What's That Wing? Fine-Grained Butterfly Species Classification Using Graph Neural Networks

Christopher Koenig
Stanford University
koenig97@stanford.edu

Abstract

As deep learning approaches for computer vision have evolved, they have been applied to progressively harder classification tasks. One such task is fine-grained visual categorization, as for distinguishing between sub-species. This study seeks to leverage a similarity network representation of butterfly sub-species to gain traction on this difficult task. We apply a Graph Convolutional Network (GCN), Graph Attention Network (GAT), and GraphSAGE Network for classification. We find that all 3 models achieve strong performance, but are limited by the size and lack of rich features of the dataset.

1 Introduction

Over the past decade, the field of computer vision has been transformed by the deep learning revolution. However, the task of fine-grained visual categorization - that is, distinguishing between subordinate-level categories such as bird species or dog species - remains a significant challenge. This is due to properties of the data, namely high levels of variance between images of the same class and relatively low levels of variance between images of different classes, particularly as compared to standard image classification tasks such as ImageNet in which the classes are distinct. These properties make it difficult for even deep learning models to achieve superior performance on the fine-grained visual categorization task.

This context poses a challenge to reframe the fine-grained visual categorization task in a manner that is more tractable for deep learning approaches. In this paper, we seek to do that by applying Graph Neural Networks (GNNs) to the task of fine-grained classification on a butterfly similarity network with the goal of leveraging the network information to make predictions on butterfly species. More specifically, our goal will be to distinguish between 10 classes of butterfly species based on similarity scores calculated from feature vectors extracted from the images. In applying Graph Convolutional Networks, GraphSAGE Networks, and Graph Attention Networks, we seek to gain insight both on how GNNs perform on fine-grained visual categorization compared to traditional deep learning methods as well as how these GNN architectures compare to each other.

This classification problem is especially urgent in the context of the ongoing environmental catastrophe. As climate change intensifies and is enhanced by the impact of such trends as pollution, deforestation, and overexploitation, worldwide biodiversity is increasingly at risk. Preserving biodiversity is thus of central importance in conservation efforts. A fundamental first step in these efforts is fine-grained species identification - such as the classification of the butterfly species we conduct in this paper - so that biodiversity can be identified and protected by environmentalists.

2 Related Work

While the advent of deep learning has been very recent, the sub-field of deep graph neural networks are especially recent, with most of the major architectures and approaches having been defined within the past 3 years. The literature on the application of graph neural networks to similarity networks or for the purposes of fine-grained visual categorization is thus very limited. We first review the critical method for denoising biological networks that led to the creation of our dataset, proceed to cover the paper introducing Graph Convolutional Network architecture, and close by examining past performance achieved on our dataset.

2.1 Network Enhancement as a General Method to Denoise Weighted Biological Networks

In this paper, Wang et al. (2018) propose a framework to denoise biological networks using network enhancement (NE). The algorithm is based on the assumption that nodes connected through paths with high-weight edges are more likely to have a direct, high-weight edge between them. Using a doubly stochastic matrix to diffuse the network, it reassigns weight to each of the edges so that edges with weak similarity are reassigned lower weights or are removed, while edges with high similarity get higher weights. The result is an updated network that retains the original network information while making the network more sparse, thus increasing the network analysis efficiency downstream and enabling more accurate detection of modules and clusters. When Wang et. al. applied Network Enhancement to a similarity network calculated on the Leeds Butterfly Dataset (which became the BioSNAP butterfly similarity dataset we use in this study), they found it substantially improved identification accuracy and the ability of the network to be clustered by species. Network Enhancement is thus likely an essential pre-processing step for any GNN-based fine-grained classification of a biological similarity network, for without this step the network will be too noisy to learn meaningful patterns.

2.2 Semi-Supervised Classification with Graph Convolutional Networks

Kipf and Welling (2016) introduce a variant of convolutional neural networks that operate directly on graph-structured data, known as Graph Convolutional Networks (GCN). This approach is derived from a first-order approximation of spectral graph convolutions. It scales linearly in the number of graph edges and thus transcends the scaling obstacles that have impeded past efforts to develop neural networks for graphs. The hidden layer representations encode both local graph structure and features of nodes in a similar way to how CNNs encode local image characteristics in their lower layers (for deep networks). While efficient, the GCNs do encounter memory limitations because the memory requirement grows linearly in the size of the dataset, so special provisions or approximations have to be made for especially large or densely connected graph datasets.

This last note is particularly relevant for the BioSNAP butterfly similarity network dataset we use in this study. As is discussed in section 3, the similarity network is originally extremely dense because every node (corresponding to a butterfly image) is connected to every other node for a total of 345696 edges for only 832 nodes. Network enhancement processing, however, reduces the number of edges from 345696 to 86528, making the application of the Graph Convolutional Networks outlined by Kipf and Welling more feasible.

