THE FUTURE IS NOW

EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs

MANAI MORTADHA

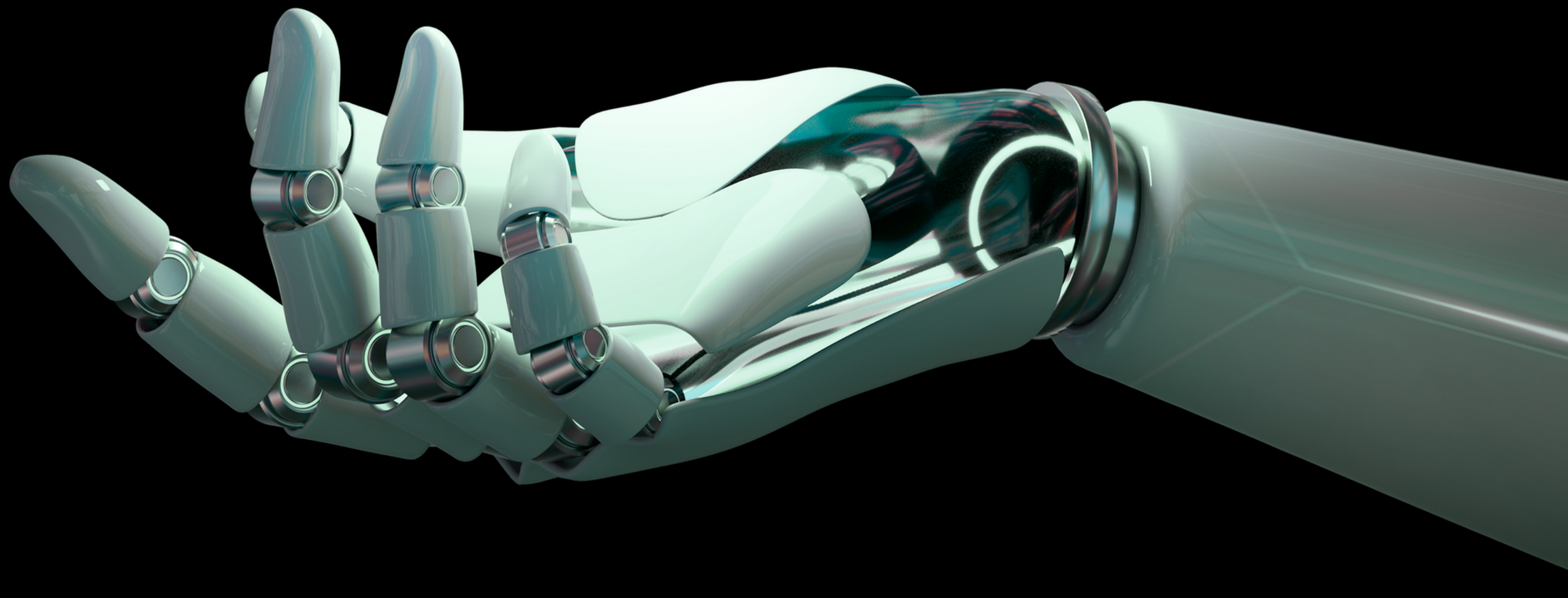
REVIEWER

AGENDA

1. Paper Overview
2. Deep Learning on Graphs : GNN , RNN ,GCN ,GCRN
3. EvolveGCN
4. Evolving Graph Convolution Unit (EGCU)
5. Data Sets
6. Some Results (Link Prediction /Edge Classification / Node Classification
7. OPINION !
8. Conclusions

PAPER Overview :

“ EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs ”



THE LANDSCAPE OF NEURAL NETWORKS TAILORED FOR GRAPH-STRUCTURED DATA HAS WITNESSED A REMARKABLE SURGE IN DIVERSE ARCHITECTURES, PROVING THEIR EFFICACY IN VARIOUS APPLICATIONS. HOWEVER, THE PRACTICAL REALITY OFTEN PRESENTS US WITH GRAPHS THAT CONTINUALLY EVOLVE, CHALLENGING THE CONVENTIONAL STATIC APPROACHES. ADDRESSING THIS, THE PIVOTAL QUESTION ARISES: HOW CAN NEURAL NETWORKS EFFECTIVELY NAVIGATE SUCH DYNAMISM? TRADITIONALLY, THE COMBINATION OF GRAPH NEURAL NETWORKS (GNNS) WITH RECURRENT NEURAL NETWORKS (RNNS) HAS BEEN A GO-TO SOLUTION, EMPLOYING GNNS FOR FEATURE EXTRACTION AND RNNS FOR LEARNING THE EVOLVING DYNAMICS FROM THESE EXTRACTED NODE FEATURES. YET, OUR NOVEL APPROACH, EVOLVEGCN, TAKES A DISTINCTIVE ROUTE. INSTEAD OF USING GNNS MERELY AS FEATURE EXTRACTORS, WE EMPLOY RNNS TO DYNAMICALLY EVOLVE THE GNN ITSELF, THEREBY ENCAPSULATING THE DYNAMISM WITHIN THE EVOLVING NETWORK PARAMETERS. THIS APPROACH OFFERS HEIGHTENED FLEXIBILITY, ALLOWING THE NETWORK TO HANDLE DYNAMIC DATA MORE EFFECTIVELY AS NODES NEED NOT PERSIST THROUGHOUT. NOTABLY, EMPIRICAL VALIDATION UNDERSCORES THE SUPERIORITY OF EVOLVEGCN OVER EXISTING METHODS ACROSS DIVERSE TASKS, INCLUDING LINK PREDICTION, EDGE CLASSIFICATION, AND NODE CLASSIFICATION.

Deep Learning on Graphs : GNN , RNN GCN ,GCRN !

HOW THEY WORK?

A Graph Neural Network (GNN) : is a type of deep learning model designed to work with graph data, which consists of nodes and edges. GNNs leverage the graph structure to learn node embeddings, which are vector representations of the nodes in the graph. These embeddings can be used for various tasks, such as node classification, link prediction, and graph clustering

Resume of all the Paper Approach

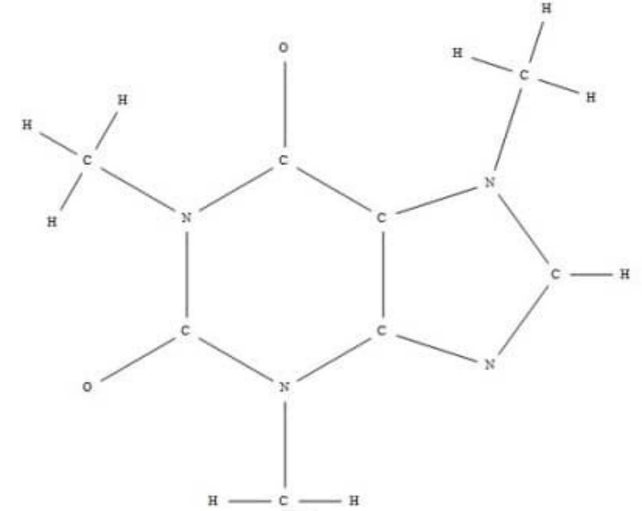
GNNs combine the strengths of graph representation learning and deep learning. They typically involve two main components:

Graph Convolutional Network (GCN): A GCN is a type of GNN that uses a neighborhood aggregation step, inspired by spectral convolution, to update node embeddings. It is simple and effective, making it a popular choice for many graph learning tasks.

