KEYPOINTS-BASED, CNN ASSISTED IMAGE REPRESENTATION LEARNING USING GRAPH NEURAL NETWORKS

Amit Lohan
Department of Electrical Engineering
Indian Institute of Technology, Bombay
amitlohan@iitb.ac.in

Avyakta Wrat
Department of Electrical Engineering
Indian Institute of Technology, Bombay
avyaktawrat@gmail.com

Amit Sethi
Department of Electrical Engineering
Indian Institute of Technology, Bombay
asethi@iitb.ac.in

June 10, 2021

ABSTRACT

We propose a framework to utilize a graph neural network assisted by a CNN for modeling the non-local interconnection information between parts of images for image representation learning. Results of experiments conducted for classification and image reconstruction tasks using the simultaneously learnt representation are presented. We also present a framework to interconnect CNNs and GNNs. To process images using a GNN, a transformation of the image or image feature maps (after some processing with a CNN) to a graph, to make it a valid GNN input is required. Similarly, to process the graph structured data using a CNN, a graph to a meaningful 3D tensor transformation may be required while preserving the possibility of gradient flow for backpropagation. In this work, two such interconnects i.e, a GNN to CNN and a CNN to GNN interconnects are also presented.

1 Introduction

Neuroscience experiments have shown [1] that very low resolution or very less information as compared to actually present in a scene or an image is required for most of the vision related tasks (such as object identification, scene recognition etc.) by the human brains. Majority of information present in the images such as regions or pixels belonging to the background, plain regions etc. are often irrelevant for tasks such as object detection, face recognition, image classification etc.

Most of the methods based on CNNs or other DL techniques discard or select the regions of importance using pooling techniques such as 2D max-pooling [2] in CNNs for image classification. However, the flexibility of these techniques to control the importance given to different regions and object parts is dictated by the size of convolutional layer kernels and number of max-pooled layers used and hence the depth of the network.

We present a framework to select key-points from images with much more flexibility and at the same time use a graph neural network to utilize the interconnections information between the selected regions for a dense vector representation learning of images.

In addition to the node features, graphs also have the connections information, whereas the CNN feature maps are represented as 3 dimensional tensors with the inherent spatial arrangement. An interconnect between a CNN and a GNN has to take care of transformation of the spatial arrangement to edge weights and vice versa in addition to assignment of features from nodes to spatial locations and vice versa.

In case of an interconnect where the entire 3D feature maps are required to be converted to a graph, every feature vector at a location (i, j) can become the feature vector of a node in the graph, i.e a tensor of size $N \times M \times L$ will produce a

graph with $N \times M$ nodes with each node having a feature vector of length L. The inverse of distance between each pair of location (i, j) in the feature map or any other measure of connectivity can become the edge weight of connection between corresponding nodes. This way, for feature maps of size $N \times M \times L$, size of adjacency matrix would be $(NM) \times (NM)$. Similarly, an inverse transformation can be applied if location of each node as in the original 3D tensor of feature maps is tracked by rearranging the nodes features in the form of a 3D tensor using a suitable way for spatial assignment.

In case of an interconnect where only a handful of locations from the 3D tensor of feature maps are required for constructing a graph, for example in case of processing features only from key-points in images, a slightly different approach than discussed in above paragraph is required. Two such interconnects for key-points based processing using a hybrid model of GNNs and CNNs are discussed in Sections 5 and 6.

2 Related Work

Deep CNNs are the most widely used method for image classification and image representation learning tasks. CNNs can very effectively detect patterns and parts of patterns in 2D image grids while preserving the spatial arrangement of such sub-patterns, giving the benefits such as translation invariance and less number of trainable parameters requirement as compared to fully connected neural networks. However, the size and complexity of patterns that CNNs can detect depends on their receptive field and hence the depth. Additionally, using pooling layers, effectively, a CNN divides image feature maps into several regions and selects or discards features belonging to centers of these region (deeper the network, more max-pools, more flexibility and more complexity of pattern that can be detected).

However, when a CNN trained on one resolution is fed an image of a significantly higher resolution, the receptive field required for detecting the patterns that the network was originally trained on falls short and a CNN may start performing poorly. In addition to that, the flexibility of CNNs to completely ignore or utilize the features computed for different regions is limited by the pooling technique e.g. [2] and number of pooling layers.

In [3], a framework to transform an image into a graph and then to run the classification algorithm on this graph is presented. The method has the flexibility to utilize the interconnections between parts of images however does not have the task specific, learnable keypoints detection.

To overcome such limitations, we explore a way of using CNNs to detect the task specific, data-defined key locations in images, transform the key information in image into a graph representation of keypoint features and then use a Graph Neural Network to process this graph to learn an image level dense vector representation. This representation may be used for image compression, image classification and other computer vision tasks such as image retrieval etc.

