Project Proposal Report: Graph Neural Network based Clustering Enhancement in VANET for Cooperative Driving (Hu and Lee 2022)

Nazanin Mehregan - 7549975

Abstract

This paper aims to improve the stability of vehicle clustering in order to enhance the lifetime of cooperative driving. Platoon-based driving is a cooperative driving pattern in which vehicles follow each other closely, forming a platoon. Clustering these platoons is a crucial task for the safety and efficiency of cooperative driving. In this paper, we propose a method to cluster platoons using graph neural networks (GNNs) in an unsupervised learning setting. Our goal is to obtain lean vehicle representations or node embeddings for downstream machine learning applications, such as K-means clustering. We use graph autoencoders to learn node embeddings and do not require any labeled data. We represent each vehicle as a node in a graph with eight features: position x, position y, velocity x, velocity y, acceleration x, acceleration y, length, and weight. We use SAGE convolutional layers to encode these features and obtain a four-dimensional embedding vector for each node. We train the GNN using cross-entropy loss and evaluate the clustering performance using K-means clustering, spectral clustering, GraphSage convolution, and graph autoencoder-K-means. Our results show that the graph autoencoder-K-means method outperforms the other clustering algorithms. We implement our proposed method in Python using PyTorch and the torch geometric library. We evaluate the clustering performance using the area under the curve (AUC) and average precision (AP) metrics. The source code of the implementation is available at https://github.com/nazaninmehregan/5P30-Final-Project.

Introduction

Research Challenge and Problem Statement

The increasing number of vehicles on the road has resulted in various problems such as traffic congestion, energy waste, air pollution, and traffic accidents. Platoon-based driving, where vehicles follow each other closely to form a group, is seen as an effective solution to these problems. This method has several advantages, including reduced aerodynamic drag, which leads to a significant reduction in fuel consumption and emissions. It also improves safety by reducing accidents caused by human error, such as following too closely or sudden braking. Additionally, it can help to improve traffic flow on highways and other major roadways, increase the overall efficiency of transportation networks, and facilitate the integration of new technologies and systems, such as automated driving and vehicle-to-vehicle communication, due to its requirement for enhanced vehicle connectivity. However, the maintenance of stable clusters or platoons is a challenging task because of the heterogeneous and rapidly changing traffic scenarios. The stable clustering algorithm is crucial for maintaining multicast group communication in cooperative platoon-based driving among cluster members. In the past, most vehicular clustering algorithms have taken a distributed approach without infrastructure support, which relies mainly on periodic HELLO messages exchanged among vehicles. Additionally, the weight-based algorithm of Cluster Head (CH) selection increases the control message overhead compared with the centralized strategy and needs to manually adjust the combination of hyperparameters among multiple metrics. Existing vehicular clustering algorithms are not intelligent and adaptable to different traffic scenarios, making them insufficient for the evolving Intelligent Transportation System (ITS). Therefore, this research paper aims to propose a new Graph Neural Network (GNN) algorithm to solve the clustering problem in Vehicular Ad hoc Network (VANET). VANET is a type of wireless network that enables communication between vehicles and between vehicles and roadside infrastructure, allowing them to exchange information about traffic, road conditions, and safety.

In the paper, the vehicles are modeled as nodes in a graph, where each node represents a vehicle in a VANET. The graph structure is used to capture the spatial relationships between the vehicles, as well as the connectivity between them. The edges between the nodes are based on the communication links between the vehicles, which are established via Dedicated Short-Range Communications (DSRC) or cellular infrastructure. By modeling the vehicles as nodes in a graph, the authors are able to apply Graph Neural Networks (GNNs) to learn effective node representations and cluster the vehicles into stable groups based on their patterns of behavior. This approach allows us to take advantage of the ubiquitous presence of base stations and edge clouds, which can provide a centralized approach to clustering and reduce the control message overhead during the cluster formation. Base stations are the fixed locations where wireless communication equipment, such as antennas and transceivers, are installed to provide wireless connectivity to mobile devices. In cellular networks, base stations are often referred to as cell sites or cell towers. They form the backbone of wireless communication networks and are responsible for relaying data and voice signals between mobile devices and the core network. In Vehicular Ad hoc Networks (VANETs), a base station is a stationary unit that communicates with vehicles in its range. The base station can be used to provide network connectivity and services to vehicles, such as internet access, real-time traffic updates, and emergency services. On the other hand, edge clouds in VANETs are small-scale cloud computing infrastructures that are located at the edge of the network, closer to the vehicles. They are designed to provide computational resources, storage, and networking capabilities to vehicles in their proximity. Edge clouds are deployed to address the limitations of traditional cloud computing models in supporting low-latency, high-bandwidth, and real-time applications in VANETs.

