Graph Convolutional Networks for Text Classification

Team -12
(Networkx)

Prateek Jaiswal - 2021701009
Pavan Baswani - 2021701035
Madan Nandiwada Santhanam- 2018900075

Yao, L., Mao, C. and Luo, Y., 2019, July. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence* (Vol. 33, No. 01, pp. 7370-7377).

Problem Statement

Text classification using Graph Neural Network for english language and extending to indic text. Where we analyze the performance of GNN models with SOTA methods like LSTM, Bi-LSTM, BERT.

Motivation

Text classification using Graph Neural Network and highlight the effectiveness of GNN in text classification.

Challenges

- Validating the datasets.
- Checking the NLP related concepts.
- Complexity involving in the graph construction.
- Checking multiple GNN models for text classification

Graph Construction

Number of Nodes = Number of Documents + Number of Unique words

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

The PMI value of a word pair i, j is computed as

$$\mathrm{PMI}(i,j) = \log \frac{p(i,j)}{p(i)p(j)}$$

$$p(i,j) = \frac{\#W(i,j)}{\#W}$$

$$p(i) = \frac{\#W(i)}{\#W}$$

PMI- Point-wise mutual information
TF-IDF term frequency-inverse document frequency
W Sliding windows count of corpus
W(i) Sliding windows count of the corpus containing the word(i)

Yuan Luo Liang Yao, Chengsheng Mao. Graph convolutional networks for text classification. AAAI Conference on Artificial Intelligence (AAAI 2019), 2019

Graph Neural Network Models

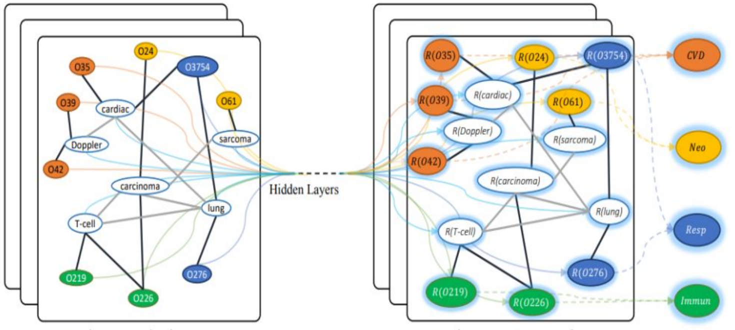
- **GRAPH CONVOLUTION NETWORK**

Graph Convolution Network,GCNs are neural networks operating on graphs and inducing features of nodes (i.e., real-valued vectors / embeddings) based on properties of their neighborhoods. GCN are multi layer neural networks that directly operate on a graph and induces embedding vector of nodes based on properties of their neighborhoods.

.

$$Z=Softmax(\tilde{A}ReLU(\tilde{A}XW_0)W_1)$$

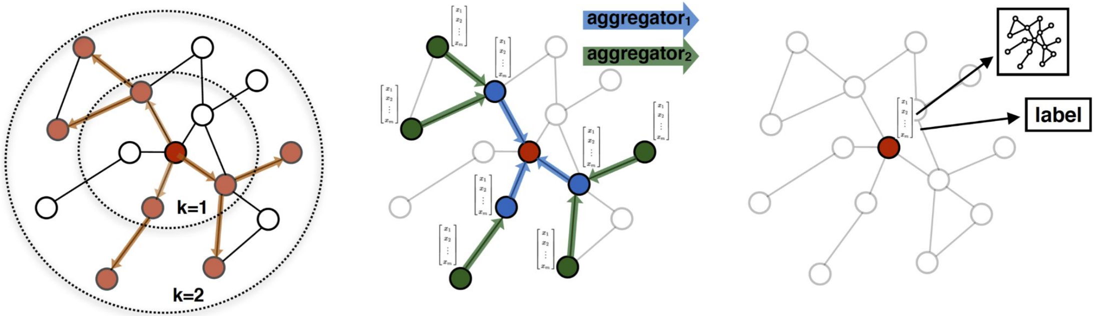
$\tilde{A}=D^{-1/2}A\,D^{1/2}$ (A is adjacency Matrix and D is the degree matrix)



Yuan Luo Liang Yao, Chengsheng Mao. Graph convolutional networks for text classification. AAAI Conference on Artificial Intelligence (AAAI 2019), 2019

Graph Neural Network Models Contd.

- **GRAPHSAGE**
  It is a technique that uses inductive approach to efficiently generate the node embeddings. Here, instead of training each node a function is used that generates embeddings by sampling and aggregation.

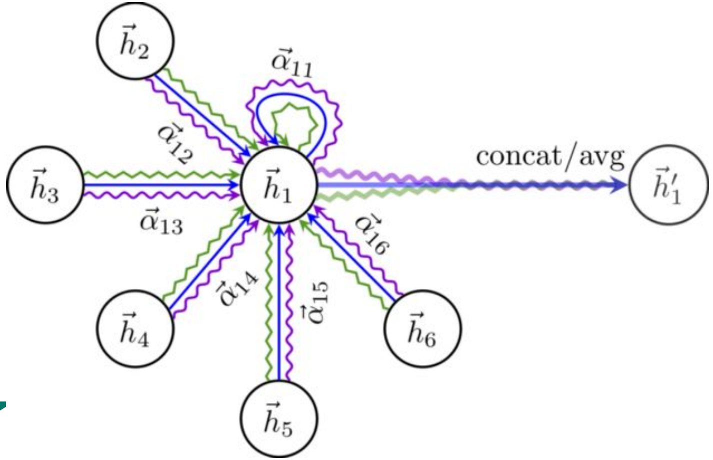


1. Sample neighborhood
2. Aggregate feature information from neighbors
3. Predict graph context and label using aggregated information

http://snap.stanford.edu/graphsage/

Graph Neural Network Models Contd.