3 Approach

We explore several ways of task based end-to-end keypoints detection. 2 shows the high level view of our algorithm. The first step of our approach is to have a method to identify a good set of key locations in the image and then extract features from these locations and their neighbourhood. This way, these features may be treated as features of nodes of a graph. As the graphs can be used to utilize the relationship between different objects which correspond to its nodes, it may be possible to learn a representation vector for each image by processing with a graph neural network which is more suitable for classification and image compression purposes. After trying out a number of approaches such as concatenating the features at the output of graph neural network and then flatten before passing through layers of a classifier neural network, the procedure shown in image 1 was found to be good training procedure for the experiments of this project. Multiple ways to detect the keypoints were experimented with.

Figure 1 shows the procedure of end-to-end training for the representation learning. Firstly, two parallel network take the image as input for features computation and keypoint location estimation. The keypoint location detector network may be implemented in a number of ways and depending on that the feature extractor network may also assume different architectures. For keypoints detection part we experimented with three different algorithms which are discussed next section.

After the keypoint locations and their respective features are computed by the combination of two leftmost networks shown in fig. 1, the next step is to represent the features corresponding to these locations as nodes of a graph. So the features at the output of feature extractor network are selected by selection based on N key locations (N is a hyperparameter) and become the node features for the graph neural network. An adjacency matrix is also calculated using the Euclidean distance between each pair of points. After the node features are processed by the graph neural network, they are put back into the locations (as predicted by keypoint detection network) of a zeros tensor. This tensor

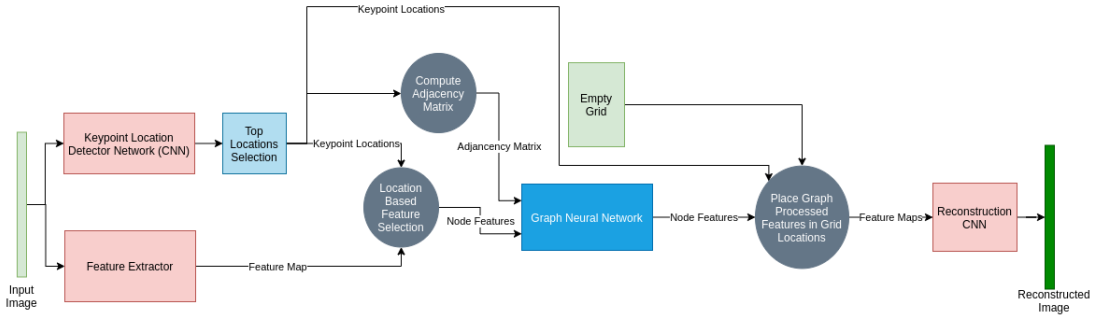

Figure 1: Procedure to learn reconstruction of images using keypoints and graph representations.

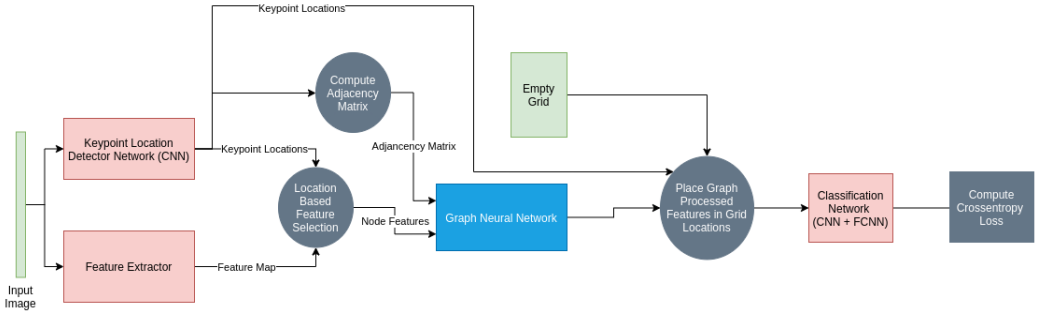

Figure 2: Procedure to learn classification of images using keypoints and graph representations.

is then further processed by a CNN to either reconstruct the image or predict the class label. The results obtained by using three different strategies for keypoint location detection are presented in the next section.

4 Keypoint Detection Methods

Following subsections include the details of three different methods that we experimented with:

4.1 Attention Network as Location Selector

One simple way of detecting keypoint locations is to predict an attention score for each pixel location and then select top N points according to the predicted attention values. This way, using a classification or reconstruction loss at the output of post processing CNN, the attention network should learn to identify the most important points in the image from classification or reconstruction point of view. The results obtained by using such an approach are shown in fig. 4.