2.3 Finding Butterfly Species Pattern: A Case Study on Butterfly Similarity Networks

While this report is a previous CS224W project report rather than a published paper, it is critical because it provides the only reference point for analysis and performance of Graph Convolutional Networks on the BioSNAP butterfly similarity network. In their study, Wang (2018) obtain embeddings for every node in the network by running the Node2Vec algorithm 3 times - once biased for BFS, once biased for DFS, and once balanced between the two. They then evaluate the GCN with each of the three embeddings, finding that the GCN obtains best performance of 77.49% with embeddings generated with BFS bias. We use the results obtained by Wang as reference for our models' performances and adopt the approach of generating Node2Vec embeddings with BFS bias.

2.4 A Survey on Deep Learning-based Fine-grained Object Classification and Semantic Segmentation

In this survey, Zhao et al. (2017) review general convolutional neural networks, part detection, ensemble of networks, and visual attention based fine-grained visual categorization approaches. They find that visual attention based deep classifiers achieve best performance, followed by ensemble of networks models. A common theme across all approaches reviewed is that the key to strong performance is incorporating some specific method for identifying fine-grained details and differences. In some models this takes place through direct part localization, in others through added classification capacity by combining multiple networks, and still others through automatically learning which regions of the images are most discriminative.

3 Dataset

Our dataset is derived from the Leeds Butterfly Dataset as detailed by Wang et al. (2009). This dataset consists of 832 images of 10 species. Each species has 55-100 images in the dataset. This dataset originally possessed no network structure.

3.1 BioSNAP Butterfly Similarity Network

The BioSNAP butterfly similarity network was created from the Leeds dataset by using two encoding methods - Fisher Vector (FV) and Vector of Linearly Aggregated Descriptors (VLAD) - to generate two vector representations of each image. These representations were then used to construct two similarity networks for which nodes represent the images and edges capture similarity between pairs of images based on their vector representations. The two resulting networks were then combined by taking their inner product via adjacency matrices. Finally, Network Enhancement processing - as described by Wang et al. and summarized in section 2.1 - was applied to generate the finalized network. This finalized network is the graph we leverage in this study. It consists of 832 nodes and 86528 weighted, undirected edges.

3.2 Network Characterization

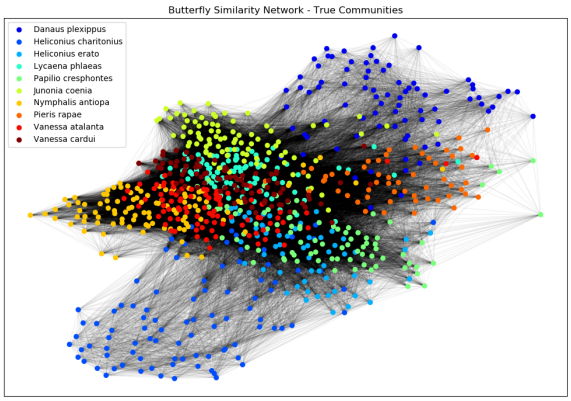


Figure 1: Clustering of species with true labels

As seen in Figure 1, while most of the clusters are decently distinct, some of the clusters are not well defined and/or seem merged with other clusters, indicating potential difficulty for a classifier to distinguish between these species. In particular, the pairs Nymphalis antiopa – Vanessa atalanta and Lycaena phlaeas – Vanessa cardui exhibit heavy overlap. The connection between these pairs is validated in the distribution of their edge weights, as demonstrated in Figure 2.

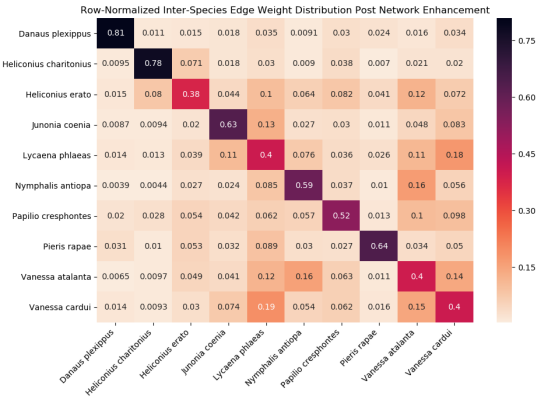


Figure 2: Edge Weight Distribution Heatmap

Figure 2 displays a row-normalized heatmap for the inter-species edge weight distribution of the network. Let S_i be the set of nodes in species i, $\mathcal{N}(i)$ be the set of neighbors of node i, and w_{ij} be the weight of the edge between nodes i and j. Each entry (i, j) above is then calculated as:

$$(i, j) = \frac{\sum_{i \in S_i} \sum_{k \in \mathcal{N}(i) \cap S_j} w_{ik}}{\sum_{i \in S_i} \sum_{j \in \mathcal{N}(i)} w_{ij}}$$

Intuitively, entry (i, j) captures the percentage of the total edge weight of nodes in species i that is derived from edges with nodes of species j. A high score at (i, i) indicates that species i has strong intra-species connections and is less similar to other species while a low score indicates that species i has weak intra-species connections and is more similar to other species. Species with high scores along the diagonal should thus be easier to distinguish from other species, and therefore easier to classify as well. Based on this criteria, Danaus plexippus and Heliconius charitonius should be the easiest to classify.