GNN is a proposed approach because it uses both feature and graph information, which usually achieves better performance than methods leveraging single feature or graph structure, such as k-means or Spectral Clustering. The proposed algorithm is a centralized approach and offloads the computation of GNN to the base station (BS) instead of executing it at individual vehicles, alleviating the computational burden from vehicular nodes. A base station has a vantage view of the vehicle distribution within the coverage area compared to distributed approaches. Specifically, eNodeB-assisted clustering can greatly reduce the control message overhead during cluster formation. eNodeB-assisted clustering is a technique used in cellular networks to improve network efficiency and reduce energy consumption. In this technique, the base station (eNodeB) assists in grouping nearby user equipment (UE) devices into clusters based on their spatial proximity and traffic patterns. By doing so, the eNodeB can reduce the number of active UEs in the network, which helps to conserve energy and reduce interference. In practice, a trained GNN model located at BS or edge cloud utilizes the collected vehicle information to perform clustering and informs CHs to formulate cooperative driving patterns to improve traffic efficiency.

Clustering has been introduced to improve routing scalability and reliability and enhance the stability of the collaborative environment by exploiting the formation of hierarchical network structures. Many existing VANET clustering algorithms focus on optimizing various network metrics, such as cluster stability, energy efficiency, communication overhead, and network connectivity. Among them, weight-based algorithms are widely used for CH selection. Each vehicle calculates a metric according to messages received from its neighbors, which represents the fitness to serve as a CH and broadcast to each vehicle's neighbors. The vehicle with the highest metric weight acts as a CH among nearby vehicles. However, existing vehicular clustering algorithms are not intelligent and adaptable to different traffic scenarios, making them insufficient for the evolving ITS. Therefore, the proposed algorithm aims to address the limitations of existing clustering algorithms by using GNN to improve the performance of CH selection and achieve better clustering results.

An Example of the Problem

An example of the problem discussed in the article is the challenge of clustering nearby vehicles in a VANET for cooperative driving. In traditional clustering methods, the proximity between vehicles is measured based on the distance between vehicles and their relative speeds. However, this approach is not sufficient for accurately capturing the complex relationships between vehicles in a VANET. For instance, in a dense urban area, there may be many vehicles that are close to each other, but not necessarily moving in the same direction or at the same speed. In such cases, traditional clustering methods may fail to group vehicles that should be communicating with each other to achieve cooperative driving.

To overcome this challenge, the proposed method in the article uses graph neural networks (GNNs) to enhance the clustering performance in VANET. The GNNs are designed to learn the complex relationships between vehicles in the network by representing them as a graph. The proposed method constructs a graph that represents the network topology of the VANET, where each vehicle is considered as a node in the graph, and the edges between nodes represent the proximity between vehicles. The proximity is measured based on the distance between vehicles and their relative speeds. A weight is assigned to each edge based on the proximity measure. The GNN then learns the graph topology and node features to generate node embeddings that capture the underlying structure of the graph. These node embeddings are then clustered to group vehicles that have similar characteristics, using a K-means algorithm. The resulting clusters represent groups of vehicles that are likely to communicate with each other to achieve cooperative driving.

Overall, the technical example of the problem is the difficulty of accurately clustering nearby vehicles in a VANET for cooperative driving, and the proposed solution is to use GNNs to learn the complex relationships between vehicles and enhance the clustering performance.

