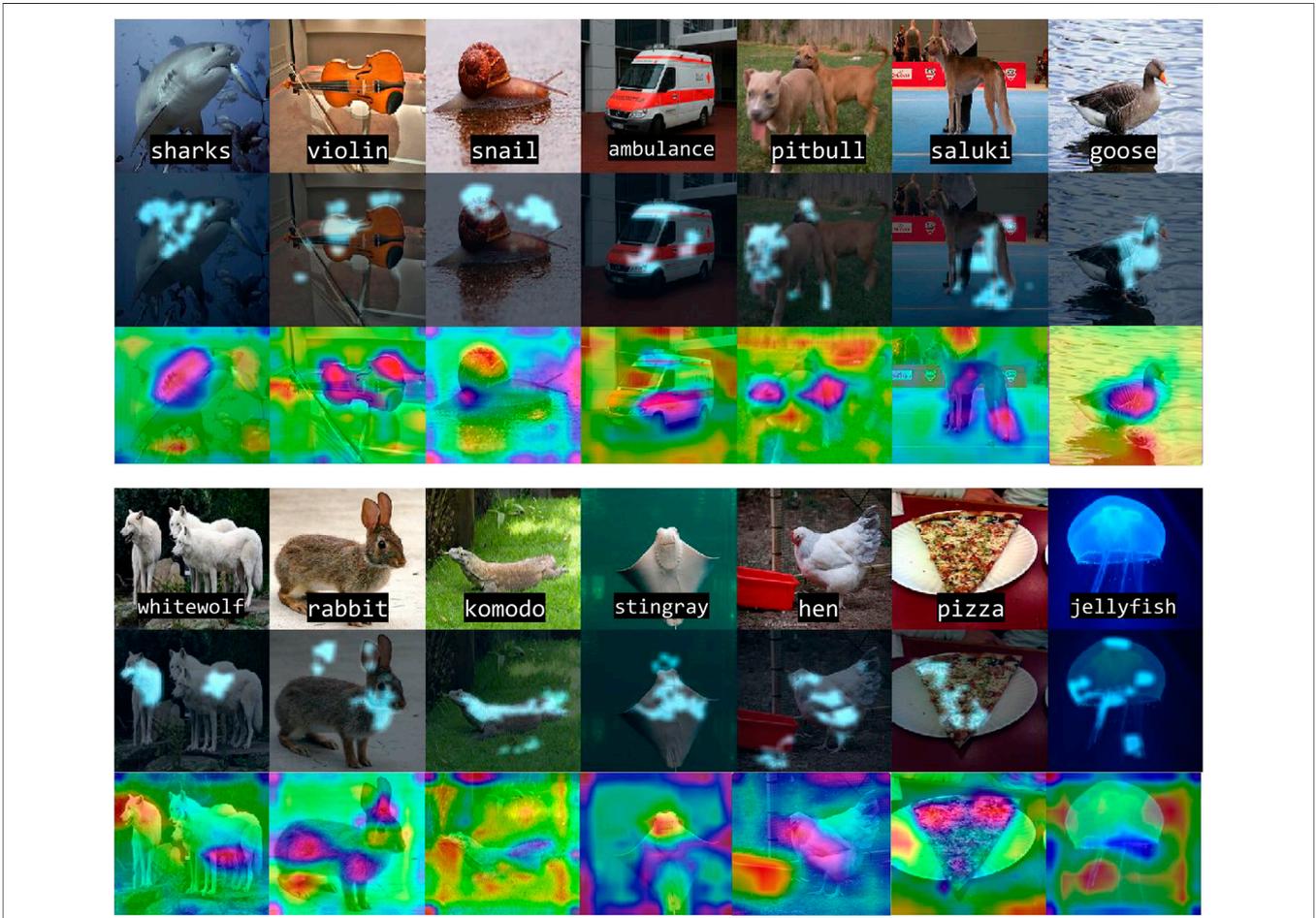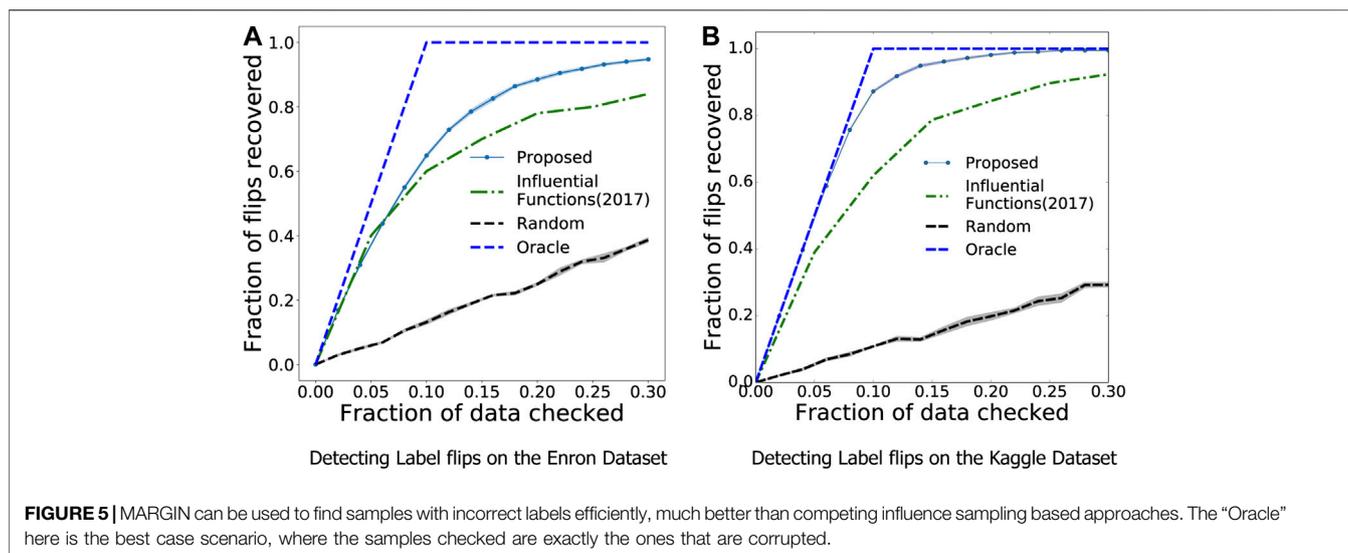

FIGURE 4 | Our approach identifies the most salient regions in different classes for image classification using AlexNet. From top to bottom: original image, MARGIN's explanation overlaid on the image, and Grad-CAM's (Selvaraju et al., 2017) explanation. Note our approach yields highly specific, and sparse explanations from different regions in the image for a given class.

$$f(i) = 1 - \frac{\sum_{j \in N_i} L(j,i)}{|N_i|}, \qquad (3)$$

where $L(j,i)$ is 1 only if nodes j and i share the same label; $|.|$ denotes the cardinality of a set.

Experiment Setup and Results

We perform our experiments on two datasets: 1) the Enron Spam Classification dataset (Metsis et al., 2006), containing 4138 training examples, with an imbalanced class split of around 70:30 (non-spam:spam), and 2) 3000 random images from

FIGURE 5 | MARGIN can be used to find samples with incorrect labels efficiently, much better than competing influence sampling based approaches. The "Oracle" here is the best case scenario, where the samples checked are exactly the ones that are corrupted.

Kaggle dog v cat classification dataset with almost equal number of images from each class[1]. Following standard practice, we randomly corrupt the labels of 10% of the samples. For the Enron Spam dataset, we extracted bag-of-words features of 500 dimensions corresponding to the most frequently occurring words. We observed these features to be noisy, so we use a simple PCA pre-processing step to reduce the dimensionality of the data down to 100. For Kaggle, we use penultimate features from AlexNet Krizhevsky et al. (2012) in order to construct a neighborhood graph. In both cases we use $k = 20$ as the number of neighbors for this purpose, we observed stable performance even when $k = 30$ or $k = 40$. The use of features instead of the data directly has become standard practice in several applications as it reduces the dimensionality of the data, while also providing a more semantically meaningful notion of neighborhood. We report average results from 10 repetitions of the experiment.

We compare our approach with three baselines: *1) Influence Functions:* We obtain the most influential samples using Influence Functions (Koh and Liang, 2017). *2) Random Sampling 3) Oracle:* The best case scenario, where the number of labels corrected is equal to the number of samples observed. Following (Koh and Liang, 2017), we vary the percentage of influential samples chosen, and compute the *recall* measure, which corresponds to the fraction of label flips recovered in the chosen subset of samples.

