COMP5046
Natural Language Processing

Lecture 12: Advanced NLP: NLP with Graph Neural Networks

Semester 1, 2020
School of Computer Science
The University of Sydney, Australia

THE UNIVERSITY OF SYDNEY

Dr Caren Han
Caren.Han@sydney.edu.au

What we learned in this course!

Week 1: Introduction to Natural Language Processing (NLP)	NLP and Machine Learning
Week 2: Word Embeddings (Word Vector for Meaning)	
Week 3: Word Classification with Machine Learning I	
Week 4: Word Classification with Machine Learning II	
Week 5: Language Fundamental	NLP Techniques
Week 6: Part of Speech Tagging	
Week 7: Dependency Parsing	
Week 8: Language Model	
Week 9: Information Extraction: Named Entity Recognition	Advanced Topic
Week 10: Advanced NLP: Attention and Reading Comprehension	
Week 11: Advanced NLP: Transformer and Machine Translation	
Week 12: Advanced NLP: Graph Neural Network with NLP	
Week 13: Future of NLP and Exam Review	

What we learned in this course!

- Week 1: Introduction to Natural Language Processing (NLP)
- Week 2: Word Embeddings (Word Vector for Meaning)
- Week 3: Word Classification with Machine Learning I
- Week 4: Word Classification with Machine Learning II

NLP and Machine Learning

- Week 5: Language Fundamental
- Week 6: Part of Speech Tagging
- Week 7: Dependency Parsing
- Week 8: Language Model

NLP Techniques

- Week 9: Information Extraction: Named Entity Recognition
- Week 10: Advanced NLP: Attention and Reading Comprehension
- Week 11: Advanced NLP: Transformer and Machine Translation
- Week 12: Advanced NLP: Graph Neural Network with NLP

Advanced Topic

- Week 13: Future of NLP and Exam Review

0 LECTURE PLAN

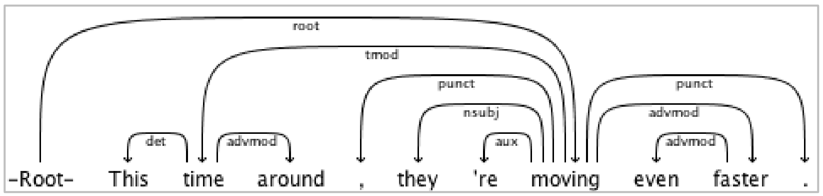
Lecture 12: NLP with Graph Neural Networks

1. Graph and Natural Language Processing
2. Graph Neural Networks
3. Graph Neural Networks – Task-based
4. Graph Neural Networks and NLP Application
    1. Text Classification
    2. Document Timestamping
    3. Text to image generation

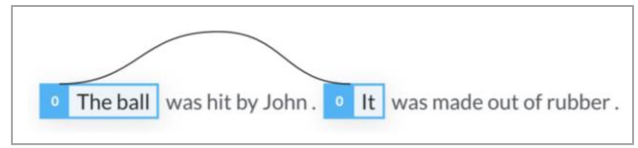
1 Graph and Natural Language Processing

Graphs are everywhere in NLP

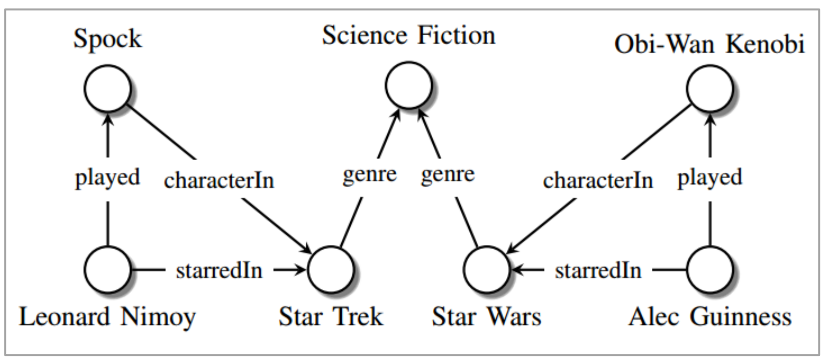
Dependency Parse Graph



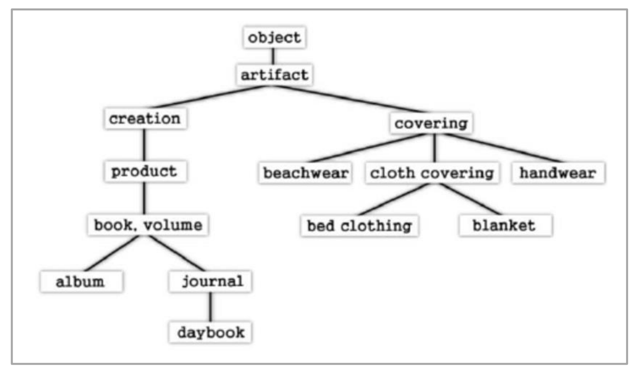
Coreference Resolution Graph



Knowledge Graph



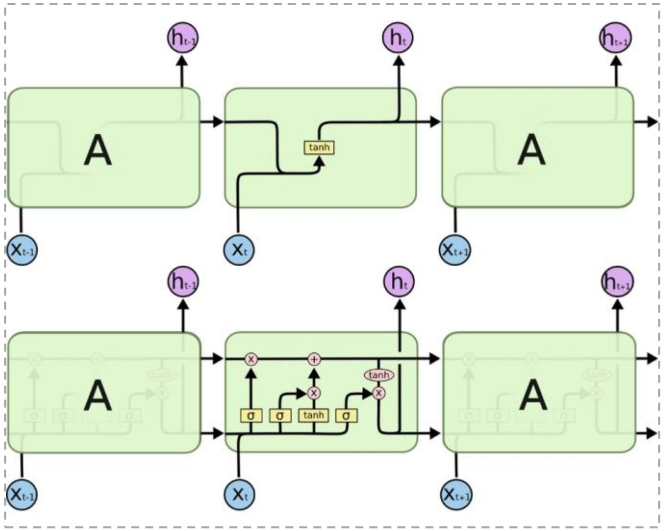
WordNet Concept Graph



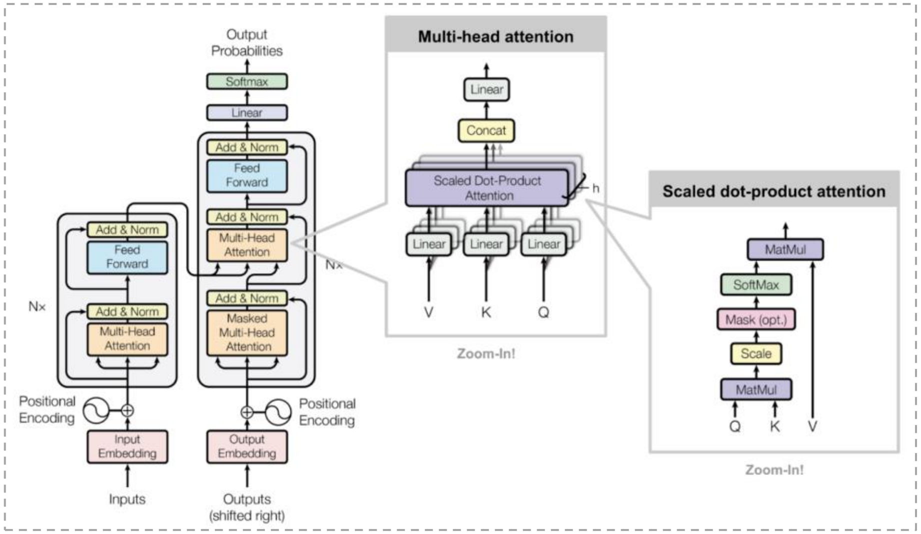
1 Graph and Natural Language Processing

Deep Learning Trend in NLP

Recurrent Neural Network



Transformers



However, it is not clear how to ***incorporate with the graph structure***

1 Graph and Natural Language Processing

Graph Neural Networks (GNN)

Graph neural networks (GNNs) *are connectionist models that capture the* ***dependence of graphs via message passing between the nodes of graphs****.*

- *Extensions of the NN models to capture the information represented as graphs.*

However, unlike the standard neural nets, ***GNNs maintain state information to capture the neighborhood properties of the nodes****. These states of the nodes of a graph can then be used to produce output labels such as a classification of the node or an arbitrary function value computed by the node*

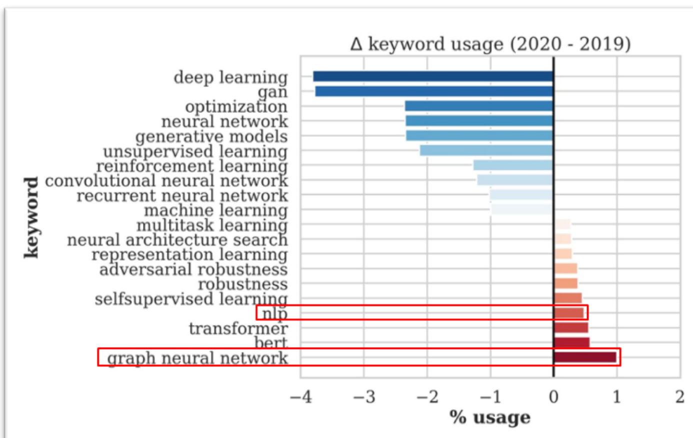
1 Graph and Natural Language Processing

Graph Neural Network(GNN) and Natural Language Processing(NLP)

***Both GNN** and **NLP** are currently receiving lots of attention!*

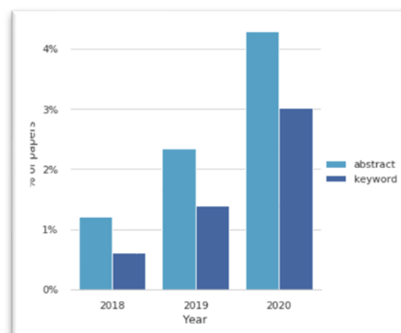
- *Extensions of the NN models to capture the information represented as graphs.*

Keywords usage analysis for ***ICLR 2020***

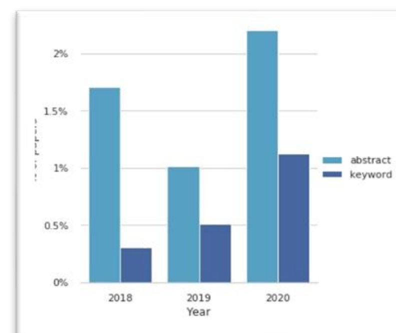


Keywords usage analysis for ***ICLR, NIPS, CVPR***

Graph Neural Networks



Natural Language Processing



Can we find the better way to apply the Graph Neural Networks in NLP tasks?

1 Graph and Natural Language Processing

What is the best way to apply Graph Neural Networks to Deep Learning NLP?

**Graph embedding is mostly used in network analysis or computer vision. However, we will not touch this part but explore more how we can use it in the deep learning NLP*

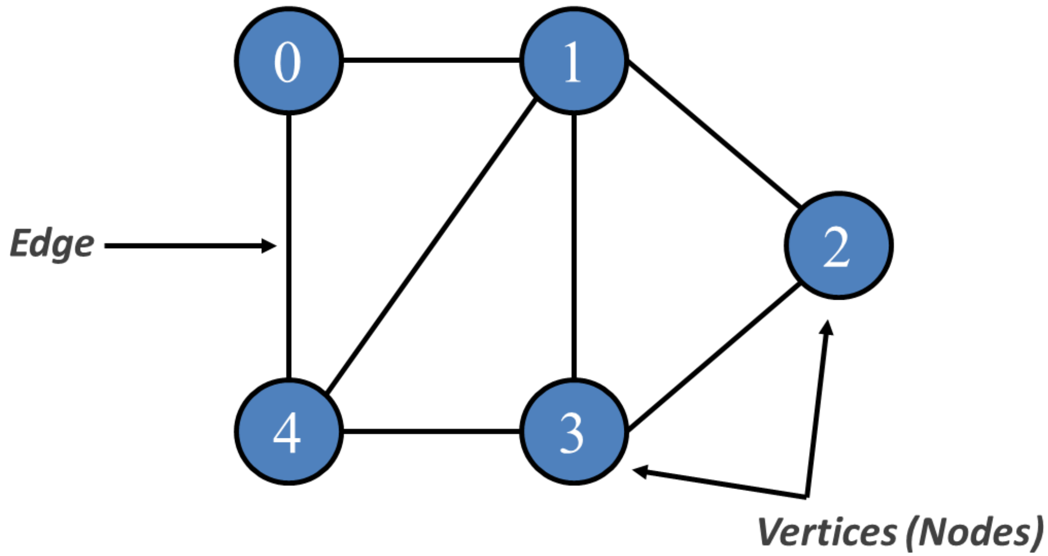
0 LECTURE PLAN

Lecture 12: NLP with Graph Neural Networks

1. Graph and Natural Language Processing
2. **Graph Neural Networks**
3. Graph Neural Networks – Task-based
4. Graph Neural Networks and NLP Application
    1. Text Classification
    2. Document Timestamping
    3. Text to image generation

2 Graph Neural Networks

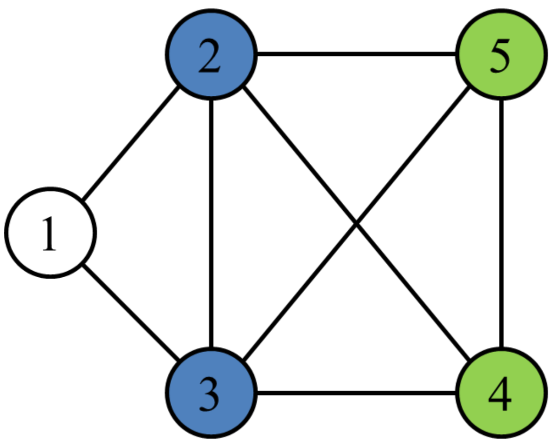
What is a Graph?