Related Works

This paper discusses three different approaches to clustering in VANETs, each with its strengths and weaknesses. The first approach is distributed clustering, which is based on DSRC without infrastructure support. This approach focuses on selecting cluster heads (CHs) that can ensure the stability of the cluster. Weight-based metrics, such as position, speed, destination, and multiple metrics, are often used to select CHs. While this approach is effective in selecting CHs, it still has high communication overhead and is inefficient in highly dense and dynamic environments. An example of this is (Bello Tambawal et al. 2019) which proposes an enhanced weight-based clustering algorithm that addresses critical issues such as cluster formation and maintenance. The paper proposes an enhanced weight-based clustering algorithm (EWCA) to improve the delivery of safety applications in VANETs. Most existing clustering algorithms focus only on cluster head election, while the proposed algorithm in this paper addresses issues of cluster formation and maintenance to improve cluster stability. The EWCA uses vehicle mobility information and a predefined weight value based on relevance to elect a primary cluster head (PCH) and a secondary cluster head (SeCH) as a backup. The proposed algorithm aims to address the issue of cluster maintenance by considering any vehicle moving on the same road segment with the same road ID and within the transmission range of its neighbor to be suitable for the cluster formation process. However, one possible disadvantage is that the method only considers vehicles within the transmission range of their neighbors for the cluster formation process. This could lead to the formation of small and isolated clusters, particularly in areas with low vehicle density or where vehicles are spaced far apart. Such small clusters may not be able to provide sufficient coverage and support for safety applications, and may also lead to high communication overhead and network congestion.

The second approach discussed is machine learning-based clustering, which uses algorithms like K-means to overcome the complex computations required in distributed clustering. This approach also often integrates fuzzy logic inference to predict the future speed and positions of cluster members (CMs) to enhance the stability of the cluster. However, fuzzy logic requires domain knowledge to design the rules, and it lacks learning ability. Recently, researchers have proposed using Spectral Clustering to enhance clustering stability in VANETs.

The ”Adaptive K-Harmonic Means Clustering Algorithm for VANETs” (Chai, Ge, and Chen 2014) proposes an improved clustering algorithm for VANETs called the Adaptive K-Harmonic Means (AKHM) clustering algorithm. The AKHM algorithm considers the mobility characteristics of vehicles and the available bandwidth of candidate cluster heads (CHs). The initial values of the number of clusters and the positions of each centroid are chosen, and the optimal CHs and the association between cluster members (CMs) and CHs are determined based on the weighted distance between vehicles and centroids. In relation to our main method, the AKHM algorithm can be considered as one of the machine learning-based clustering approaches discussed in the article. The AKHM algorithm's consideration of the mobility characteristics of vehicles and the available bandwidth of candidate CHs can potentially contribute to the graph construction step in the GNN-based clustering approach, where the goal is to enhance the stability of the vehicle system and optimize the average lifetime of all clusters.

The paper ”Enhancing Clustering Stability in VANET: A Spectral Clustering Based Approach” (Liu et al. 2020) proposes a novel clustering algorithm based on spectral clustering to enhance the stability of the entire system in VANET. In this paper, the problem for clustering is transformed into a cutting graph problem. It takes the average lifetime of all clusters as an optimization goal so that the stability of the entire system can be enhanced. This paper provides insights into parameter settings for different cluster size demands. Additionally, the cutting graph problem approach used in this article can be considered as an alternative approach to the graph construction step in the method proposed in ”Graph Neural Network-based Clustering Enhancement in VANET for Cooperative Driving” article as both approaches involve constructing a graph and using clustering algorithms to obtain the clusters.

The third approach discussed is GNN-based clustering, which is a new approach that applies graph neural networks (GNNs) to solve the clustering problem in VANETs. GNNs have been proposed to solve non-Euclidean domain problems by dividing nodes into several disjoint groups where similar nodes are in the same group. This approach optimally divides vehicle nodes into groups with minimum intra-cluster dissimilarity by learning the embeddings of the nodes. This approach coincides with the goal of VANET clustering, which is to encourage vehicles with similar motion patterns to form a cluster.