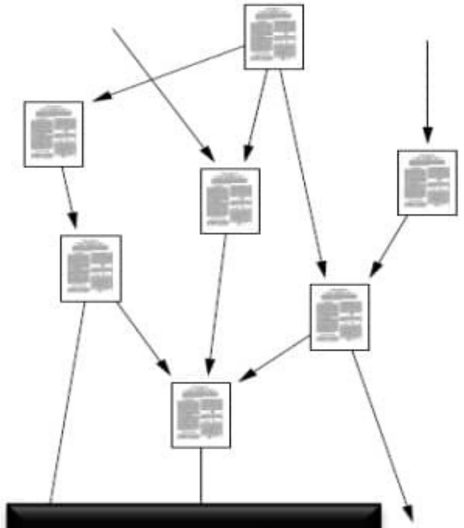
Recurrent Neural Network (RNN): RNNs are used to handle the temporal dynamics of the graph, as they can process sequences of data and update their weights over time

In the context of GNNs, RNNs are often used in combination with GCNs to create models that can adapt to changing graph structures and node features over time. One example of a GNN that handles dynamic graphs is EvolveGCN (Evolving Graph Convolutional Network)

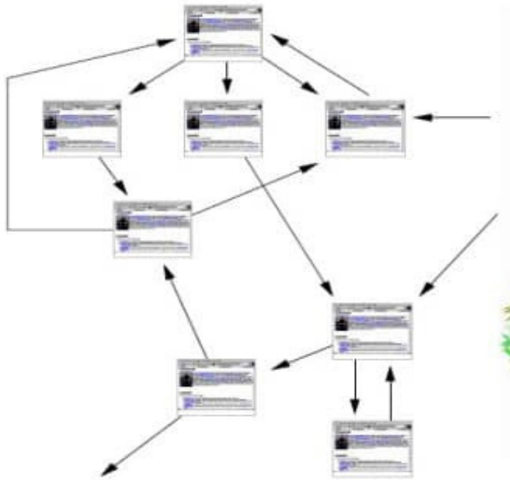
EvolveGCN captures the dynamism of the graph sequence by using an RNN to evolve the GCN parameters, allowing it to adapt to changing node sets and graph structures. This approach has been shown to generally outperform related methods for various tasks, such as link prediction, edge classification, and node classification.