2 Graph Neural Networks

Graph Neural Networks (GNN)

Network



Adjacency matrix A

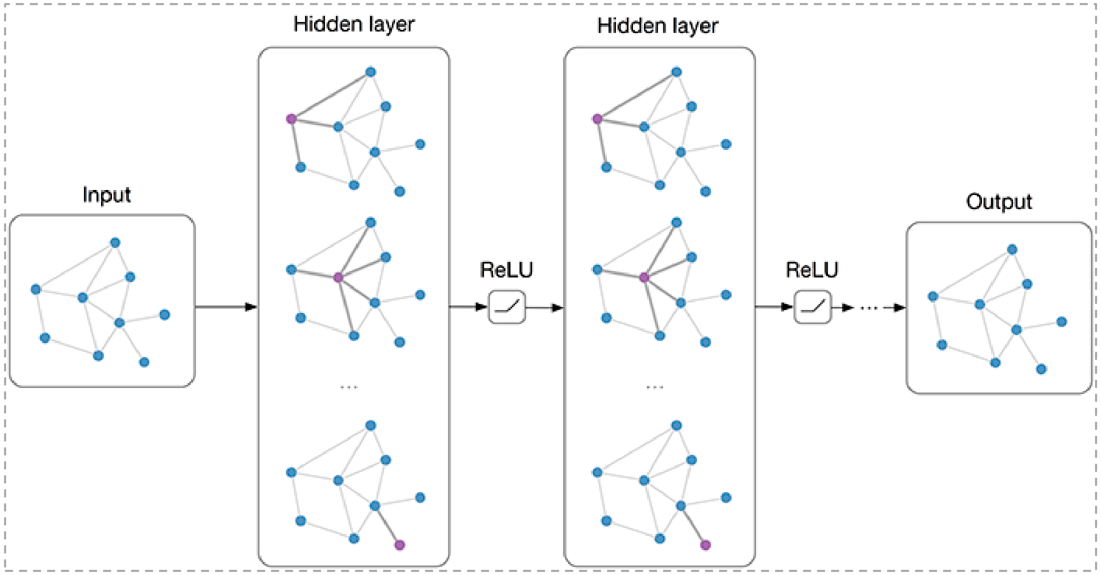
$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

Feature matrix A+I

$$\begin{array}{l} \text{node 1} \\ \text{node 2} \\ \text{node 3} \\ \text{node 4} \\ \text{node 5} \end{array} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

2 Graph Neural Networks

Graph Convolutional Networks (GCN)

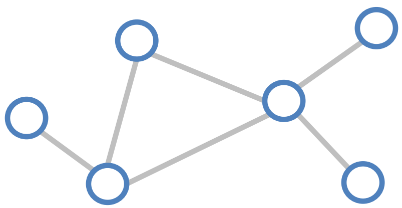


Graph Convolutional Networks

2 Graph Neural Networks

Graph Convolutional Networks (GCN)

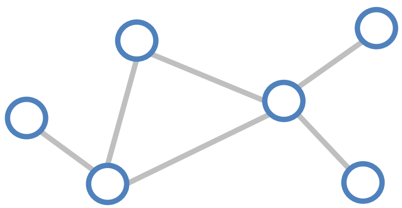
Consider this undirected graph:



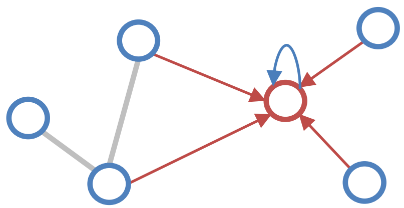
2 Graph Neural Networks

Graph Convolutional Networks (GCN)

Consider this undirected graph

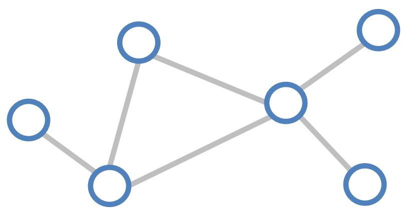


Consider update for node in red

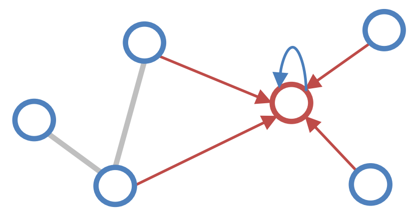


Graph Convolutional Networks (GCN)

Consider this undirected graph



Consider update for node in red



Desirable properties:

- *Weight sharing over all locations*
- *Invariance to permutations*
- *Linear Complexity O(E)*
- *Applicable both in transudative and inductive settings*

Update rule: $$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

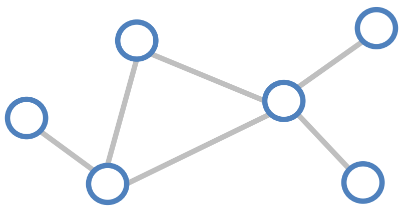
N_i: neighbour indices

c_{ij}: norm. constant (fixed/trainable)

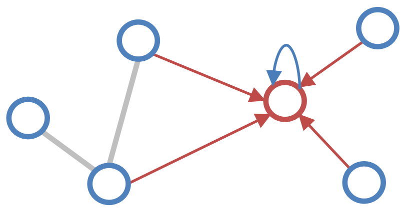
2 Graph Neural Networks

Graph Neural Networks with Edge Embeddings

Consider this undirected graph



Consider update for node in red



Desirable properties:

- *Weight sharing over all locations*
- *Invariance to permutations*
- *Linear Complexity O(E)*
- *Applicable both in transudative and inductive settings*

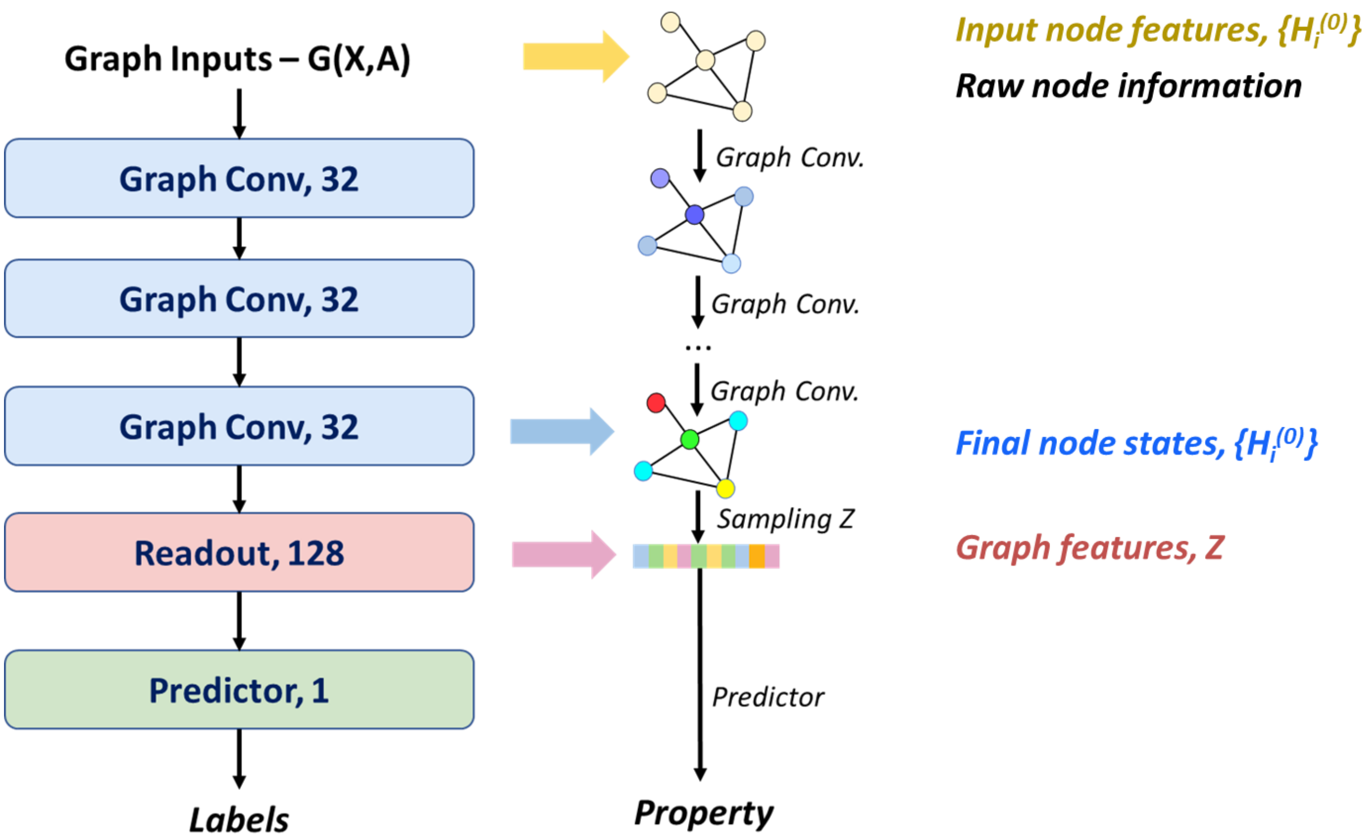
Update rule: $$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

N_i: neighbour indices

c_{ij}: norm. constant (fixed/trainable)

2 Graph Neural Networks

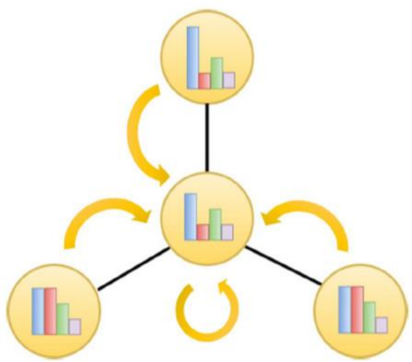
Graph Convolutional Networks (GCN)



Graph Neural Networks

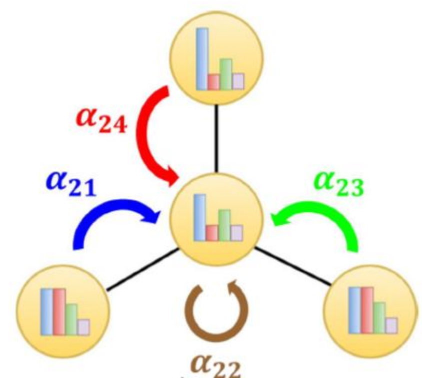
Graph Neural Networks (GNN) with Attention

Vanilla GCN updates information of neighbor atoms **with same importance**.



$$H^{(l+1)} = \sigma\left(\sum_{j \in N(i)} H_j^{(l)} W^{(l)}\right)$$

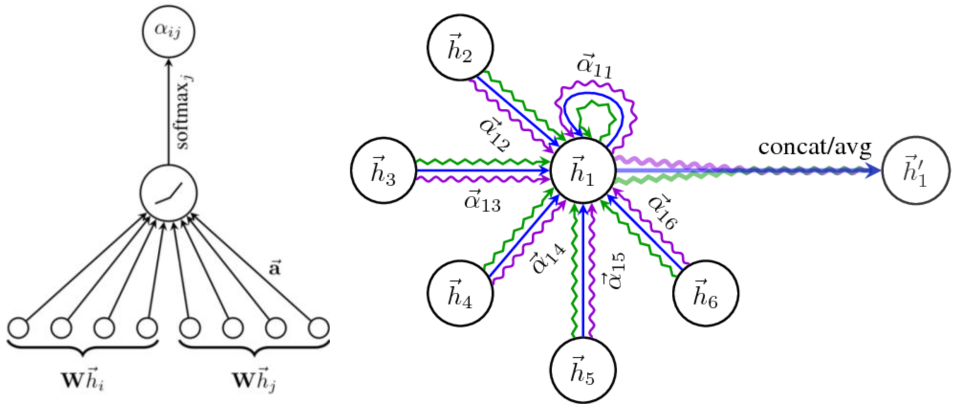
Attention mechanism enables it to update nodes **with different importance**



$$H^{(l+1)} = \sigma\left(\sum_{j \in N(i)} \alpha_{ij} H_j^{(l)} W^{(l)}\right)$$

Graph Neural Networks (GNN) with Attention

“Uses self-attention for GCNs”



Input features: $h = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_n\}, \vec{h}_i \in \mathbb{R}^F$

Importance of v_j to v_i: $e_{ij} = a(\boldsymbol{W}\vec{h}_i, \boldsymbol{W}\vec{h}_j)$ $\quad a_{ij} = softmax_j(e_{ij})$

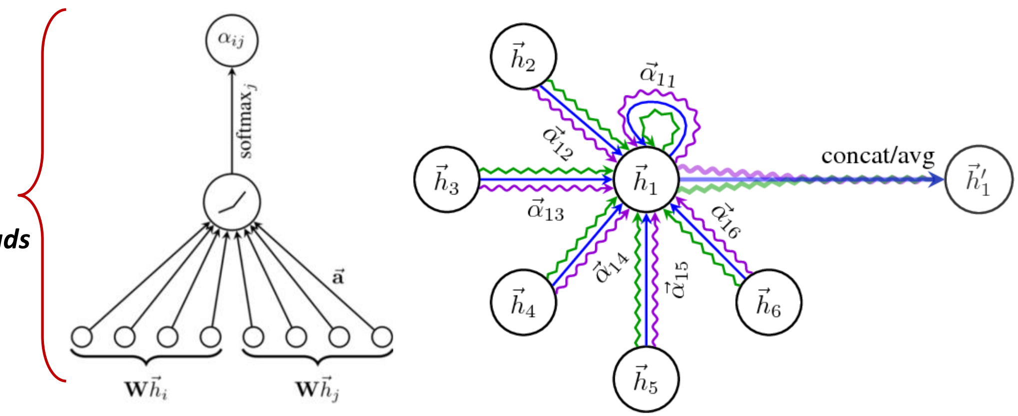
2 Graph Neural Networks

Graph Neural Networks (GNN) with Attention

“Uses self-attention for GCNs”

