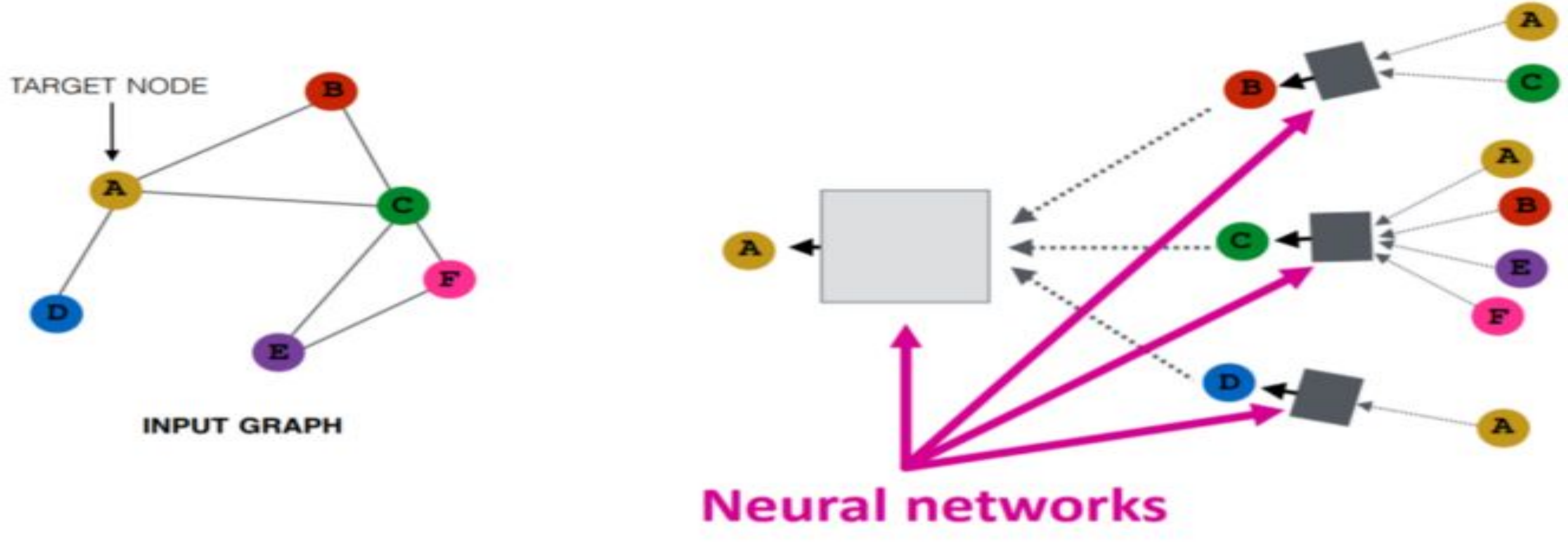
Graph Attention Network for Classification of Financial Knowledge Graphs

Sujit Khanna

Graph Neural Networks: Introduction

- Traditional Deep Learning models are ill equipped for datasets with non-euclidean geometry
- Many real world applications are represented by unstructured datasets
- Biological networks, social networks, financial networks etc. are well represented as graphs
- GNNs extends existing deep neural networks for processing data represented as graphs
- GNNs may potentially excel in applications where traditional ML models fail



Source: https://medium.com/neuralspace/graphs-neural-networks-in-nlp-dc475eb089de

Graph Neural Networks: Applications

- Existing applications of graph neural networks include
  - Node Classification: Classifying all/subset of nodes in a graph
  - Graph Classification: Classifying the entire graph
  - Link Prediction Problems: predicting an edge between two edges
  - Generating Graph Embeddings: High dimensional graph into a low dimensional embedding
  - Generating valid molecules using GraphVAE, Simonovsky and Komodakis [2018]
  - Modelling Stock Relationships, Li et al. [2020]
- The potential of GNNs for Node classification of financial knowledge graphs is explored in detail
- We also discuss the potential direction of future research in this field