As seen in **Figure 5**, we see that our method is nearly 10 percentage points better than the state-of-the-art Influence Functions, achieving a recall of nearly 0.95 by observing just 30% of the samples. This difference is further improved when observing a balanced dataset like the Kaggle dogs v cats, as seen in **Figure 5B** where MARGIN outperforms Influence functions signficantly. On examining how MARGIN picks the samples, we see a clear trend which indicates a strong preference for samples that lie farther away from the classification boundary. In

other words, this corresponds strongly to correcting the least number of samples which can lead to the most gain in validation performance when using a trained model.

Case Study IV—Interpreting Decision Boundaries

While studying black-box models, it is crucial to obtain a holistic understanding of their strengths, and more importantly, their weaknesses. Conventionally, this has been carried out by characterizing the decision surfaces of the resulting classifiers. In this experiment, we demonstrate how MARGIN can be utilized to identify samples that are the most confusing to a model, or more precisely those examples which are likely to be mis-classified by a pre-trained classifier. By definition these are images that are closest to the decision boundary inferred by the classifier.
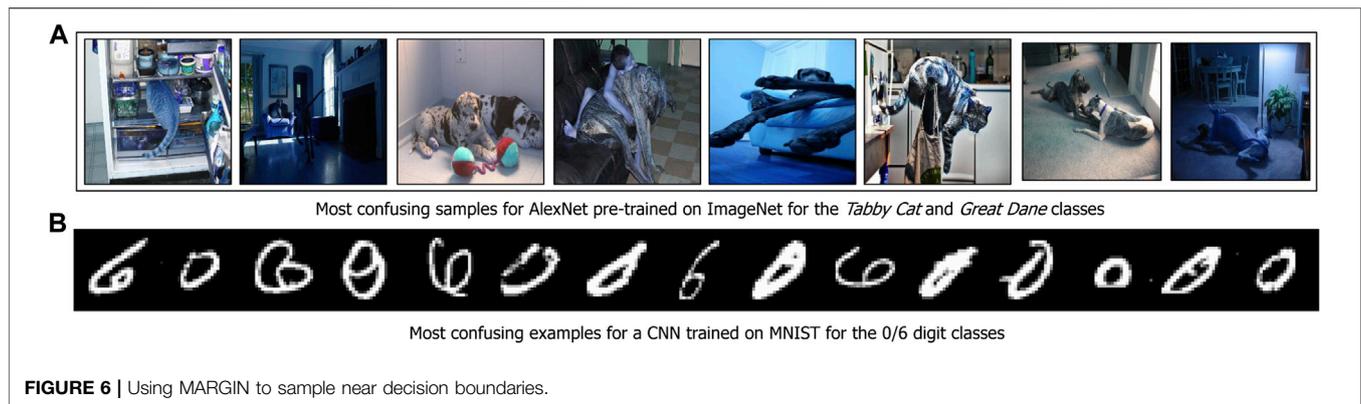
Formulation

In order to adopt MARGIN for analyzing a specific model, we construct a nearest neighbor graph ($k = 30$) using latent representations inferred from the pre-trained classifier in consideration. This achieves two things–it gives us a semantic similarity measure as interpreted by the classifier, i.e., which similarities are considered important for the classification task. More importantly for this case study, this automatically distills the information regarding confusing samples into the graph that is constructed, since these samples are likely to be in regions of the neighborhood with high prediction uncertainty. Next, since the decision surface characterization is similar to case Study III, we use the local label agreement measure in **(3)** as the explanation function. This disagreement between the function and the neighborhood shows up as high frequency information which is exploited by MARGIN to identify the decision surface.

Experiment Setup

We perform an experiment on 2-class datasets extracted from ImageNet and MNIST. More specifically, in the case of ImageNet,

[1]https://www.kaggle.com/c/dogs-vs-cats/data

FIGURE 6 | Using MARGIN to sample near decision boundaries.

we perform decision surface characterization on the classes *Tabby Cat* and *Great Dane*. We used the features from a pre-trained AlexNet's penultimate layer to construct the graph. For the MNIST dataset, we considered data samples from digits '0' and '6', and we used the latent space produced using a convolutional neural network for the analysis. A selected subset of samples characterizing the decision surfaces of both datasets are shown in **Figure 6**.