K x

K = # Heads



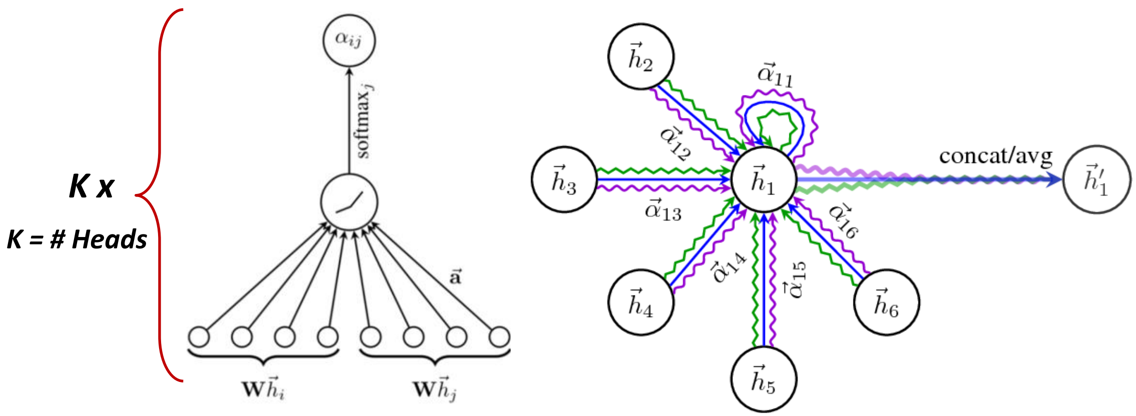
Input features: $h = \{\vec{h}_1, \vec{h}_2, \ldots, \vec{h}_n\}, \vec{h}_i \in \mathbb{R}^F$

Importance of v_j to v_i: $e_{ij} = a(\boldsymbol{W}\vec{h}_i, \boldsymbol{W}\vec{h}_j)$ $\quad a_{ij} = softmax_j(e_{ij})$

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right)$$

Graph Neural Networks (GNN) with Attention

"Uses self-attention for GCNs"



Input features: $h = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_n\}, \vec{h}_i \in \mathbb{R}^F$

Importance of v_j to v_i: $e_{ij} = a(W\vec{h}_i, W\vec{h}_j) \quad a_{ij} = softmax_j(e_{ij})$

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k \mathbf{W}^k \vec{h}_j\right) \qquad \alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

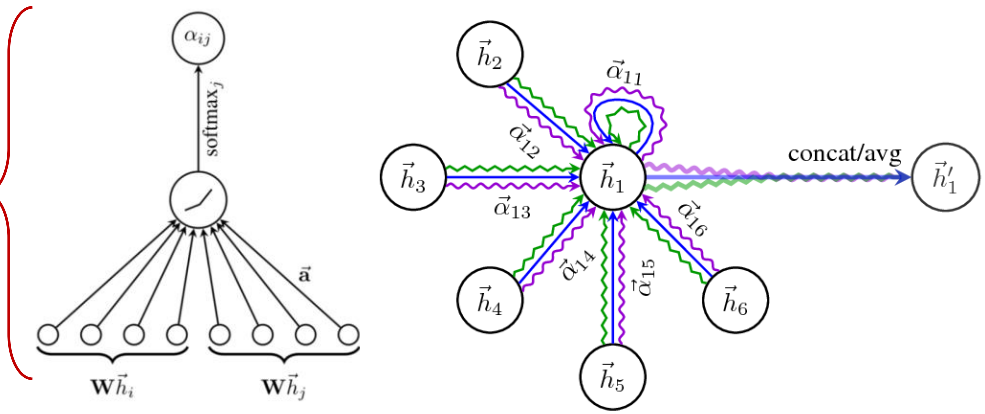
THE UNIVERSITY OF SYDNEY

Graph Neural Networks (GNN) with Attention

“Uses self-attention for GCNs”

K x

K = # Heads



Pros:
- No need to store intermediate edge-based activation vectors (when using dot-product attn.)
- Slower than GCNs but faster than GNNs with edge embeddings

Cons:
- (Most likely) less expressive than GNNs with edge embeddings
- Can be more difficult to optimise

Input features: $h = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_n\}, \vec{h}_i \in \mathbb{R}^F$

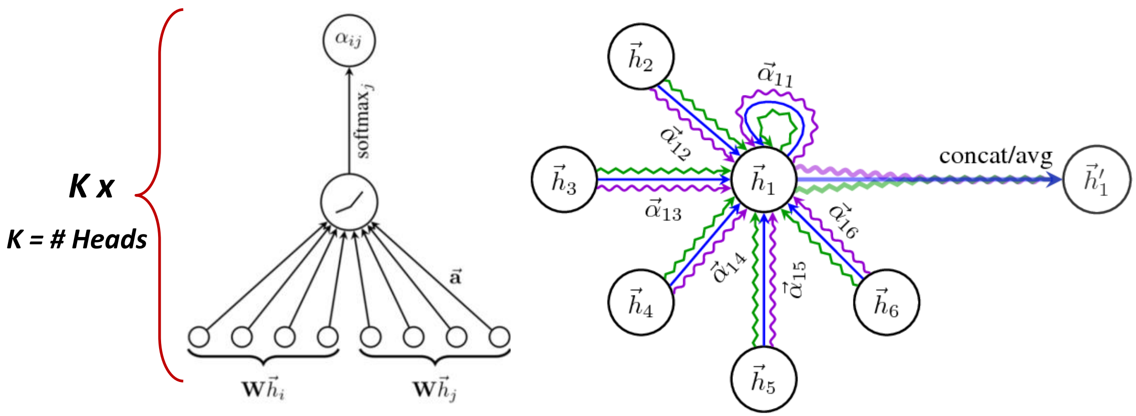
Importance of v_j to v_i: $e_{ij} = a(W\vec{h}_i, W\vec{h}_j)$ $\quad a_{ij} = softmax_j(e_{ij})$

$$\vec{h}'_i = \sigma\left(\frac{1}{K}\sum_{k=1}^{K}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^k\mathbf{W}^k\vec{h}_j\right) \qquad \alpha_{ij} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_j]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\mathrm{LeakyReLU}\left(\vec{\mathbf{a}}^T[\mathbf{W}\vec{h}_i \| \mathbf{W}\vec{h}_k]\right)\right)}$$

2 Graph Neural Networks

Graph Neural Networks (GNN) with Attention

"Uses self-attention for GCNs"



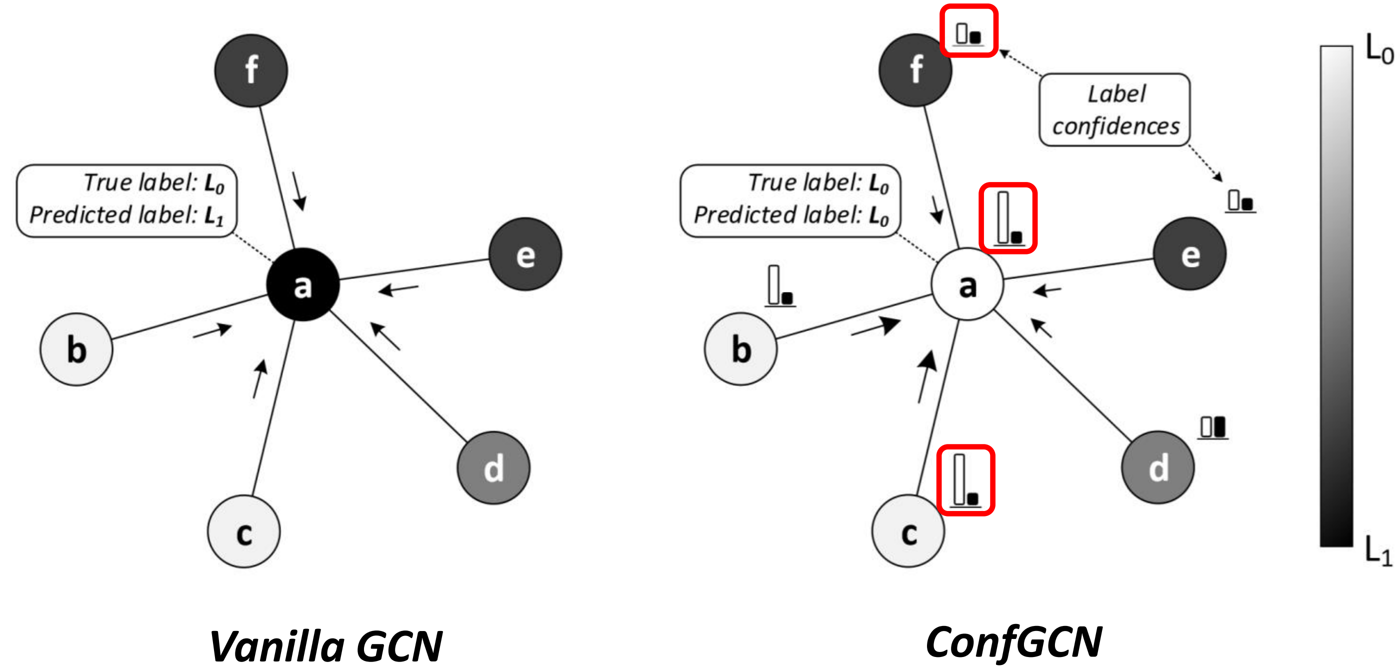
Input features: $h = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_n\}, \vec{h}_i \in \mathbb{R}^F$

Importance of v_j to v_i: $e_{ij} = a(\boldsymbol{W}\vec{h}_i, \boldsymbol{W}\vec{h}_j)$ $\quad a_{ij} = softmax_j(e_{ij})$

Method	Cora	Citeseer	Pubmed
GCN	81.5%	70.3%	**79.0**%
GAT	**83.0** $\pm$ 0.7%	**72.5** $\pm$ 0.7%	**79.0**% $\pm$ 0.3%

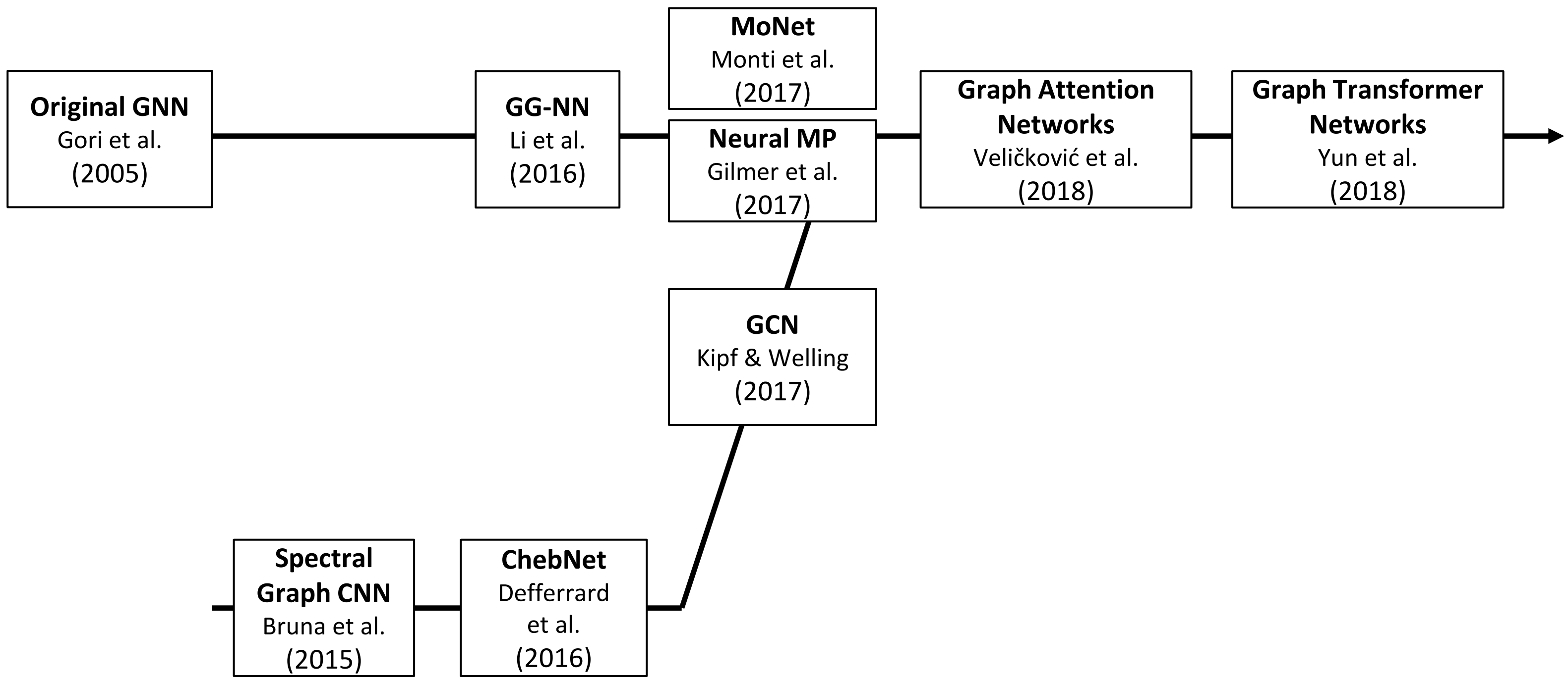
Confidence-based GCN

“Comparison with standard GCN model”