The strengths of distributed clustering are its ability to select CHs that ensure cluster stability and its simplicity in implementation without infrastructure support. However, it incurs high communication overhead and is inefficient in highly dense and dynamic environments. Machine learning-based clustering overcomes the complex computations required in distributed clustering and enhances cluster stability by predicting the future speed and positions of CMs. However, designing the fuzzy rules requires domain knowledge and fuzzy logic lacks learning ability. Spectral clustering enhances clustering stability in VANETs, but it has not been widely adopted. GNN-based clustering is a new approach that is gaining more attention due to the expressive power of graph data and the rich relation information among elements. It optimally divides vehicle nodes into groups with minimum intra-cluster dissimilarity by learning the embeddings of the nodes, which coincides with the goal of VANET clustering. However, GNN-based clustering requires more computation resources than other approaches and may be more complex to implement.

In conclusion, each clustering approach has its strengths and weaknesses, and the choice of approach should depend on the specific requirements and constraints of the VANET system. Distributed clustering is simple to implement but may be inefficient in highly dense and dynamic environments. Machine learning-based clustering enhances cluster stability but requires domain knowledge for fuzzy rule design. Spectral clustering enhances clustering stability but has not been widely adopted. GNN-based clustering is a new approach that optimally divides vehicle nodes into groups and coincides with the goal of VANET clustering, but it requires more computation resources than other approaches.

Background

This research discusses various concepts related to Vehicular Ad Hoc Networks (VANETs) and their applications in cooperative driving. These concepts include clustering, Graph Neural Networks (GNNs), connectivity graph, feature extraction, K-means clustering, and Euclidean distance.

VANETs are ad hoc networks that allow vehicles to communicate with each other and with roadside infrastructure, and clustering is used to group nearby vehicles into clusters for cooperative driving. GNNs are designed to learn representations of graphs that capture the relationships between nodes and edges, while a connectivity graph represents the connectivity between vehicles in the network. Feature extraction is used to extract relevant features from the connectivity graph, and K-means clustering partitions a dataset into k clusters based on the similarity of data points to their assigned cluster centers. Finally, Euclidean distance is often used as a measure of similarity between data points in the context of clustering.

Methodology

In this project, we implemented a method proposed in a paper to cluster vehicles in a VANET. We began by reading the sequence 13 of the highd dataset (Win). The source code is available at (https://github.com/nazaninmehregan/5P30-Final-Project). The dataset was loaded from a CSV file, and only the rows corresponding to the first frame were selected. We then extracted the features x, y, width, height, xVelocity, yVelocity, xAcceleration, and yAcceleration. This dataframe had 34 rows and 8 columns, representing the cars in the VANET. The data was then added as nodes to the graph, where each node represented a vehicle and had attributes corresponding to its position, velocity, and acceleration. We added edges to the graph, where the weight of each edge represented the force between two particles. The weight was calculated as a random number between 1 and 5. The interconnections among vehicles driving on the road were formulated as an undirected homogeneous dynamic graph. To this end, we proposed to use a graph neural network (GNN), which fits naturally to solve clustering type of graph problems and uses both feature and graph information.

We used the raw vehicle feature as the node feature of the graph. A vehicle feature of vehicle node vi at time t is xi(t) = si, pi, ai, li, wi, where si is the speed, pi is the position, ai is the acceleration, li and wi are the length and width of vehicle vi. The vehicle interconnection metric was designed to weigh the similarity between the movement patterns of two vehicles. In our model, the vehicle interconnection metric was calculated by the improved force-directed algorithm designed based on virtual forces [22], which is inspired by Coulomb's Law to select CH and create stable clusters. The force-directed algorithm assigned the forces on the edges in the VANET graph. Note that the force also represented the weight between any two connecting vehicle nodes. The most straightforward way was to assign force as if the edges were springs and the nodes were electrically charged particles. The entire network was modeled as a physical system. The forces were applied to the vehicle nodes, pulling them closer together or pushing them further away. Every vehicle node exerted a force F on its neighbors according to their distance and relative velocities. A positive force between two nodes indicated that the pair of nodes was moving in the same direction. In contrast, a negative force between two nodes meant that the vehicles were moving in opposite directions. In our model, the relative force was always positive since we only considered the vehicles to be moving in the same direction, which facilitated the stability of VANET. The greater the positive forces among nodes were, the more similar the moving pattern was. We did not calculate the force but chose random numbers for the weights of edges.