Results

From **Figure 6A**, we see that the model gets confused whenever the animal's face is not visible, or if it is in a position facing away from the camera. This is reasonable since we are only measuring the most confusing samples between the *Tabby Cat* and *Great Dane* classes which share a lot of semantic similarity. Similarly, in the MNIST dataset, the examples shown depict atypical ways in which the digits '0' and '6' can be written. These results suggest that MARGIN is effective in identifying these examples, with a combination of the appropriate neighborhoods (in the latent space of the model) and labels.

Case Study V−−Characterizing Statistics of Adversarial Examples

In this application, we examine the problem of quantifying the statistical properties of adversarial examples using MARGIN. Adversarial samples (Biggio et al., 2013; Szegedy et al., 2013) refer to examples that have been specially crafted, such that a particular trained model is 'tricked' into misclassifying them. This is done typically by perturbing a sample, sometimes in ways imperceptible to humans, while maximizing misclassification rates. In order to better understand the behavior of such adversarial examples, there have been studies in the past to show that adversarial examples are statistically different from normal test examples. For example, an MMD score between distributions is proposed in (Grosse et al., 2017), and a kernel density estimator (KDE) in (Feinman et al., 2017). However, these measures are global, and provide little insight into individual samples. We propose to use MARGINto develop these statistical measures at a sample level, and study how individual adversarial samples differ from regular samples.

Formulation

As in other case studies, MARGIN constructs a graph, where each node corresponds to an example that is either adversarial or harmless, and the edges are constructed using neighbors in the latent space of the model, against which the adversarial examples have been designed. We consider two kinds of functions in this experiment: 1)

MMD Global

Similar to *Case Study I—Prototypes and Criticisms*, we use the MMD score between the whole set, and the set without a particular sample and its neighbors. This provides a way to capture statistically rarer samples in the dataset; 2)
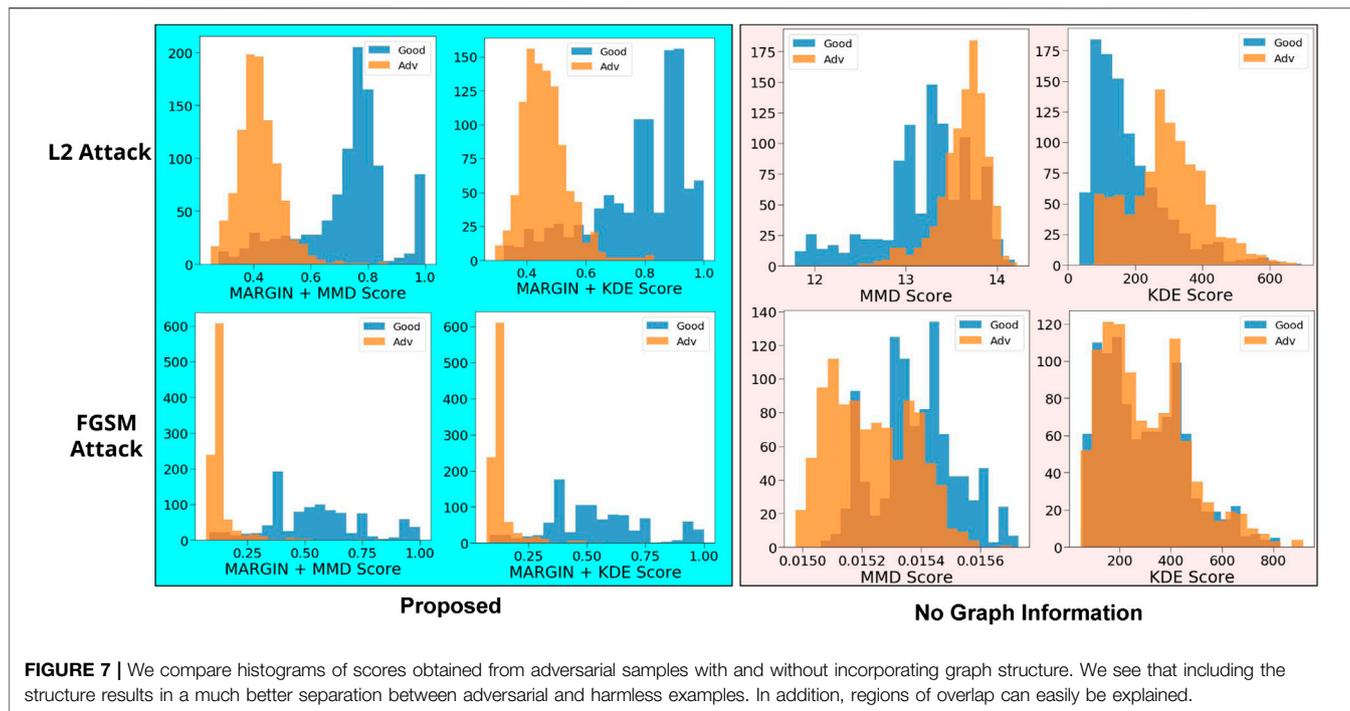
Kernel Density Estimator

