Exploring Explainability Methods for Graph Neural Networks

Harsh Patel HARSH.PATEL@IITGN.AC.IN
Computer Science and Engineering

Shivam Sahni SHIVAM.SAHNI@IITGN.AC.IN
Computer Science and Engineering

Abstract

With the growing use of deep learning methods, particularly graph neural networks, which encode intricate interconnectedness information, for a variety of real tasks, there is a necessity for explainability in such settings. In this project, we demonstrate the applicability of popular explainability approaches on Graph Attention Networks (GAT) for a graph-based super-pixel image classification task. We assess the qualitative and quantitative performance of these techniques on three different datasets and describe our findings. The results shed a fresh light on the notion of explainability in GNNs, particularly GATs.

Keywords: Graph Neural Networks, Explainable AI, Super-pixel Image Classification

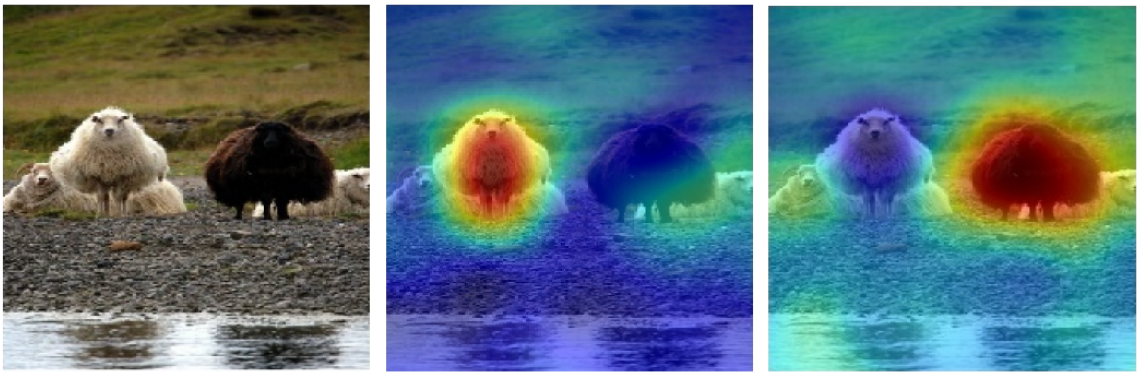
1. Introduction

Deep Neural Networks have been criticized for being black boxes of information. To explain the results generated by these models, a variety of techniques have been used to enhance their interpretability. For example in Figure 1 (b),(c), the heat maps show the explainability results on a Convolutional Neural Network (CNN), ResNet-50, which performs a classification task (Petsiuk et al. (2018)). As can be seen from the importance map of cow, the model confuses the black sheep for a cow and thus mispredicts its presence in the image. Such answers, obtained using explainability methods, can be useful in rectifying the mistakes of a model. Graph Neural Networks (GNNs) are deep learning models which have proven to be a powerful tool for exploiting graphical information. They do so by combining the node feature information with the graph structure by passing messages along the edges of the graph which makes GNN a complex model. GNN explainability can increase trust in GNNs, improve transparency of model's working and allow users to identify and improve GNN's mistakes.

Traditional convolutional networks use kernels that are limited to domains which use rectangular grids such as 2D images. However, the more complicated sources of visualisations like panoramas capture 360 degree view of a place. Additionally, point cloud classification relies on spatially unstructured data which again cannot be represented through a rectangular grid. We can exploit GNNs to solve the above two problems. The use of graphs allows us to model images using pixel level representation or super pixel level representations of images, which are a group of pixels that share common characteristics (like pixel

intensity). Superpixels have a perceptual meaning since pixels belonging to a given superpixel share similar visual properties. They provide a convenient and compact representation of images that can be very useful for computationally demanding problems.

In this project, we choose to perform superpixel image classification with a broader aim to provide explainability to the GNN predictions. We then compare the explainability results on varying the number of superpixels in the image. Finally, we look at a way to measure these explainability results known as Fidelity.



