Superpixel Image Classification with Graph Attention Networks

Pedro H. C. Avelar*†, Anderson R. Tavares*, Thiago L. T. da Silveira‡, Cláudio R. Jung* and Luís C. Lamb*

*Institute of Informatics
Federal University of Rio Grande do Sul
Porto Alegre, Brazil
Emails: {phcavelar,artavares,crjung,lamb}@inf.ufrgs.br

†Data Science Brigade
Porto Alegre, Brazil

‡Center of Computational Sciences
Federal University of Rio Grande
Rio Grande, Brazil
Email: tltsilveira@furg.br

***Abstract*—This paper presents a methodology for image classification using Graph Neural Network (GNN) models. We transform the input images into region adjacency graphs (RAGs), in which regions are superpixels and edges connect neighboring superpixels. Our experiments suggest that Graph Attention Networks (GATs), which combine graph convolutions with self-attention mechanisms, outperforms other GNN models. Although raw image classifiers perform better than GATs due to information loss during the RAG generation, our methodology opens an interesting avenue of research on deep learning beyond rectangular-gridded images, such as 360-degree field of view panoramas. Traditional convolutional kernels of current state-of-the-art methods cannot handle panoramas, whereas the adapted superpixel algorithms and the resulting region adjacency graphs can naturally feed a GNN, without topology issues.**

I. Introduction

The generic image classification problem consists of determining what object classes (typically from a set of pre-defined categories) are present in an input image. Early approaches followed the traditional pipeline of extracting image features (e.g., colour, texture, etc.) and feeding them to a classifier. The seminal work by Krizhevsky and colleagues [1] explored deep neural networks for image classification, winning the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 by a large margin and setting a turning point for research on image classification. The datasets became more challenging and the networks grew deeper, with the GoogLeNet architecture [2] winning the ILSVRC2014 challenge and "Squeeze-and-Excitation" layers being introduced in [3] to win the ILSVRC2017 challenge, with a top-5 error rate of 2.251%.

Despite the recent advances both in terms of datasets and network architectures, using traditional convolutional kernels limits the applications of these networks in problems that do not present a domain based on rectangular grids. For example, panoramas capture a full 360-degree field of view. Although the equirectangular representation does use a rectangular domain, sampling is highly non-uniform. To handle these issues, some authors proposed networks designed to adapt to the spherical domain have been designed, such as [4], while others propose to learn how to adapt convolutional layers to the spherical domain, as [5]. As another example, we can mention point-cloud classification, in which spatially unstructured data cannot be represented using a rectangular domain. In this context, some authors either explore a voxelized representation of the scene with 3D networks [6] or directly use 3D points as input to a network.

Graph-based representations can be used to model a variety of problems and domains. Furthermore, they naturally allow several "multiresolution" representations of the same object. For example, both pixel-level and superpixel-level representations of the same image might be modeled using graphs. In fact, superpixel-based representations have the advantage of reducing the input size, and potentially allowing different domains (e.g. pinhole and spherical images) to be represented as the same (or similar) graph. Furthermore, there are several recent advances toward the development of Graph Neural Networks (GNNs) [7], which could bridge the gap between different domains. In this paper, we explore Graph Attention Networks (GATs) [8] to classify images based on superpixel representations. GATs are a Graph Neural Network model that combine ideas of graph convolutions [9], which allows graph nodes to aggregate information from their irregular neighbourhoods, with self-attention mechanisms [10], which allows nodes to learn the relative importance of each neighbour during the aggregation process.

Our methodology comprises the following steps: (i) generate a superpixel representation of the input image; (ii) create a region adjacency graph (RAG) from the superpixel representation, by connecting neighbouring superpixels; (iii) feed the RAG to the GAT, which will predict the class. Experiments on several datasets show that the GAT outperforms other RAG-based GNN classifiers, but the RAG provides much less information than the raw image, so that the GAT's performance is inferior compared to the raw-image classifiers.