Financial Knowledge Graphs

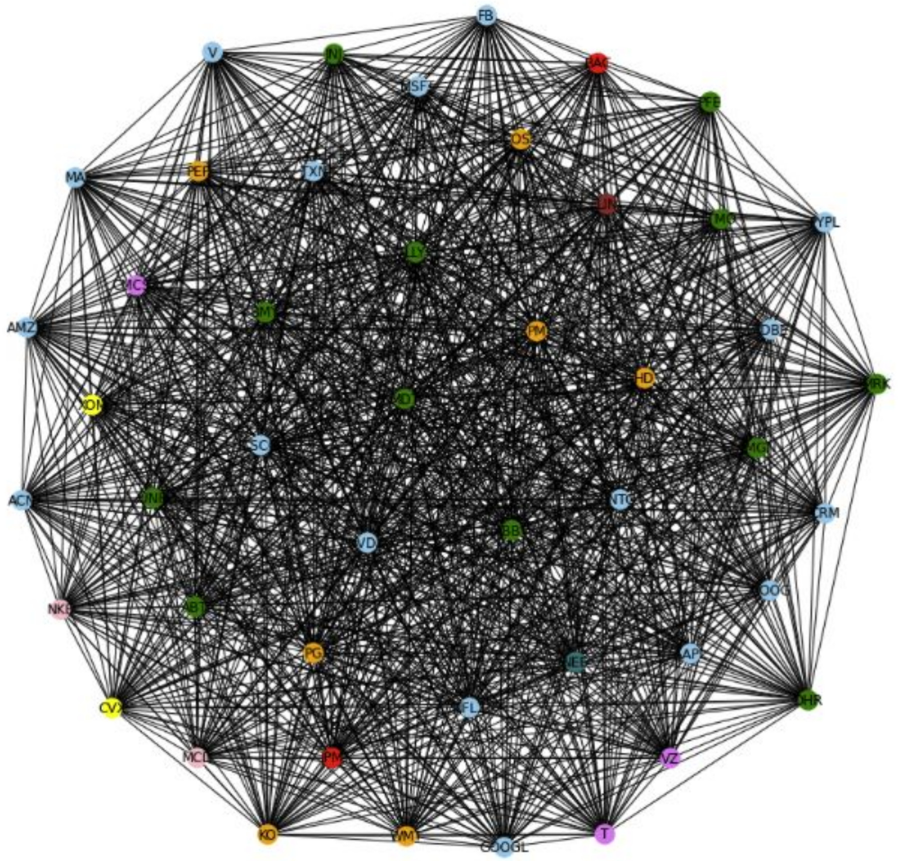
- Financial markets: Multiple players, Stocks, Asset Classes
- The components of financial markets interact with each other in a causal manner
  - i.e. Different components are dependent on each other's behavior
- Such a relationship can be effectively modeled as a network or a knowledge graph
- Financial datasets exhibit high non-linearity and non-euclidean geometry in the
- No known model in academics that can accurately predict financial prices out-of-sample
- Graph based models may offer a better way to model these intricate relationships
- For our problem Individual stocks are nodes, and dependency between stocks are edges
- We aim to predict the label of each node (label indicates future price move)