2 Graph Neural Networks

A **Brief History** of Graph Neural Networks



0 LECTURE PLAN

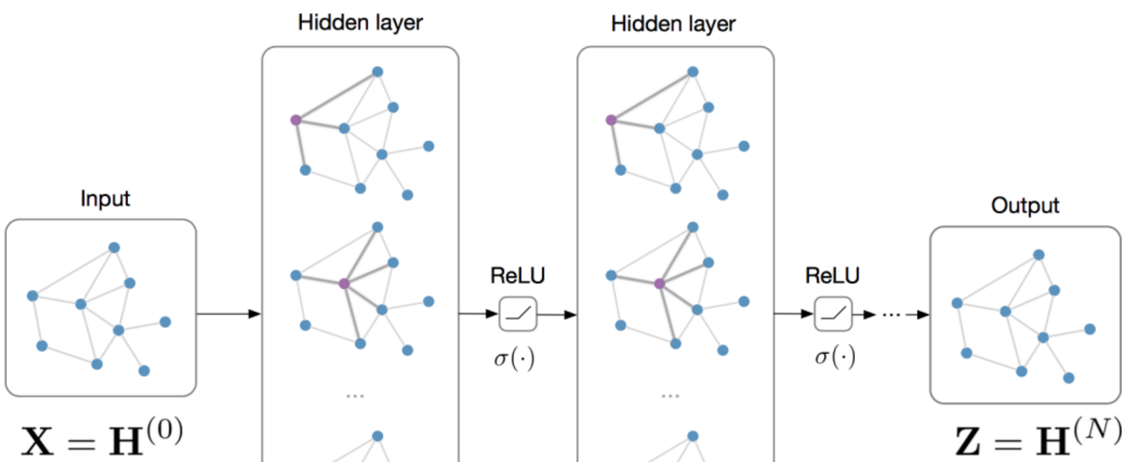
Lecture 12: NLP with Graph Neural Networks

1. Graph and Natural Language Processing
2. Graph Neural Networks
3. **Graph Neural Networks – Task-based**
4. Graph Neural Networks and NLP Application
    1. Text Classification
    2. Document Timestamping
    3. Text to image generation

Graph Neural Networks

Classification and Link Prediction with Graph Neural Networks

Input: Feature matrix $X \in R^{N\ X\ E}$, preprocessed adjacency matrix $\hat{A}$

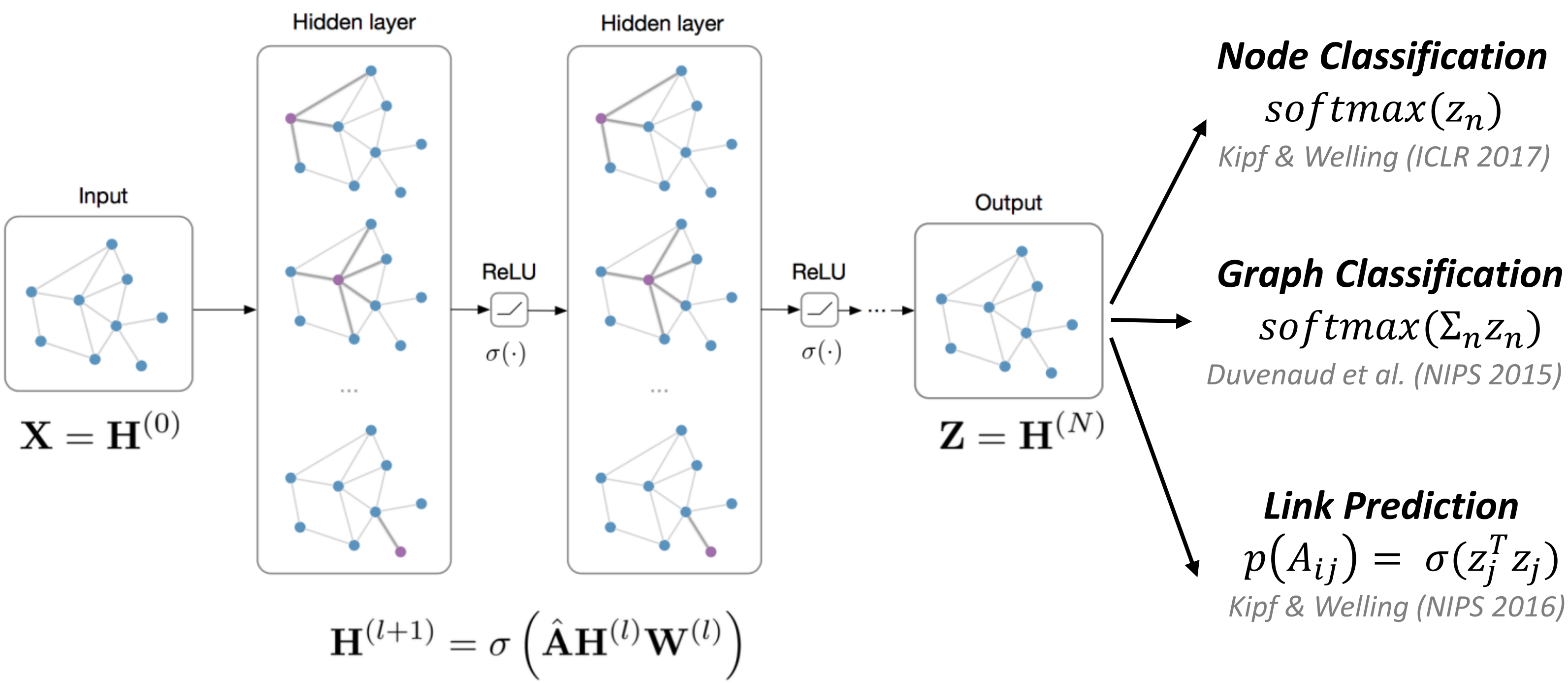


$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

2 Graph Neural Networks

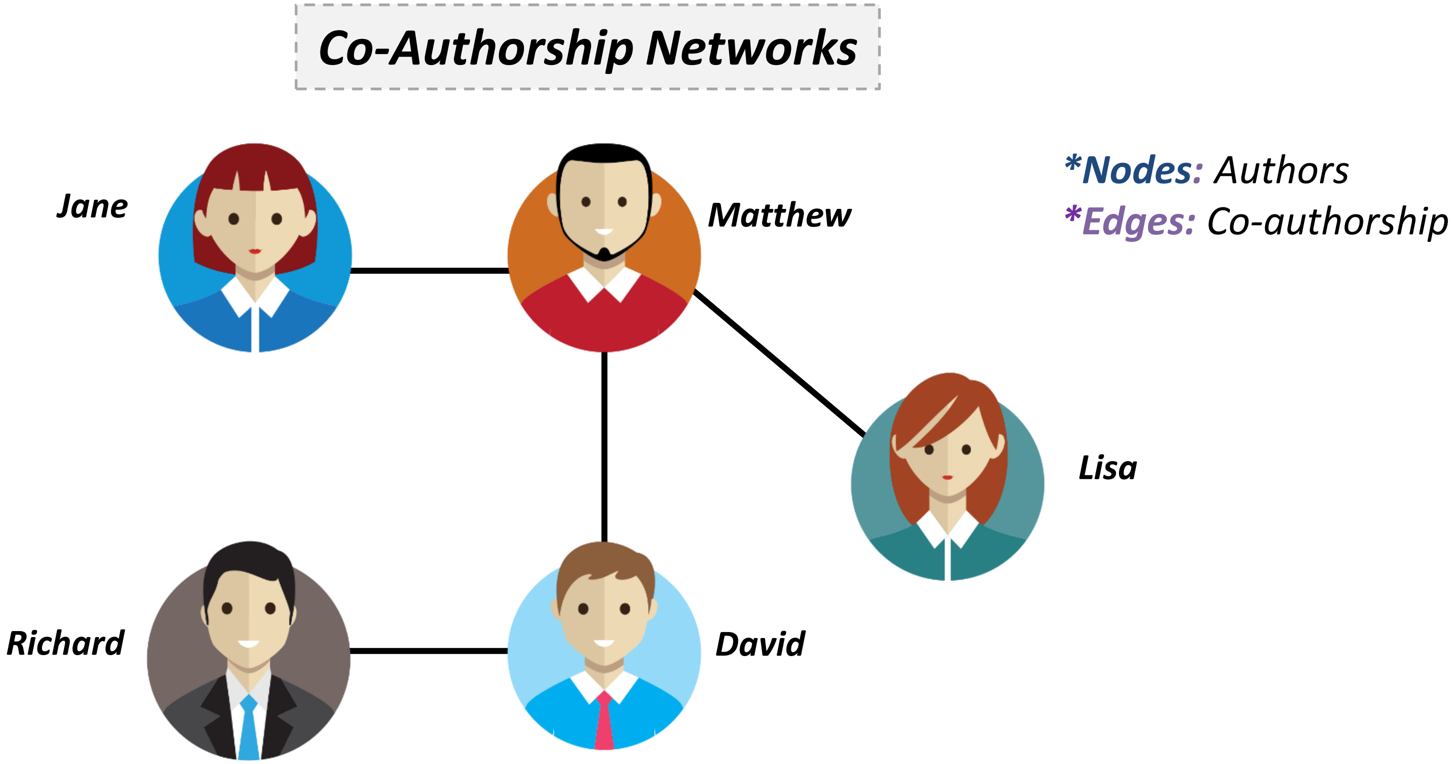
Classification and Link Prediction with Graph Neural Networks

Input: Feature matrix $X \in R^{N \times E}$, preprocessed adjacency matrix $\hat{A}$



