MARGIN: Uncovering Deep Neural Networks using Graph Signal Analysis

Seminar: Explainable Machine Learning

Thorsten Bernhard Wünsche (3148266)

September 10, 2018

Supervisor: Dr. Ullrich Köthe

Contents

1 Introduction

While most machine learning algorithms can deliver impressive results, even their creators often don't entirely understand why certain predictions were made. Explaining the results may not only enable research to improve more specific parts of these algorithms. It could also give end users more confidence in the predictions and may even be required by law.

This leads to a wide array of interpretability questions, such as 'Why did a sample receive a certain classification?' or 'Which training samples were particularly useful?'. Rather than use specialized tools for each individual task, Rushil Anirudh, Jayaraman J. Thiagarajan, Rahul Sridhar and Peer-Timo Bremer suggest an approach to solve all these questions in the paper 'MARGIN: Uncovering Deep Neural Networks using Graph Signal Analysis'[1], by constructing a graph specific to each task and finding its most influential nodes.

In chapter 2, the basic approach is explained and summarized. The five different case studies are presented in chapter 3. Chapter 4 summarizes this report.

2 Generic Protocol

MARGIN is not a tool to be used out of the box, but a generic protocol, which has to be adapted to each dataset, neural network and task individually. The approach itself remains mostly unchanged and consists of the five steps shown in figure 2.1.

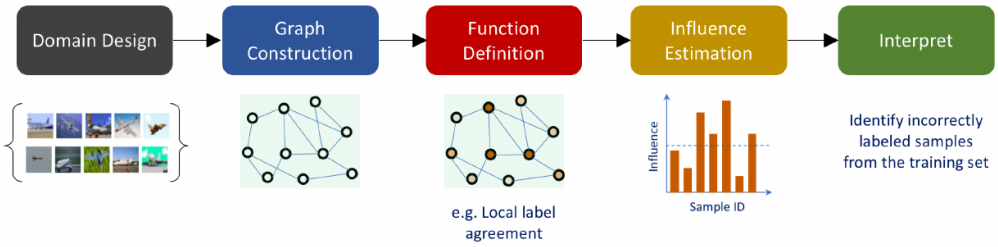


Figure 2.1: The five steps of MARGIN, here used to find incorrectly labeled samples.

2.1 Domain Design

MARGIN uses a graph to solve interpretability tasks. The first step towards the creation of this graph is choosing the correct domain. This determines, what the nodes of the graph represent. The output of MARGIN is always a list of nodes in order of their influence score, therefore the domain design determines the answer MARGIN can give.

The most common domain is the entire dataset (or its subsets), with each sample being represented by a single node. This will yield a ranked list of samples, which is for instance useful when searching for influential or mislabeled samples.

It is also possible to define the domain as a single sample, turning its attributes into nodes. Using superpixels of one image as nodes can be used to create a saliency map, which can explain the classification of that sample.

While these are the most common domains and the only ones covered in the case studies, with some creativity other domains could be used to solve new interpretability tasks.

2.2 Graph Construction

Using the nodes defined in section 2.1, the next step is to connect them into an undirected weighted graph. For most purposes a k-nearest neighbor domain graph is sufficient. Choosing the metric for the graph is important, as it determines the target of the evaluation: By judging the closeness of nodes by their features or other knowledge about

the domain, only the dataset is evaluated. This should be used for instance when searching for mislabeled samples. To gain knowledge about a particular neural network, the metric can use the samples latent representation in the network. This is the only step in MARGIN, that connects it to the machine learning model. Therefore any model specific analysis has to include some information from the model in the graphs construction.

2.3 Function Definition

The next step is to find an explanation function that measures how well each node supports our hypothesis. The hypothesis depends on the task: When looking for mislabeled samples for example, we assume that most or all of its neighbors have different labels. This hypothesis can be encoded by the local label agreement function.

The nodes that most define this function will have a higher influence score. This step requires some creativity, but given a suitable explanation function, MARGIN can be used for any interpretability task.