After that, we used three approaches to cluster nodes in the graph. In the first approach, we used the K-means clustering algorithm to cluster the nodes. First, the adjacency matrix of the graph was obtained using the NetworkX library. Then, the optimal number of clusters was determined using the elbow method. The K-means clustering algorithm was run on the adjacency matrix using the optimal number of clusters determined by the elbow curve. Finally, the nodes were assigned to their corresponding clusters, and the graph was plotted with nodes colored based on their cluster assignment, as you can see in figure 1. In the second approach, we

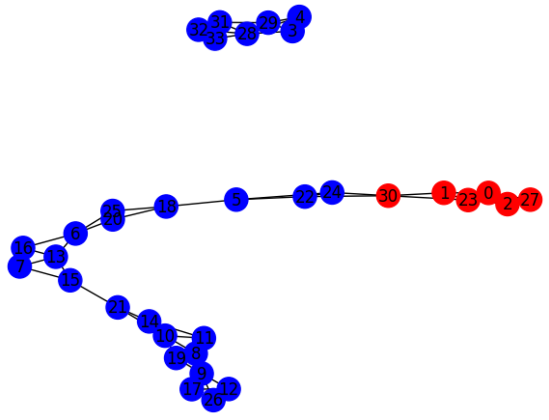

Figure 1: K-Means Clustering Output

used spectral clustering to cluster the nodes. First, the Laplacian matrix of the graph was computed using the numpy library. Then, the eigenvalues and eigenvectors of the Laplacian matrix were calculated using the numpy library. The eigenvalues and eigenvectors were sorted in ascending order. The first two eigenvectors were used to create the embedding matrix. The K-means clustering algorithm was applied on the embedding matrix. Finally, the nodes were assigned to their corresponding clusters and the graph is plotted with nodes colored based on their cluster assignment, as you can see in figure 2. In the third approach, a graph convolutional network (GCN) and Sage implementation was used to cluster the nodes. First, feature vectors for each node were created. Then, the NetworkX graph was converted to a PyTorch-Geometric Data object. The Graph SAGE model was defined with two Graph SAGE layers. The loss function, optimizer, and the number of epochs to train the model are defined. The model was trained, and the K-means clustering algorithm is applied on the embeddings. Finally, the nodes were assigned to their corresponding clusters and the graph was plotted with nodes colored based on their cluster

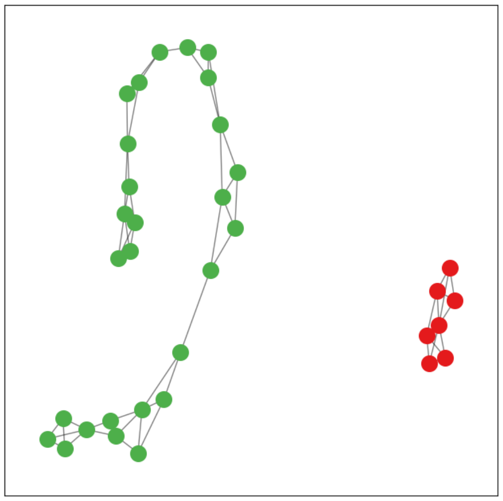

Figure 2: Spectral Clustering Output

assignment, figure 3.