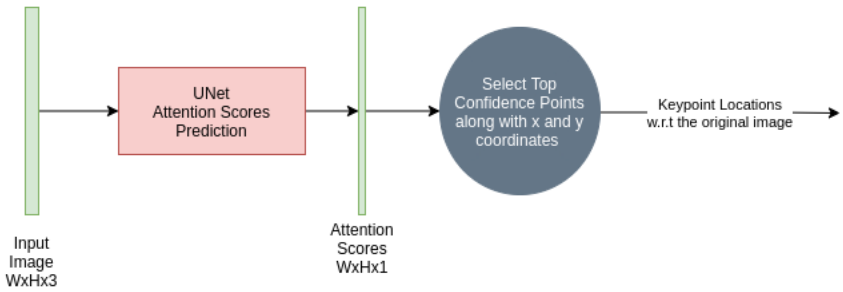

Figure 3: Procedure to identify keypoints with attention network based approach.

It can be seen in the images in fig. 4 that the detected keypoints using the attention network based method alway lie in bunches or groups. Two or more keypoints lying in the close proximity of each other are undesirable as the features corresponding to these points will be quite similar but will take different nodes in the graph network. So most of the features taken from such locations will be redundant and will wast lots of computation. This has been observed to be a serious problem with the attention network based keypoint detector network. To overcome this problem two more keypoint location detection strategies were developed and are discussed in section 4.2.

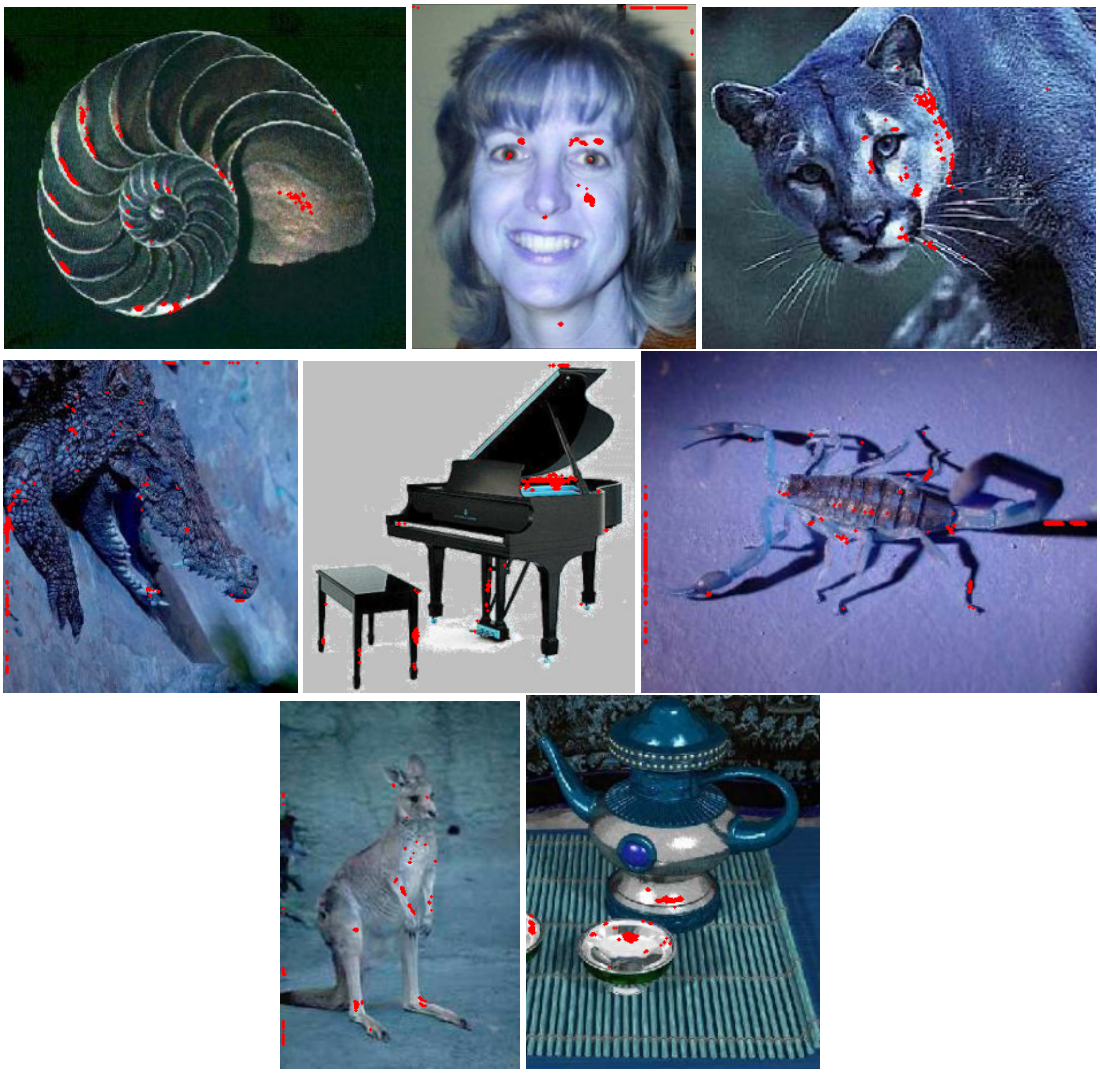

Figure 4: Detected key points (red) with attention network based approach

4.2 Keypoints Coordinates Prediction with a Neural Network

One possible way to overcome the problem posed by the attention network based method in section 4.1 is to have a network directly predict the coordinate values of keypoint locations and pre-train it with a dataset of images by computing ground truth at the training time. The ground truth locations for pretraining can be obtained by detecting key locations with SIFT or ORB keypoint detector algorithms. This way the keypoint coordinate prediction network will learn to identify the key locations to some extent before it is used in the main training loop.