We also use the KDE of each sample, as proposed in (Feinman et al., 2017), where we measure the discrepancy of each sample against the training samples from its predicted class. While these measures on their own may not be very illustrative, they are useful functions to determine influences within MARGIN.

Experiment Setup and Results

We perform experiments on 2000 randomly sampled test images from the MNIST dataset (LeCun, 1998), of which we adversarially perturb 1000 images. We measure MARGIN scores using both MMD Global, and KDE, against two popular attacks–the Fast Gradient Sign Method (FGSM) attack (Goodfellow et al., 2014), and the L2-attack (Carlini and Wagner, 2017b). We use the same setup as in (Carlini and Wagner, 2017a), including the network architecture for MNIST. The resulting MARGINscore determined using **Algorithm 1**, is more discriminative, as seen in **Figure 7**. As noted in (Carlini and Wagner, 2017a), the MMD and KDE measures were not very effective against stronger attacks such as the L2-attack. This is reflected to a much lower degree even in our approach, where there is a small overlap in the distributions. We also find that the overlapping regions correspond to samples from the training set that are extremely rare to begin with (like criticisms from *Case Study I—Prototypes and Criticisms*).

Case Study VI—Active Learning on Graphs

To demonstrate the applicability of MARGIN to work with graph structured data, we study the problem of active learning on

FIGURE 7 | We compare histograms of scores obtained from adversarial samples with and without incorporating graph structure. We see that including the structure results in a much better separation between adversarial and harmless examples. In addition, regions of overlap can easily be explained.

graphs, or in other words, generating highly influential samples for a label propagation task. Label propagation is a semi-supervised learning problem, where the task is to propagate labels from a small set of nodes to all the other nodes of the graph. In order to evaluate the samples chosen using our method, we study the test accuracies for varying sizes of the training set. In order to perform the semi-supervised learning, we use the Graph Convolutional Network (GCN) implementation by Kipf and Welling (2017), with 3 graph convolutional layers comprising 16 graph filters each, and a learning rate of 0.01. The rest of the hyper-parameters are those recommended in the GCN implementation[2].

Formulation

Since the attributes are independently defined on each node, they do not contain information about the neighborhoods in the graph and therefore do not directly provide us a notion of influence. Instead, we first embed the attributes using a graph convolutional autoencoder Kipf and Welling (2016), and compute the explanation function f as the as the norm of each latent feature at each node. Next, using MARGIN we compute the influences of the training samples alone, and sort them in decreasing order.

Datasets and Baselines

We consider two popularly used citation network datasets–Cora and Citeseer Sen et al. (2008). The Cora dataset consists of 2,708 nodes and 5,429 edges, while the Citeseer dataset consists of 3,327 nodes and 4,732 edges. The attributes at each node are comprised
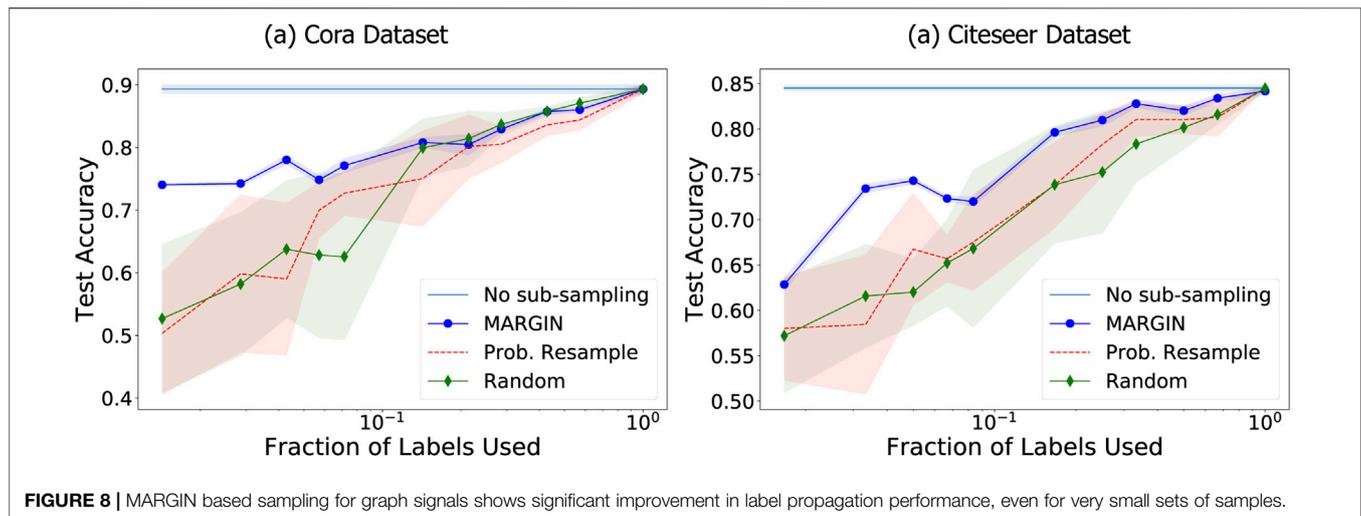
of a sparse bag-of-words feature vector with 3,703 dimensions for Citeseer, and 1,433 dimensions for Cora.