2.4 Influence Estimation

The result of MARGIN is a list of nodes ranked by their influence score. This score is determined by how much a particular node defines the explanation function. To calculate this influence score, MARGIN uses graph signal analysis, which is an active field of research itself. The influence estimation requires only a basic algorithm, though any more advanced findings from graph signal analysis can be used to enhance this step.

Intuitively, the influence is estimated by calculating the values of the explanation function for the entire graph. Then, the nodes are replaced one at a time by a linear combination of their neighbors and the explanation function is recomputed. A low influence node can be replaced without any large changes, but the more a node defines the function, the more its values will differ. The larger the difference, the higher the influence score of the node.

2.5 Interpretation

The final step involves interpreting the results. Depending on the hypothesis, this step can be very straight forward, or it may require an additional strategy to make the influence scores easily understandable.

3 Case Studies

This chapter presents five case studies to illustrate the variety of MARGIN's applications.

3.1 Prototypes and Criticisms

Prototypes are samples, that are especially representative of their class. On the other end of the spectrum are criticisms. To find both types of samples using MARGIN, one can use a simple neighborhood graph based on Euclidean distance, as this analysis concerns only the dataset. Each sample in the dataset is represented by one node, the explanation function Maximum Mean Discrepancy is used to estimate how well a node represents its label.

This is done by comparing the mean value of the complete dataset (global) or all samples of the corresponding class (local) with the mean value of the same set after removing a node and all of its neighbors. The hypothesis is, that prototypes should have no or very little influence on the mean value of the class, whereas criticisms would show a significant difference.

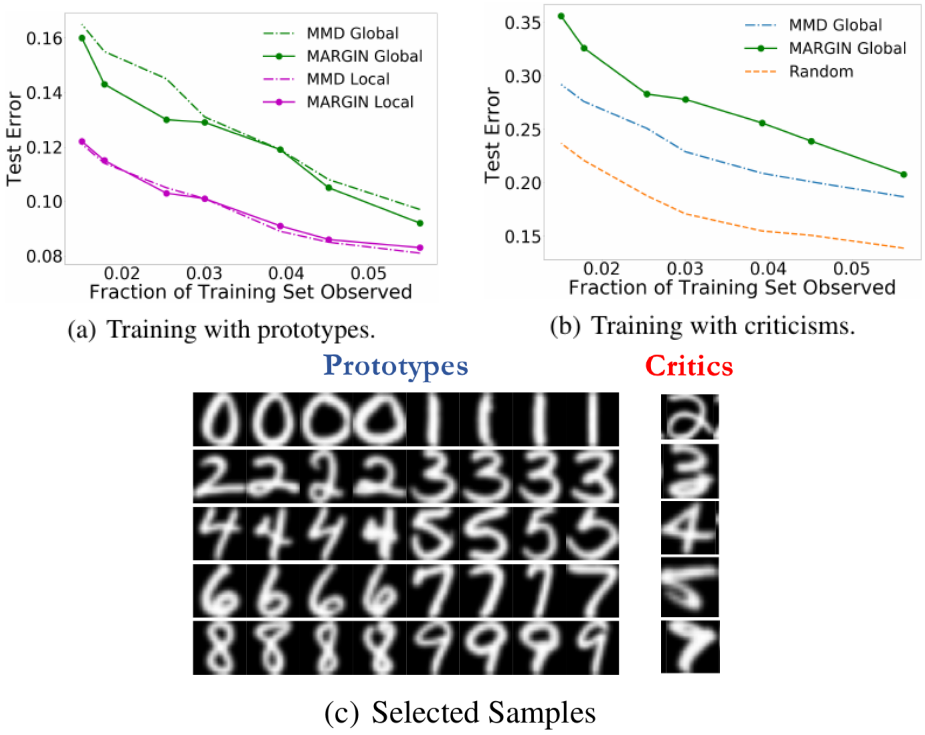
To evaluate the quality of these prototypes and criticisms, they are used as training samples for a simple 1-nearest neighbor classifier. The dataset is the USPS handwritten digits data. As prototypes are supposed to generalize well, a low error-rate is desired. Criticisms on the other hand should result in a higher error rate, the better they are chosen.