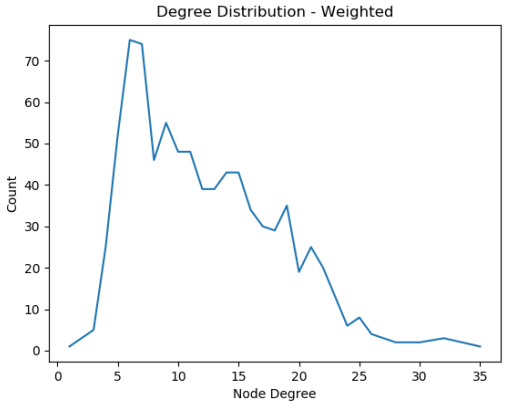


Figure 3: Weighted Degree Distribution

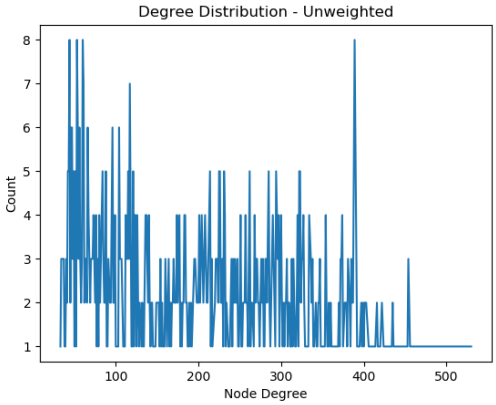


Figure 4: Unweighted Degree Distribution

In contrast, we see that Heliconius erato, Lycaena phlaeas, Vanessa atalanta, and Vanessa cardui have the lowest scores along the diagonal, so these species will likely be the hardest to classify. In particular, Lycaena phlaeas has a large portion of its edge weight from edges with Vanessa cardui, and vice versa. The same holds for Nymphalis antiopa and Vanessa atalanta. The edge weight distribution thus reinforces the overlap trends between communities in Figure 1.

Finally, we find that the network's weighted degree distribution follows a standard exponentially decreasing trend. However, the unweighted degree distribution is unusal. Figure 4 demonstrates a wide spread of degree values from less than 10 to over 500, and significant numbers of nodes with degree values everywhere in between. Looking at Figure 5, we see that the 4 species with the lowest scores along the diagonal have 4 of the highest average degree scores. Nymphalis antiopa, discussed above, also has a high average degree.

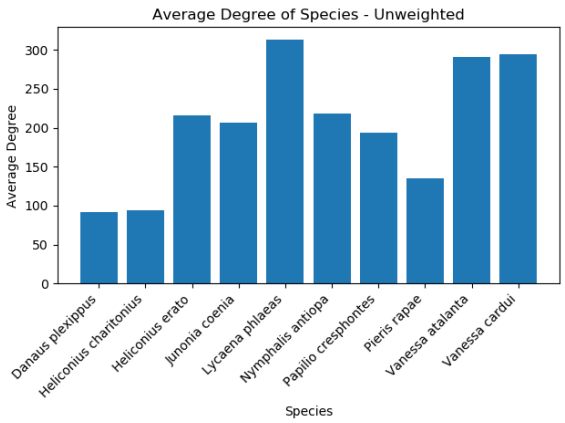


Figure 5: Average Unweighted Degree By Species

In contrast, the 3 species with the highest scores along the diagonal - Danaus plexippus, Heliconius charitonius, and Pieris rapae - have the 3 lowest average degrees. This aligns with our earlier observations from the edge weight distribution heatmap. Namely, the weaker a species' intra-species connections, the higher the number of inter-species edges. Tangibly, this means that when a butterfly looks like many other butterflies, it is going to have many edges to other species in the similarity network.

3.3 Community Detection

The trends discussed above manifest in the results of the Louvain community detection algorithm. This algorithm clusters nodes into communities with the heuristic of optimizing modularity. The algorithm iteratively performs 2 steps: (1) For each node i, the change in modularity is calculated for removing i from its own community and moving it into the community of each neighbor j of i. and i is then placed into the community that yields the greatest modularity increase. (2) Nodes in the same community are contracted into a "super" node, generating a new network composed of the resulting super nodes.