(a) Sheep - 26%, Cow - 17% (b) Importance map of '*sheep*' (c) Importance map of '*cow*'

Figure 1: CNN explainability visualisation from Petsiuk et al. (2018)

2. Related Work

The difficulty in interpretation and explanation of performance of deep neural networks has been a long standing problem (Pope et al. (2019)). Several explainability methods have been developed for deep neural networks, specially CNNs (Simonyan et al. (2014), Selvaraju et al. (2020)). Methods like Gradient-weighted Class Activation Mapping (Grad-CAM), and Excitation Back-Propagation (EB) have proven to be quite effective on CNNs for image classification tasks. More recently, the work (Pope et al. (2019)) demonstrates tools for explaining the performance of Graph Convolutional Neural Networks (GCNNs) on visual scene graphs and molecular graphs. By considering each pixel as a node, graph-based approaches can be directly applied to images. Lower-level image representation (e.g., Super-pixel segmentation) can, on the other hand, be more effective in classification tasks due to the smaller graphs that are formed. Avelar et al. (2020) present a methodology for super-pixel image classification using Graph Attention Networks.

3. Problem Description

A superpixel is a collection of pixels that share similar features like color, pixel intensity, etc. These superpixels give a simple and compact visual representation of an image without losing its perceptual meaning. Many fundamental tasks in computer vision can be applied to these over-segmented and simplified images, including image classification (Long et al.

(2021)). These superpixels can be generated using several approaches like Simple Linear Iterative Clustering (SLIC, Achanta et al. (2012)), Superpixels Extracted via Energy-Driven Sampling (SEEDS, den Bergh et al. (2013)). We use the SLIC algorithm for this task which uses an iterative approach to perform the segments for a desired number of equally sized superpixels. Figure 2(b) shows the superpixeled version of the image 2(a). This superpixeled image is further formulated into a graph structure by a simple region adjacency graph generation approach. Each superpixel segment is representative of a node in the graph and all the neighbouring segments form the edges with this node. Figure 2(c) shows the generated graph for the image. This entire graph acts as an input to our GNN model which performs the task of superpixel image classification.

(a) (b) (c)

Figure 2: Superpixel graph generation - a) shows the original image, b) shows the superpixels nodes formed using the SLIC algorithm, and c) neighbouring superpixel nodes form an edge which are shown in blue and the nodes are shown in red

4. Dataset

We use the following three standard classification datasets in our experimentation. Each of these datasets have 50,000 training, 10,000 testing examples and include 10 classes.

- MNIST (Lecun et al. (1998))
- Fashion-MNIST (Xiao et al. (2017))
- CIFAR-10 (Krizhevsky (2009))

5. Model: Multi-headed Graph Attention Network (GAT)

We use a multi-headed graph attention network to perform the task of super pixel image classification. Similar to a GNN, this model generates contextualised node embeddings based on the information that its constitutes (Adjacency matrix, Node Features, Edge Features of the input graph). Graph attention network is a generalised version of a Graph Convolution Network (Kipf and Welling (2017)), which allows the graphs nodes to generate their embeddings by aggregating information from each of their neighbors.

GATs expand this basic aggregation method by using self-attention mechanisms (Vaswani et al. (2017)) that learn the relative importance of each neighbor's contribution, thus improving its learning capacity. An attention coefficient is calculated for each edge in the input

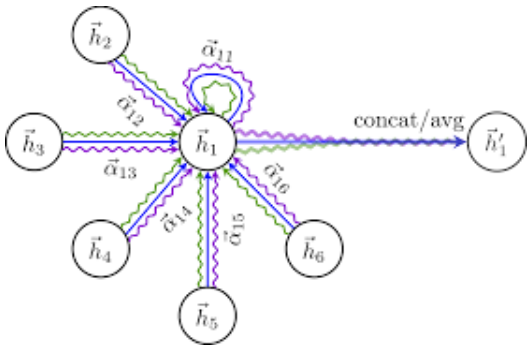


Figure 3: Attention mechanism on node 1 by its neighbourhood (Velickovic et al. (2018))

graph, so as to improve the aggregation step. Also, similar to the Vaswani et al. (2017) transformer architecture, multi-headed attentions are used in the Graph attention networks for a stabilised learning. Figure 3 illustrates how the attention mechanism aggregates the contribution of each neighbor for the node embedding $\vec{h}_1$.