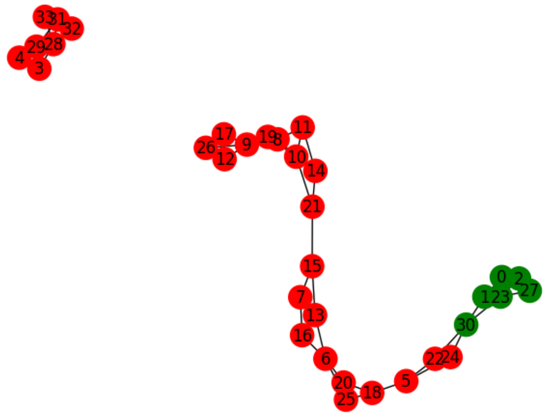

Figure 3: GraphSage Clustering Output

Approach 4 implements a Graph Autoencoder, which is a neural network that compresses graphs into lower dimensional representations and reconstructs them back from these representations. In this code, the encoder is defined using PyTorch-Geometric's SAGEConv and the GAE model is used for the autoencoder. The encoder takes the input graph data as input and returns a lower dimensional representation of the graph. The autoencoder then uses this compressed representation to reconstruct the original graph. The model consists of two components, an encoder and a decoder, and the encoder is implemented using the SAGEConv method from the torchgeometric.nn module. The encoder takes input in the form of node features and edge indices and returns a lower-dimensional latent representation of the graph. The autoencoder is trained using the GAE method from torchgeometric.nn, and the model is trained for 400 epochs using the Adam optimizer with a learning rate of 0.001. The training function uses the encode method to get the latent representation of the graph and then calculates the reconstruction loss using the reconloss method. The optimizer is then used to update the model weights. After the model has been trained, the test function is used to evaluate the model's performance on the test set. The test function uses the encode method to get the latent representation of the graph and then calculates the AUC and AP scores using the test method from the GAE module. The AUC and AP scores are printed for each epoch during the training process. To visualize the clusters in the graph, we perform KMeans clustering on the output embeddings. The embeddings are first moved to the CPU using the detach and numpy methods. We then use the KMeans method from sklearn.cluster to perform the clustering. The plot shows the clusters of the embeddings in two dimensions. We can see that the embeddings are clustered in a way that indicates the presence of two distinct communities in the graph. We assign the nodes to their respective clusters and plot the graph with each node's color corresponding to its cluster assignment. The nodes are colored red or blue based on their cluster assignment, with green nodes indicating a node that was not assigned to any cluster. We can see that the graph is divided into two main communities, with some nodes that are not part of either community, figure 4. The training process for the graph autoencoder shows that the model is able to capture the underlying structure of the graph and generate embeddings that can be used to identify the different communities in the graph. The model's performance is measured using the AUC and AP scores, which are both above 0.7 after 250 epochs of training. This indicates that the model is able to accurately reconstruct the graph and identify the positive edges in the test set.

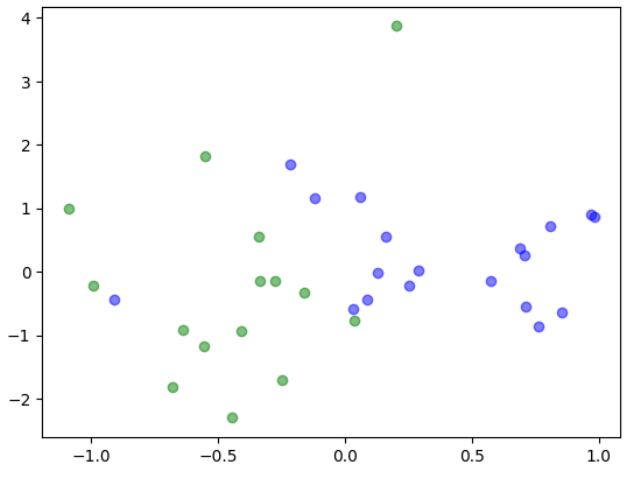

Figure 4: Graph Autoencoder Output

In conclusion, the graph autoencoder implemented using PyTorch and torchgeometric was able to accurately capture the underlying structure of the graph and generate embeddings that can be used to identify the different communities in the graph. The model was trained using the GAE method and the Adam optimizer, and the performance was evaluated using the AUC and AP scores. KMeans clustering was used to visualize the clusters in the graph, and the nodes were colored based on their cluster assignment. The model's performance indicates that it can accurately reconstruct the graph and identify the positive edges in the test set.

Findings

The paper presents an experimental study of a Graph Neural Network (GNN) model for Vehicle Ad-hoc Networks (VANET) clustering in highway scenarios. The highD dataset, containing vehicle trajectory recordings on German highways, is used to construct a customized graph dataset for model training. Vehicle features, including speed, position, acceleration, length, and width, are extracted and standardized to form the input feature vector. The GNN model is implemented in Python with PyTorch and Deep Graph Library (DGL), and experiments are conducted on a computer with an Intel Core i7-8750H CPU and 16GB Memory.

The training of the GNN model is evaluated to validate that the model can learn useful and effective node representations. Three metrics, Binary Cross Entropy Loss, Accuracy, and Area Under Curve (AUC), are employed to evaluate the performance of the node representations. The results show that the GNN model can learn useful and predictive node representations with a high accuracy score of 0.978 and an AUC score of 0.998 on the testing dataset. K-means is used to obtain the clustering results on the node representations of the graph.