As can be seen in figure 3.1, MARGIN shows some mild improvement compared to regular MMD when choosing prototypes in the global case and significant improvements in the choice of criticisms. Another advantage is, that MARGIN computes criticisms and prototypes at the same time, by simply picking the top and bottom of a ranked list. Regular MMD on the other hand requires two separate runs.

3.2 Explanations for Image Classification

The second task is to explain why an image has been classified in a certain way. This is typically done though saliency maps, highlighting relevant areas of the image. Since MARGIN has no direct way to interact with an image, super-pixels chosen by a separate method are used as nodes in the graph. By examining the difference in class probability with and without these super-pixels a dense saliency map is created (figure 3.2 second image from the left).

Note, that up to this point, MARGIN was not involved in the process. It is now used to add a sparsity requirement: The hypothesis is, that smaller explanations are more



(a) Training with prototypes.

(b) Training with criticisms.

(c) Selected Samples

Figure 3.1: Using MARGIN to select prototypes and criticisms

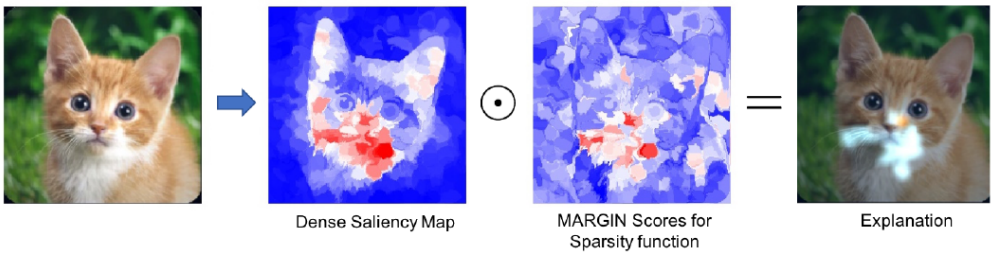


Figure 3.2: Constructing a saliency map using MARGIN.

useful, to prevent them from gaining a high score simply due to covering more of the image.

The graph is created based on the super-pixels impact on the probability density, the explanation function is the relative size of the current node to the largest super-pixel in the graph.

This way, MARGIN can be used to enhance existing explanation attempts by enforcing sparsity. The quality of the results are very dependent on the potential explanations chosen as nodes, as MARGIN cannot create these on its own.

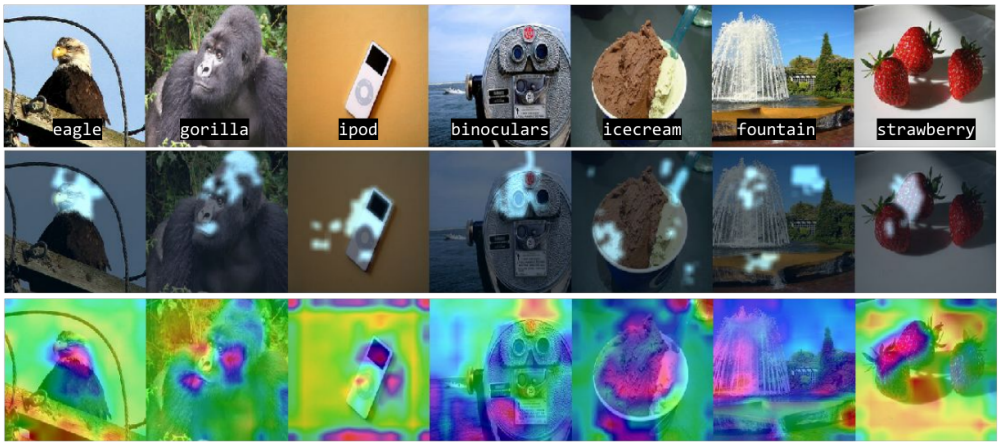
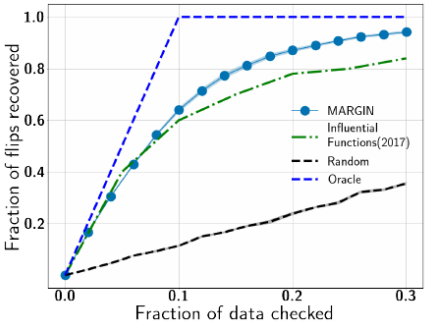