Figure 6 shows such a setup to predict coordinate locations. The output of predictor network should be a LxMx(3h) vector where L<=W and M<=H and L and M are number of cells along rows and columns. Each cell is responsible for detecting keypoints in a mxm square patch. The values of m in the experiments included in this section has been 8 and 16. Each cell can detect a maximum of h keypoints in the square patch it is responsible for. Each 3h numbers along the axis 2 of output tensor predict three things: x coordinate within the cell, y coordinate within the cell and confidence of

prediction of each keypoint. In the training loop, confidences are multiplied with the selected feature from output of feature extractor network and in addition to providing confidence values for selection, they also act as attention values.

The results obtained using this approach are shown in figure 4.2. As can be seen, the network is able to track the keypoints on object quite well. The only problem that was observed with this approach was that after training the network for enough number of epochs, the keypoints are predicted very close to or on the boundaries of cells. The probable reason for that sounds to be the use of sigmoid to normalize the output in the range of 0 to 1. As the gradients at extreme points of sigmoid are quite low and very high around 0, the keypoints find it very unlikey around the center of the image. For classification task, this network was able to train quite quickly to 99.9% train accuracy but did not give better than 60% accuracy for 10 classes of imagenet. So far, I haven't been able to solve the problem of keypoints moving close to boundaries.