- **GRAPH ATTENTION MECHANISM**
  As attention mechanism has become powerful tool in almost all the sequence based task Dealing with variable sized inputs, focusing on the most relevant parts of the input to make decisions are some benefits of using attention mechanism. When an attention mechanism is used to compute a representation of a single sequence is known as self-attention.



Individual contribution to Project

	Task	Work Distribution	Start and End Dates
Phase One	Report preparation	Madan, Pavan, Prateek	(Feb 22 – Mar 8)
	Understanding GCN	Madan, Prateek ,Pavan	
	EDA	Prateek, Pavan	
Phase Two	Baseline	Prateek, Madan & Pavan	(Mar 9 - Mar 22)
	Methodology	Prateek, Madan & Pavan	
	Description	Prateek, Madan & Pavan	
Phase Three	Coding (Baseline)	Prateek, Madan & Pavan	(Mar 23 – Apr 5)
	Comparing with transformer models	Prateek, Pavan	
	Reporting results	Prateek, Madan & Pavan	
	Error analysis	Prateek, Pavan	
	Changing embedding	Prateek, Pavan	
Phase Four	Future works	Prateek, Madan & Pawan	(Apr 6 - Apr 18)
	Conclusion	Prateek, Madan & Pawan	
	References	Prateek, Madan & Pawan	

EXPERIMENTS (DATASETS)

Datasets	Docs	Training	Test	Words	Nodes	Classes	Average length
20NG	18846	11314	7532	42757	61603	20	221.26
R8	7674	5485	2189	7688	15362	8	65.72
R52	9100	6532	2568	8892	17992	52	69.82
Ohsumed	7400	3357	4043	14157	21557	23	135.82
MR	10662	7108	3554	18764	29426	2	20.39
Hindi	38121	36214	1907	17383	42661	5	471.18
Marathi	15884	15099	785	26125	23306	7	742.10
Bengali	20541	19500	1041	43725	30876	8	561.05
Telugu	38526	36599	1927	20674	43231	9	310.98
Tamil	35591	33811	1780	34694	42461	12	543.46
Malayalam	38104	36201	1903	31793	45975	5	708.96
Kannada	36647	34814	1833	35006	44660	9	469.17
Gujarathi	36922	35079	1843	50028	47871	11	545.40

Novel Ideas Worked:

- Indic dataset (8 languages) crawled from the new articles to do text classification..
- GNN's for datasets with less data able to give better accuracy compared to SOTA methos like LSTM, Bi-LSTM, BERT
- GCN always perform better than GAT while doing experimenting .

Parameters:

- For all the models we have trained on early stopping with 100 epochs.
- We have used two layers of GNN with RELU activation.
- Loss function: Cross entropy
- Dropout rate = 0
- DGL graph library for graph creation.
- Learning rate = 0.02
- Window size = 10 (while word pair count)

Model Results (F1-score)

Dataset	LSTM	Bi-LSTM	GCN	GAT	SAGE	BERT
20ng	0.62	0.74	**0.85**	0.42	0.74	0.16
oshumed	-	0.44	0.68	0.87	**0.97**	0.21
R8	0.89	0.95	**0.96**	0.90	0.94	0.93
R52	-	-	0.93	0.87	**0.96**	-
MR	0.72	0.72	**0.77**	0.76	0.76	0.62
Hindi	0.92	0.94	**0.89**	**0.84**	0.54	**0.94**
Telugu	0.85	**0.86**	0.63	**0.73**	0.69	0.54
Kannada	**0.85**	0.84	0.78	0.76	**0.78**	0.76
Bengali	0.53	0.54	0.68	0.73	0.67	**0.76**
Marathi	0.53	0.42	**0.699**	0.695	0.68	0.68
Tamil	0.85	**0.86**	0.81	0.68	0.43	0.82
Malyalam	0.61	0.62	-	**0.6269**	-	0.62
Gujrathi	0.92	0.92	-	-	-	**0.93**

Observations

1. Most of the indic languages (that we have used) don't have the packages to preprocess. Due to unavailability of standard packages, the data contain some noisy text like emojis, urls (in indic script), unnecessary punctuations (which will not match with string.punctuation). This impact on graph construction (as the graph purely constructed on word-word and doc-word relations)
2. Indic-scripts are morphologically rich languages. This result in out of vocabulary words which slight change in the syntax of any word. Also, this impact on graph construction by considering the sub-words instead of words (increasing the number of nodes). This can be resolved with the language specific tokenizer.
3. For some of the indic-languages (hindi, telugu, kannada, and gujarati), we have used almost 1/4th of original data size due to "**Out of memory**" error for all GNN models. Whereas for NLP models we have used entire data size for all languages.
4. Compared to the NLP models, GNN models performing better even though we perform the experiments with very less data.

Visualization