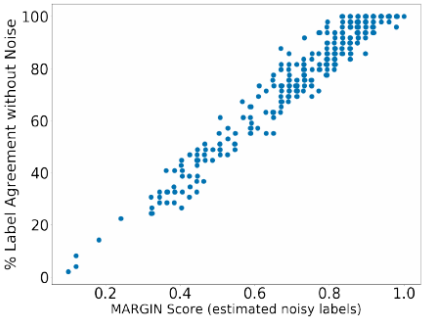
Figure 3.3: Explaining image classification using MARGIN (middle row) and Grad-CAM (bottom row)

3.3 Detecting Incorrectly Labeled Samples

This task has already be used as an example in chapter 2. It involves a two-class dataset (Enron Spam Classification dataset), in which some percentage of samples has their labels flipped. Using a bag of words, the neighborhoods for the graph are determined. Local label agreement is used as an explanation function, the hypothesis being, that corrupted samples will most likely be in a cluster of their real class. The more their neighbors disagree with a samples label, the more likely it is to be mislabeled.



(a) Detecting label flips in the Enron dataset (Metsis et al., 2006).



(b) Examining the incorrectly labeled samples with their influence score.

Figure 3.4: Using MARGIN to find incorrectly labeled samples

Figure 3.4 (a) shows, that MARGIN is almost 10 percentage points ahead of the baseline. In (b), the correlation between the MARGIN score and the local label agreement without incorrect labels can be seen.

3.4 Interpreting Decision Boundaries

MARGIN can be used to find decision boundaries, by selecting samples, that are hard to classify for a neural network. The process is very similar to case study 3.3. The differences are, that the labels in this case study are all correct, and the graph is constructed using latent representations from AlexNet pre-trained on ImageNet to distinguish Tabby Cats from Great Danes as well as a CNN trained to tell 0 and 6 from MNIST apart. By including information directly from the model, we are no longer analyzing the dataset, but the model itself.

Local label agreement can still be used, as samples close to the decision boundary will naturally have more disagreement than those that are easy to classify. Figures 3.5 and 3.6 show the most confusing samples for their respective networks. In the case of the animals, obscured faces and unusual poses cause th most confusion. In the case of the numbers, most of the chosen samples would require guessing even from a human.

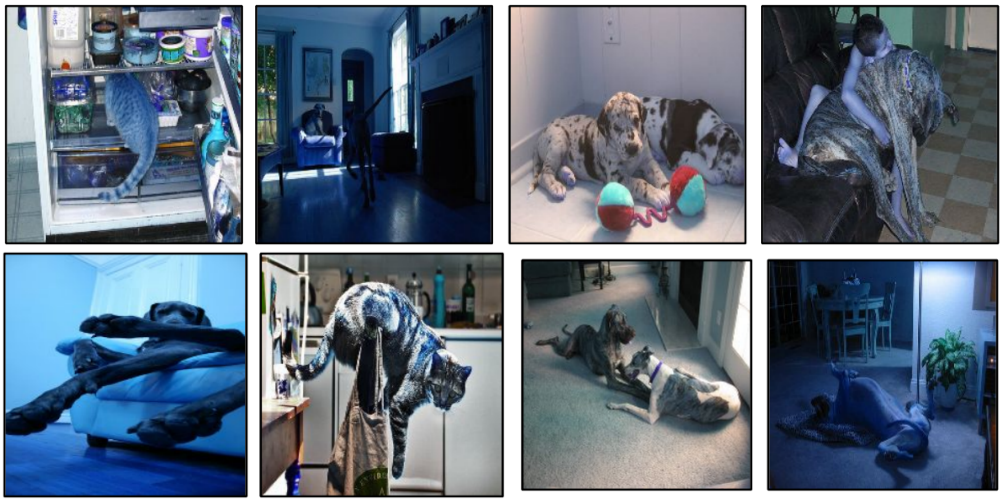


Figure 3.5: Most confusing samples for AlexNet pre-trained on ImageNet for the Tabby Cat and Great Dane classes