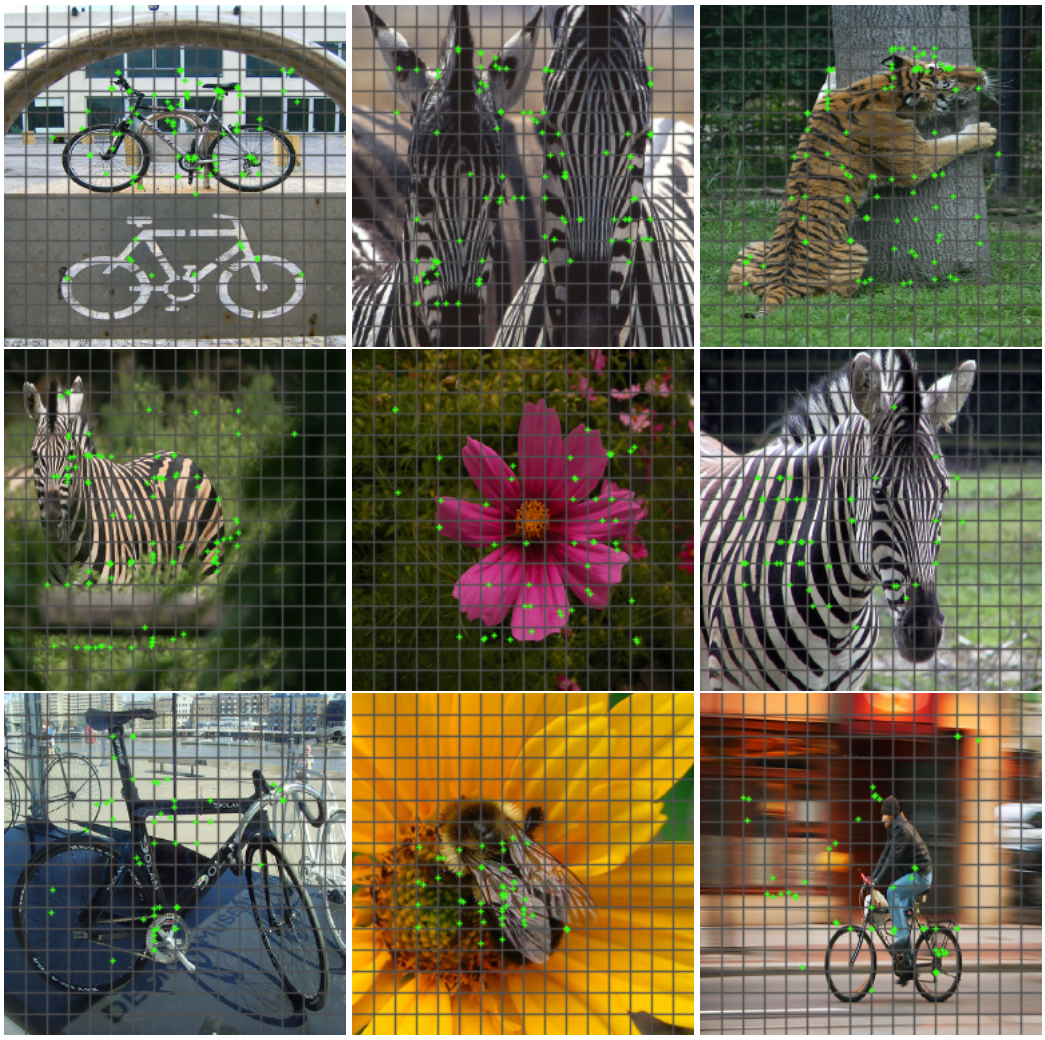

Figure 5: Keypoint locations identified by a coordinate prediction (regression) network based on the concept of [**?**].

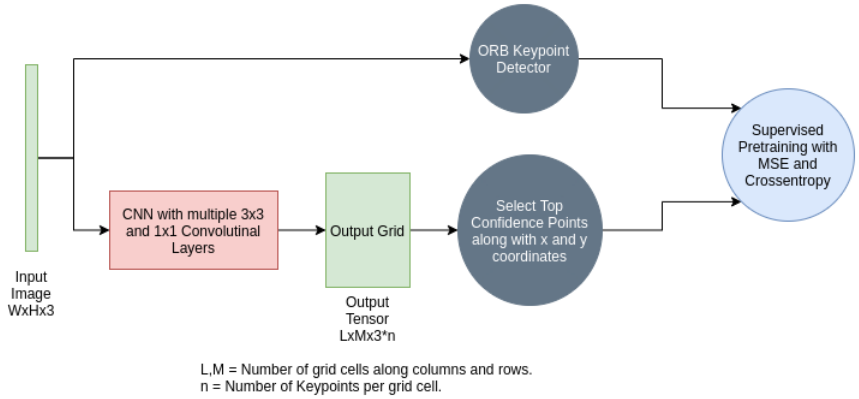

Figure 6: Procedure to identify keypoints with coordinates prediction network based approach.

4.3 Large Maxpool as Non-max Suppression

Another way that I worked out for non-max suppression in the experiment presented in the section 4.1 is to use the procedure shown in figure 7. A large maxpool of 8x8 or 16x16 is used to avoid the possibility of detecting points in close clusters and the locations of maxpooled in points are also saved (relative to the original predicted mask). Top N locations are then selected from the maxpooled feature map based on the attention scores and the respective locations are also selected from the set of locations previously saved. This way, we get a set of top keypoints with non-max suppression and spatial sparsity. The results obtained using the classification loss in this method are presented in figure 8.

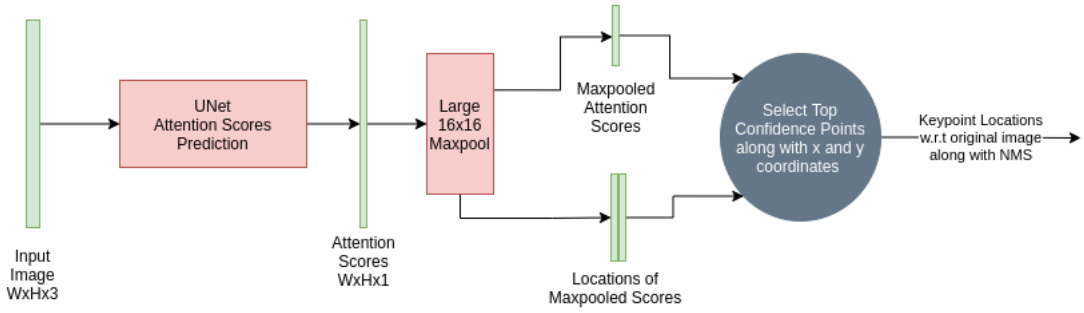

Figure 7: Procedure to detect keypoints using maxpool as non-max suppression.

In addition to the classification approach, training experiments were also conducted with the reconstruction objective with imagenet and a faces dataset. The reconstruction did not work well with the imagenet, the possible reason perhaps being high variablity of images. The results obtained with faces dataset are presented in figure 12. Average mse of 0.06 for images normalised in 0 to 1 was obtained for the reconstruction.

5 CNN to Graph Interconnect

To transform the output feature maps of a convolutional layer, at any stage in CNN, into a graph, two pieces of information are required:

- Node features (The features from CNN output, which will become the features of Nodes of the graph).
- Connections information between the nodes.