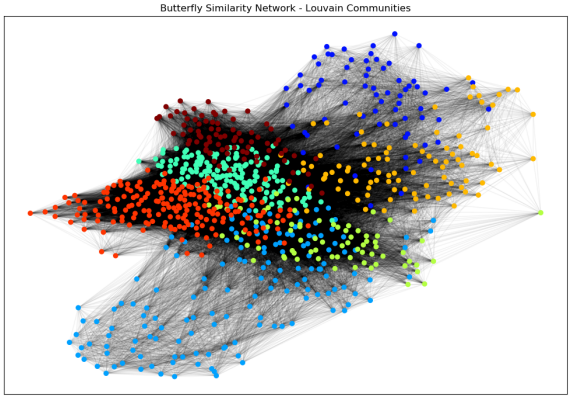


Figure 6: Communities Identified with Louvain

Running this algorithm results in the identification of 7 communities. Examining Figure 6, we see that Nymphalis antiopa – Vanessa atalanta, Lycaena phlaeas – Vanessa cardui, and Heliconius charitonius – Heliconius erato have been largely merged together. This reinforces the intuition built from Figures 2, 4, and 5 as these pairs of species share a high proportion of edge weight compared to other pairs. It is thus highly likely that these pairs will be hardest to classify for our models.

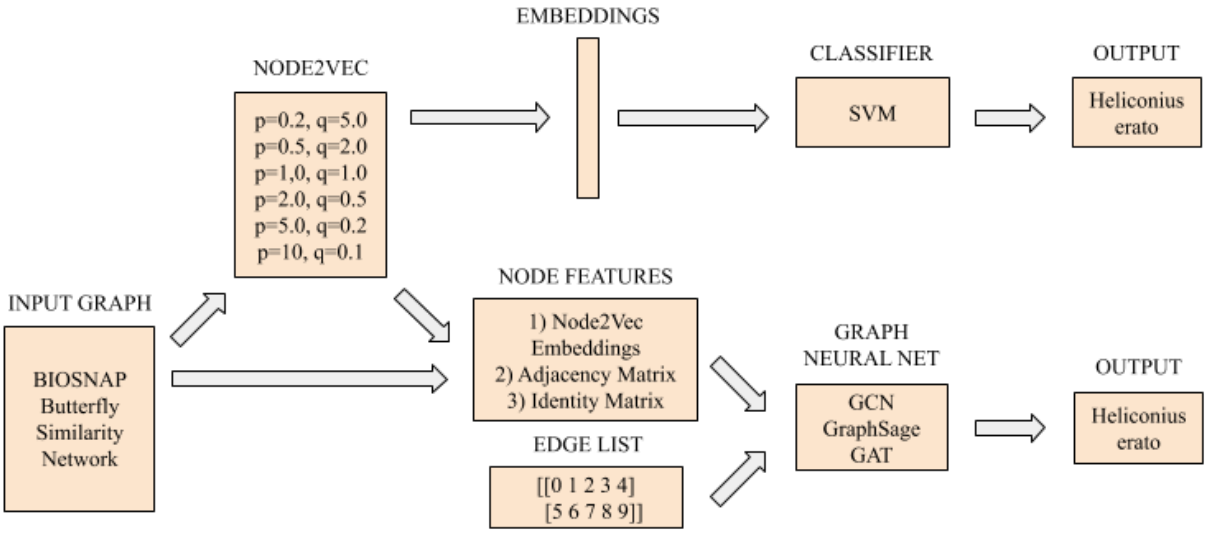


Figure 7: Visual Summary of Classification Pipeline

4 Model & Approach

As discussed in section 2.3, Wang (2018) achieved strong performance on this dataset with a GCN accuracy of 77.49% compared to 51.81% with a Multilayer-Perceptron baseline. Looking to build on this success, we implement a GCN, GraphSAGE, and Graph Attention Network (GAT) for the task of node classification with our dataset. Because the BIOSNAP dataset does not include the original butterfly images nor a mapping of node ids to images from the original Leeds dataset, the similarity network does not contain any node features. It only contains weighted edges post Network Enhancement. Therefore we experiment with 3 types of node features for our GNNs - the identity matrix, the adjacency matrix, and a matrix of Node2Vec embeddings. We also evaluate the performance of a SVM classifier on the Node2Vec embeddings to provide a reference point for the performance of the GNN models. A visual summary of this pipeline is given in figure 7.

4.1 Node2Vec

Node2Vec is an algorithm developed by Grover and Leskovec (2016) that encodes nodes in a graph to a low-dimensional embedding space via random walks on the graph such that the similarity of nodes in the embedding space approximates similarity of nodes in the graph. The key property of Node2Vec is the expressivity it allows for in the types of random walks that determine a node's "neighborhood" (and consequently the similarity between 2 nodes in the graph). Specifically, Node2Vec allows for trade-off between local and global views of the network via 2 parameters: the return parameter p and the in-out parameter q. A low value of p and high value of q corresponds to a "BFS" (Breadth-first search) like random walk that provides a more local definition of a node's neighborhood. In contrast, a high value of p and low value of q corresponds to a "DFS" (Depth-first search) like random walk that provides a more global definition of a node's neighborhood, capturing the structure of the node in the network. In our approach, we generate a range of 128-dimensional BFS-like and DFS-like embeddings to evaluate how the different types of information captured impact performance.

