GCN for Text Classification

[Group:NINJAS]
Shubham
Shaantanu Mishra
Sushil Kumar

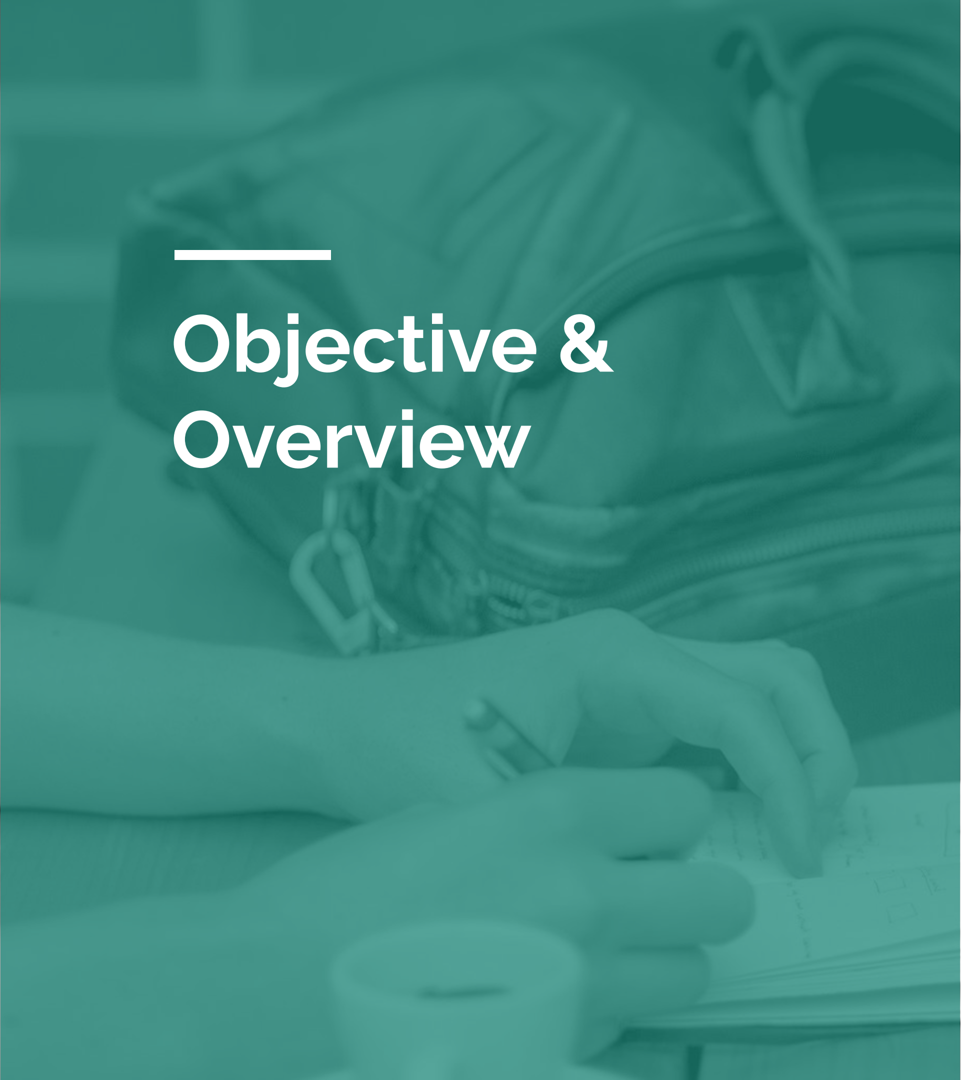
Outline

Introduction

- Text classification is an important and classical problem in natural language processing(NLP).
- The goal of text classification is to automatically analyze text and then assign a set of predefined tags or categories based on its context.
- There are numerous applications of text classification such as document organization, news filtering, spam detection, opinion mining, and computational phenotyping

Related Work

- Traditional Text Classification
- Deep Learning for Text Classification
- Graph Neural Network



Objective & Overview

- Generating Predictive Word & Document Embedding.
- To Classify the text based on the embedding generated(using GCN).

- This paper is about Text Classification using Graph Convolution Network model.
- Whole corpus converted into a single heterogeneous graph
- Edges between word-word and word-document
- Transformed Text classification into a node classification problem.

Applied Method Details.

- Text GCN is just an Improvisation upon GCN idea which was given by kipf and Welling in 2017.
- Text GCN is a semi supervised learning Concept that is able to very accurately infer the labels of some unknown textual data given related known labeled textual data.
- In above Model a Graph is Constructed using Entire Corpus in which both words and Documents were Considered as Node (First study to model whole corpus as Heterogeneous Graph)

continued..