Figure 3.6: Most confusing samples for a CNN trained on MNIST for the classes 0 and 6

3.5 Characterizing Statistics of Adversarial Examples

Adversarial samples are specifically crafted to trick a particular machine learning model, usually by locating the closest decision boundary and changing the sample very slightly

to move it across. On a global level, the properties of these samples can be determined by methods such as MMD score between distributions and Kernel Density Estimation. MARGIN can be used to apply these methods at the level of individual samples.

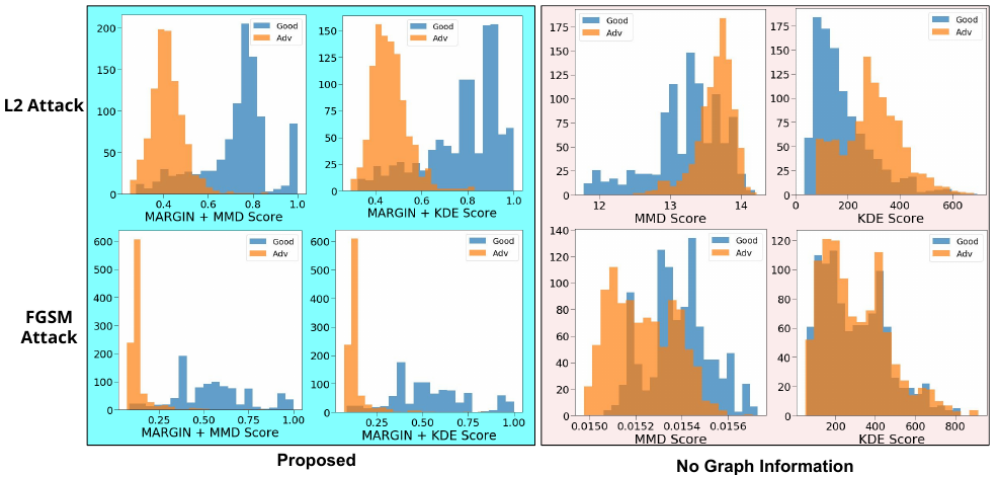


Figure 3.7: Comparison of statistical scores to find adversarial examples with and without the graph provided by MARGIN

Using both harmless and adversarial samples as nodes and latent representations of the model, against which the samples were designed, a graph can be constructed. Using MMD and KDE as explanation functions makes it possible to estimate the influence of individual samples.

By enhancing these two existing methods with the graph structure of MARGIN, there is significantly less overlap between harmless and adversarial samples, as shown in figure 3.7. The overlap can also be explained, as it corresponds to rare cases, similar to critics in section 3.1.

4 Conclusion

The protocol MARGIN can be used to deal with a wide array of common interpretability tasks and can even be used to approach entirely new questions. It can analyze both datasets and machine learning models. While neural networks are the focus, any model that can provide latent representations of samples can be encoded into the graph. There is also no need to train MARGIN and as such it can be used on models that are already trained. The lack of particularly time-consuming calculations makes it a very fast method.

MARGIN is both flexible and delivers competitive results, the only downside is the customization required: Each task requires its own explanation function, each dataset and model a new graph. Once the modeling stage is complete though, the same analysis can easily be performed again. Due to this, MARGIN benefits heavily from examples and case studies, which provide useful explanation functions and graph designs.

Overall, MARGIN is both fast and flexible. The case studies in the original paper provide explanation function and graph modeling strategies, that cover some common tasks and the ability to use MARGIN on completely new problems makes it a useful tool for researchers. Even end users might benefit from MARGIN, provided the interpretation of the influence scores presents them in a suitable format.

Bibliography

[1] Rushil Anirudh, Jayaraman J. Thiagarajan, Rahul Sridhar, Peer-Timo Bremer, *MARGIN: Uncovering Deep Neural Networks using Graph Signal Analysis* `https://arxiv.org/abs/1711.05407`.