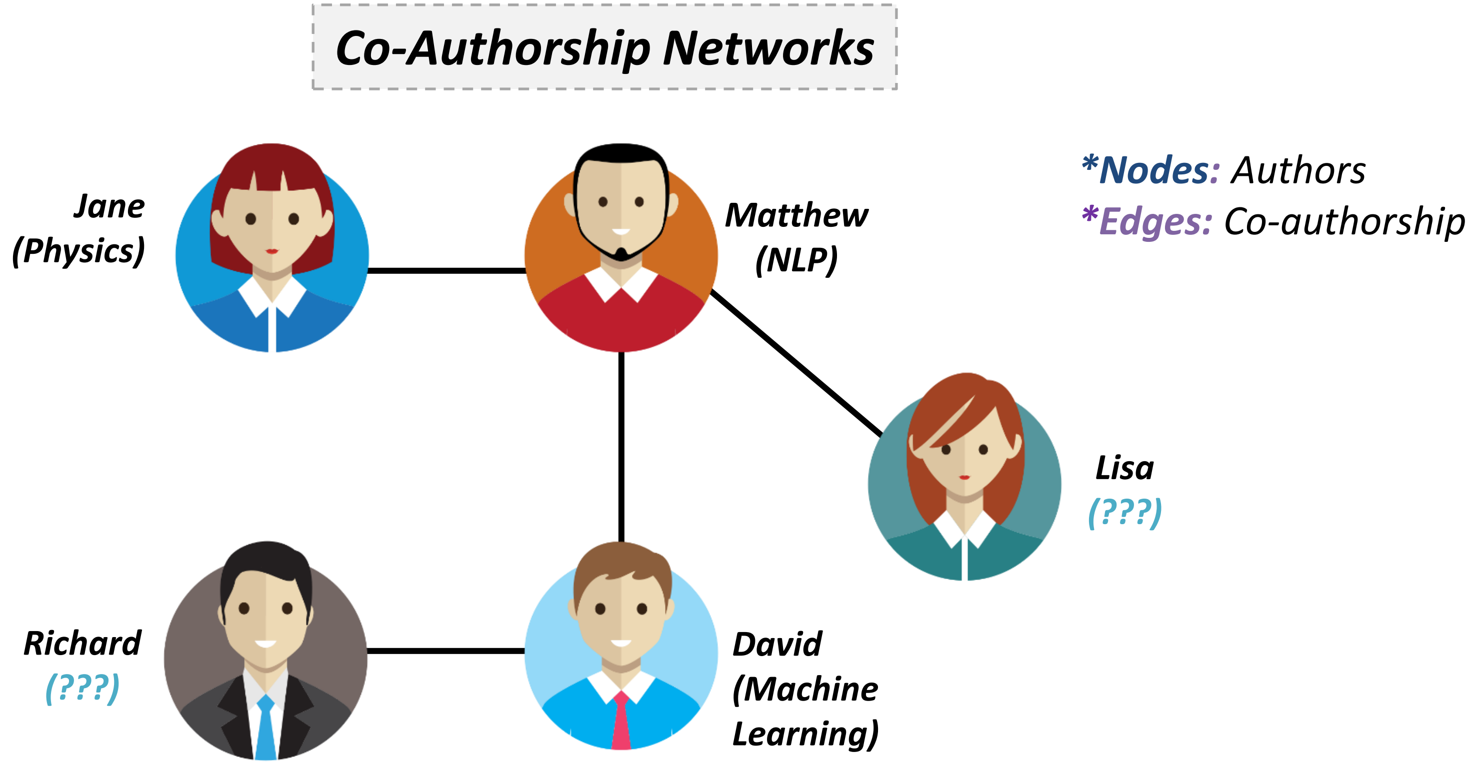
Graph Neural Networks – Task-based

Graph Neural Networks



3 Graph Neural Networks – Task-based

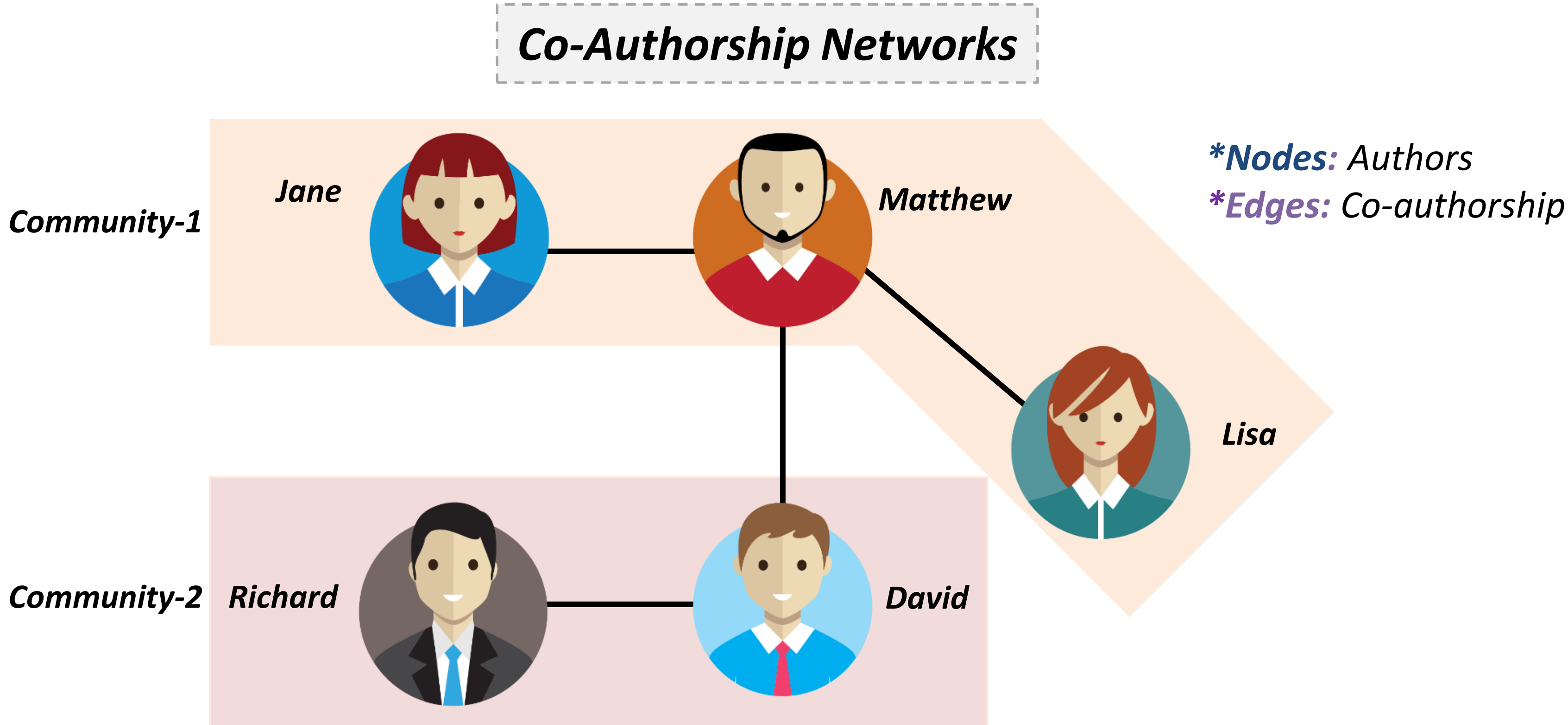
Graph Neural Networks – 1) Semi-supervised Learning



***1) Node Classification** (Semi-supervised learning): Predict research area of **unlabeled** authors*

3 Graph Neural Networks – Task-based

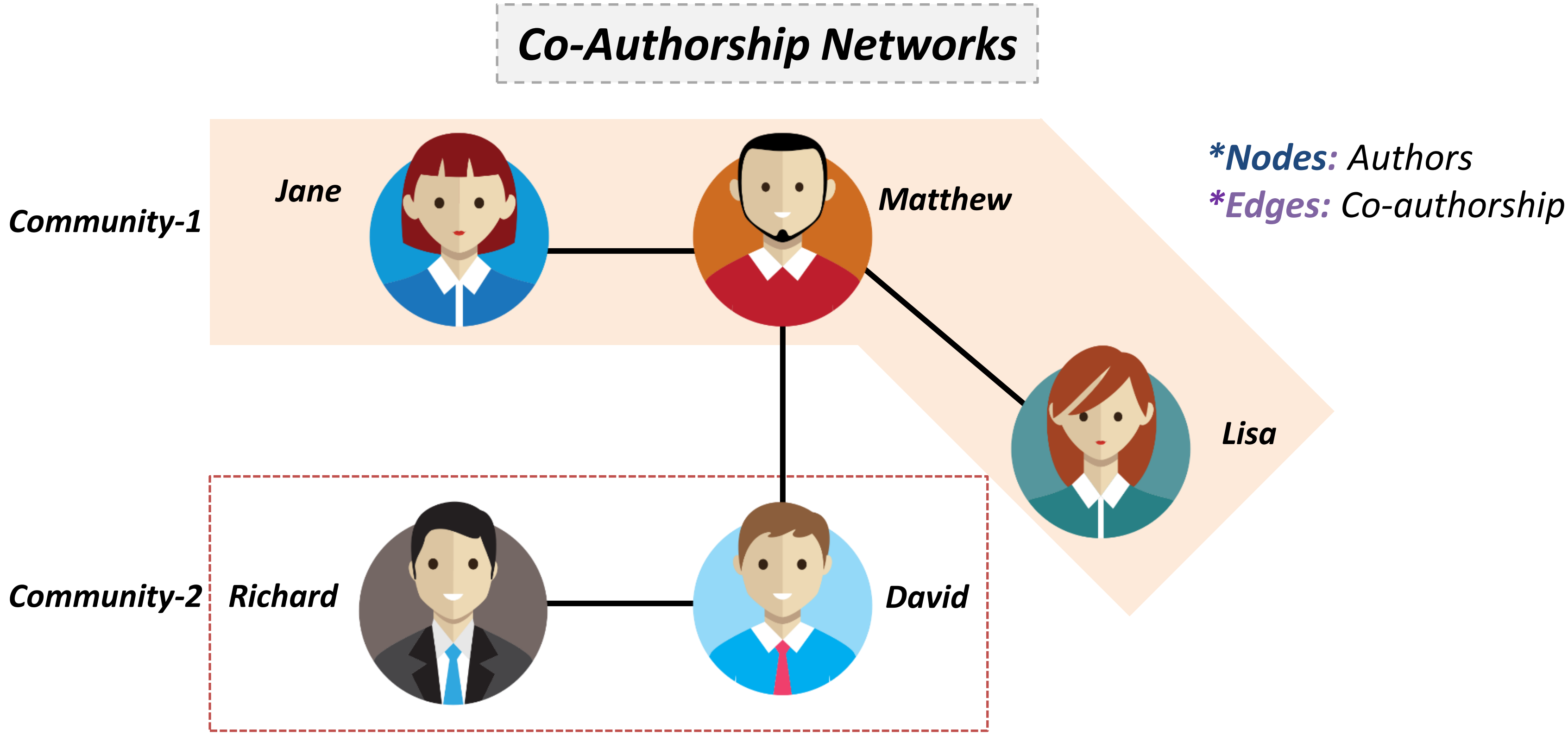
Graph Neural Networks – 2) Unsupervised Learning



2) Community Identification *(Unsupervised learning): Grouping authors with similar research interests*

3 Graph Neural Networks – Task-based

Graph Neural Networks – 3) Supervised Learning



***3) Graph Classification** (Supervised learning): Identifying class of each community*

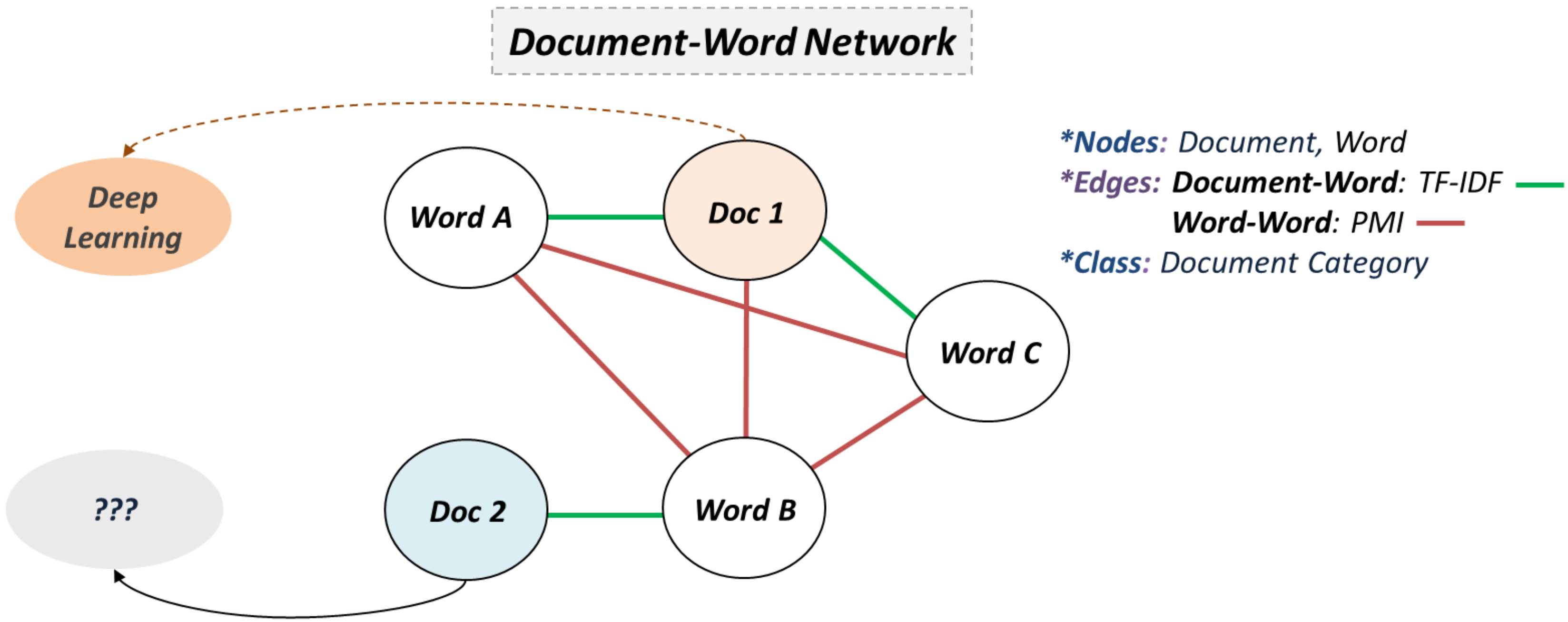
0 LECTURE PLAN

Lecture 12: NLP with Graph Neural Networks

1. Graph and Natural Language Processing
2. Graph Neural Networks
3. Graph Neural Networks – Task-based
4. **Graph Neural Networks and NLP Application**
    1. **Text Classification**
    2. Document Timestamping
    3. Text to image generation

4 Graph Neural Networks and NLP Application

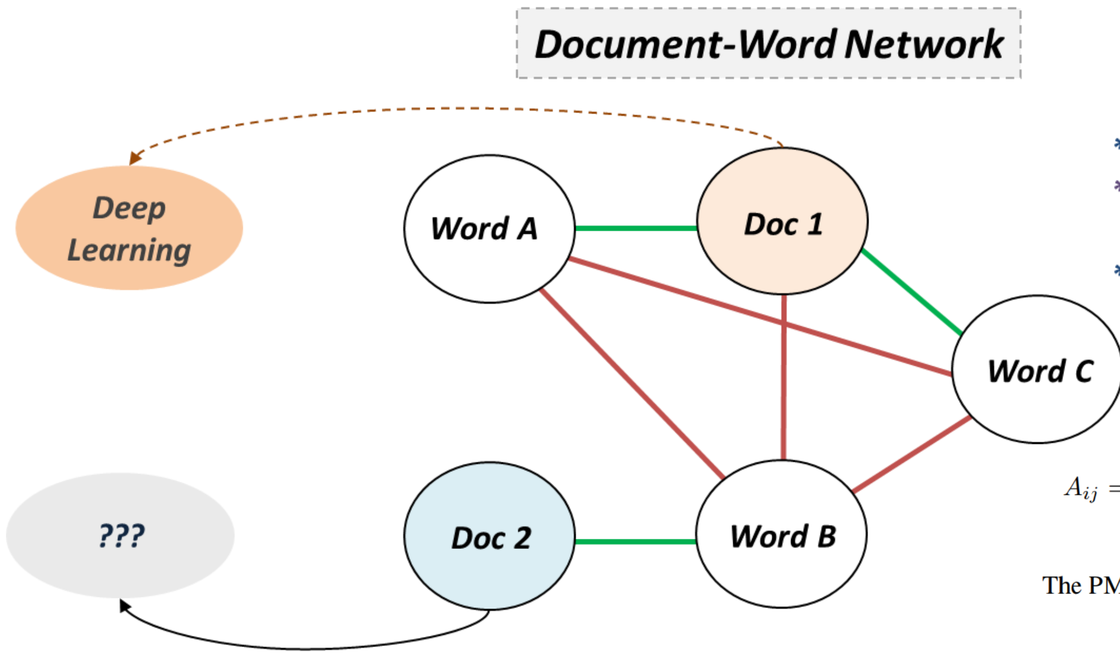
Text Classification with Graph Neural Networks