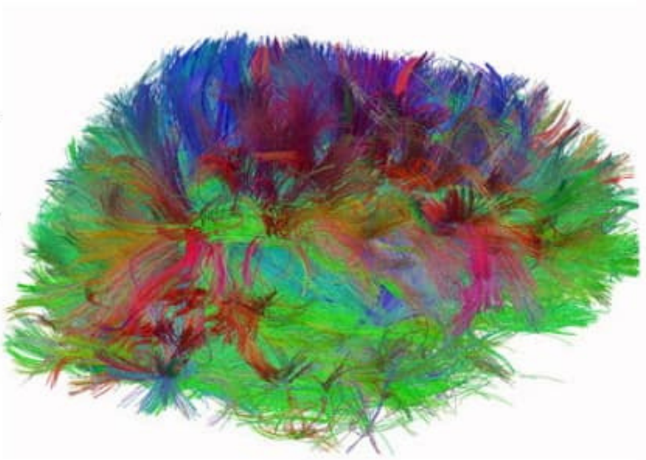
Molecules



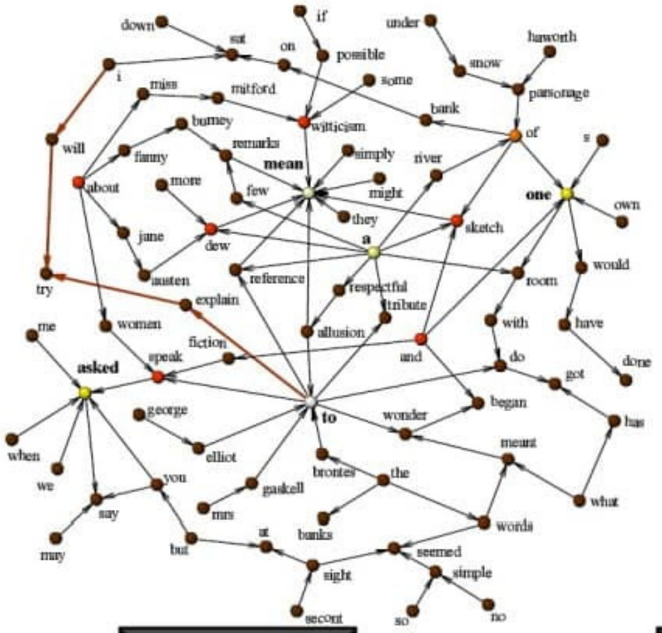
Knowledge



Information



Brain/neurons

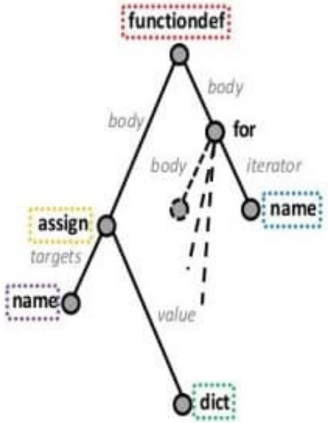


Genes

Communication

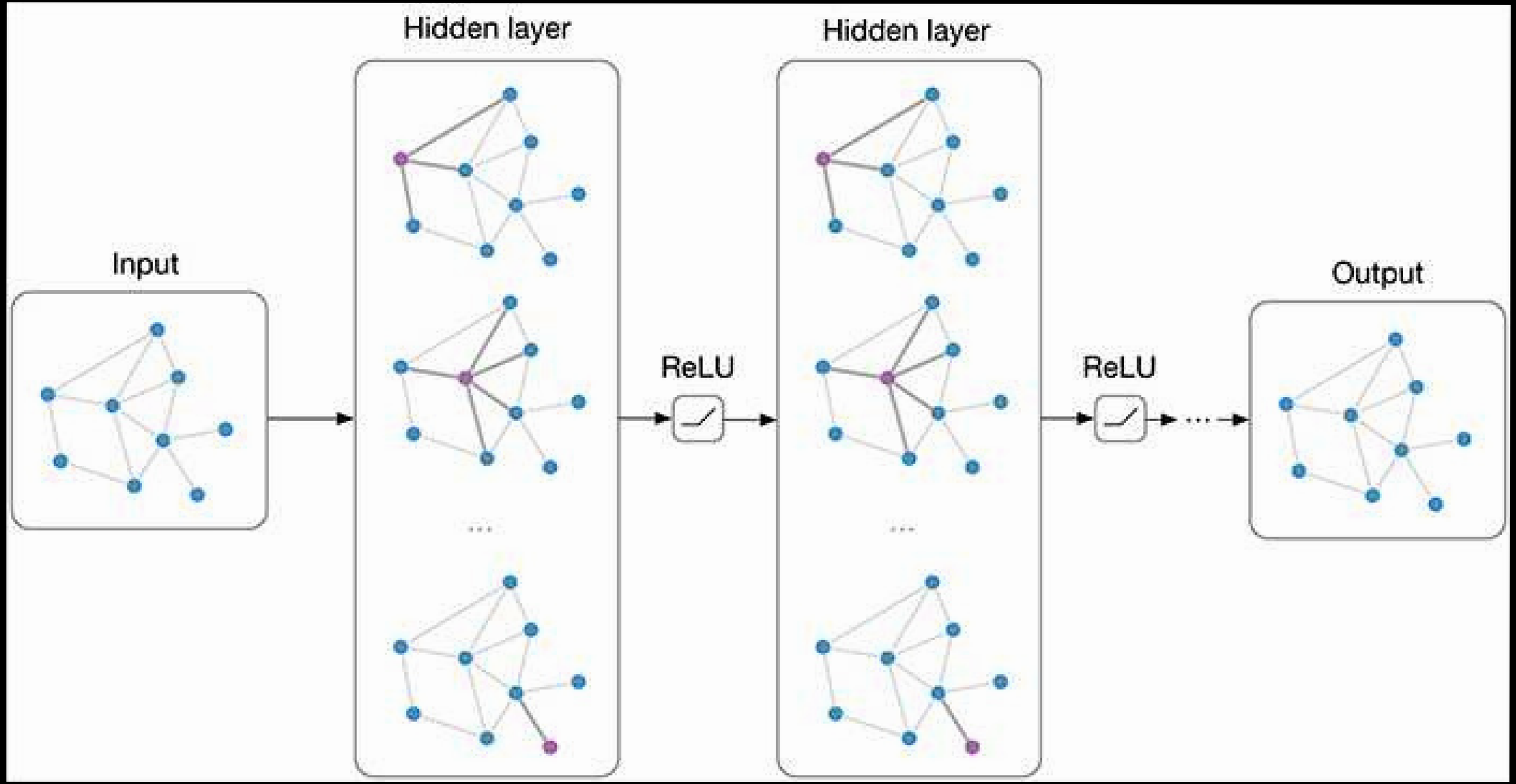
```
def encode(obj):
    """
    Encode a (possibly nested)
    dictionary containing complex values
    into a form that can be serialized
    using JSON.
    """
    e = {}
    for key,value in obj.items():
        if isinstance(value,dict):
            e[key] = encode(value)
        elif isinstance(value,complex):
            e[key] = {'type' : 'complex',
                      'r' : value.real,
                      'i' : value.imag}
    return e
```

```
import ast
tree = ast.parse("...")
...
```

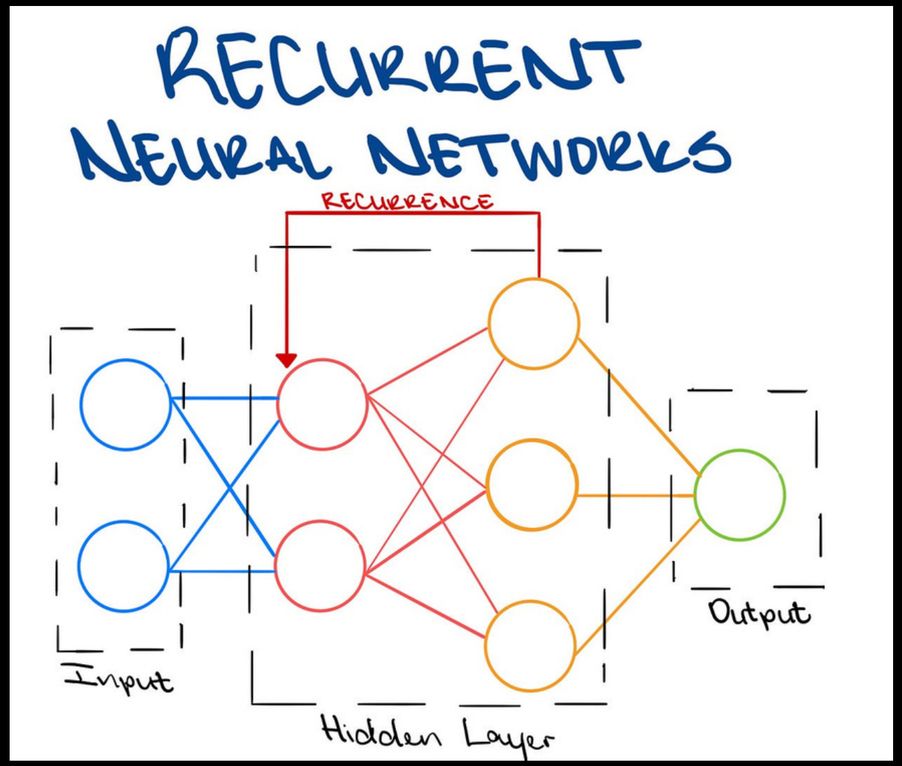


Software

Social


Input
Hidden layer
ReLU
Hidden layer
ReLU
Output

A RECURRENT NEURAL NETWORK (RNN) IS A TYPE OF ARTIFICIAL NEURAL NETWORK THAT CAN PROCESS AND PREDICT DATA IN A SEQUENCE, MAKING IT SUITABLE FOR TASKS INVOLVING SEQUENCES OR TIME-DEPENDENT DATA. RNNS ARE DESIGNED TO MAINTAIN A HIDDEN STATE, WHICH ALLOWS THEM TO REMEMBER AND PROCESS THE INPUT DATA OVER A LONG SEQUENCE. THEY USE A GATING MECHANISM TO CONTROL THE FLOW OF INFORMATION THROUGH THE NETWORK, ENSURING THAT THE NETWORK CAN LEARN AND REMEMBER INFORMATION OVER LONG SEQUENCES



THERE ARE SEVERAL TYPES OF RNNS, INCLUDING:

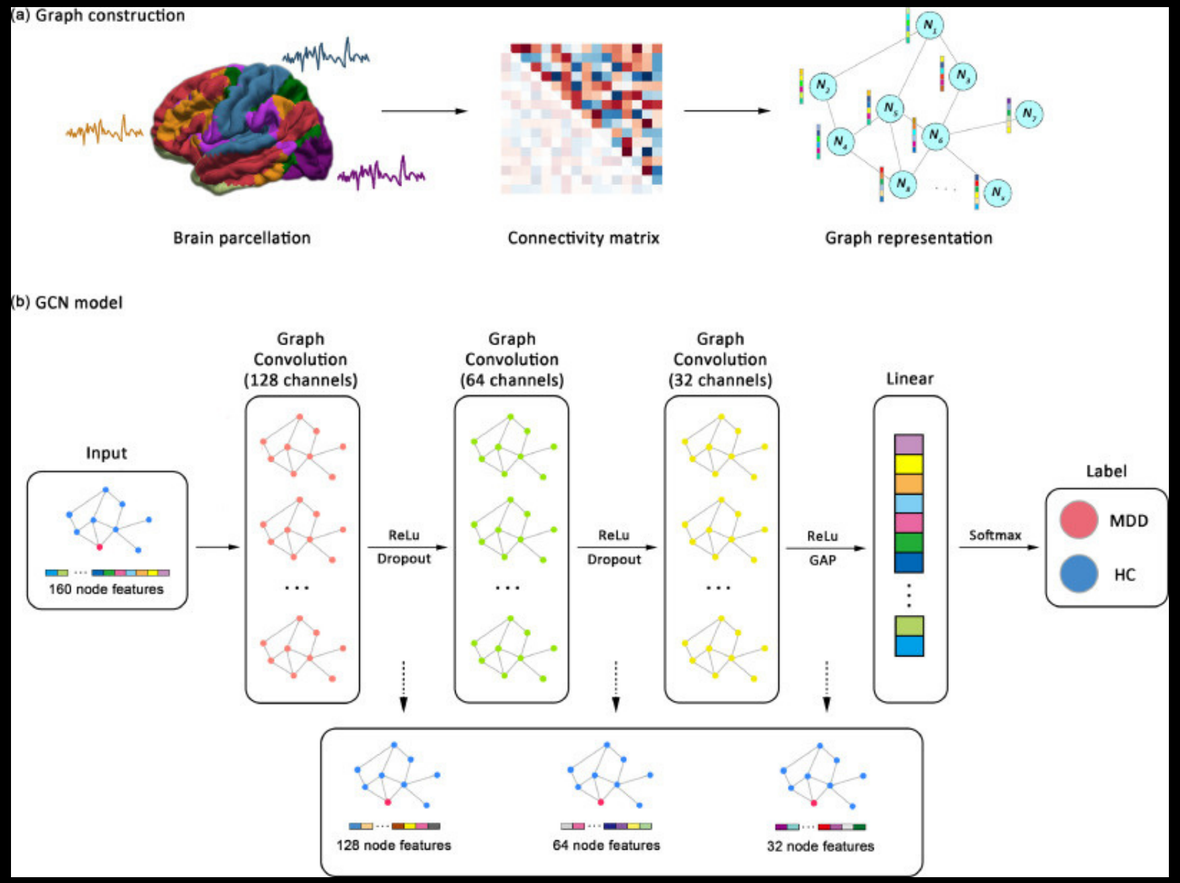
SIMPLE RNN: A BASIC RNN MODEL THAT MAINTAINS A HIDDEN STATE AND PROCESSES INPUT DATA IN A SEQUENCE

LSTM (LONG SHORT-TERM MEMORY): A SPECIFIC TYPE OF RNN THAT USES GATING MECHANISMS TO ALLOW IT TO REMEMBER INFORMATION OVER LONG SEQUENCES

GRU (GATED RECURRENT UNIT): ANOTHER TYPE OF RNN THAT USES GATING MECHANISMS, SIMILAR TO LSTMS, TO CONTROL THE FLOW OF INFORMATION THROUGH THE NETWORK

GRAPH CONVOLUTIONAL NETWORK (GCN):

A GRAPH CONVOLUTIONAL NETWORK GCN:
IS A TYPE OF NEURAL NETWORK SPECIFICALLY DESIGNED TO OPERATE ON GRAPH-STRUCTURED DATA. GCNS ARE TAILORED TO LEARN REPRESENTATIONS OF NODES IN A GRAPH BY CONSIDERING THE RELATIONSHIPS WITH THEIR NEIGHBORING NODES. THEY LEVERAGE CONVOLUTIONAL OPERATIONS SIMILAR TO THOSE IN CONVOLUTIONAL NEURAL NETWORKS (CNNS) BUT ADAPTED FOR GRAPH DATA.



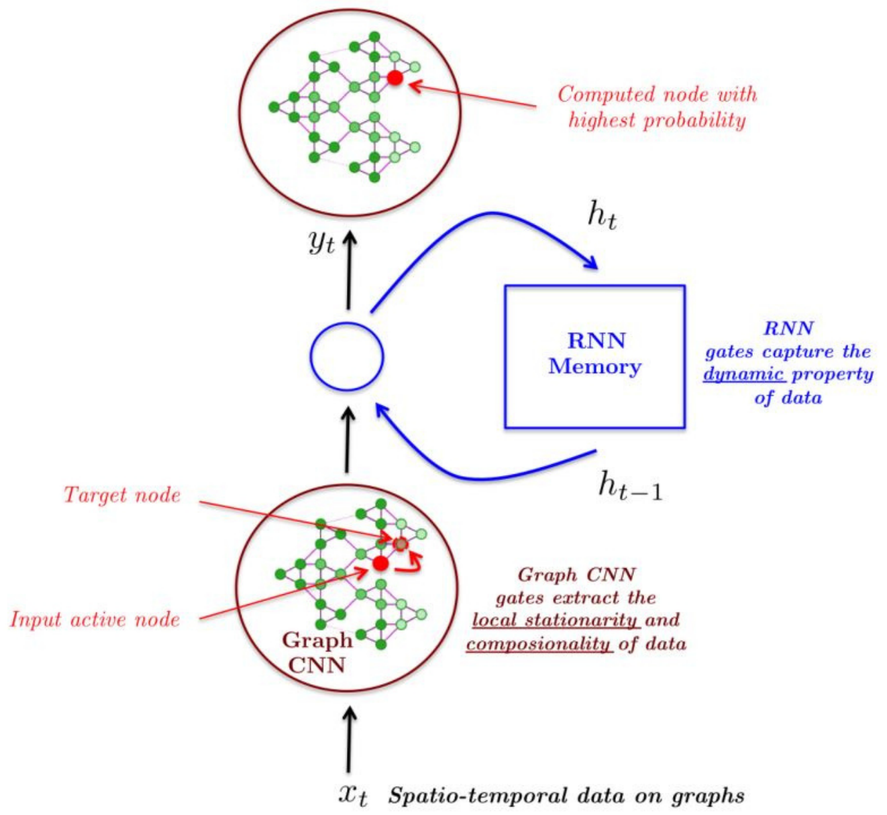


Figure 1: Illustration of the proposed GCRN model for spatio-temporal prediction of graph-structured data. The technique combines at the same time CNN on graphs and RNN. RNN can be easily exchanged with LSTM or GRU networks.