4.2 Node Features

We experiment with 3 options for GNN node features: an identity matrix, the similarity network adjacency matrix, and a matrix of Node2Vec embeddings. The identity matrix will have size 832 x 832 and provides no node information, so the GNNs must learn embeddings using only the edge list. The adjacency matrix has size 832 x 832 as well and provides the GNNs with all edge weights. Lastly, the Node2Vec embedding matrices will have size 832 x 128 and provide the GNNs with the local or global information learned about each node. Because the identity and adjacency matrix have size n x n, they scale exponentially with the number of nodes in the graph. Thus with larger graphs, the much smaller size of the embedding matrix is an advantage, as it scales linearly with the number of nodes in the graph n.

4.3 Support Vector Classifier

We utilize a Support Vector Classifier (SVC) for classifying the Node2Vec embeddings because support vector machines are considered by many to be the best "off-the-shelf" machine learning classifiers. Unlike simple artificial neural networks (ANN), SVCs have only a few parameters, clear convergence criterion, and are guaranteed to train quickly for small datasets. The simple but effective nature of SVCs thus makes them a good candidate for a reference model against which to evaluate the performance of the GNN models.

4.4 Graph Neural Networks

Finally, the main area of our evaluation in this study is the performance of 3 models of graph neural networks on the BIOSNAP data. The Graph Convolutional Network (GCN) was introduced in section 2.2. GraphSAGE, developed by Hamilton et al. (2017), built on GCN by generalizing the aggregation of node features during the neighborhood aggregation phase of the GCN learning process beyond weighted average, allowing for approaches such as pooling, applying a LSTM to learn how to aggregate node features, etc. The Graph Attention Network (GAT) from Veličković et al. (2017) further augmented the basic idea of the GCN by introducing an attention strategy into the learning process. Through the attention strategy, weights are implicitly specified for the node features in a node's neighborhood, allowing a node to learn the importance of each of its neighbor's information. We hypothesize that the GAT will achieve best performance on the dataset given the weighted nature of the BIOSNAP similarity network. While the other GNN models can likely learn the relative importance of each node's neighbors with time, the ability to directly model weights seems like it grants the GAT expressive power that is particularly relevant for this task.

The 3 models increase in expressive power in the order of GCN, GraphSAGE, and GAT. We chose these models to evaluate (as opposed to other graph convolutional layer types) because we hope that this variation in expressive power allows for a fuller understanding of the performance and limitations of GNNs on our fine-grained visual categorization task. It is also important to recognize that 832 nodes is a small amount of data for a deep learning model. Future work could explore augmentation of the dataset for better learning.

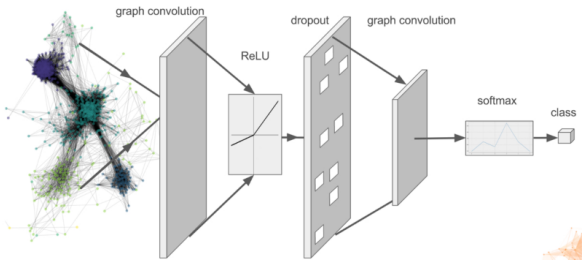


Figure 8: GCN Model Architecture (credit for the image to https://www.experoinc.com/)

5 Experiments & Results

5.1 Evaluation

For our evaluation metrics, we report the accuracy and confusion matrix. Accuracy provides a high-level overview while a confusion matrix provides a nuanced insight into per-class performance.

5.2 Baseline

Our baseline model uses the Louvain communities in Figure 9. We make predictions by using the majority label in each community as the predicted species for every node in that community. This approach yields an accuracy of 66.23%. Notably, this accuracy is nearly 15% better than the MLP baseline reported by Wang (2018).

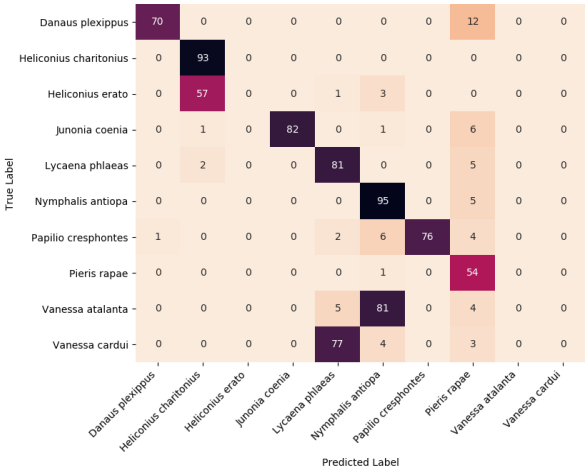


Figure 9: Majority-Label Louvain Classifications

From figure 9, the majority-label Louvain classifier makes predictable errors. As discussed in section 3.3, the algorithm merges 3 pairs of species, so understandably there are no classifications for Vanessa atalanta, Vanessa cardui, and Heliconius erato. Additionally, in line with the heatmap in Figure 2, the misclassifications occur exactly where expect them to - between pairs of species with high levels of shared edge weight.