*** Document Classification** *(Document-Word Graph): Predict document category of unlabeled documents*

4 Graph Neural Networks and NLP Application

Text Classification with Graph Neural Networks



****Nodes:*** *Document, Word*
****Edges:*** ***Document-Word****: TF-IDF*
Word-Word*: PMI*
****Class:*** *Document Category*

$$A_{ij} = \begin{cases} \mathrm{PMI}(i,j) & i,j \text{ are words, } \mathrm{PMI}(i,j) > 0 \\ \text{TF-IDF}_{ij} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

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

$$\mathrm{PMI}(i,j) = \log \frac{p(i,j)}{p(i)p(j)} \quad (4)$$

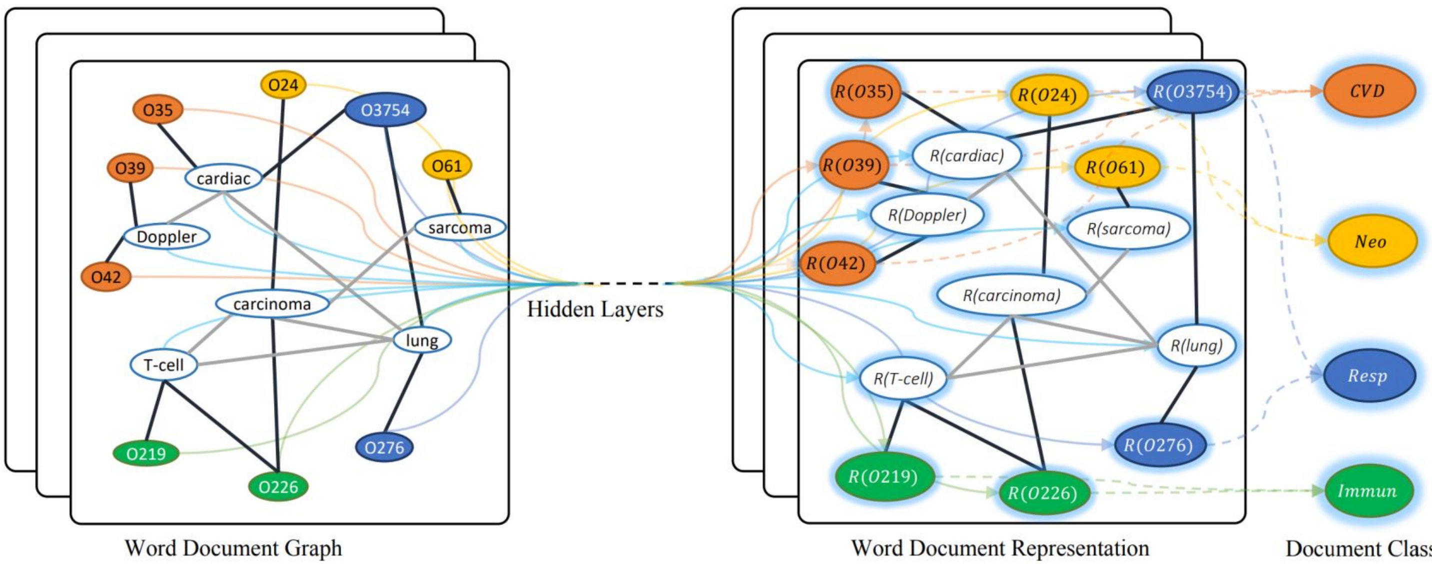
$$p(i,j) = \frac{\#W(i,j)}{\#W} \quad (5)$$

$$p(i) = \frac{\#W(i)}{\#W} \quad (6)$$

*** Document Classification*** *(Document-Word Graph): Predict document category of unlabeled documents*

4 Graph Neural Networks and NLP Application

Text Classification with Graph Neural Networks



$$A_{ij} = \begin{cases} \mathrm{PMI}(i,j) & i,j \text{ are words, } \mathrm{PMI}(i,j) > 0 \\ \text{TF-IDF}_{ij} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

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

$$\mathrm{PMI}(i,j) = \log \frac{p(i,j)}{p(i)p(j)} \tag{4}$$

$$p(i,j) = \frac{\#W(i,j)}{\#W} \tag{5}$$

$$p(i) = \frac{\#W(i)}{\#W} \tag{6}$$

$$Z = \mathrm{softmax}(\tilde{A}\,\mathrm{ReLU}(\tilde{A}XW_0)W_1) \tag{7}$$

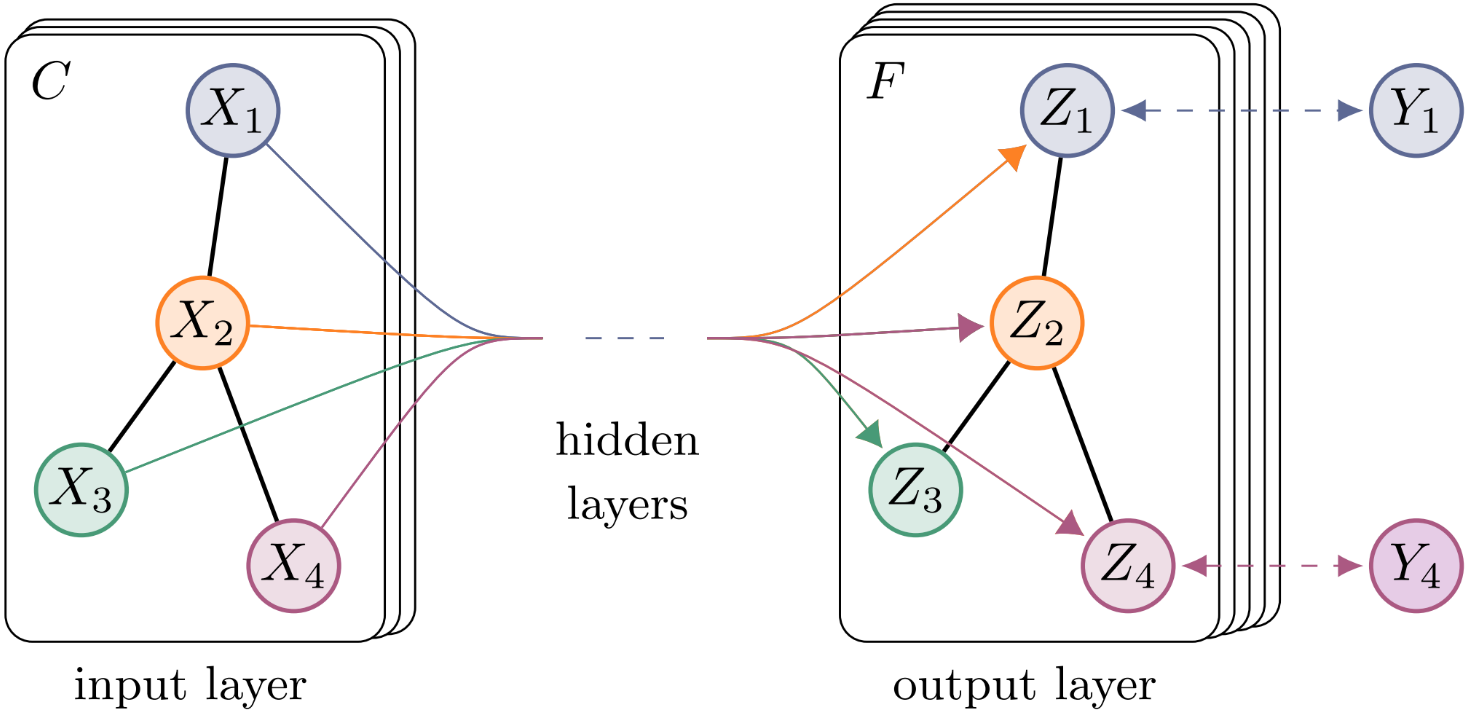
where $\tilde{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ is the same as in equation 1, and $\mathrm{softmax}(x_i) = \frac{1}{\mathcal{Z}}\exp(x_i)$ with $\mathcal{Z} = \sum_i \exp(x_i)$. The loss function is defined as the cross-entropy error over all labeled documents:

$$\mathcal{L} = -\sum_{d \in \mathcal{Y}_D} \sum_{f=1}^{F} Y_{df} \ln Z_{df} \tag{8}$$

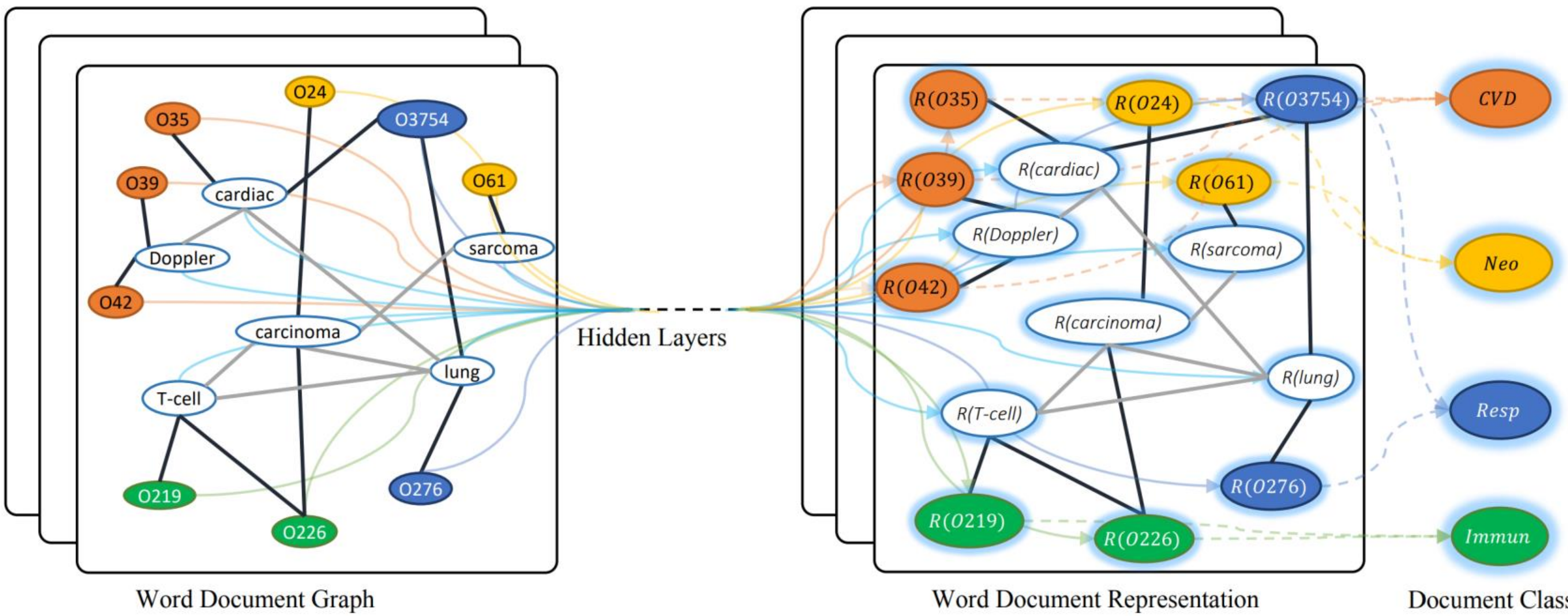
where $\mathcal{Y}_D$ is the set of document indices that have labels and F is the dimension of the output features, which is equal to the number of classes. Y is the label indicator

4 Graph Neural Networks and NLP Application

Text Classification with Graph Neural Networks



Text Classification with Graph Neural Networks



THE UNIVERSITY OF SYDNEY

Text Classification with Graph Neural Networks

Table 1: Summary statistics of datasets.

Dataset	# Docs	# Training	# Test	# Words	# Nodes	# Classes	Average Length
20NG	18,846	11,314	7,532	42,757	61,603	20	221.26
R8	7,674	5,485	2,189	7,688	15,362	8	65.72
R52	9,100	6,532	2,568	8,892	17,992	52	69.82
Ohsumed	7,400	3,357	4,043	14,157	21,557	23	135.82
MR	10,662	7,108	3,554	18,764	29,426	2	20.39

- **20NG**: 20 News Group. 18,846 documents evenly categorized into 20 different categories. In total, 11,314 documents are in the training set and 7,532 documents are in the test set.
- **Ohsumed**: MEDLINE database, which is a bibliographic database of important medical literature maintained by the National Library of Medicine.
- **R52 and R8**: two subsets of the Reuters 21578 dataset. R8 has 8 categories, and R52 has 52 categories.
- **MR**: movie review dataset for binary sentiment classification, in which each review only contains one sentence

Graph Neural Networks and NLP Application

Text Classification with Graph Neural Networks

Table 2: Test Accuracy on document classification task. We run all models 10 times and report mean $\pm$ standard deviation. Text GCN significantly outperforms baselines on 20NG, R8, R52 and Ohsumed based on student t-test ($p < 0.05$).