A GRAPH CONVOLUTIONAL RECURRENT NETWORK (GCRN) :

IS A HYBRID ARCHITECTURE THAT COMBINES THE CAPABILITIES OF GRAPH CONVOLUTIONAL NETWORKS (GCNS) WITH RECURRENT NEURAL NETWORKS (RNNS). THIS COMBINATION ALLOWS THE MODEL TO PROCESS BOTH THE TEMPORAL DEPENDENCIES IN SEQUENCES AND THE STRUCTURAL INFORMATION INHERENT IN GRAPH-STRUCTURED DATA.

KEY CHARACTERISTICS OF GCRNS INCLUDE:

1. **SEQUENTIAL AND GRAPH INFORMATION:** GCRNS HANDLE SEQUENCES OF DATA WHILE CONSIDERING THE GRAPH STRUCTURE. THIS IS PARTICULARLY USEFUL IN TASKS WHERE BOTH TEMPORAL RELATIONSHIPS AND GRAPH CONNECTIONS PLAY A CRUCIAL ROLE.
2. **INFORMATION FUSION:** GCRNS FUSE INFORMATION FROM BOTH THE RECURRENT CONNECTIONS (FOR SEQUENTIAL PATTERNS) AND THE GRAPH CONNECTIONS (FOR RELATIONAL INFORMATION) TO LEARN REPRESENTATIONS THAT CAPTURE BOTH ASPECTS EFFECTIVELY.

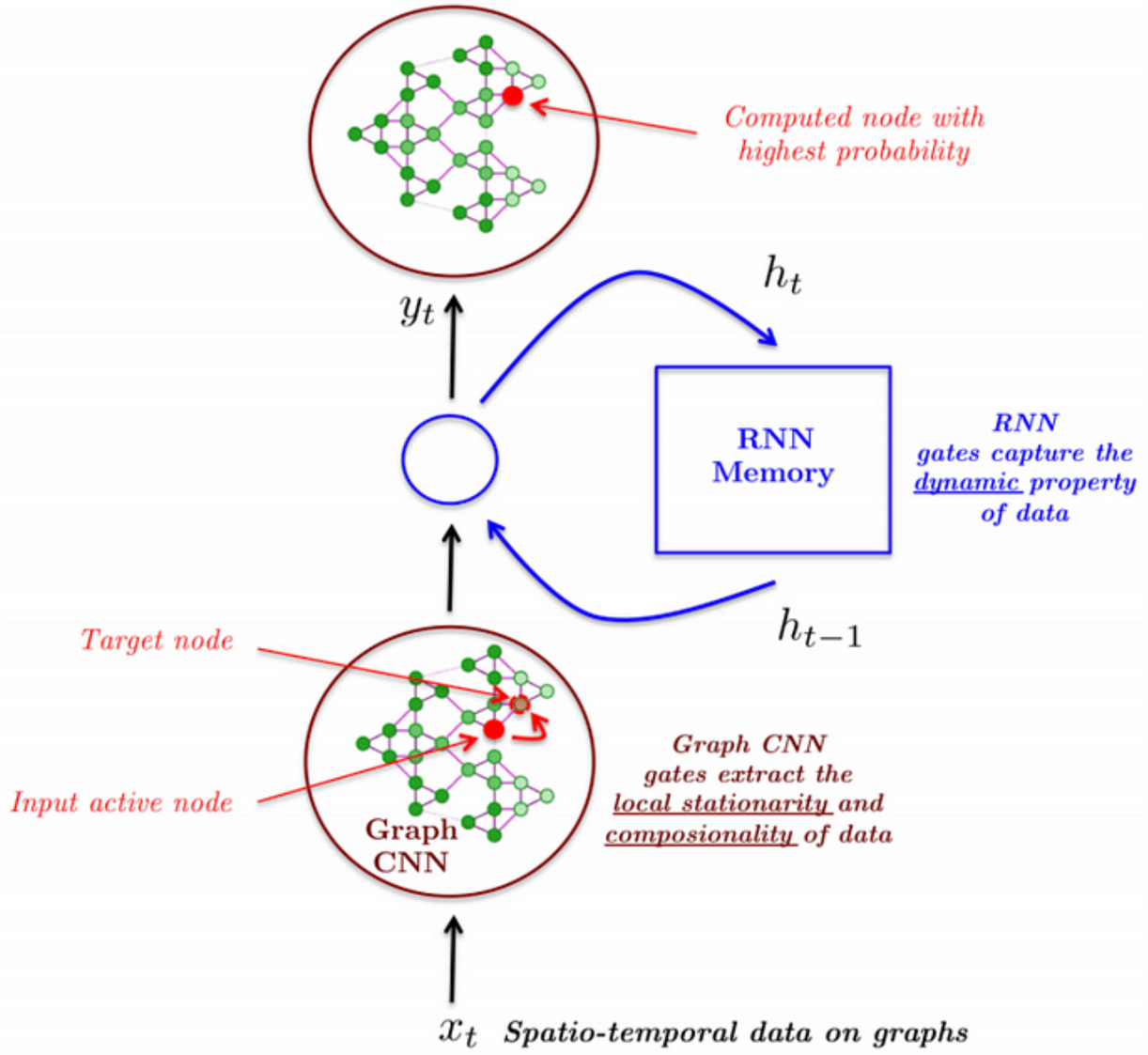
LSTM model

$$
\begin{aligned}
i &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + w_{ci} \odot c_{t-1} + b_i),\\
f &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + w_{cf} \odot c_{t-1} + b_f),\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c),\\
o &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + w_{co} \odot c_t + b_o),\\
h_t &= o \odot \tanh(c_t),
\end{aligned}
$$

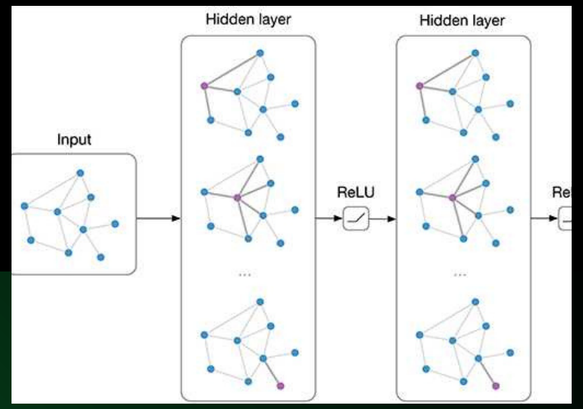
GCRN model

$$
\begin{aligned}
i &= \sigma(W_{xi} *_{\mathcal{G}} x_t + W_{hi} *_{\mathcal{G}} h_{t-1} + w_{ci} \odot c_{t-1} + b_i),\\
f &= \sigma(W_{xf} *_{\mathcal{G}} x_t + W_{hf} *_{\mathcal{G}} h_{t-1} + w_{cf} \odot c_{t-1} + b_f),\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc} *_{\mathcal{G}} x_t + W_{hc} *_{\mathcal{G}} h_{t-1} + b_c),\\
o &= \sigma(W_{xo} *_{\mathcal{G}} x_t + W_{ho} *_{\mathcal{G}} h_{t-1} + w_{co} \odot c_t + b_o),\\
h_t &= o \odot \tanh(c_t).
\end{aligned}
$$