We compare with two baselines: *1) Probabilistic resampling on graphs:* The resampling strategy was proposed in Chen et al. (2017) as a way to efficiently resample dense point clouds. In this approach, the magnitude of the features at each node after a high pass filtering is directly used as a probability of influence at that node, $p(n)$. This is followed by a resampling of the nodes on the graph according to $p(n)$. While it is an effective strategy to resample dense point clouds, it tends to be less reliable for the label propagation experiment, as shown in **Figure 8**. Since we are sampling from a distribution, we sample 10 times, and report the mean and standard deviation. *2) Random sampling:* We also randomly sample from each class on the graph, and repeat this 10 times, while reporting the mean and standard deviation.

Results

In all cases, the accuracy of label propagation is measured on a test set of size 1,000 samples, by training on only 10-100s of samples. **Figure 8** shows the accuracy of label propagation for varying number of training set sizes. It is clear that our proposed sampling achieves state-of-theart performance on the graph. The performance is around 10–15% points higher in accuracy compared to the baseline techniques, especially in small training set regimes. While MARGIN's resampling method is deterministic, we repeat the other baselines 5 times and report average and standard deviation. As we observe in **Figure 8**, the influence computed by MARGIN is significantly better and more stable than the influence obtained by directly using the attributes as the function, as done in the case of probabilistic resampling. It is also interesting to note that this probabilistic method is highly unstable for a very low number of

---

[2]https://github.com/tkipf/gcn

FIGURE 8 | MARGIN based sampling for graph signals shows significant improvement in label propagation performance, even for very small sets of samples.

samples, as it was originally proposed to resample dense point clouds. Finally, random sampling itself is a competitive baseline as the number of samples under consideration is very small.

CONCLUSION

We proposed a generic framework called MARGIN that is able to provide explanations to popular interpretability tasks in machine learning. These range from identifying prototypical samples in a dataset that might be most helpful for training, to explaining salient regions in an image for classification. In this regard, MARGIN exploits ideas rooted in graph signal processing to identify the most influential nodes in a graph, which are nodes that maximally affect the graph function. While the framework is extremely simple, it is highly general in that it allows a practitioner to include rich semantic information easily in three crucial ways–defining the domain (intra-sample vs inter-sample), edges (pre-defined/native/ model latent space), and finally a function defined at each node. The graph based analysis easily scales to very sparse graphs with tens of thousands of nodes, and opens up several opportunities to study problems in interpretable machine learning.