The input to our GAT layer (Figure4) is a set of node features h, where N is the number of nodes, and F is the number of features in each node.

$$h = \left\{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_N\right\}, \vec{h}_i \in \mathbb{R}^F$$

A linear layer is then used to transform the input features into high-level features using a weight matrix W. Now, for each edge (i, j) in our input graph, we use a linear attention mechanism (a) to compute its attention coefficient (e_{ij}) that represents its importance in the message passing step of this graph neural network. The input to this attention mechanism is the concatenation of the node embedding of the involved nodes in the edge.

$$e_{ij}^{(l)} = ReLU\left(\vec{a}^{(l)^T}(W^{(l)}h_i^{(l)} \| W^{(l)}h_j^{(l)})\right), W^{(l)} \in \mathbb{R}^{F' \times F}$$

These attention coefficients, for each node i, are normalised using a softmax operation across all its neighbors (js).

$$\alpha_{ij}^{(l)} = \frac{\exp\left(e_{ij}^{(l)}\right)}{\sum_{k \in \mathcal{N}(i)} \exp\left(e_{ik}^{(l)}\right)}$$

The output features for each node are then aggregated using a weighted linear combination of its neighbors' features (the weight being the calculated attention coefficients). In case of multiple heads, we concatenate the scaled attention scores obtained from each independent attention mechanism to generate the final output embeddings.

$$h_i^{(l+1)} = ||_{k=1}^{K} \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right)$$

We thus obtain highly contextualised embeddings for each node in the graph which can be further be passed through feed-forward layers to achieve the task of classification.

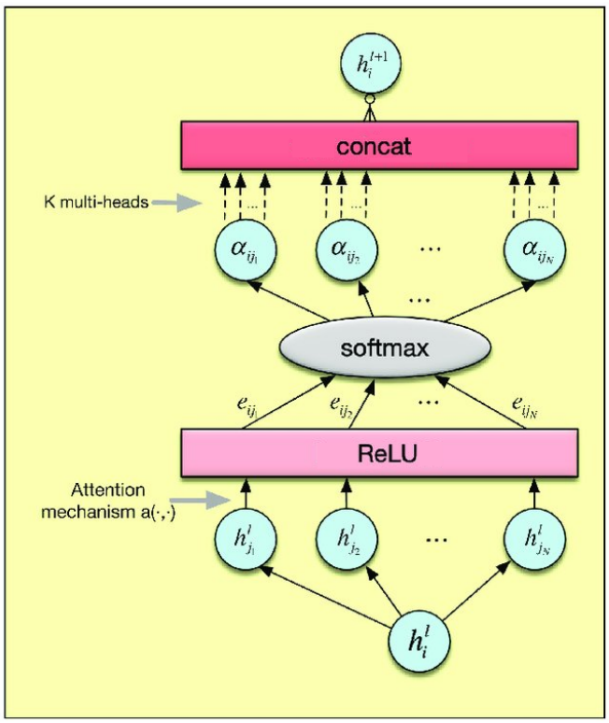


Figure 4: GAT Layer, figure inspired from Zhang et al. (2019)

Dataset	Accuracy (%)
CIFAR-10	49.73
MNIST (pretrained)	97.8
Fashion MNIST (pretrained)	89.6

Table 1: Model Accuracies

6. Experimental Settings: Training and Evaluation

For each dataset, we set the number of desired super-pixel segments to be 75 and train them on a three-headed Graph Attention Network. All experiments are conducted on GPUs. The training on CIFAR-10 dataset for 250 epochs takes around 10 days. Figure 5 shows the variations in classification loss, training and validation accuracy with number of epochs. Table 1 shows the testing accuracies obtained on the three datasets using their corresponding trained models. We use these trained models to perform a variety of explainability methods in an attempt to retrieve useful interpretations and insights on their performance and learning behavior.