graph convolution

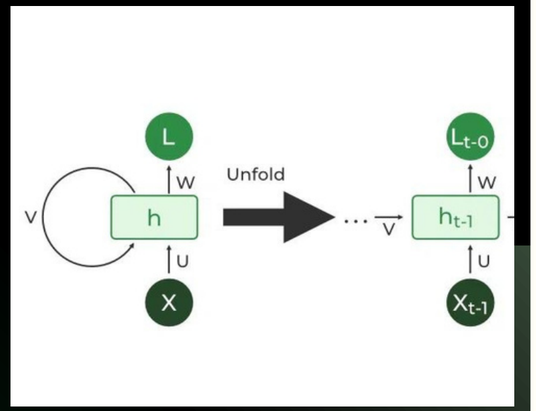


PURPOSE & FUNCTIONALITY !



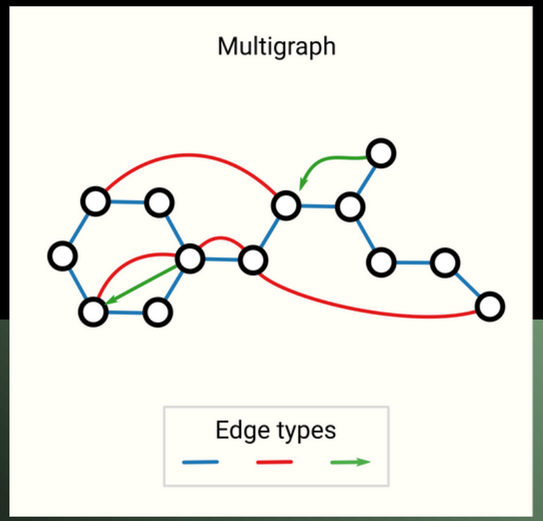
GNN

- Purpose: GNNs are specialized neural networks designed to work with graph-structured data, enabling learning on graphs.
- Functionality: GNNs process nodes and edges in a graph by propagating information through message passing. They learn representations by considering the connections and features of neighboring nodes, enabling tasks like node classification, link prediction, and graph-level tasks.



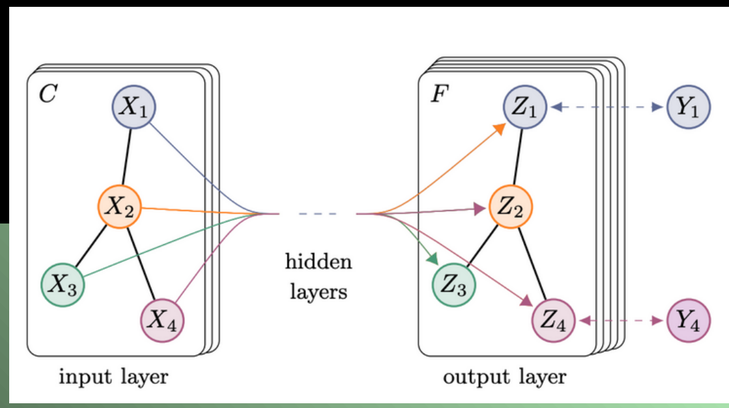
RNN

- Purpose: RNNs are neural networks designed to handle sequential data, allowing them to consider the temporal dependencies in sequences.
- Functionality: RNNs have a feedback loop in their architecture, allowing them to retain information about previous inputs. This makes them suitable for tasks like natural language processing (NLP), time series prediction, and sequence generation.



GCN

- Purpose: GCNs are a type of GNN specifically focused on performing convolutions on graph-structured data, allowing for efficient learning of node representations.
- Functionality: GCNs apply convolutional operations on graphs, similar to how CNNs (Convolutional Neural Networks) operate on grid-structured data like images. They aggregate information from neighboring nodes to update node representations.