Model	**20NG**	**R8**	**R52**	**Ohsumed**	**MR**
TF-IDF + LR	0.8319 ± 0.0000	0.9374 ± 0.0000	0.8695 ± 0.0000	0.5466 ± 0.0000	0.7459 ± 0.0000
CNN-rand	0.7693 ± 0.0061	0.9402 ± 0.0057	0.8537 ± 0.0047	0.4387 ± 0.0100	0.7498 ± 0.0070
CNN-non-static	0.8215 ± 0.0052	0.9571 ± 0.0052	0.8759 ± 0.0048	0.5844 ± 0.0106	**0.7775 ± 0.0072**
LSTM	0.6571 ± 0.0152	0.9368 ± 0.0082	0.8554 ± 0.0113	0.4113 ± 0.0117	0.7506 ± 0.0044
LSTM (pretrain)	0.7543 ± 0.0172	0.9609 ± 0.0019	0.9048 ± 0.0086	0.5110 ± 0.0150	0.7733 ± 0.0089
Bi-LSTM	0.7318 ± 0.0185	0.9631 ± 0.0033	0.9054 ± 0.0091	0.4927 ± 0.0107	0.7768 ± 0.0086
PV-DBOW	0.7436 ± 0.0018	0.8587 ± 0.0010	0.7829 ± 0.0011	0.4665 ± 0.0019	0.6109 ± 0.0010
PV-DM	0.5114 ± 0.0022	0.5207 ± 0.0004	0.4492 ± 0.0005	0.2950 ± 0.0007	0.5947 ± 0.0038
PTE	0.7674 ± 0.0029	0.9669 ± 0.0013	0.9071 ± 0.0014	0.5358 ± 0.0029	0.7023 ± 0.0036
fastText	0.7938 ± 0.0030	0.9613 ± 0.0021	0.9281 ± 0.0009	0.5770 ± 0.0049	0.7514 ± 0.0020
fastText (bigrams)	0.7967 ± 0.0029	0.9474 ± 0.0011	0.9099 ± 0.0005	0.5569 ± 0.0039	0.7624 ± 0.0012
SWEM	0.8516 ± 0.0029	0.9532 ± 0.0026	0.9294 ± 0.0024	0.6312 ± 0.0055	0.7665 ± 0.0063
LEAM	0.8191 ± 0.0024	0.9331 ± 0.0024	0.9184 ± 0.0023	0.5858 ± 0.0079	0.7695 ± 0.0045
Graph-CNN-C	0.8142 ± 0.0032	0.9699 ± 0.0012	0.9275 ± 0.0022	0.6386 ± 0.0053	0.7722 ± 0.0027
Graph-CNN-S	–	0.9680 ± 0.0020	0.9274 ± 0.0024	0.6282 ± 0.0037	0.7699 ± 0.0014
Graph-CNN-F	–	0.9689 ± 0.0006	0.9320 ± 0.0004	0.6304 ± 0.0077	0.7674 ± 0.0021
Text GCN	**0.8634 ± 0.0009**	**0.9707 ± 0.0010**	**0.9356 ± 0.0018**	**0.6836 ± 0.0056**	0.7674 ± 0.0020

0 LECTURE PLAN

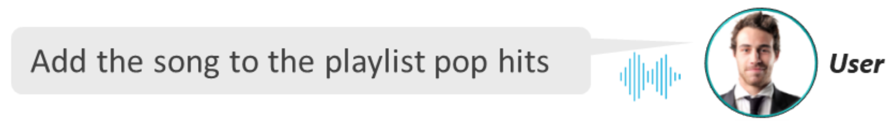
Lecture 12: NLP with Graph Neural Networks

1. Graph and Natural Language Processing
2. Graph Neural Networks
3. Graph Neural Networks – Task-based
4. **Graph Neural Networks and NLP Application**
    1. Text Classification
    2. **Text Classification and Sequence Labeling**
    3. Text to image generation

4 Graph Neural Networks and NLP Application

Natural Language Understanding with Graph Neural Networks

*To build the successful spoken dialogue model, **Natural Language Understanding** is the first crucial step to pass but considered as an AI-hard problem*



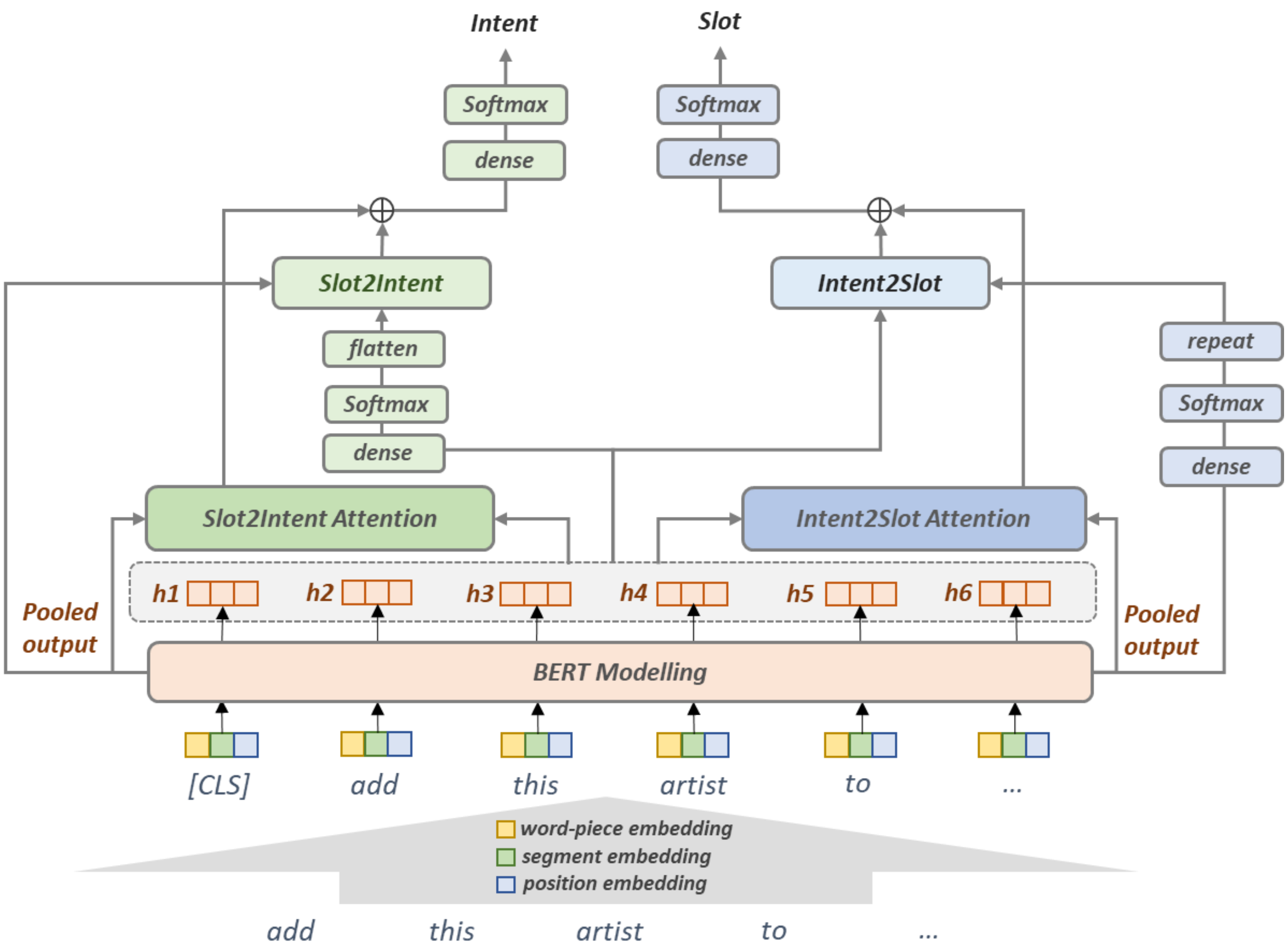
Language Understanding Tasks

1) Interprets the semantic context (Slot Labels) that the user communicates and 2) classifies it into proper intents

Utterance	*Add*	*this*	*song*	*to*	*the*	*playlist*	*pophits*
Task 1) Slot Filling	**O**	**O**	**B-mitem**	**O**	**O**	**O**	**B-playlist**
Task 2) Intent Classification	**AddPlaylist**						

Graph Neural Networks and NLP Application

Natural Language Understanding with Graph Neural Networks



Natural Language Understanding with Graph Neural Networks

Used the same metrics as the baselines: Slot Filling F1, Intent Classification Accuracy, and Sentence Accuracy

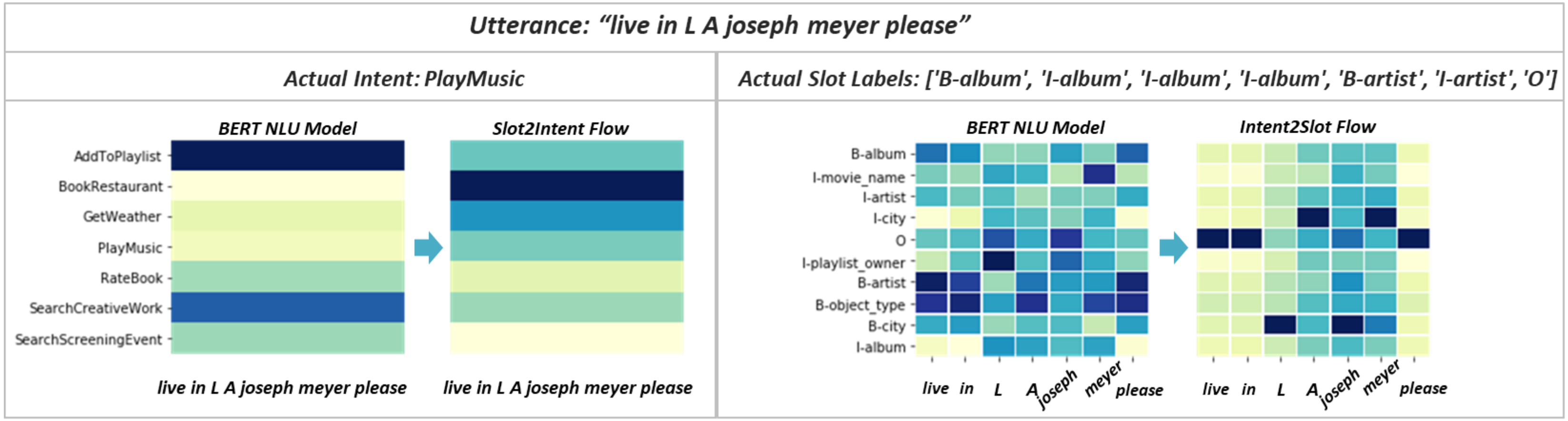
- *Achieved the best prediction performance on all tasks for both datasets*

Model	ATIS (10 epoch)			SNIPS (20 epoch)		
	Slot (f1)	**Intent (acc)**	**Sentence (acc)**	**Slot (f1)**	**Intent (acc)**	**Sentence (acc)**
Joint Seq (Hakkani-Tür et al., 2016)	94.3	92.6	80.7	87.3	96.9	73.2
Atten.-based (Liu and Lane, 2016)	94.2	91.1	78.9	87.8	96.7	74.1
Slot-Gated Full Atten (Goo et al., 2018)	94.8	93.6	82.2	88.8	97.0	75.5
Slot-Gated Intent Atten (Goo et al., 2018)	95.2	94.1	82.6	88.3	96.8	74.6
Capsule NN (Zhang et al., 2019)	95.2	95.0	83.4	91.8	97.3	80.9
Bi-LSTM-CRF (Haihong et al., 2019)	95.8	97.8	86.8	91.4	97.4	80.6
Our Model with Bi-flow	95.9	98.0	87.7	95.6	98.7	89.3
Our Model with Bi-flow, atten.	96.0	98.0	87.9	95.6	98.7	89.4

Graph Neural Networks and NLP Application

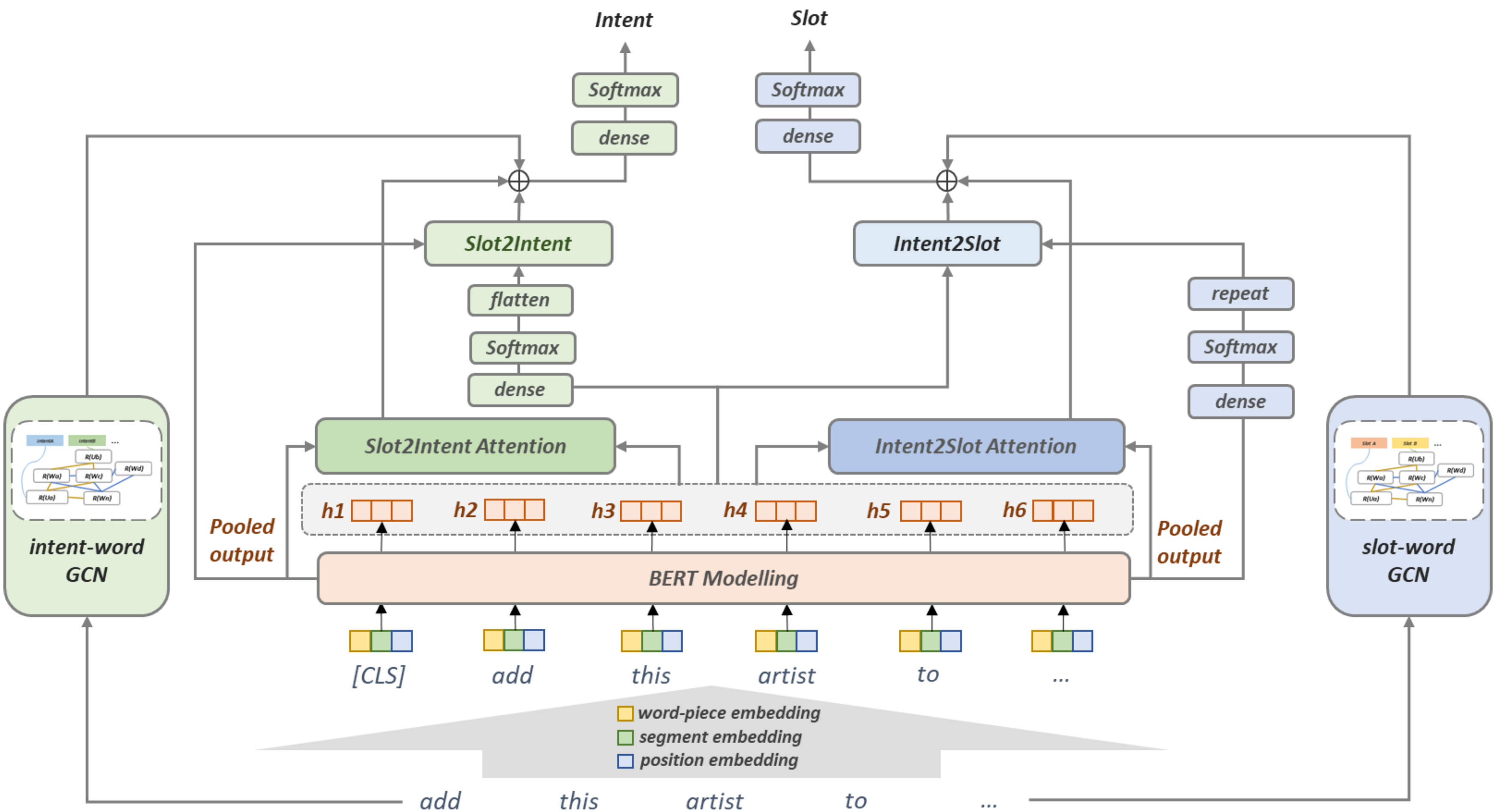
Natural Language Understanding with Graph Neural Networks

ISSUE: *Slot Label Confusion, Tokenization (Preprocessing) Issue , or Benchmark Data Annotation Issue*



How can we solve this problem?