7. Explainability Methods

We utilise four different explainability methods which were originally intended to work on CNN and can be extended to GCN (Pope et al. (2019)). The methods are Contrastive Gradient-based Saliency Maps (CGSM) Simonyan et al. (2014), Class Activation Mapping

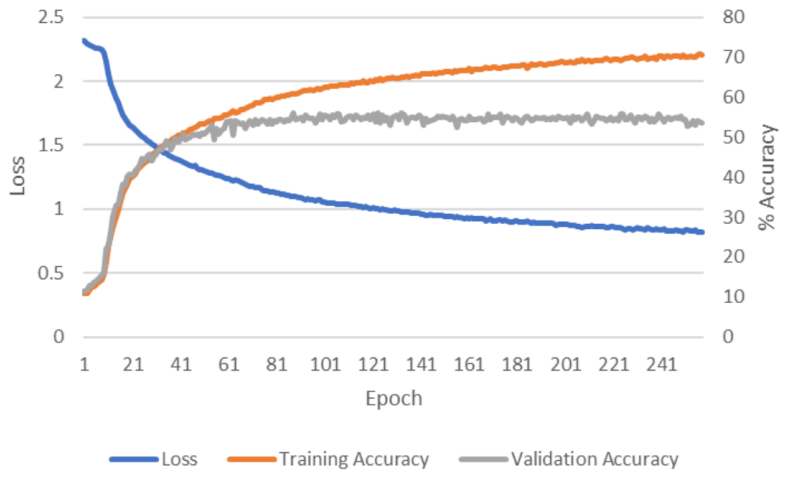


Figure 5: Training curves on CIFAR-10 dataset

(CAM) Zhou et al. (2015), Grad-Class Activation Mapping (Grad-CAM) Selvaraju et al. (2020), Guided-Backpropagation (GBP) Springenberg et al. (2014).

7.1 Contrastive Gradient-based Saliency Maps (CGSM)

CGSM is the most straightforward way to implement explainability. This method relies on the derivative of the output of the model with respect to the input image. We apply ReLU to the gradients to discard negative gradients.

$$L^c_{\text{Gradient}} = \left\| \text{ReLU}\left(\frac{\partial y^c}{\partial x}\right) \right\|$$

where y^c is the score of class c and x is the input. It is argued that CGSM represents noise more than signals.

7.2 Class Activation Mapping (CAM)

CAM is based on the motivation that the class features at the last convolutional layer are more meaningful. CAM has an architectural constraint in which it requires the layer immediately before the final softmax classifier to be a convolutional layer followed by a global average pooling layer. Assume, F_k to be the k^{th} feature map of the output from last convolutional layer. The global average pooling is defined as:

$$e_k = \frac{1}{Z} \sum_i \sum_j F_{k,i,j}$$

The global average pooled output is then passed to a softmax classifier to get the output class scores.

$$y^c = \sum_k w^c_k e_k,$$

The importance of a pixel in CAM is defined as:

$$L^c_{CAM}[i,j] = \text{ReLU}\left(\sum_k w^c_k F_{k,i,j}\right)$$

7.3 Grad-Class Activation Mapping (Grad-CAM)

Grad-CAM relaxes the assumption the architectural restriction put forward by CAM by replacing the feature map weights with the back propagated gradients.

$$\alpha^c_k = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial F_{k,i,j}}$$

$$L^c_{Grad-CAM}[i,j] = \text{ReLU}\left(\sum_k \alpha^c_k F_{k,i,j}\right)$$

7.4 Guided-Backpropagation (GBP)

The motivation behind Guided-Backpropagation is to discard neagtive gradients during backpropagation as negative gradients play the role of suppressing the output. As shown in Fig 6 we compute gradients, then zero out the negative ones and continue to backpropagate.