This paper is organised as follows: in Section II, we present related works and peering approaches. Section III revises existing superpixel segmentation techniques and how they are used to represent graphs, exposing the differences with the competing approach. Section IV describes the proposed method, while Section V shows the experimental setup and obtained results. Finally, the conclusions are drawn in Section VI.

II. Related Work

Monti et al. [11] provided, to the best of our knowledge, the first application of Graph Neural Networks (GNNs) to image classification, as well as proposing the MoNET framework for dealing with geometric data in general. Their framework works by weighting the neighbourhood aggregation through a learnt scaling factor based on geometric distances.

Velickovic et al. [8] proposed a model using self-attention for weighting the neighbourhood aggregation in GNNs, recognising that this model could be seen as a sub-model of the MoNET framework, nonetheless providing extraordinary results on other datasets, namely Cora and Citeseer, two famous citation networks [12], and on the FAUST humans dataset [13].

Although graph-based methods can be applied directly to images by considering each pixel a node of the graph, as in the seminal paper of Shi and Malik for image segmentation [14], lower-level representations generate smaller graphs. Using each region produced by a segmentation result might be a natural choice, but generating accurate segmentation results is still an open problem. A compromise solution between using individual pixels and object-related regions is *superpixels*. Superpixels group pixels similar in colour and other low-level properties, like location, into perceptually meaningful representation units (regions or segments) [15]. These over-segmented, simplified, images can be applied in a number of common tasks in computer vision, including depth estimation, segmentation, and object localization [16]. A comprehensive survey on superpixels can be found in [15].

The abovementioned work on using GNNs on images, alongside the work on adapting self-attention for GNNs and the works for generating superpixels of images form the pillars on which we based our experiments.

Two other models later came to our knowledge, which extended or could be seen as sub-models of the MoNET framework, using geometric information to weight neighbourhood aggregation, and provided results for the MNIST dataset. One of those is the SplineCNN model [17], which leverages properties of B-spline bases in their neighbourhood aggregation procedure. The other is the Geo-GCN model [18], which is a MoNET sub-model with a differently engineered learned distance function performing data augmentation using rotations and conformations.

Another technique for using GNNs with image data is to use them as a form of semi-supervised augmentation for classification, as in [19]. The main difference between their method and ours is that while they extract a CNN feature descriptor for each image with a (possibly pretrained) convolutional network, and then build a graph on which their model is used (akin to how Graph Convolutional Networks are used for semi-supervised classification in bag-of-words in [9]), we use the GAT as a classifier for a graph representing an image directly. Although their technique is useful for semi-supervised learning, the technique of using the network for superpixel classification has some possible advantages of its own, and they are not directly comparable.

Algorithm 1 Region Adjacency Graph (RAG) generation

procedure SUPERPIXEL2GRAPH(Image I of width w, height h and k channels, Superpixel segmentation technique S, and node feature builder F)
  $s, \mathcal{N} \leftarrow S(I)$ ▷ s returns the superpixel $n \in \mathcal{N}$ of a (x,y) pixel
  $x(n) \leftarrow F(I, s, n) \forall n \in \mathcal{N}$ ▷ $x(n)$ is the feature vector of node n
  $\mathcal{E} \leftarrow \{\}$
  for $1 \leq x \leq w$ **do**
    for $1 \leq y \leq h$ **do**
      if $s(x, y) \neq s(x+1, y)$ **then**
        $\mathcal{E} \leftarrow \mathcal{E} \cup \{(s(x, y), s(x+1, y))\}$
      end if
      if $s(x, y) \neq s(x, y+1)$ **then**
        $\mathcal{E} \leftarrow \mathcal{E} \cup \{(s(x, y), s(x, y+1))\}$
      end if
    end for
  end for
  return $\mathcal{G} = (\mathcal{N}, \mathcal{E}), x$
end procedure

III. Superpixel Graphs

A number of techniques exist to generate superpixels from images, such as SLIC [16], SNIC [20], SEEDS [21], ETPS [22], and the hierarchical approach from [23]. For our experiments, we chose to use SLIC [16] since it was readily available and had a spatial component in its superpixel segmentation. SLIC is stable and it is still recommended among other state-of-the-art oversegmentation algorithms [15]. Nonetheless, we believe that other segmentation techniques with similar characteristics could be used.