Nodes, Edges and Labels

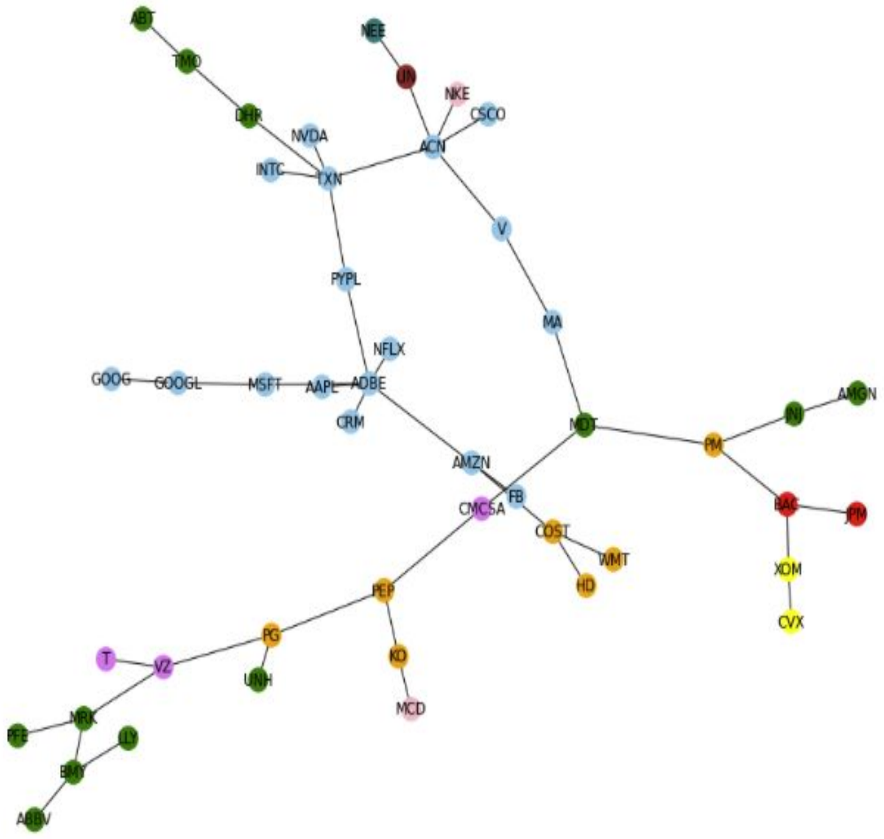
- Dataset:
  - Top 50 largest stocks in S&P 500, Time period: “2017-01-01” to “2021-04-30”
  - Graphs constructed using a rolling window, with lookback=125 days, and lookahead=5 days, i.e. node features and edges are computed using history of 125 days, and labels are constructed based on price moves over next 5 days i.e. day 126 to day 130
- Node Features:
  - Technical Indicators: Relative Strength Index, Moving Average Convergence Divergence, Average Directional Index
  - Fundamental Indicators: Price/Earning per share, Company Ratings
- Edge Features:
  - Dependency between different nodes is captured based on a angular correlation distance given by $d_{i,j}$=sqrt(0.5*(1-abs($\rho_{i,j}$))
  - Higher correlation means shorter distance in a graph
- Node Labels:
  - Node Labels to classify prices over next 5 days, is calculated as (thresh = 0.05%)

$$Label_t = \begin{cases} buy(1) & if\ price_{t+5}/price_{t+1} - 1 > thresh \\ sell(-1) & elif\ price_{t+5}/price_{t+1} - 1 < -thresh \\ neutral(0) & otherwise \end{cases}$$