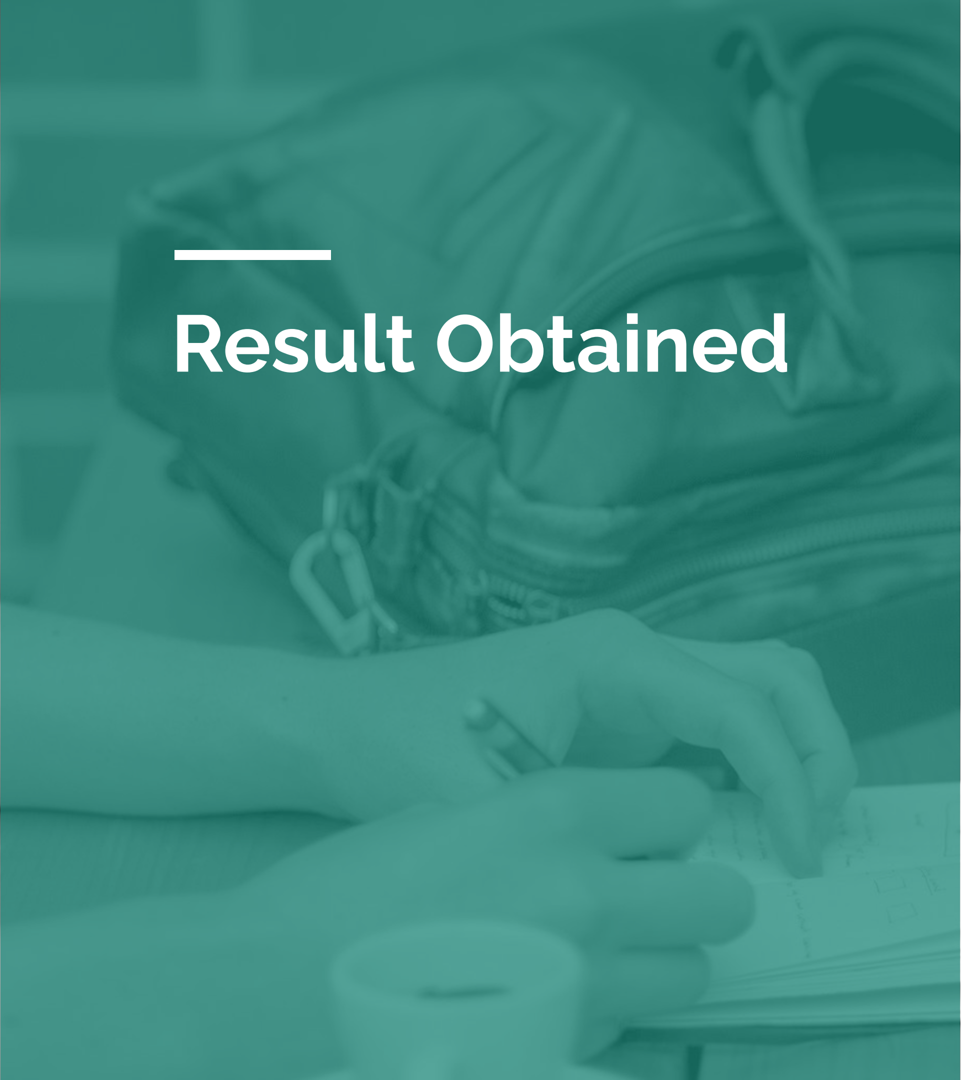
- Relation / weight between word - word and word - document in Adjacency Matrix is established using Relation as Shown in Below :

$$A_{ij} = \begin{cases} \mathrm{PMI}(i,j) & i,j \text{ are words, } \mathrm{PMI}(i,j) > 0 \\ \text{TF-IDF}_{ij} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases}$$

- Building The Text Graph we Simply feed the Graph G that in 2 Layer GCN where 2nd Layer Embedding has the same Size as Label set so that after The Final Feature Vector is Fed in Softmax Classifier we can Classify appropriately.

Implementation Overview

- **DataSet details.**
  - R52 dataset
  - R8 dataset
  - MR dataset
  - 20ng dataset
- **Settings Used**
  - Sliding Window Size is 20.
  - Embedding size of first Convolution layer as 200.
  - Learning rate to 0.02 & dropout rate to 0.5
  - Validation set size is 10% .