After applying a superpixel segmentation technique, we generate a Region Adjacency Graph (RAG) by treating each superpixel as a node and adding edges between all directly adjacent superpixels (1-neighborhood connection). Note that this differs from the approach adopted in [11], since their superpixel graphs have connections that span more than one neighbour level, with edges formed with the K nearest neighbours. Each graph node can have associated features, providing an aggregate information based on characteristics of the superpixel itself. Algorithm 1 describes the adopted procedure, whereas Fig. 1 depicts the generation of a RAG from an image.

There are many possibilities for building the features related to each node. For example, statistics about the colour and position of a superpixel, such as the mean, standard deviation, and correlation matrices of its pixels are readily available from the superpixel segmentation. In the case of images defined on rectangular domains, positional information relates to a 2D point. However, this concept can be easily adapted to omnidirectional images or point clouds, so that a single

(a) (b) (c)

Fig. 1. From left to right, the image to be converted into a RAG (a), the image with the superpixel segmentation being shown (b), and the image with the superpixel segmentation and the generated region adjacency graph overlayed on top of it (c).

topology based on graphs can be used in different applications. We do not build features for the edges, since we use an attention-based technique, and believe that the edge feature will be learned accordingly from the attention mechanism using both nodes' features.

In this work in particular, we apply this procedure to the well-known MNIST [24], FashionMNIST [25], Street View House Numbers (SVHN) [26] and CIFAR-10 [27] datasets. The first two datasets contain grayscale images of 28×28 pixels and 10 classes, and the last two contain RGB images of 32×32 pixels, both also having 10 classes.

Monti and colleagues [11] used the MNIST dataset and converted it into a graph-based format by using a superpixel-based representation. But whereas they connected nodes through a K-nearest neighbour procedure, we do so using RAGs. Hence, our dataset presents a lower-connectivity graph, which could impair information flow and make the classification problem harder. We also provide results for the RAG representation of the FashionMNIST dataset, since it is a more challenging dataset for which the information loss from the superpixel representation could impact more significantly the model. Since these two datasets contain only grayscale images, we build each superpixel's feature vector as the concatenation of the average luminosity of the pixels in a superpixel and the geometric centroid.

The SVHN and CIFAR datasets, however, both contain RGB channels in their images, and a natural extension for the feature vector is to use the concatenation of the average value for each colour channel and the geometric centroid. These datasets were used to see how the model would perform both with simple and complex colour images.

IV. Our Model

We transform the Undirected Graph produced from the over-segmented image's RAG into a Directed Graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, and feed it to a Neural Network model that operates on Graphs. More specifically, we use GAT layers stacked on top of each other using the same adjacency graph on each layer.

Our model is a version of the GAT model by Velickovic et al. [8], roughly based on the implementation by Nathani et al. [28]. Attention is implemented by scattering the source and target nodes' input features into their respective edges, making

Algorithm 2 Implemented GAT Layer

procedure GAT-FORWARD(Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, Node Features $x(n) \forall n \in \mathcal{N}$, learnable transition function f and learnable attention function a)
  $M_{tgt}(t_e, e) \leftarrow 1\{e = (s_e, t_e)\} \forall e \in \mathcal{E}$
  $h_{src}(s_e) \leftarrow x(s_e) \forall e \in \mathcal{E}$
  $h_{tgt}(t_e) \leftarrow x(t_e) \forall e \in \mathcal{E}$
  $h(e) \leftarrow h_{src}(s_e) \| h_{tgt}(t_e) \forall e = (s_e, t_e) \in \mathcal{E}$
  $y(e) \leftarrow f(h(e)) \forall e \in \mathcal{E}$
  $\alpha(e) \leftarrow a(h(e)) \forall e \in \mathcal{E}$
  $\alpha_{base}(e) \leftarrow \max_{e \in \mathcal{E}} \alpha(e)$
  $\alpha_{norm}(e) \leftarrow \alpha(e) - \alpha_{base}(e) \forall e \in \mathcal{E}$
  $\alpha_{exp}(e) \leftarrow e^{\alpha_{norm}(e)} \forall e \in \mathcal{E}$
  $\alpha_{sum} \leftarrow (M_{tgt} \times \alpha_{exp}(e)) + \epsilon$
  $y_\alpha(e) \leftarrow y(e) \alpha_{exp}(e) \forall e \in \mathcal{E}$
  return $o = (M_{tgt}, y_\alpha) / \alpha_{sum}$
end procedure