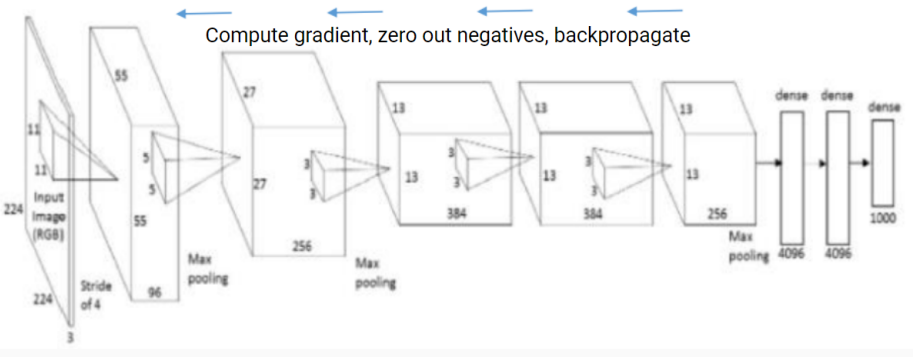


Figure 6: Intuitive explanation of guided backpropagation (GBP)

8. Explainability Quantification: Fidelity

We use the Fidelity score as the quantification metric for all the explanaiblity methods. The key idea behind Fedality is that the occlusion of important features, discovered through the obtained explanations, on the original image would lead to a reduction in classification accuracy. It is defined as the difference in accuracy obtained by excluding all nodes with a saliency value greater than 0.01 (on a scale 0 to 1). Figure 7 shows the fidelity scores of each of the above discussed explainability methods with varying threshold on the MNIST dataset. We observe that out of all the 4 methods, Guided-Backpropagation (Section 7.4) shows the best fidelity, i.e. occluding its strongly identified regions from the image, leads to higher number of misclassifications. We also observe that on increasing the threshold (i.e., occluding lesser number of nodes), the fidelity score decreases because occluding less number

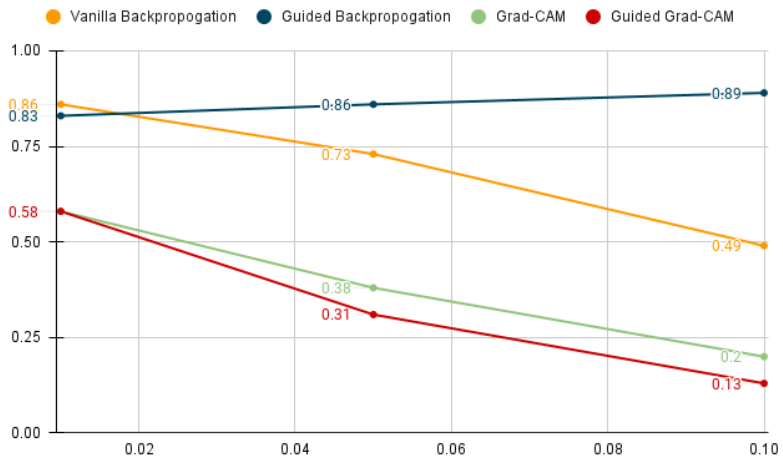


Figure 7: Fidelity metric results on the four explainability methods by varying occlusion threshold

of nodes would generally lead to a very less change in the classification accuracy. However, in Guided-Backpropagation, the fidelity scores do not decrease with increasing threshold. The gradients in this method are backpropogated only through the relevant neurons, which means that the contributing pixels have a direct and significant correlation with the classification output. Thus, even on occluding lesser number of nodes, the misclassifications are still observed because of the high saliency values of the occluded nodes.