p	q	SVC	GCN	GraphSAGE	GAT
0.2	5.0	**.9581**	.9281	**.9341**	**.9162**
0.5	2.0	.9521	.9042	.9222	.9042
1.0	1.0	.9461	**.9341**	.9222	.9042
2.0	0.5	.9281	.9161	.9281	.8862
5.0	0.2	.9222	.8862	.8922	.8802
10	0.1	.8922	.8982	.8802	.8443

Table 1: SVC and GNN Test Accuracies with Balanced, BFS-Biased and DFS-Biased Node2Vec Embeddings

5.3 Training Splits

We create a 70/10/20 train/validation/test split of the data. We also set the random seed so that the set of test nodes for SVC and GNN models is the same. When training the SVC, we combine the train and evaluation sets because during development we utilize cross-validation as our optimizing metric. For GNN training, the validation set is used to determine appropriate hyperparameters.

5.4 Support Vector Classifier

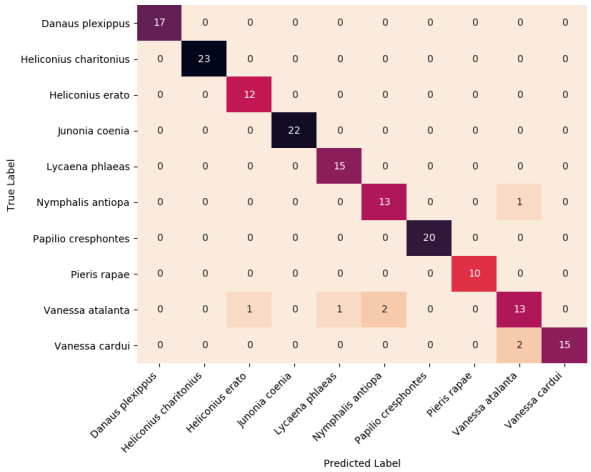


Figure 10: SVC Classifications when $p = 0.2, q = 5.0$

For the SVC, our input is a 832 x 128 matrix of Node2Vec embeddings. We standardize the input embeddings to zero mean and unit variance and optimize over a small hyperparameter space of SVM kernels and regularization strengths using 5-fold cross-validation as our optimizing metric. The test accuracy of the best estimator obtained from hyperparameter search for each of the Node2Vec embeddings generated from different p and q values is provided in Table 1. The best SVC performance is achieved with $p = 0.2, q = 5.0$.

	Identity	Adjacency	Embedding
GCN	0.9759	0.9398	0.9277
GraphSAGE	0.9639	0.9277	0.9036
GAT	0.8554	0.9398	0.9398

Table 2: GNN Eval Accuracies with Various Node Features

	Identity	Adjacency	Embedding
GCN	0.9281	0.9162	**0.9341**
GraphSAGE	0.9281	0.9222	**0.9341**
GAT	0.9042	0.9102	**0.9162**

Table 3: GNN Test Accuracies with Various Node Features

5.5 Graph Neural Networks

As specified in section 4, the input to our GNNs is a node feature matrix that is either the identity matrix, BIOSNAP butterfly similarity network graph adjacency matrix, or a matrix of Node2Vec embeddings. Each GNN is trained with 1 layer, 32 hidden dimensions, 0.2 dropout rate, and early stopping. We conducted hyperparameter search with higher numbers of layers and hidden dimensions, but found that these values resulted in overfitting while the minimal choices above still provided sufficient expressive power to achieve strong performance on the dataset. The evaluation and testing accuracies of the GCN, GraphSAGE, and GAT models using identity, adjacency, and embedding features are given in tables 2 and 3, respectively. The score for the embedding column in these tables corresponds to the best accuracy achieved by the model from the embedding parameters of p and q in table 1. In addition, confusion matrices for the top performing GNN models are given in Figure 11. For all models, we find that training converges between 100 to 200 epochs.