the transition and activation function on both these inputs and then summing them up over each target node through the edges.

Therefore, for each layer with input dimension d_i and output dimension d_o we learn two functions. The transition function $f : \mathbb{R}^{2d_i} \rightarrow \mathbb{R}^{d_o}$, composed of a linear layer followed by a nonlinearity, and the attention function $a : \mathbb{R}^{2d_i} \rightarrow \mathbb{R}$ that tells how much the target node of an edge should attend to the source node's information, also composed of a linear layer. The values produced by the attention function are activated using softmax for each target node. On the implementation, we take advantage of the fact that $\sigma(\mathbf{z} + \mathbf{c})_j = \sigma(\mathbf{z})_j$.

In summary, given a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, with edges $e = (s_e, t_e) \in \mathcal{E}$ and node features $x(n) \forall n \in \mathcal{N}$, let $\mathcal{T}(t)$ be the set of nodes with an edge towards t, the attention model can be summarised by Equations (1) and (2) below, where $\|$ denotes vector/tensor concatenation.

$$\alpha(s,t) = \frac{e^{a(x(s)\|x(t))}}{\sum_{s' \in \mathcal{T}(t)} e^{a(x(s')\|x(t))}} \tag{1}$$

$$o(t) = \sum_{s \in \mathcal{T}(t)} \alpha(s,t) f(x(s)\|x(t)) \tag{2}$$

The detailed algorithm showing the optimisation can be seen in Algorithm 2. Each of these layers can be arranged in a multi-head model by concatenating their outputs after the forward pass of each layer. That is, given k heads, the joint output of the k-headed layer, where each head has its own transition and attention functions f_i and a_i (as well as the intermediary α_i), would be as in Equation (3), where $\Big\|_{i=1}^{k} a_k$ is the concatenation of all vectors/tensors a_k:

$$o(t) = \sum_{s \in \mathcal{T}(t)} \Big\|_{i=1}^{k} \alpha_i(s,t) f_i(x(s) \| x(t)) \tag{3}$$

The output of the final GAT layer can then be sum-pooled, having all the values added, and then passed through a MultiLayer Perceptron (MLP) for the final prediction. The Python/Pytorch implementation in its fullest can be seen in the provided GitHub repository[1] as well. Most operations have been parallelized as much as the authors could fathom, with some operations done in a preprocessing phase to avoid overload.

V. Experiments

In this section, we show the potential of our technique on four datasets: MNIST [24], FashionMNIST [25], SVHN [26] and CIFAR-10 [27]. The superpixel algorithm has a target of 75 regions, so that the analyzed graphs present approximately 75 nodes each.

All experiments were ran either in a computer with a NVIDIA Quadro P6000 or one with a NVIDIA GTX 1070 Mobile. Both computers have 32GB of RAM. For development, we adopted the Pytorch library, version 1.X, using CUDA.

For all experiments we set a budget of 100 epochs for optimisation, with a batch size of 32 images, using a $90/10$ split for training and validation in the dataset's original training data. We use Adam as the optimiser, with a learning rate of 0.001, $\beta_1 = 0.9$, and $\beta_2 = 0.999$, using the model with the best validation accuracy on the test dataset. If the model failed to leave a baseline accuracy for the first 10 epochs, we restarted the training procedure from scratch.