In case of a key-points processing based CNN-to-graph interconnect, as shown in Fig. 10, the feature vectors belonging to the key-point locations become the features of nodes and the inverse of distances between every pair of these

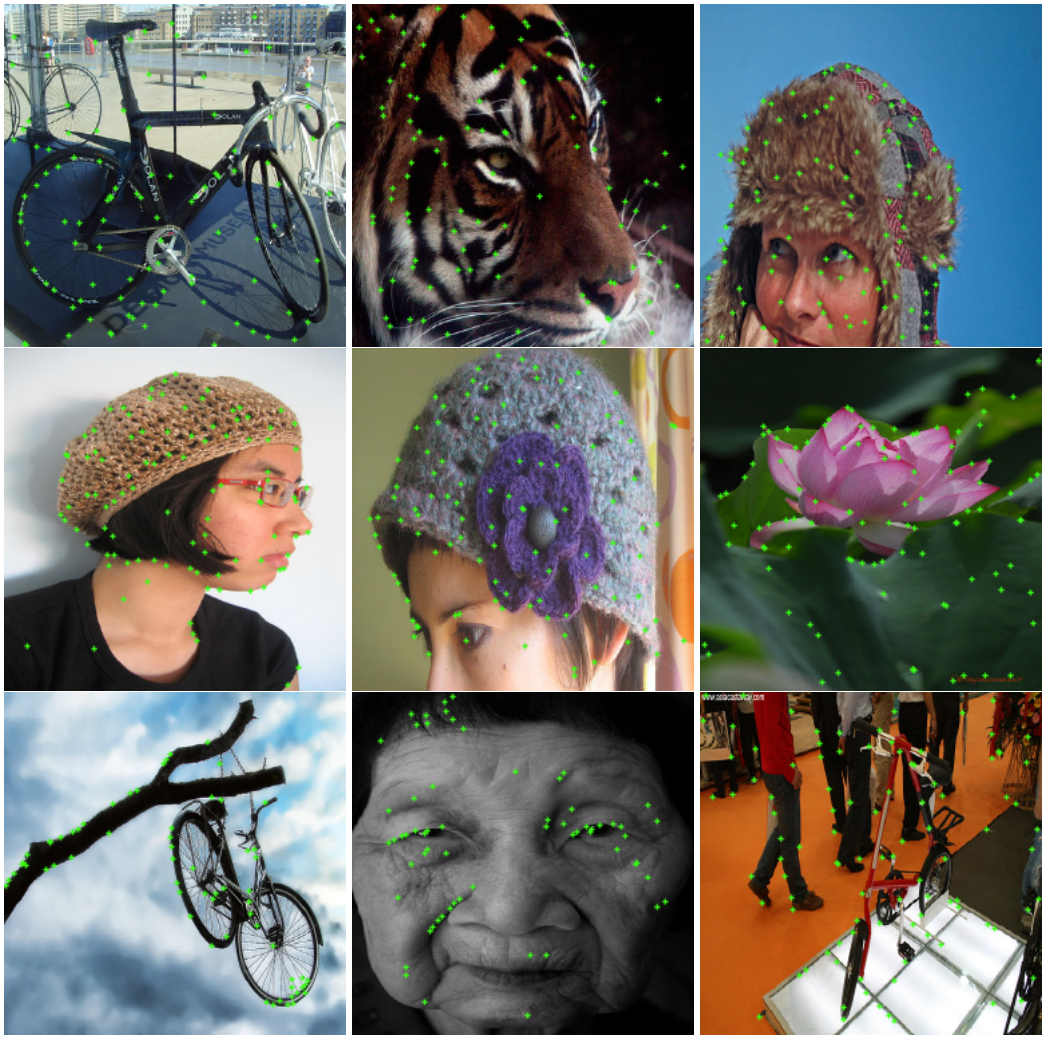

Figure 8: Keypoints detected by the model in section 4.3.

key-points becomes the strength of interconnection between the corresponding pair of nodes in the graph. This way if there are N key-points then the adjacency matrix will have dimensions of $N \times N$.

The implementation of this type of interconnect is available at:
`https://github.com/amitlohan/nn_interconnects`

6 Graph to CNN Interconnect

For transforming a graph into a tensor of CNN feature maps, we require node feature vectors and a way to arrange them spatially in a grid. To arrange the features correctly into a 2D grid, the original key-point locations are required. As shown in Fig. 11, first of all we take a 3D tensor of zeros with dimensions $N \times H \times W$, where N is the size of feature vectors of the nodes of graph, and W and H are respectively the width and height of the CNN feature maps to be obtained, then the features are assigned to the 2D locations as given by original locations of key-points which was used for conversion of feature maps to graph. This way a valid CNN features map is obtained.