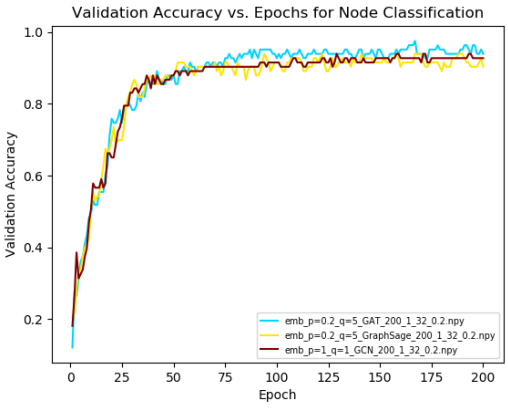
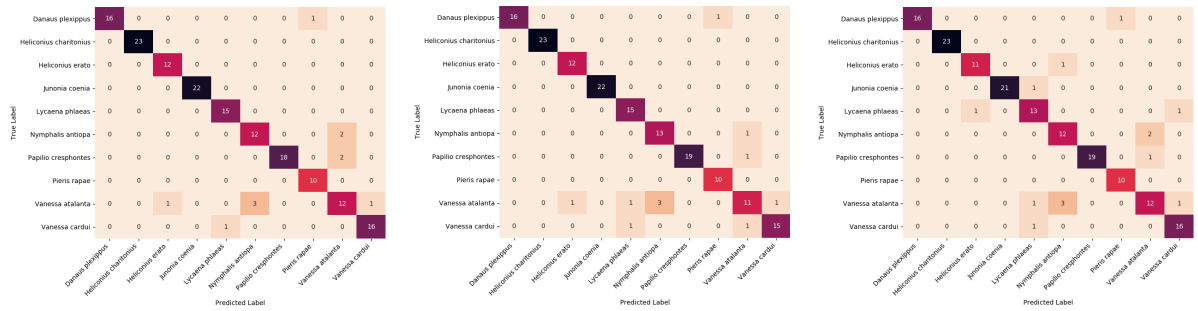


Figure 12: Learning Curves for Top GNN Models



GCN GraphSAGE GAT

Figure 11: Confusion Matrices for Top-Performing GNN Models

6 Discussion

6.1 Node2Vec Embeddings

From table 1, we see that the models achieve better performance with BFS-biased than DFS-biased embeddings, with over 6% better performance with the most BFS-biased (p=0.2, q=5.0) as compared to the most DFS-biased (p=10, q=0.1) embeddings. These findings make sense given the structure of our network. As discussed in section 3.1, Network Enhancement processing removes or reassigns weak edges lower weight while reassigning strong edges higher weight. Thus the presence of an edge between 2 nodes means they had a high similarity score and is a strong indicator those nodes belong to the same class. As BFS-biased walks enable a local, micro-view of the network, they emphasize the 1-2 hop neighborhood. This is well-suited for our network, as nearly all nodes of a given class are likely to be within 1-2 hops of each other due to Network Enhancement.

DFS-biased walks, in contrast, capture a global, macro-view of the network, more strongly emphasizing the structural role of a node such as a bridge between communities or a fringe member of a group. However, our network does not necessarily have ”roles” in the way other networks do because edges do not signify real-world relationships (ie friendship, follows, etc.), but rather similarity scores between nodes. After Network Enhancement, the structure of each node is similar, with many connections to nodes of the same class and a few to moderate number of edges to nodes of other classes. Therefore outward-oriented exploration won't find meaningful structural significance because there is little variation in structure between nodes of different classes. This means a global view of a node in our network does not offer useful information for node classification.

6.2 Node Features

From table 3, we see that there is little variation in test accuracies among the GNNs for node features of identity, adjacency matrix, and Node2Vec embeddings, with all test accuracies falling withing a range between 90% - 94%. The Node2Vec embeddings result in the highest scores for all 3 models, but only slightly. In conversation with the results in table 1, this tells us 2 things. The first is a reinforcement of GNNs remarkable ability to generate high-quality embeddings. Specifically, for the identity node features, the GCN and GraphSAGE models achieve 92.81% test accuracy to the 95.81% test accuracy of the SVC with the best BFS-biased embeddings (p=0.2, q=5.0). The fact that GCN and GraphSAGE achieve comparable performance despite having no node features and no access to the edge weights of the similarity network means that these models are able to generate node embeddings of similar quality to Node2Vec from scratch, leveraging only the neighbors of each node and 32 dimensions instead of the 128 dimensions of the Node2Vec embeddings (because we set the number layers to 1 and number hidden dimensions to 32 during training).

However, the fact that the best-performing SVC was able to achieve 95.81% accuracy indicates that it almost perfectly learned the problem space despite its relative simplicity. This finding is reinforced by the fact that the GNNs do not perform meaningfully better with the adjacency matrix or Node2Vec embeddings. Even though the GNNs are able to generate high-quality embeddings with no node features (ie the identity matrix), the information from the adjacency matrix and Node2Vec embeddings should only improve learning and classification performance. The fact that performance does not improve significantly - combined with the low number of layers (1)

Lycaena Phlaeas | Vanessa Cardui | Nymphalis Antiopa | Vanessa Atalanta

Figure 13: Images of Butterfly Species Commonly Confused by Classifiers

and hidden dimensions (32) needed to achieve the scores reported - indicates that the performance of the GNNs is capped by the simplicity of the dataset as reflected by its small size (832 nodes) and lack of node features. The similarity network is derived from the Leeds Butterfly Dataset which contains images for all 832 data points. However, the mapping from the original images to the nodes they are associated with is not provided in the BIOSNAP dataset. If the GNNs were able to take advantage of the images and an expanded dataset, their expressive power could be better leveraged.