9. Results

In this section, we show explainability results on the three datasets and provide a qualitative analysis of the results obtained. The figure 8 shows the visualisations from all the four methods on an example from MNIST dataset representing the number '6'. We observe that Guided Backpropagation seems to capture all the nodes involved in making the number look like the number '6', and hence performs the best. We also observe that the Vanilla Backpropagation method has the maximum noise and the Guided Grad-CAM has the minimum noise, however they both fail to fully capture all the important nodes.

The figure 10 shows the visualisations by varying the number of superpixels in the image. From the figure we observe that as the total number of superpixel increases from 25-150, the total amount of region enclosed outside '0' decreases, thus we can say that the noise in prediction decreases.

Figure 11 shows an exmaple image from a MNIST dataset super-pixeled with varying number of segments. We also compare the % classification accuracy of the GAT model with varying number of nodes in the input graph. We observe that using a very few number of nodes leads to loss of information which leads to the classification accuracy of 35% on the MNIST dataset, whereas increasing the number of nodes to 150, leads to increased

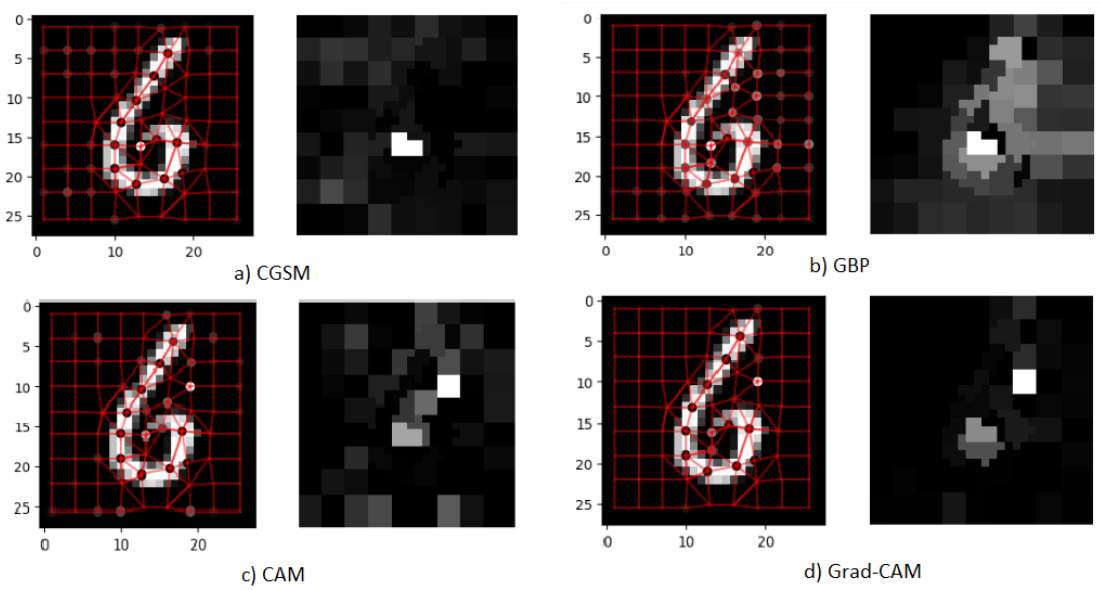


Figure 8: Visualizations of the four explainability methods on '6' from MNIST dataset