The implementation of this type of interconnect is available at: `https://github.com/amitlohan/nn_interconnects`

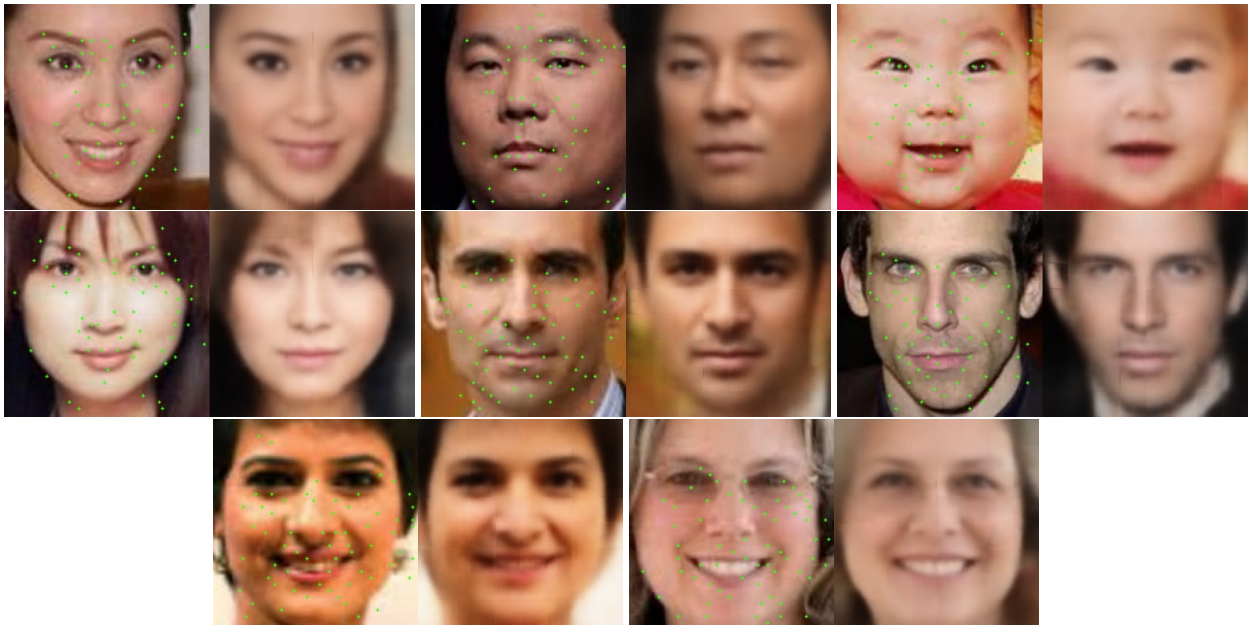

Figure 9: Image reconstruction results and keypoints detected by the model in section 4.3.

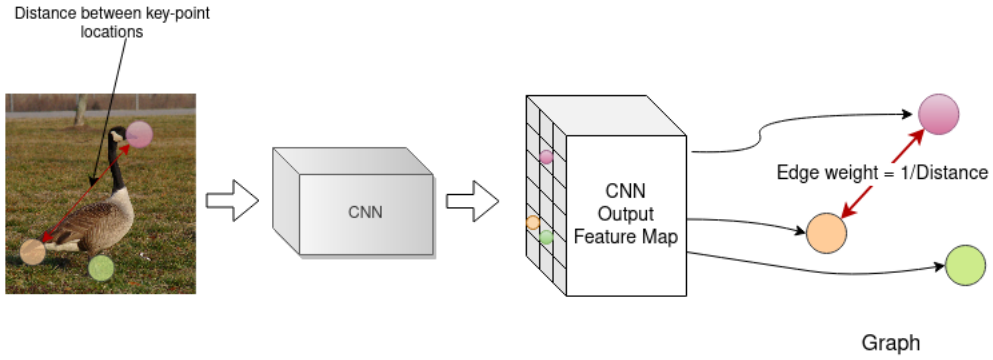

Figure 10: Sample figure caption.

Exp. No.	GNN Architecture	No. of Graph Layers	No. of Key Points	No of Classes	Keypoint Score Activation	No. of trainable parameters	Test Accuracy
1	CNN Only		64	10	Linear	3.29M	51.60
2	GCN	4	64	10	Linear	3.31M	78.15
3	GCN	5	64	10	Linear	3.31M	74.35
4	Diffpool	5	64	10	Sigmoid	5.11M	98.1
5	GCN	5	64	10	Sigmoid	6.48M	98.4

Table 1: Imagenet Classification Experiments

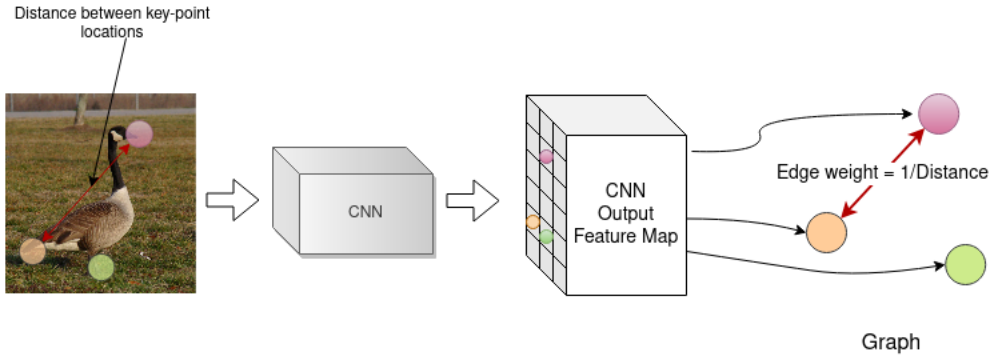

Figure 11: Sample figure caption.