A. MNIST

We trained two versions of the GAT model: A single-headed GAT with 3 layers, with $32, 64$ and 64 neurons, and a two-headed GAT, where each head has the same amount of neurons as the single-headed model. Both models used sum-pooling and a MLP with two layers of 32 and d_o neurons for the final classification, where $d_o = 10$ is the number of classes in MNIST. All neurons use ReLU activations, except for those at the last layer of the classification MLP, which use softmax activations. We did not use any regularisation technique.

All dataset images are converted to a corresponding RAG, using SLICO [29], a zero-parameter variant the SLIC algorithm. We set the target number of superpixels as 75, but the generated RAGs are not guaranteed to have exactly 75 nodes due to how the SLIC algorithm works.

Table I shows that both GAT models performed better than the MoNET model [11], showing that a learned representation of the geometric distance can lead to better performance than the fixed one of [11]. Note that our model has also to deal with graphs that are sparser than the ones used in the baselines [11], [17], [18], since in their graph edges are formed through K-nearest neighbours and ours use only directly adjacent nodes. Although one could expect worse accuracy, the results suggest that our approach is able to learn relevant geometric information relating the features from all neighbouring superpixels. We also report the performance of the graph-based approaches SplineCNN [17] and Geo-GCN [18], as well as recent alternative approaches such as the Generative Tensor Network Classifier (GTNC) [30] and the best performing method based on Support Vector Classification (SVC) as reported in the dataset homepage[2]. Although our approach does not reach state-of-the-art results, it is competitive and the best among graph-based methods.

B. FashionMNIST

We trained the single-headed and multi-headed GAT models with the same configuration of Section V-A. Since the FashionMNIST dataset also has 10 classes, the number of output neurons is $d_o = 10$ as well. The RAGs were also generated as in Section V-A.

Since none of the found graph-based papers present results for both MNIST and FashionMNIST, we provide a performance comparison of our models to GTNC [30] and the two best classifiers from the FashionMNIST benchmark [25] in Table I. The gap on the performance between the traditional ML models that used the full features of the dataset and our models, which use a reduced representation based on the oversegmented image, is higher on FashionMNIST. This shows how much harder the FashionMNIST dataset is for oversegmented images, where the information loss is greater with the aggregation of the features of the pixels in the superpixel.

Another interesting fact to note is that the multi-headed model performed worse than the single-headed one. This was further confirmed when we tried learning with a 4-headed model, which performed slightly worse than the 2-headed one.

C. Street View House Numbers

To check the performance of the model with multi-channel data, we trained and tested the two-headed model on the Street View House Numbers (SVHN) dataset [26], using the 32×32 cropped version and the same parameters for oversegmentation as in the previous sections. We achieved a similar performance with the two-headed GAT model on the FashionMNIST dataset, with 80.72% test accuracy. This comes to show that our model works even with full colour data, as well as the confounders present in the SVHN dataset.

[1] https://github.com/machine-reasoning-ufrgs/spixel-gat

[2] https://github.com/zalandoresearch/fashion-mnist

TABLE I

TEST ACCURACY FOR THE TESTED MODELS ON MNIST AND FASHIONMNIST, PROCESSED AS RAGS WITH APPROXIMATELY 75 NODES (CALLED MNIST-75 AND FASHIONMNIST-75), COMPARED TO THE BASELINE MODELS. BOLD VALUES SHOW THE BEST OF THE GRAPH-BASED MODELS. WE ALSO PRESENT THE MEAN ACCURACIES OF THE TWO BEST CLASSIFIERS FOR THE NON-OVERSEGMENTED MNIST AND FASHIONMNIST DATASETS, AVAILABLE IN THE FASHIONMNIST BENCHMARK.