Fully Connected vs MST

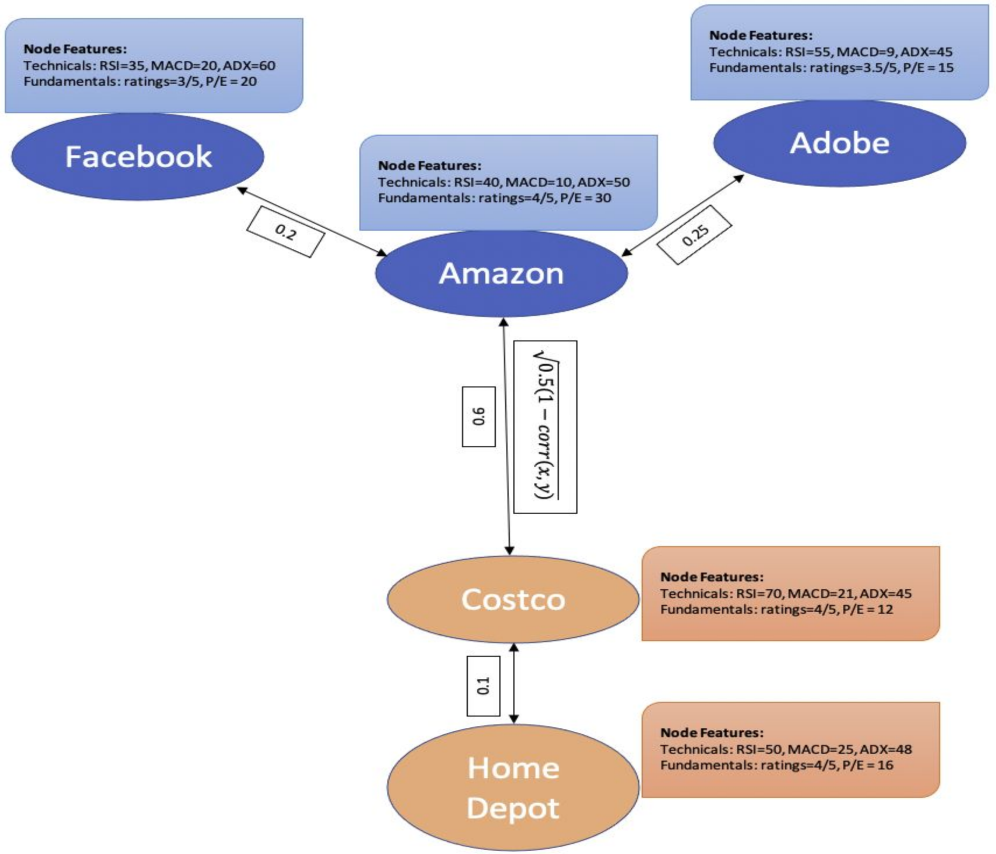


Fully Connected Graph



Minimum Spanning Tree

Exhaustive Graphs Representation

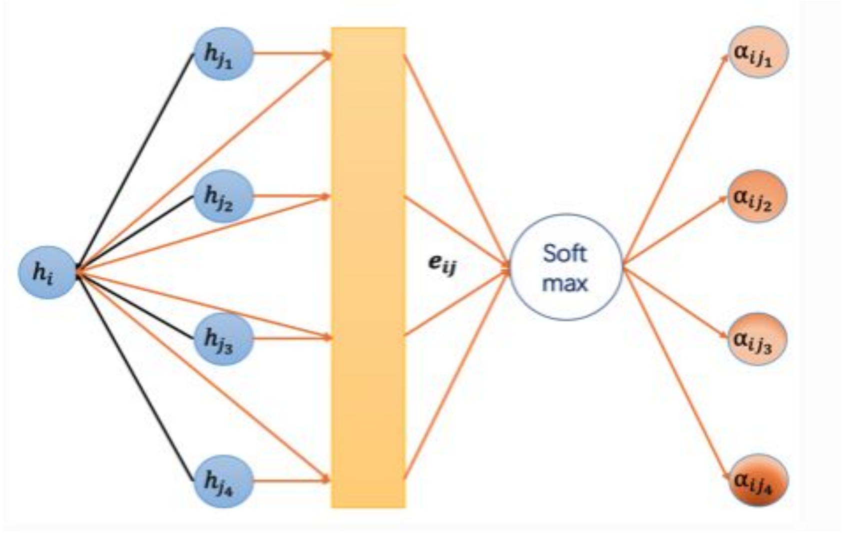

Node Features:
Technicals: RSI=35, MACD=20, ADX=60
Fundamentals: ratings=3/5, P/E = 20
Facebook
Node Features:
Technicals: RSI=55, MACD=9, ADX=45
Fundamentals: ratings=3.5/5, P/E = 15
Adobe
Node Features:
Technicals: RSI=40, MACD=10, ADX=50
Fundamentals: ratings=4/5, P/E = 30
Amazon
0.2
0.25
0.6
$\sqrt{0.5(1 - corr(x, y))}$
Costco
Node Features:
Technicals: RSI=70, MACD=21, ADX=45
Fundamentals: ratings=4/5, P/E = 12
0.1
Home Depot
Node Features:
Technicals: RSI=50, MACD=25, ADX=48
Fundamentals: ratings=4/5, P/E = 16

Sub-Graph visual representation of the final graph structure

- The figure shows the expanded sub-graph of the MST shown in the last slide
- Tech. Stocks are colored Blue, and consumer staples orange
- The angular distance within tech and consumer staple stocks are small
- The distance increase for edge connecting amazon and costco
- Square boxes adjacent to each node represents the feature space of each stock

Graph Attention Network

- Graph Convolutional Networks (GCN) have been traditionally used for node classification
- Shortcomings of GCNs include:
    - Eigendecomposition of adjacency matrix requires matrix inversion
    - Training is numerically unstable and computationally expensive
    - Filters are non-spatially localized
    - Assume a graph structure and therefore fail at transductive problems
- Graph Attention Networks overcomes the above shortcomings, and provides attention mechanism
- The self-attention strategy of GATs makes it an ideal choice for sequential problems like financial time-series data



$$z_i^{(l)} = W^{(l)} h_i^{(l)}, \tag{1}$$

$$e_{ij}^{(l)} = \text{LeakyReLU}(\vec{a}^{(l)^T}(z_i^{(l)} || z_j^{(l)})), \tag{2}$$