6.3 GNN Comparison

It is somewhat surprising to note from table 3 that there is little variation in performance among the GCN, GraphSAGE, and GAT models. Particularly, it is intriguing that the GAT achieves the lowest performance among the three models by about 1% - 2% depending on what node features are utilized. We had expected GAT to perform the best among the 3 models because our similarity network is weighted, so we hypothesized that the GAT would be able to automatically learn the importance of each neighbor according to its edge weight via the attention mechanism. However, this hypothesis was not upheld in our experiments. It is important to note that we also explored a large hyperparameter space of number of epochs, number of layers, number of hidden dimensions, and dropout rates, so the low performance of the GAT models is not due to lack of expressive power or failure to train long enough. Rather, related to the preceding discussion around capped GNN performance due to dataset limitations, it is likely that there is little variation in performance between GNN types due to the simplicity of the dataset. Specifically, if each model type is powerful enough to learn the problem space well, then none of the GNN types have the opportunity to distinguish themselves.

6.4 Error Analysis

In section 3.3, we predicted that the majority of misclassifications would occur on the species Nymphalis antiopa, Vanessa atalanta, Lycaena phlaeas, Vanessa cardui. We find that this hypothesis holds in our results. In the confusion matrix for the top-performing SVC (figure 10), we see that misclassifications involving pairs of these species accounted for 6 of the 7 errors made by the model. In the confusion matrices in figure 11, misclassifications involving pairs of these species account for over 50% of the errors made by the top GCN, GraphSAGE and GAT models. Given the strong visual similarity between these species as showcased in figure 13, the predominance of misclassifications involving pairs of these species makes sense. However, the persistence of these errors informs us that while the graph-based classification techniques we have explored have achieved substantial success on the dataset, the most challenging fine-grained visual categorization tasks are still not perfectly mastered by our approaches.

7 Conclusion & Future Work

In this study, we sought to gain traction on the difficult task of fine-grained visual categorization through leveraging graph classification techniques. Working with the BIOSNAP butterfly similarity network dataset, we first ran Node2Vec to generate balanced, BFS-biased, and DFS-biased embeddings, then leveraged these embeddings in downstream SVC and GNN models. In addition to using these embeddings as node features for our GNN models, we also developed GNN models with no node features and with the similarity network edge weights as features. We find that both our SVC and GNN models are able to achieve excellent accuracy in the range of 90% - 95% on our test set - an improvement of more than 17%

as compared to the top GNN model reported by Wang (2018) - with accurate predictions across all species. Regarding Node2Vec embeddings, we find that BFS-biased embeddings are optimal for this task due to the characteristics of our network, namely a lack of distinctive global structure that could contribute to the identification of "roles" in the network. Our GCN, GraphSAGE, and GAT models achieve similar performance even with different node features (identity, adjacency, and embeddings), indicating that the expressive power of our GNN models is not able to be fully exercised due to the lack of node features (e.g., original butterfly images) and small size of the dataset.

Future work on this task would benefit greatly from an expanded dataset and a mapping of the node ids in the BIOSNAP dataset to the original images in the Leeds Butterfly Dataset so that the Graph Neural Networks could take advantage of much richer node features. In addition, it would be worthwhile to further explore GAT architecture choices, as the GAT attention mechanism should in theory be able to better model the weights of the network than other GNN models.

Contributions

As I am the only member of my team, I completed all of the work detailed above for this project, including writing all code as well as this report.

Acknowledgements

We would like to thank Jure Leskovec and Michele Catasta for their excellent instruction and provision of resources throughout this study.

References

Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. *CoRR* abs/1607.00653. http://arxiv.org/abs/1607.00653.

William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *CoRR* abs/1706.02216. http://arxiv.org/abs/1706.02216.

Thomas N. Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks .

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks.

Bo Wang, Armin Pourshafeie, Marinka Zitnik, Junjie Zhu, Carlos D. Bustamante, Serafim Batzoglou, and Jure Leskovec. 2018. Network enhancement as a general method to denoise weighted biological networks. In *Nature Communications*.

Josiah Wang, Katja Markert, and Mark Everingham. 2009. Learning models for object recognition from natural language descriptions. In *Proceedings of the British Machine Vision Conference*.

Qiwen Wang. 2018. Finding butterfly species pattern: a case study on butterfly similarity networks.

Bo Zhao, Jiashi Feng, Xiao Wu, and Shuicheng Yan. 2017. A survey on deep learning-based fine-grained object classification and semantic segmentation. *International Journal of Automation and Computing* 14(2):119–135. https://doi.org/10.1007/s11633-017-1053-3.

Code for this project is available at:
https://github.com/koenig125/224W-final-project