	MNIST-75	FashionMNIST-75
MoNET [11]	91.11%	-
SplineCNN [17]	95.22%	-
GeoGCN [18]	95.95%	-
GAT-1Head	95.83%	**83.07%**
GAT-2Head	**96.19%**	81.40%
	MNIST	FashionMNIST
GTNC [30]	97.6%	88.2%
SVC(10,poly)	97.6%*	89.7%*
SVC(100,poly)	97.8%*	89.6%*

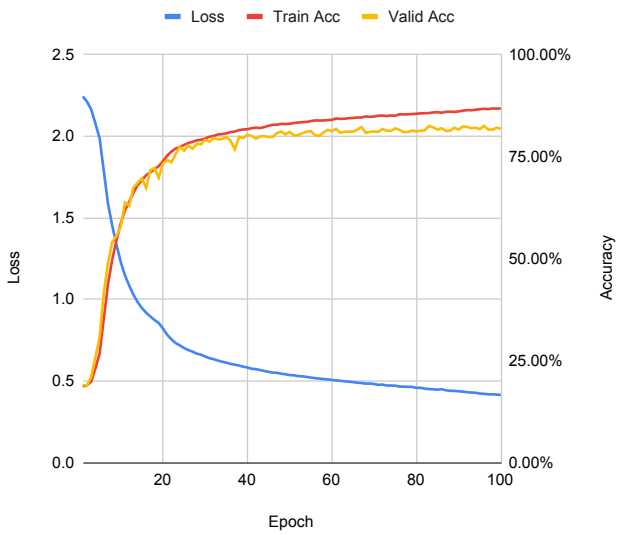

Fig. 2. Training Curve for the Street View House Numbers 32×32 dataset

That is, the classifier has learned both to prioritise the centermost superpixels and to identify which structure it contains by comparing changes in colour tone, instead of simple changes in luminosity, proving that, although it does not reach state-of-the-art performance, the model can achieve relatively good accuracy even working with less expressive data.

D. CIFAR-10

The CIFAR-10 dataset [27] contains 50,000 32×32 colour images distributed in 10 classes. We used the same parameters for oversegmentation as in the previous sections. The results we achieved through a network with the same architecture as GAT-2Head was 45.93% accuracy on the test set – very distant from what we achieved on the MNIST and FashionMNIST datasets. The training and validation accuracies were not impressive either, being 58.61% and 53.40% respectively.

As a baseline, we considered the VGG11 [31] architecture, with and without batch normalisation. We were unable to train the model without batch normalisation, whereas with batch normalisation we achieved 62.86% validation accuracy, which shows the heavy information loss during the RAG transformation procedure. However the comparison of the GAT with the VGG model is still unfair, in terms both of information available to the model and the number of parameters.

While the VGG model has access to the oversegmented image, with each segment's pixel having the averaged RGB values for each superpixel – approaching also *geometry* information – the GAT model only has access to the average colour and the centroid position, not knowing anything about the superpixel's shape. More precisely, while the VGG model has access to the middle images in Fig. 3, the GNN model has only access to the graph on the right images, with each node containing the average pixel value and position.

As for the model size, the VGG11 network has 132,868,840 parameters, while the GAT has only 55,364. The VGG network also consumed almost twice as much VRAM as the GAT-2Head architecture on the CIFAR images. VGG11 consumed 4,109MiB for training and 1,055MiB for testing, while the GAT model expended 1,067MiB for training and 485MiB for testing. For the sake of illustration, the current state-of-the-art result on CIFAR-10 [32] is reached with an AmoebaNet-B (550M parameters) pre-trained on ImageNet and fine-tuned on CIFAR-10.

VI. CONCLUSION

In this paper, we have investigated the application and interplay between Graph Attention Networks (GATs) [8] and image classification problems. In order to do so, we have used Region Adjacency Graphs (RAGs) computed from an image segmented using a superpixel algorithm, SLICO. We showed that using attention-based graph neural networks on a feature space that contains the geometric information can be improved by weighting the edges of a superpixel graph using a learned function which operates solely on the geometric information.

However, this approach to image classification has its shortcomings. The information loss in the pixel aggregation for more complex images can result in significant performance degradation when compared to using the full image. Also, graph-based approaches may come with the same limitations intrinsic to the models they use, and in our case the GNN-based architecture imposes some limitations in terms of memory usage for larger graphs due to the batching procedure (and thus finer segmentations), despite the smaller number of parameters the model itself had. Training in small batches lead to an unreliable training pattern, further aggravating such issues.