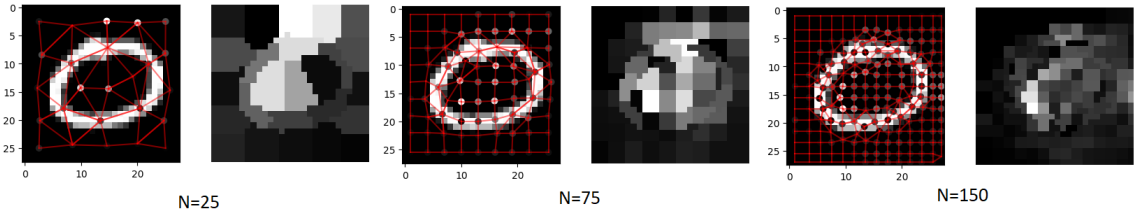


Figure 9: Visualizations

Figure 10: Visualization of GBP by varying the number of superpixels on '0' from MNIST dataset

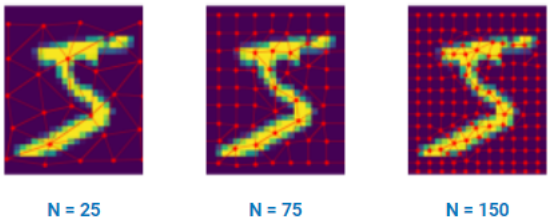


Figure 11: Variation in the superpixel graph on varying the number of desired superpixels

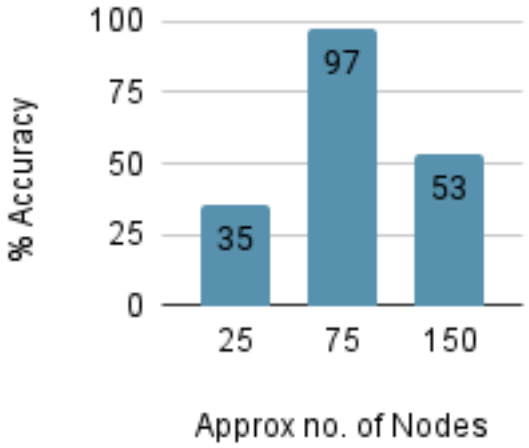


Figure 12: Accuracy vs Varying no. of superpixels

complexity in message passing and also leads to the formation of super pixels that do not carry forward a perceptual meaning. This again leads to a decrease in the overall accuracy.

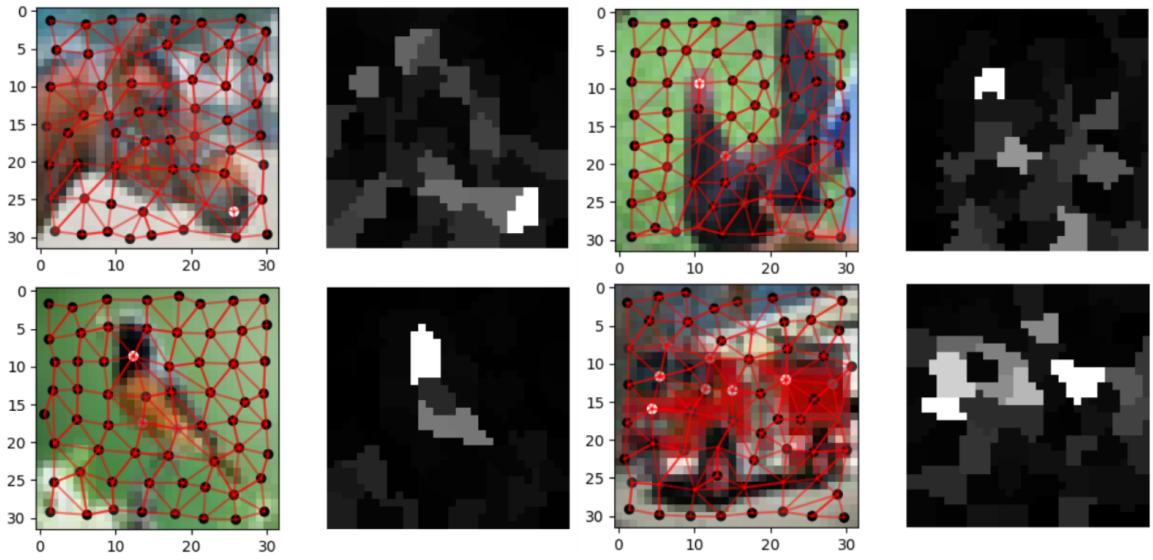


Figure 13: Visualizations Results on CIFAR 10 dataset

The figure 13 shows some explainability results on the CIFAR-10 dataset. We observe that the silhouette of the explainability visualizations match the objects depicted in the image. This provides assurance on the quality of the explainability visualizations we get.