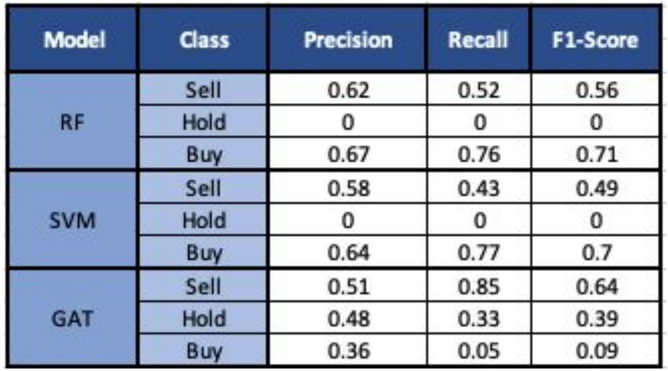
$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}^{(l)})}, \tag{3}$$

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right), \tag{4}$$

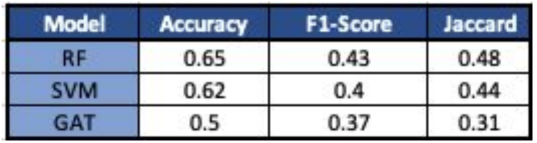
Source: https://docs.dgl.ai/tutorials/models/1_gnn/9_gat.html?highlight=gat

Results

- Train = (2017-01-01 to 2018-12-31), Valid = (2019-01-01 to 2018-08-31), Test Split = (2019-09-01 to 2021-04-31)
- Baseline Models: Random Forest and Support Vector Machines
- Baseline models cannot be directly compared to GAT, but provides a approximate benchmark
- Individual baseline models (i.e. 50) are trained for each node using the same node features, instead of training one broad model to predict price moves of all stocks at a time



Model	Class	Precision	Recall	F1-Score
RF	Sell	0.62	0.52	0.56
	Hold	0	0	0
	Buy	0.67	0.76	0.71
SVM	Sell	0.58	0.43	0.49
	Hold	0	0	0
	Buy	0.64	0.77	0.7
GAT	Sell	0.51	0.85	0.64
	Hold	0.48	0.33	0.39
	Buy	0.36	0.05	0.09



Model	Accuracy	F1-Score	Jaccard
RF	0.65	0.43	0.48
SVM	0.62	0.4	0.44
GAT	0.5	0.37	0.31

Remarks:

- Classic baseline models outperformed GAT by a large margin
- GATs overweighs “Hold” class and underweighs Buy class
- GATs found Task of predicting 50 nodes at a time daunting
- Fully connected graphs may fare better at the cost of computational complexity
- Based on the results, GNNs might be a better fit for graph classification application of financial networks
- Transductive problems are challenging

Pros and Cons of GNNs

- Pros
  - New field, new findings everyday, promising.
  - Seem to work really well for specific applications.
  - High flexibility as GNNs can work on transductive and Inductive problems, as well as on homogeneous and heterogeneous graphs
  - Ability to solve different problems like link, and node predictions
- Cons
  - Little generalized work and current framework is hard to use.
  - Relative works also require progress like dataset.
  - Computing complexity too high (based on current framework).
  - Not suitable for some tasks, not comparative with some other mature models.
  - We shall create some new structures for GNNs, not only try to transplant models from NN.

Conclusion and Future Scope

- GNN fit concrete dataset better, especially when the data points has some kind of relationship between each other.
- However GNNs do not generalize well for all applications exhibiting relational structure and non-euclidean geometry
- Using Fully Connected Structures enables the model to extract more information at the cost of computational complexity
- We believe that future research should focus on making GNNs applicable to a large set of applications
- Reducing computational complexity of GNNs should also be a key focus
- Need more graphical datasets to test generalization and effectiveness of the different GNN architectures
- Future research should also focus on explicitly identifying causal relationships between different nodes of a graph