These limitations are, however, venues for future work. It has been shown that architectures based on graph convolutional networks, such as the GAT, suffer from an over-smoothing of node-level information, thus acting like low-pass

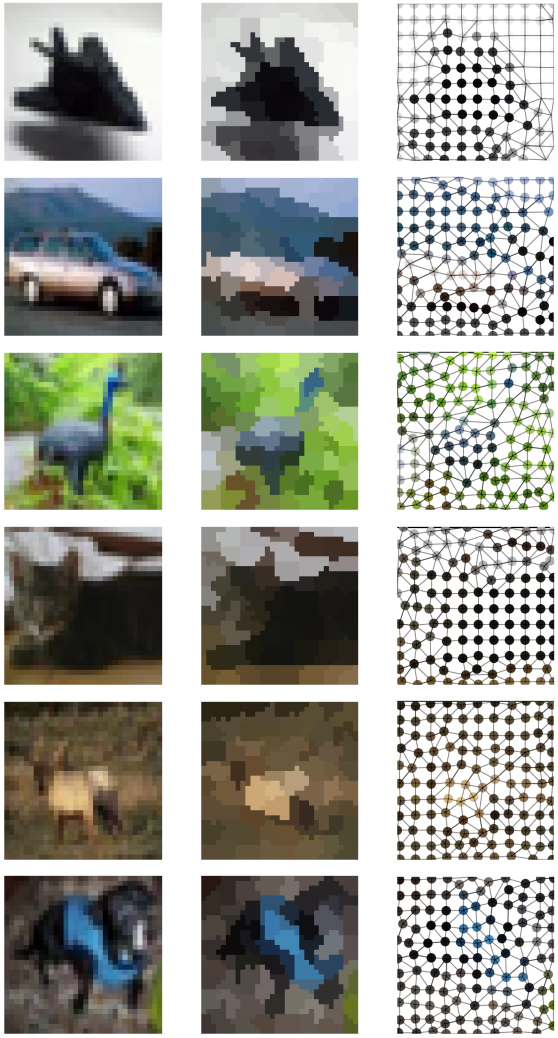

Fig. 3. Examples showing the loss of information in the RAG procedure when using only the average of each channel, which negatively affects the performance of the network.

filters [33]. While GATs might not be subject to the same limitation, this could be investigated to allow deeper GATs with potentially better performance in this domain. Another venue is helping scaling Graph Neural Networks, of which GAT is a representative, to larger graphs (and thus larger images) or to make them work in an online manner, or with smaller batches.

Also, our models used no regularization whatsoever, and investigating regularization techniques for these models could incur in better performance. Lastly, investigating different node feature vectors could provide the network with richer information and lesser the information loss due to the RAG procedure, possibly with information to recustruct the superpixel's components.

These graph-based approaches to image classification are also a prime example of application to non-euclidean images, such as omnidirectional images [34]. The flexibility of a graph-based approach could be more invariant to the domain of the image, possibly allowing pre-training on planar images and transfer to spherical images.

Acknowledgements

We would like to thank NVIDIA Corporation for the Quadro GPU granted to our research group. This work is partly supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) – Finance Code 001 and by the Brazilian Research Council CNPq. We would also like to thank the Pytorch developers for their library.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[3] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[4] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical cnns," *arXiv preprint arXiv:1801.10130*, 2018.

[5] Y.-C. Su and K. Grauman, "Kernel transformer networks for compact spherical convolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9442–9451.