7 Image classification experiments

Table 1, shows the results of classification experiments. It can be observed that when a CNN only architecture, with graph neural network being absent is used, a 51.60 % accuracy is obtained, which is much less as compared to the flow-graphs where a graph neural network is present in the loop. In experiment no. 2, when a graph neural network is inserted, in addition to the classification CNN, it can be seen that with the increase of only a small number of parameters, the accuracy increases quite significantly.

While ranking the importance of key-points, two types of activations can be used for calculating the final score, a linear activation and a sigmoid activation. In table 1 Experiments 3 and 5 show the difference of accuracies obtained when a sigmoid is used instead of a linear activation. It was observed that using the Diffpool architecture gives almost as good accuracy as the plain GCN network architecture. Additionally, diffpool architecture trains faster than the fully connected graph neural network architecture.

8 Image reconstruction experiments

For reconstruction, an upsampling convolutional network was used to reconstruct the image from tensor filled with node features. The loss function used for reconstruction was MSE and a modified MSE (termed as conf. MSE). For calculation of conf. MSE loss, the square error loss at each location is also multi with the attention score (output of keypoint detection network) or the confidence of that point/pixel being a key-point.

Figure 12: Image reconstruction results and keypoints detected for experiments of table 2.

This way, the potential background regions are not given more importance for reconstruction and only the object regions, where keypoints are meant to be detected are given more importance for training. Results of reconstruction experiments are given in table 2.

Exp. No.	Graph Architecture	No. of Graph Layers	No. of Key Points	Keypoint Score Activation	No. of trainable parameters	Loss Function	Loss Value
1	GCN	5	48	Linear	14.3M	MSE	0.01720
2	GCN	9	64	Linear	14.42	MSE	0.03128
3	GCN	5	64	Sigmoid	17.48	MSE	0.00928
4	Diffpool	5	64	Sigmoid	5.11M	Conf. MSE	0.11712

Table 2: Imagenet Reconstruction Experiments

9 Conclusions and future work

It can be concluded from the experiments in section 4.1 that a vanilla attention based network is not a good solution to the problem of keypoints detection as it does not give a spatial sparsity of keypoints and introduces lots of redundance of data in the learnt representation, which is a contradiction with the very objective of this project. Observing the results in section 4.2, we can say that the coordinate prediction network is quite successful in locating the key-points on primary objects in the image and is able to effectively discard the points on background. The only problem with this approach seems to be its inability to track points in the central areas of grid cells, possibly because of the use of sigmoid at the output. As sigmoid seems to be the only available option of a activation function which is both differentiable as well as, is capable of mapping a real number to [0,1] range. A solution to this problem may help detect good points and might improve the classification and reconstruction results significantly.

The results of experiment in section 4.3 may be said to support the conclusion that a max-pool based non-max suppression is working quite effectively and key-points are being detected at appropriate locations. A network trained on features extracted from these locations should be able to give a good classification accuracy. Now the test accuracy obtained on 10 classes of Imagenet using the representations learnt with this approach was not more than 65% although the model was able to smoothly train to get close to 100% accuracy on the test dataset.

In the second stage of experiments in this project, after the keypoint-detection and feature extractor networks were combined, the overfitting reduced significantly as can be seen from results included in table 1. Also it can be observed that using a sigmoid activation instead of a linear activation to obtain final key-point scores given much better classification and reconstruction results as can be observed from tables 1 and 2.

Overall it can be concluded that the representation learning framework employed for reconstruction and classification tasks works well and is able to give 98.5% classification accuracy for a classification task and good enough representation for a reconstruction task for a dataset constructed from 10 classes of Imagenet. The method, however, failed to obtain good classification accuracy for 100 classes of Imagenet. Significant overfitting was observed for a dataset constructed with 100 classes as the model failed to obtain more than 35% accuracy for 100 class dataset. Further investigation may be done to find a solution to the problem of overfitting when the number of classes is increased from 10 to 100. Using a deeper model for keypoint detection, or using more number of key-points for a 100 classes dataset may result in improvement of overall classification accuracy as 48 or 64 key-points may not be able to capture enough distinguishable information for classification or reconstruction when training is done on a dataset of 100 classes.

References

[1] David Eagleman. *The Brain, The Story of You - Chapter 2.* Pantheon, 2016.

[2] Marc' aurelio Ranzato, Y-lan Boureau, and Yann Cun. Sparse feature learning for deep belief networks. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

[3] Yin Li and Abhinav Gupta. Beyond grids: Learning graph representations for visual recognition. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.