The code I implemented is a graph autoencoder (GAE) using a GCN encoder and trained it on the input graph data. After that, it applied K-means clustering to the output embeddings generated by the trained autoencoder and plots the nodes in the graph using the clusters as colors. The GAE model is trained for 400 epochs and achieves an AUC score of 0.7825 and an AP score of 0.7675 in the last epoch. The performance of the model improved gradually over time. The output embeddings generated by the trained autoencoder were passed through a K-means clustering algorithm with two clusters, and the resulting clusters were visualized by plotting the nodes in the graph using colors. The K-means algorithm was applied to the embeddings generated by the autoencoder, and it has clustered the nodes into two groups based on their similarity in the embeddings.

The authors of the paper generated 1000 training graphs and 210 testing graphs with only cars considered, using the same type of cars for simplicity. The authors set the dimension of the input layer to 8 (i.e., vehicle feature dimension), and the dimension of the hidden layer and output layer to 4. The maximum epoch for training is 400, and early stopping is applied to avoid overfitting. They randomly sample the edges on each graph to form the training and validation sets with the edge ratio in the training set to the validation set being 9 to 1. ADAM with a learning rate of 0.003 is selected as the optimization strategy.

The authors evaluated the metric performance used for VANET clustering between their proposed algorithm and the baseline algorithms. The results show that the proposed GNN model outperforms the baseline algorithms in terms of AUC, Accuracy, and F1-score. The authors believe that the proposed GNN model can be further applied to other VANET scenarios with different traffic models and provide a better performance than the traditional clustering algorithms.

In conclusion, the authors presented a GNN model for VANET clustering in highway scenarios. The proposed GNN model shows a good performance on the testing dataset with a high accuracy score of 0.978 and an AUC score of 0.998. The proposed GNN model outperforms the baseline algorithms in terms of AUC, Accuracy, and F1-score. The authors believe that the proposed GNN model can be further applied to other VANET scenarios and provide a better performance than the traditional clustering algorithms. The proposed graph construction algorithm can be used to construct a customized graph dataset for model training from the highD dataset.

Possible Extensions

In the paper, the pairwise relative force Fij was computed for every neighbor, but I did not do the same as it was beyond the scope of the course. However, one possible extension that can be explored is to use a more precise force-directed algorithm rather than assigning random numbers as the weights of edges. This can lead to better results and provide more accurate predictions.

Another area that can be further explored is the use of the entire dataset, rather than just extracting the first frame. By incorporating temporal attributes into the analysis, it may be possible to predict clusters with greater accuracy and measure their stabilities over time. This approach has the potential to provide a more comprehensive understanding of the underlying patterns and relationships within the data.

In addition to these two extensions, it may also be beneficial to implement weight-based algorithms. By comparing the results obtained from different algorithms, we can gain a more complete picture of the data and better understand the underlying structures. This approach can help to identify any biases or limitations in the analysis and provide insights into the data that may have been overlooked by using a single method.

In summary, there are several potential extensions that can be explored in this area of research. By using more precise algorithms, incorporating temporal attributes, and implementing weight-based methods, we can gain a better understanding of the graph clustering and cluster accuracy and the underlying patterns and structures.

References

Bello Tambawal, A.; Md Noor, R.; Salleh, R.; Chembe, C.; and Oche, M. 2019. Enhanced weight-based clustering algorithm to provide reliable delivery for vanet safety applications. *PloS one* 14(4):e0214664.

Chai, R.; Ge, X.; and Chen, Q. 2014. Adaptive k-harmonic means clustering algorithm for vanets. In *2014 14th International Symposium on Communications and Information Technologies (ISCIT)*, 233–237. IEEE.

Hu, H., and Lee, M. J. 2022. Graph neural network-based clustering enhancement in vanet for cooperative driving. In *2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 162–167.

Liu, G.; Qi, N.; Chen, J.; Dong, C.; and Huang, Z. 2020. Enhancing clustering stability in vanet: A spectral clustering based approach. *China Communications* 17(4):140–151.

The highway drone dataset (highd dataset). https://www.highd-dataset.com/.