[6] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[7] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[8] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[9] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[11] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 5425–5434. [Online]. Available: https://doi.org/10.1109/CVPR.2017.576

[12] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Magazine*, vol. 29, no. 3, pp. 93–106, 2008. [Online]. Available: http://www.aaai.org/ojs/index.php/aimagazine/article/view/2157

[13] F. Bogo, J. Romero, M. Loper, and M. J. Black, "FAUST: dataset and evaluation for 3d mesh registration," in *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*. IEEE Computer Society, 2014, pp. 3794–3801. [Online]. Available: https://doi.org/10.1109/CVPR.2014.491

[14] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[15] D. Stutz, A. Hermans, and B. Leibe, "Superpixels: An evaluation of the state-of-the-art," *Computer Vision and Image Understanding*, vol. 166, pp. 1–27, 2018. [Online]. Available: https://doi.org/10.1016/j.cviu.2017.03.007

[16] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, 2012. [Online]. Available: https://doi.org/10.1109/TPAMI.2012.120

[17] M. Fey, J. E. Lenssen, F. Weichert, and H. Müller, "Splinecnn: Fast geometric deep learning with continuous b-spline kernels," in *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018, pp. 869–877. [Online]. Available: http://openaccess.thecvf.com/content_cvpr_2018/html/Fey_SplineCNN_Fast_Geometric_CVPR_2018_paper.html

[18] P. Spurek, T. Danel, J. Tabor, M. Smieja, L. Struski, A. Slowik, and L. Maziarka, "Geometric graph convolutional neural networks," *CoRR*, vol. abs/1909.05310, 2019. [Online]. Available: https://arxiv.org/abs/1909.05310v1

[19] B. Jiang, D. Lin, J. Tang, and B. Luo, "Data representation and learning with graph diffusion-embedding networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 414–10 423.

[20] R. Achanta and S. Süsstrunk, "Superpixels and polygons using simple non-iterative clustering," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 4895–4904. [Online]. Available: https://doi.org/10.1109/CVPR.2017.520

[21] M. V. den Bergh, X. Boix, G. Roig, B. de Capitani, and L. V. Gool, "SEEDS: superpixels extracted via energy-driven sampling," in *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VII*, ser. Lecture Notes in Computer Science, A. W. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., vol. 7578. Springer, 2012, pp. 13–26. [Online]. Available: https://doi.org/10.1007/978-3-642-33786-4_2

[22] J. Yao, M. Boben, S. Fidler, and R. Urtasun, "Real-time coarse-to-fine topologically preserving segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*. IEEE Computer Society, 2015, pp. 2947–2955. [Online]. Available: https://doi.org/10.1109/CVPR.2015.7298913

[23] X. Wei, Q. Yang, Y. Gong, N. Ahuja, and M. Yang, "Superpixel hierarchy," *IEEE Trans. Image Processing*, vol. 27, no. 10, pp. 4838–4849, 2018. [Online]. Available: https://doi.org/10.1109/TIP.2018.2836300

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[25] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[26] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[27] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[28] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul, "Learning attention-based embeddings for relation prediction in knowledge graphs," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, A. Korhonen, D. R. Traum, and L. Màrquez, Eds. Association for Computational Linguistics, 2019, pp. 4710–4723. [Online]. Available: https://doi.org/10.18653/v1/p19-1466

[29] "SLIC superpixels: SLICO," available at https://www.epfl.ch/labs/ivrl/research/slic-superpixels/#SLICO.

[30] Z.-Z. Sun, C. Peng, D. Liu, S.-J. Ran, and G. Su, "Generative tensor network classification model for supervised machine learning," *Physical Review B*, vol. 101, no. 7, p. 075135, 2020.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[32] Y. Huang, Y. Cheng, A. Bapna, O. Firat, D. Chen, M. Chen, H. Lee, J. Ngiam, Q. V. Le, Y. Wu *et al.*, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," in *Advances in neural information processing systems*, 2019, pp. 103–112.

[33] H. NT and T. Maehara, "Revisiting graph neural networks: All we have is low-pass filters," *CoRR*, vol. abs/1905.09550, 2019. [Online]. Available: http://arxiv.org/abs/1905.09550

[34] Y. K. Lee, J. Jeong, J. S. Yun, W. Cho, and K. Yoon, "Spherephd: Applying cnns on a spherical polyhedron representation of 360deg images," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 9181–9189.