PYTHON IMPLEMENTATION OF MARGIN

The graph analysis based influence estimation in MARGIN is extremely simple, in that it can be implemented using a few lines of python code.

```
import numpy as np
import networkx as nx
import scipy.sparse as sp
'''
Inputs:
adj: adjacency matrix
f: function defined at each node
p: number of hops from each node
for filtering
Output:
I: Influence score per node
'''
def MARGIN(adj,f,p=1):
    G = nx.Graph(adj) #graph object
    N = adj.shape[0] #number of nodes
    degree = G.degree()
    deg = [1./d[1] for d in degree.items()]
    tmp = np.zeros((N,N))
    Dinv = sp.csr_matrix(tmp)
    idx0,idx1 = np.diag_indices(N)
    Dinv[idx0,idx1] = deg
    A_n = np.sqrt(Dinv)*adj*np.sqrt(Dinv)
    P = np.power(A_n,p)
    M = np.sum(P>0,axis=1,dtype=np.float)
    f_0 = np.matrix(f)
    f_1 = (P*f_0)/M
    f_filter = f_1-(P*f_1)/M
    I = np.abs(f_filter)
    I = I/np.max(I)
    return I.A
```

DATA AVAILABILITY STATEMENT

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

AUTHOR CONTRIBUTIONS

FUNDING

REFERENCES

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* 34, 2274–2282. doi:10.1109/tpami. 2012.120

Anirudh, R., Thiagarajan, J. J., Sridhar, R., and Bremer, T. (2017). Margin: uncovering deep neural networks using graph signal analysis. Available at: http://arxiv.org/abs/1711.05407 (Accessed November 15, 2017).

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., et al. (2013). "Evasion attacks against machine learning at test time." in Joint European conference on machine learning and knowledge discovery in databases (Berlin, Germany: Springer), 387–402. doi:10.1007/978-3-642-40994-3_25

Carlini, N., and Wagner, D. (2017b). Towards evaluating the robustness of neural networks, in Security and Privacy (SP), 2017 IEEE Symposium on (Piscataway, NJ: IEEE), 39–57.

Chakraborty, S., Tomsett, R., Raghavendra, R., Harborne, D., Alzantot, M., Cerutti, F., et al. (2017). Interpretability of deep learning models: a survey of results. in Proceedings of the 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced and Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/ SCI), San Francisco, CA, 4–8 August, 2017. Piscataway, NJ: IEEE, 1–6. doi:10.1109/UIC-ATC.2017.8397411

Chen, S., Tian, D., Feng, C., Vetro, A., and Kovačević, J. (2017). "Fast resampling of 3d point clouds via graphs." Available at: http://arxiv.org/abs/1702.06397 (Accessed Febrauary 11, 2017).doi:10.1109/icassp.2017.7952695

Chen, S., Varma, R., Sandryhaila, A., and Kovacevic, J. (2015). Discrete signal processing on graphs: sampling theory. *IEEE Trans. Signal. Process.* 63, 6510–6523. doi:10.1109/tsp.2015.2469645

Doshi-Velez, F., and Kim, B. (2017). "A roadmap for a rigorous science of interpretability." Available at: http://arxiv.org/abs/1702.08608.

Feinman, R., Curtin, R. R., Shintre, S., and Gardner, A. B. (2017). "Detecting adversarial samples from artifacts." Available at: http://arxiv.org/abs/1703. 00410

Fong, R., and Vedaldi, A. (2017). "Interpretable explanations of black boxes by meaningful perturbation." Available at: http://arxiv.org/abs/1704.03296. doi:10. 1109/iccv.2017.371

Gadde, A., Anis, A., and Ortega, A. (2014). Active semi-supervised learning using sampling theory for graph signals. in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining NY, United states: (ACM), 492–501.

Giacinto, N., and Wagner, D. (2017a). Adversarial examples are not easily detected: Bypassing ten detection methods, in Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security Newyork, United states: ACM, 3–14.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). "Explaining and Harnessing Adversarial Examples." Available at: http://arxiv.org/abs/1412.6572.

Grosse, K., Manoharan, P., Papernot, N., Backes, M., and McDaniel, P. (2017). "On the (Statistical) Detection of Adversarial Examples." Available at: http://arxiv. org/abs/1702.06280.

Huang, A., and Moura, J. M. F. (2013). Discrete Signal Processing on Graphs. *IEEE Trans. Signal. Process.* 61, 1644–1656. doi:10.1109/tsp.2013.2238935

Kim, B., Khanna, R., and Koyejo, O. O. (2016). "Examples Are Not Enough, Learn to Criticize! Criticism for Interpretability." in Advances in Neural Information Processing Systems (NIPS), 2280–2288.

Kipf, T. N., and Welling, M. (2017). Semi-supervised Classification with Graph Convolutional Networks, in International Conference on Learning Representations. (Ethiopia, United States: ICLR).