Result Obtained

- On R8 dataset Test Accuracy of :0.93731
- On R52 dataset Test Accuracy of:0.86352
- On MR dataset Test Accuracy of :0.76084
- On 20 ng dataset Test Accuracy of :0.97031

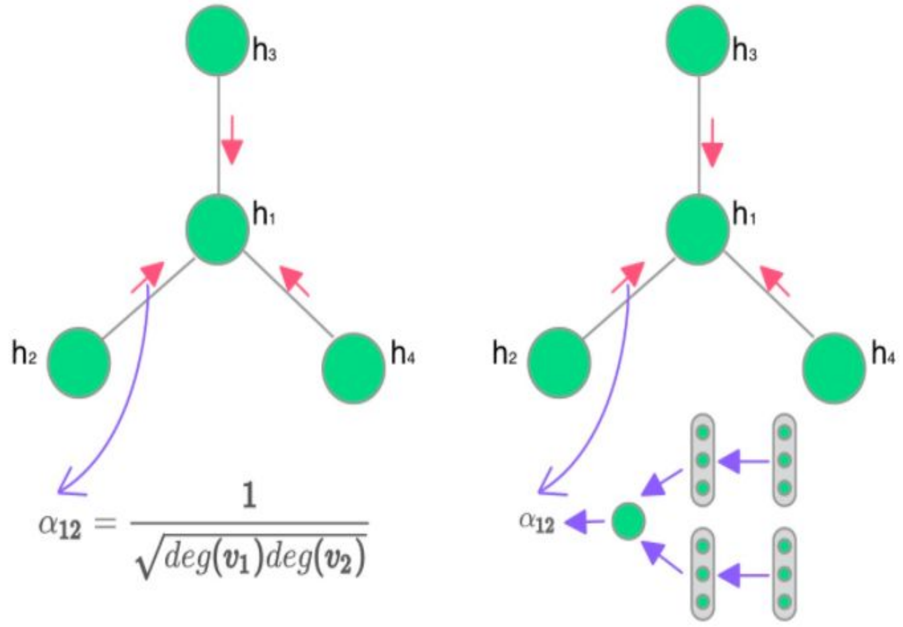
All the testing accuracy obtained are similar to Testing accuracy given in the Paper.

Can we do better than GCNs ?

- From Graph Convolutional Network (GCN), we learnt that combining local graph structure and node-level features yields good performance on node classification task. However, the way GCN aggregates messages is structure-dependent, which may hurt its generalizability.
- GAT (Graph Attention Network), is a novel neural network architecture that operate on graph-structured data, leveraging masked self-attentional layers to address the shortcomings of prior methods based on graph convolutions or their approximations.
- Analyzing and Visualizing the learned attentional weights also lead to a more interpretable model in terms of importance of neighbors.

Can we do better than GCNs ?

GCN vs GAT

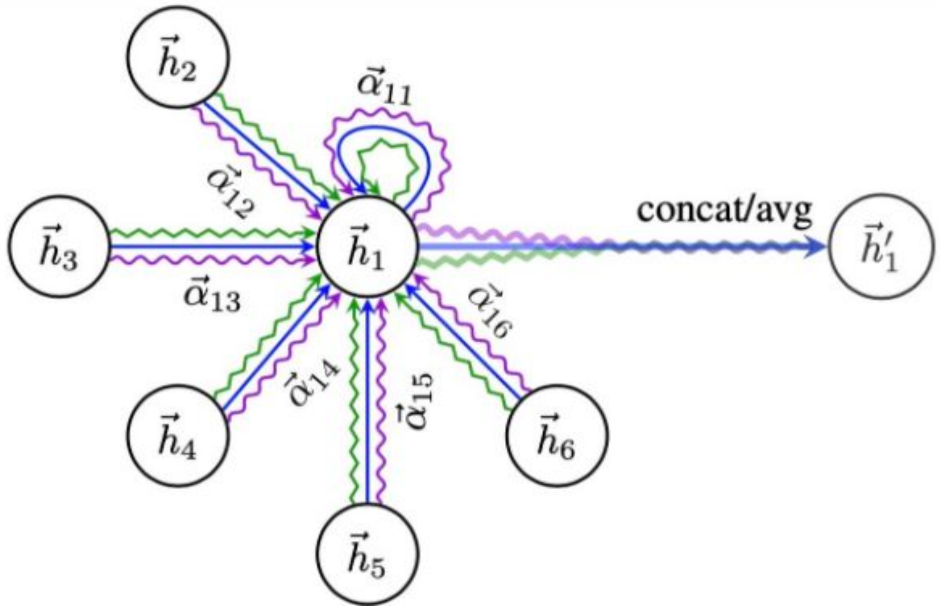


GCN explicitly assigns non-parametric weight $\alpha_{ij} = \frac{1}{\sqrt{deg(v_i)deg(v_j)}}$, via the normalization function during neighborhood aggregation.

GAT implicitly captures the weight α_{ij}, via the attention mechanism, so that more important nodes receive higher weight during neighborhood aggregation.

How does the GAT layer work?

Multi-head Attention



Improvement

- For the improvisation part we have selected only “mr” dataset.
- We used GAT model that uses attentions mechanism and modified it further to make it Compatible for our Dataset to further Improve Accuracy.

Result Comparison:

- Earlier Test Accuracy: 0.7604
- After improvisation Test Accuracy:0.7802

Thank You !!