Natural Language Understanding with Graph Neural Networks



THE UNIVERSITY OF SYDNEY

Natural Language Understanding with Graph Neural Networks

Used the same metrics as the baselines: Slot Filling F1, Intent Classification Accuracy, and Sentence Accuracy

- *Achieved the best prediction performance on all tasks for both datasets*

Model	ATIS (10 epoch)			SNIPS (20 epoch)		
	Slot (f1)	Intent (acc)	Sentence (acc)	Slot (f1)	Intent (acc)	Sentence (acc)
Joint Seq (Hakkani-Tür et al., 2016)	94.3	92.6	80.7	87.3	96.9	73.2
Atten.-based (Liu and Lane, 2016)	94.2	91.1	78.9	87.8	96.7	74.1
Slot-Gated Full Atten (Goo et al., 2018)	94.8	93.6	82.2	88.8	97.0	75.5
Slot-Gated Intent Atten (Goo et al., 2018)	95.2	94.1	82.6	88.3	96.8	74.6
Capsule NN (Zhang et al., 2019)	95.2	95.0	83.4	91.8	97.3	80.9
Bi-LSTM-CRF (Haihong et al., 2019)	95.8	97.8	86.8	91.4	97.4	80.6
Our Model with Bi-flow	95.9	98.0	87.7	95.6	98.7	89.3
Our Model with Bi-flow, atten.	96.0	98.0	87.9	95.6	98.7	89.4

4 Graph Neural Networks and NLP Application

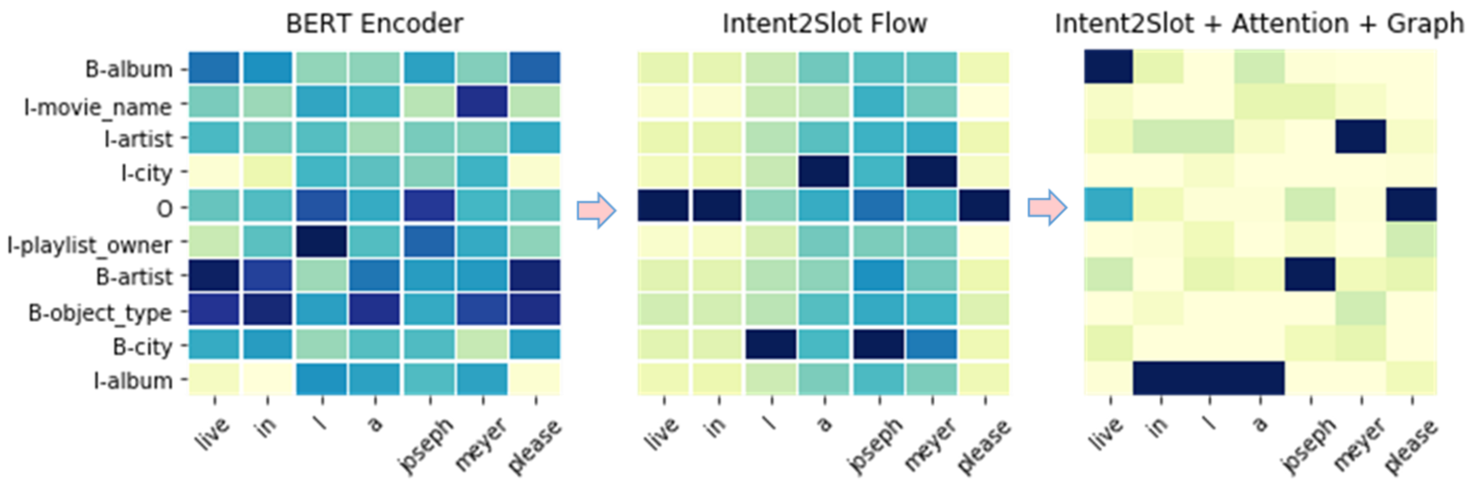
Natural Language Understanding with Graph Neural Networks

Used the same metrics as the baselines: Slot Filling F1, Intent Classification Accuracy, and Sentence Accuracy

- *Achieved the best prediction performance on all tasks for both datasets*

Model	ATIS (10 epoch)			SNIPS (20 epoch)		
	Slot (f1)	**Intent (acc)**	**Sentence (acc)**	**Slot (f1)**	**Intent (acc)**	**Sentence (acc)**
Joint Seq (Hakkani-Tür et al., 2016)	94.3	92.6	80.7	87.3	96.9	73.2
Atten.-based (Liu and Lane, 2016)	94.2	91.1	78.9	87.8	96.7	74.1
Slot-Gated Full Atten (Goo et al., 2018)	94.8	93.6	82.2	88.8	97.0	75.5
Slot-Gated Intent Atten (Goo et al., 2018)	95.2	94.1	82.6	88.3	96.8	74.6
Capsule NN (Zhang et al., 2019)	95.2	95.0	83.4	91.8	97.3	80.9
Bi-LSTM-CRF (Haihong et al., 2019)	95.8	97.8	86.8	91.4	97.4	80.6
Our Model with Bi-flow	95.9	98.0	87.7	95.6	98.7	89.3
Our Model with Bi-flow, atten.	96.0	98.0	87.9	95.6	98.7	89.4
Our Model with Bi-flow, atten., graph	**96.3**	**98.6**	**88.2**	**96.1**	**99.2**	**89.8**

Natural Language Understanding with Graph Neural Networks



0 LECTURE PLAN

Lecture 12: NLP with Graph Neural Networks

1. Graph and Natural Language Processing
2. Graph Neural Networks
3. Graph Neural Networks – Task-based
4. **Graph Neural Networks and NLP Application**
   1. Text Classification
   2. Document Timestamping
   3. **Text to image generation**

Graph Neural Networks and NLP Application

Text-to-Image Generation- 1) Text to Structured Graph

Text description
Both of the men are riding their horses

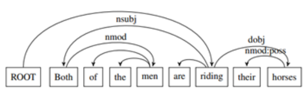
↓

Text Preprocessing
(incl. stemming – VBG issue,)

↓

Semantic graph extraction

1. dependency parser
(incl. Universal dependencies)



1-1. Quantificational modifiers
(light nouns)
e.g. both of the cars are blue

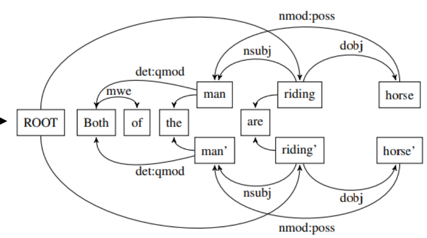
Blue car
Blue car

1-2. Pronoun Resolution <u>Coreference resolution</u>
(personal nouns)

1-3. Plural Nouns
(copy individual nodes of the graph based on the value of their numeric modifier)

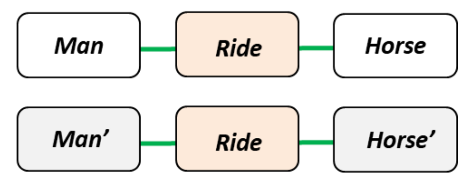
1-4. Tense Resolution
(Past, ing present tense, passive to)

Final semantic graph



Transformer Scene Graph Parser

0, 1, 2, 3
Objects: *man, man', horse, horse'*
Relationships: *(**0**, **<u>ride</u>**, **2**), (**1**, **<u>ride</u>**, **3**)*



4 Graph Neural Networks and NLP Application

Text-to-Image Generation – 2) Positional Text Embedding

Position: Train the word embedding based on the position of the (relation and attribute) word

A man rides a skateboard with a dog outside

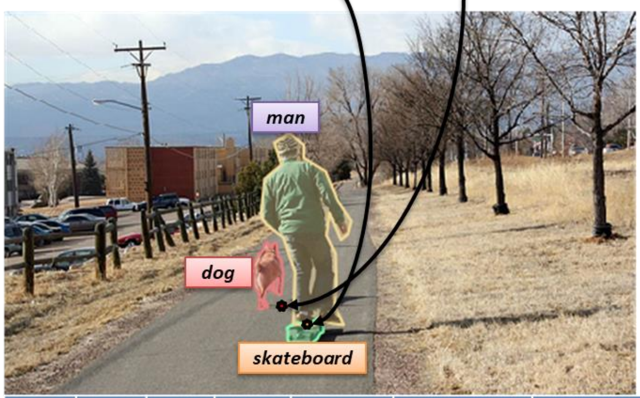
Scene Graph parser

Check which sentence achieves the better scene graph extraction

0 1 2

***Objects:** man, skateboard, dog*

***Relationships:** (0, ride, 1), (1, with, 2)*



Imgid	capid	objid	object	Bounding Box	Segmentation Box	hypernyms	Matched_object
1	1	0	man	..	..	...	Person
1	1	..	..	..	..	...	...
1	1	2	dog	..	..	....	dog

man

man,ride,skateboard

woman,walk,man

4 Graph Neural Networks and NLP Application

Text-to-Image Generation – 2) Positional Text Embedding

**Position: Train the word embedding based on the position of the (relation and attribute) word*

A man rides a skateboard with a dog outside

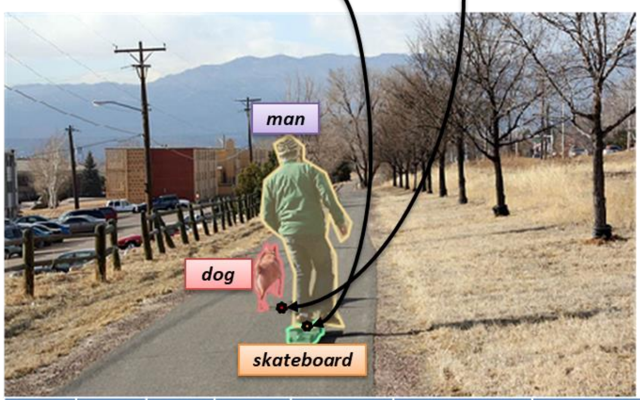
↓

Scene Graph parser

Check which sentence achieves the better scene graph extraction

Objects: *man* (0), *skateboard* (1), *dog* (2)

Relationships: *(0, ride, 1), (1, with, 2)*



Imgid	capid	objid	object	Bounding Box	Segmentation Box	hypernyms	Matched_object
1	1	0	man	..	..	...	Person
1	1	..	..	..	..	...	...
1	1	2	dog	..	..	....	dog

Methods \ Metrics	Inception Score ↑	FID ↓	R-precision ↑
StackGAN (Zhang et al., 2017)	8.45 ± .03	-	-
StackGAN-VL	+1.93	-	-
AttnGAN (Xu et al., 2018)	25.89 ± .47	35.49	85.47 ± 3.69
AttnGAN-VL	+2.29	-5.54	+0.92
DM-GAN (Zhu et al., 2019)	30.49 ± .57	32.64	88.56 ± 0.28
DM-GAN-VL	+1.65	-0.27	+1.81

What we learned in this course!

Weeks	Theme
Week 1: Introduction to Natural Language Processing (NLP) Week 2: Word Embeddings (Word Vector for Meaning) Week 3: Word Classification with Machine Learning I Week 4: Word Classification with Machine Learning II	NLP and Machine Learning
Week 5: Language Fundamental Week 6: Part of Speech Tagging Week 7: Dependency Parsing Week 8: Language Model	NLP Techniques
Week 9: Information Extraction: Named Entity Recognition Week 10: Advanced NLP: Attention and Reading Comprehension Week 11: Advanced NLP: Transformer and Machine Translation Week 12: Advanced NLP: Graph Neural Network with NLP	Advanced Topic
Week 13: Future of NLP and Exam Review	

/ Reference

Reference for this lecture

- Deng, L., & Liu, Y. (Eds.). (2018). Deep Learning in Natural Language Processing. Springer.
- Rao, D., & McMahan, B. (2019). Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning. " O'Reilly Media, Inc.".
- Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.
- Manning, C 2018, Natural Language Processing with Deep Learning, lecture notes, Stanford University

- Yao, L., Mao, C., & Luo, Y. (2019, July). Graph convolutional networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, pp. 7370-7377).
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903.
- Yun, S., Jeong, M., Kim, R., Kang, J., & Kim, H. J. (2019). Graph Transformer Networks. In Advances in Neural Information Processing Systems (pp. 11960-11970).
- Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.
- Vashishth, S., Yadav, P., Bhandari, M., & Talukdar, P. (2019). Confidence-based graph convolutional networks for semi-supervised learning. arXiv preprint arXiv:1901.08255.