Kipf, T. N., and Welling, M. (2016). "Variational Graph Auto-Encoders." Available at: http://arxiv.org/abs/1611.07308.

Koh, P. W., and Liang, P. (2017). Understanding Black-Box Predictions via Influence Functions, in International Conference on Machine Learning, (Vienna, Austria. ICML).

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet Classification with Deep Convolutional Neural Networks, in Advances in neural information processing systems. 1097–1105.

LeCun, Y. (1998). The Mnist Database of Handwritten Digits. Available at: http: //yann lecun.com/exdb/mnist/.

Lipton, Z. C. (2016). "The Mythos of Model Interpretability." Available at: http:// arxiv.org/abs/1606.03490.

Lundberg, S. M., and Lee, S. I.. (2017). "A Unified Approach to Interpreting Model Predictions." in Advances in neural information processing systems, 4765–4774.

Metsis, V., Androutsopoulos, I., and Paliouras, G. (2006). Spam Filtering with Naive Bayes-Which Naive Bayes?. *CEAS* 17, 28–69.

Pan, P., Swaroop, S., Immer, A., Eschenhagen, R., Turner, R. E., and Khan, M. E. (2020). "Continual Deep Learning by Functional Regularisation of Memorable Past." Available at: http://arxiv.org/abs/2004.14070.

Pascucci, V., Scorzelli, G., Bremer, P.-T., and Mascarenhas, A. (2007). Robust On-Line Computation of Reeb Graphs: Simplicity and Speed papers. (NY, United states: ACM SIGGRAPH), 58. doi:10.1145/1275808.1276449

Pesenson, I. (2008). Sampling in Paley-Wiener Spaces on Combinatorial Graphs. *Trans. Amer. Math. Soc.* 360, 5603–5627. doi:10.1090/s0002-9947-08-04511-x

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why Should I Trust You?: Explaining the Predictions of Any Classifier, in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. NY, United states: ACM, 1135–1144.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). Imagenet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* 115, 211–252. doi:10.07/s11263-015-0816-y

Schwab, P., and Karlen, W. (2019). Explain: Causal Explanations for Model Interpretation under Uncertainty. in Advances in Neural Information Processing Systems. 10220–10230.

Schwab, P., Miladinovic, D., and Karlen, W. (2019). Granger-causal Attentive Mixtures of Experts: Learning Important Features with Neural Networks, in Proceedings of the AAAI Conference on Artificial Intelligence, CA, United States. AAAI, 4846–4853. doi:10.1609/aaai.v33i01.33014846

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D.. (2017). "Grad-cam: Visual Explanations from Deep Networks via Gradient-Based Localization". in. Proceedings of the IEEE international conference on computer vision, 22–29 October, 2017. (Venice, Italy IEEE), 618–626.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. (2008). Collective Classification in Network Data. *AIMag* 29, 93. doi:10.1609/aimag. v29i3.2157

Shrikumar, A., Greenside, P., and Kundaje, A. (2017). Learning Important Features through Propagating Activation Differences. *Proc. Machine Learn. Res.* 70, 3145–3153.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2013). The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal. Process. Mag.* 30, 83–98. doi:10.1109/msp.2012.2235192

Simonyan, K., Vedaldi, A., and Zisserman, A. (2013). "Deep inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps." Available at: http://arxiv.org/abs/1312.6034. doi:10.5244/c.27.8

Sundararajan, M., Taly, A., and Yan, Q. (2017). "Axiomatic Attribution for Deep Networks." Available at: http://arxiv.org/abs/1703.01365.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., et al. (2013). "Intriguing Properties of Neural Networks." Available at: http://arxiv.org/abs/1312.6199.

Thiagarajan, J. J., Narayanaswamy, V., Anirudh, R., Bremer, P.-T., and Spanias, A. (2020). "Accurate and Robust Feature Importance Estimation under Distribution Shifts." Available at: http://arxiv.org/abs/2009.14454 doi:10.2172/1670557

Zeiler, M. D., and Fergus, R. (2014). Visualizing and Understanding Convolutional Networks. European conference on computer vision. Berlin, Germany: Springer, 818–833. doi:10.1007/978-3-319-10590-1_53

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2014). "Object Detectors Emerge in Deep Scene Cnns." Available at: http://arxiv.org/abs/1412.6856.

Conflict of Interest: Author RS portion of the work was done during a summer internship at LLNL. He is currently employed by the company Walmart Labs. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.