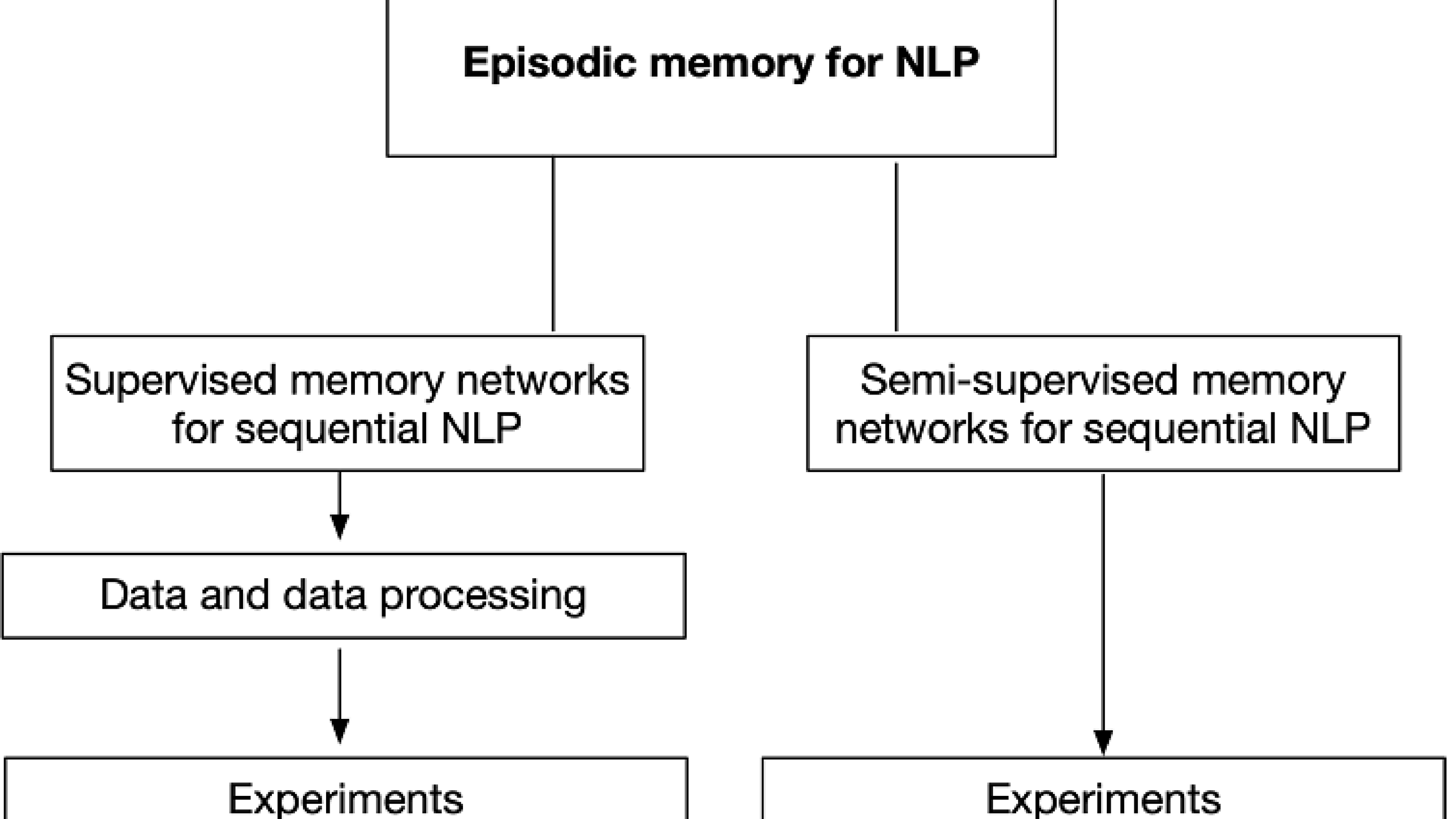
GCRN

Purpose: The primary purpose of a GCRN is to model and analyze data that exhibit both temporal sequences and relationships represented as graphs. By integrating the capabilities of GCNs and RNNs, GCRNs aim to capture and learn from the temporal dependencies in sequential data while also considering the structural information inherent in graph-based data.

Functionality:

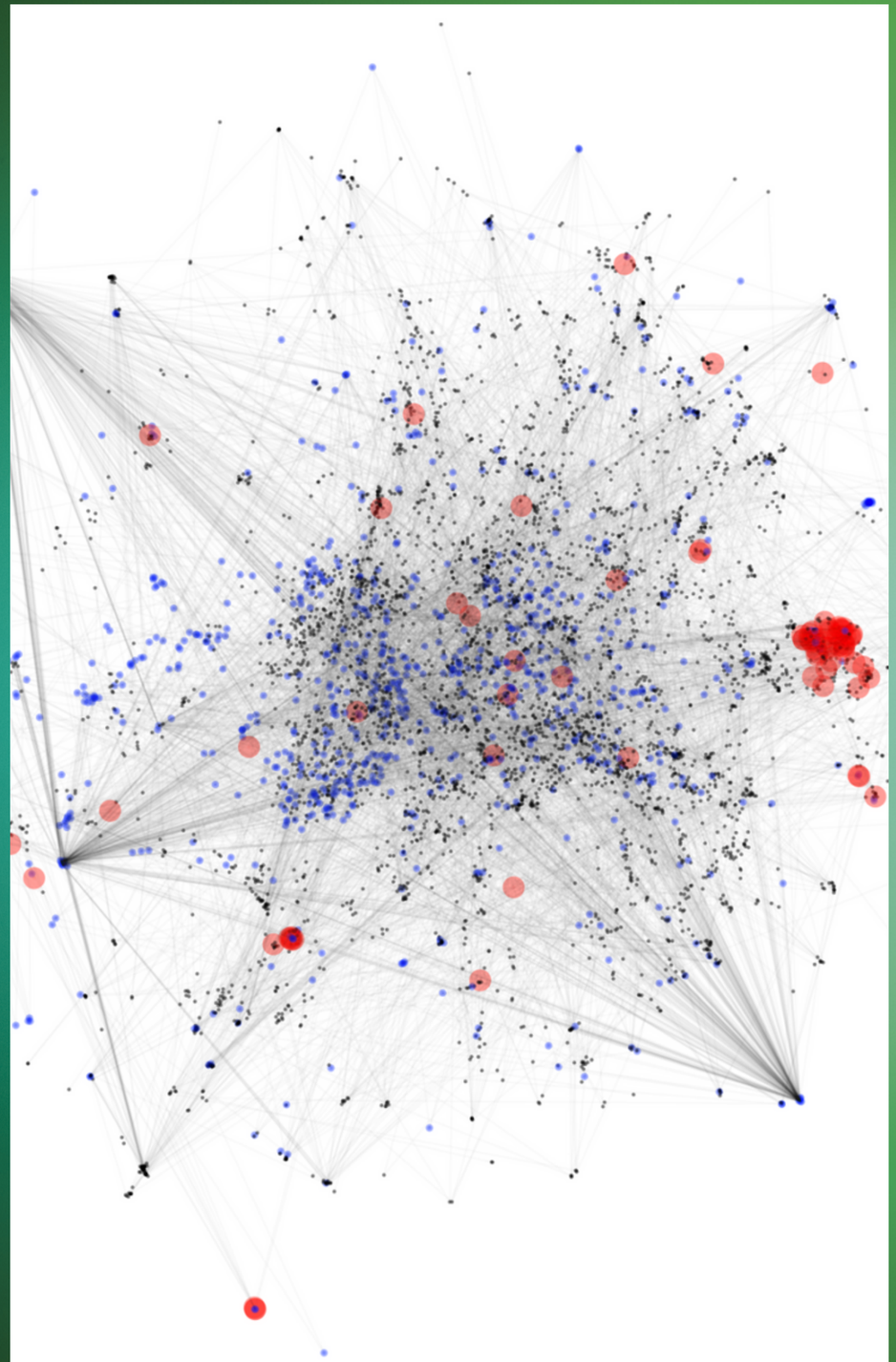
1.Handling Temporal Sequences: GCRNs leverage the recurrent nature of RNNs to capture sequential patterns or time-dependent information present in data sequences. This allows the model to understand and learn from the temporal dynamics within the data.

2. Utilizing Graph Structures: GCRNs utilize the graph convolutional capabilities to process and interpret the relational information encoded in the graph structure. This enables the network to understand and extract features based on the connections between entities in the graph.