10. Conclusion

In this project, we performed superpixel image classification task on three standard classification datasets and visualized various explainability methods in Graph Neural Network settings, particularly Graph Attention Networks (GATs). We characterized the explanations with the fidelity (quantification metric) and found that Guided-Backpropagation method shows the best performance amongst all the methods. We also provide performance analysis of the explainability methods with varying number of superpixels in the images during the classification task.

References

Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. doi: 10.1109/TPAMI.2012.120.

Pedro H. C. Avelar, Anderson R. Tavares, Thiago L. T. da Silveira, Cláudio R. Jung, and Luís C. Lamb. Superpixel image classification with graph attention networks, 2020. URL `https://arxiv.org/abs/2002.05544`.

Michael Van den Bergh, Xavier Boix, Gemma Roig, and Luc Van Gool. SEEDS: superpixels extracted via energy-driven sampling. *CoRR*, abs/1309.3848, 2013. URL `http://arxiv.org/abs/1309.3848`.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907 [cs, stat]*, Feb 2017. URL `http://arxiv.org/abs/1609.02907`. arXiv: 1609.02907.

A. Krizhevsky. Learning multiple layers of features from tiny images. *undefined*, 2009. URL `https://www.semanticscholar.org/paper/Learning-Multiple-Layers-of-Features-from-Tiny-Krizhevsky/5d90f06bb70a0a3dced62413346235c02b1aa086`.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998. ISSN 1558-2256. doi: 10.1109/5.726791.

Jianwu Long, Zeran yan, and Hongfa chen. A graph neural network for superpixel image classification. *Journal of Physics: Conference Series*, 1871(1):012071, Apr 2021. ISSN 1742-6588, 1742-6596. doi: 10.1088/1742-6596/1871/1/012071. URL `https://iopscience.iop.org/article/10.1088/1742-6596/1871/1/012071`.

Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models, 2018. URL `https://arxiv.org/abs/1806.07421`.

Phillip E. Pope, Soheil Kolouri, Mohammad Rostami, Charles E. Martin, and Heiko Hoffmann. Explainability methods for graph convolutional neural networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 10764–10773, Jun 2019. doi: 10.1109/CVPR.2019.01103.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *International Journal of Computer Vision*, 128(2):336–359, Feb 2020. ISSN 0920-5691, 1573-1405. doi: 10.1007/s11263-019-01228-7. URL `http://arxiv.org/abs/1610.02391`. arXiv: 1610.02391.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv:1312.6034 [cs]*, Apr 2014. URL `http://arxiv.org/abs/1312.6034`. arXiv: 1312.6034.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net, 2014. URL https://arxiv.org/abs/1412.6806.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv:1706.03762 [cs]*, Dec 2017. URL http://arxiv.org/abs/1706.03762. arXiv: 1706.03762.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio', and Yoshua Bengio. Graph attention networks. *ArXiv*, abs/1710.10903, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv:1708.07747 [cs, stat]*, Sep 2017. URL http://arxiv.org/abs/1708.07747. arXiv: 1708.07747.

Chenhan Zhang, James J.Q. Yu, and Yi Liu. Spatial-temporal graph attention networks: A deep learning approach for traffic forecasting. *IEEE Access*, 7:1–1, 01 2019. doi: 10.1109/ACCESS.2019.2953888.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization, 2015. URL https://arxiv.org/abs/1512.04150.