Episodic memory for NLP
Supervised memory networks for sequential NLP
Semi-supervised memory networks for sequential NLP
Data and data processing
Experiments
Experiments

EvolveGCN

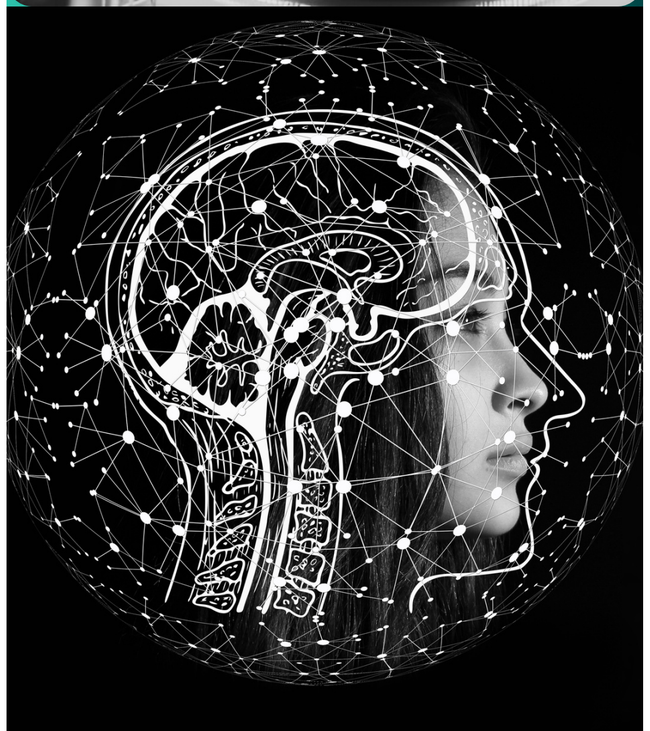
The attached files appear to contain code snippets related to the implementation of the Evolving Graph Convolution Unit (EGCU). The EGCU is a type of neural network layer that adapts the graph convolutional network (GCN) model to handle evolutionary dynamism over time.The first file provides an implementation of the EGCU layer, which uses an RNN to evolve the GCN parameters over time, allowing it to adapt to changing graph structures and node features. The second file provides an example of how to use the EGCU layer in a model for a specific task.The EGCU has been applied in various domains, such as social network analysis, where the graph structure and node features can change over time. By leveraging the strengths of GCNs and RNNs, the EGCU can effectively capture the structural and temporal information of the graph, making it suitable for tasks that involve dynamic graph data.In summary, the EGCU is a type of neural network layer that adapts the GCN model to handle evolutionary dynamism over time. The attached files provide code snippets related to the implementation of the EGCU layer and an example of how to use it in a model.



EVOLVING GRAPH CONVOLUTION UNIT (EGCU): THE ATTACHED FILES PROVIDE CODE SNIPPETS RELATED TO THE IMPLEMENTATION OF THE EVOLVING GRAPH CONVOLUTION UNIT (EGCU). THE EGCU IS A TYPE OF NEURAL NETWORK LAYER THAT ADAPTS THE GRAPH CONVOLUTIONAL NETWORK (GCN) MODEL TO HANDLE EVOLUTIONARY DYNAMISM OVER TIME.THE FIRST FILE PROVIDES AN IMPLEMENTATION OF THE EGCU LAYER, WHICH USES AN RNN TO EVOLVE THE GCN PARAMETERS OVER TIME, ALLOWING IT TO ADAPT TO CHANGING GRAPH STRUCTURES AND NODE FEATURES. THE SECOND FILE PROVIDES AN EXAMPLE OF HOW TO USE THE EGCU LAYER IN A MODEL FOR A SPECIFIC TASK.THE EGCU HAS BEEN APPLIED IN VARIOUS DOMAINS, SUCH AS SOCIAL NETWORK ANALYSIS, WHERE THE GRAPH STRUCTURE AND NODE FEATURES CAN CHANGE OVER TIME. BY LEVERAGING THE STRENGTHS OF GCNS AND RNNS, THE EGCU CAN EFFECTIVELY CAPTURE THE STRUCTURAL AND TEMPORAL INFORMATION OF THE GRAPH, MAKING IT SUITABLE FOR TASKS THAT INVOLVE DYNAMIC GRAPH DATA.IN SUMMARY, THE EGCU IS A TYPE OF NEURAL NETWORK LAYER THAT ADAPTS THE GCN MODEL TO HANDLE EVOLUTIONARY DYNAMISM OVER TIME. THE ATTACHED FILES PROVIDE CODE SNIPPETS RELATED TO THE IMPLEMENTATION OF THE EGCU LAYER AND AN EXAMPLE OF HOW TO USE IT IN A MODEL.

Which Version to Use ??

Choosing the right version is data set dependent. When node features are informative, the -H version may be more effec- tive, because it incorporates additionally node embedding in the recurrent network. On the other hand, if the node fea- tures are not much informative but the graph structure plays a more vital role, the -O version focuses on the change of the structure and may be more effective.



Implementation of the -H Version

The -H version can be implemented by using a standard GRU, with two extensions: (a) extending the inputs and hid- den states from vectors to matrices (because the hidden state is now the GCN weight matrices); and (b) matching the col- umn dimension of the input with that of the hidden state.

Implementation of the -O Version

Implementing the -O version requires only a straightforward extension of the standard LSTM from the vector version to the matrix version. The following is the pseudocode, where note again that all named variables are only local variables and they are not to be confused with the mathematical nota- tions we have been using so far. We use these local variable names so that the reader easily recognizes the LSTM func- tionality.

Data Sets !!

- Stochastic Block Model.
- Bitcoin OTC
- Bitcoin Alpha
- UC Irvine messages
- Autonomous systems.
- Reddit Hyperlink Network
- Elliptic

SOME RESULTS OF :

LINK PREDICTION /EDGE CLASSIFICATION / NODE CLASSIFICATION

WE WILL EXPLAIN THEM DIRECTLY ON THE PAPER WITH THE GRAPHS ! 📈 📊 📉

OPINION !

The paper "EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs" introduces a novel method for handling dynamic graphs in machine learning. The key contributions of the paper are:Problem Statement: The paper addresses the challenge of learning on dynamic graphs, which evolve over time due to changes in node attributes, node connections, and graph topology

Methodology: The authors propose the EvolveGCN (Evolving Graph Convolutional Network) model, which adapts Graph Convolutional Networks (GCNs) to handle dynamic graphs

The EvolveGCN model consists of two main components:A Graph Convolutional Network (GCN) that learns node embeddings based on the graph structure and node features

An RNN that evolves the GCN parameters over time, allowing it to adapt to changing graph structures and node features

Experiments: The authors conduct experiments on various datasets to evaluate the performance of the EvolveGCN model for tasks such as node classification, link prediction, and graph clustering

. The results show that EvolveGCN generally outperforms related methods for these tasks

Impact: The paper demonstrates that the EvolveGCN model can effectively handle dynamic graphs and adapt to changes in the graph structure and node features, making it a valuable tool for various machine learning applications

CONCLUSIONS

IN SUMMARY, THE PAPER "EVOLVEGCN: EVOLVING GRAPH CONVOLUTIONAL NETWORKS FOR DYNAMIC GRAPHS" INTRODUCES A NOVEL METHOD FOR HANDLING DYNAMIC GRAPHS IN MACHINE LEARNING, WHICH HAS THE POTENTIAL TO IMPROVE THE PERFORMANCE OF GCNS IN VARIOUS TASKS. THE AUTHORS HAVE CONDUCTED EXPERIMENTS ON VARIOUS DATASETS TO DEMONSTRATE THE EFFECTIVENESS OF THEIR APPROACH AND HAVE SHOWN THAT EVOLVEGCN GENERALLY OUTPERFORMS RELATED METHODS

THANK YOU FOR YOUR ATTENTION !

@Dr.Mohamed Drira