COMP4332 Group 4 :
Project 1, Project 2

CHAN Chun Hin, CHAN Long Ki,
LEE Seungho, WAN Nga Chi

PROJECT 1
Sentiment Analysis Report

<u>DATA PREPROCESSING</u>

Biased Data set

- Biased Towards positive and neutral rating
  → Poor testing accuracy and F1 Score

Not applied stopword(nltk library) filtering: Not for sentiment analysis

Possible Ratings	Possible words in reviews
1	Very not good
2	Not good
3	Not so good
4	Good
5	Very good

Stopword Filtering

After Stopword Filtering
Good
Good
Good
Good
Good

CNN MODEL (Selected Model)

WHY?

→ Effectiveness in handling text input

Employed a kernel size of 1, 2, and 3 to extract unigram, bigram, and trigram

→ Further N-grams not useful (Can capture fewer words)

Used softmax function for output layer

→ Adam Optimizer for multi-label classification

The Model Layout:

Model: "model_9"

Layer (type)	Output Shape	Param #	Connected to
input_10 (InputLayer)	[(None, 100)]	0	[]
embedding_9 (Embedding)	(None, 100, 100)	454600	['input_10[0][0]']
conv1d_27 (Conv1D)	(None, 100, 100)	10100	['embedding_9[0][0]']
conv1d_28 (Conv1D)	(None, 99, 100)	20100	['embedding_9[0][0]']
conv1d_29 (Conv1D)	(None, 98, 100)	30100	['embedding_9[0][0]']
activation_37 (Activation)	(None, 100, 100)	0	['conv1d_27[0][0]']
activation_38 (Activation)	(None, 99, 100)	0	['conv1d_28[0][0]']
activation_39 (Activation)	(None, 98, 100)	0	['conv1d_29[0][0]']
max_pooling1d_27 (MaxPooling1D)	(None, 1, 100)	0	['activation_37[0][0]']
max_pooling1d_28 (MaxPooling1D)	(None, 1, 100)	0	['activation_38[0][0]']
max_pooling1d_29 (MaxPooling1D)	(None, 1, 100)	0	['activation_39[0][0]']
flatten_27 (Flatten)	(None, 100)	0	['max_pooling1d_27[0][0]']
flatten_28 (Flatten)	(None, 100)	0	['max_pooling1d_28[0][0]']
flatten_29 (Flatten)	(None, 100)	0	['max_pooling1d_29[0][0]']
concatenate_9 (Concatenate)	(None, 300)	0	['flatten_27[0][0]', 'flatten_28[0][0]', 'flatten_29[0][0]']
dropout_9 (Dropout)	(None, 300)	0	['concatenate_9[0][0]']
dense_19 (Dense)	(None, 100)	30100	['dropout_9[0][0]']
activation_40 (Activation)	(None, 100)	0	['dense_19[0][0]']
dense_20 (Dense)	(None, 5)	505	['activation_40[0][0]']

Total params: 545505 (2.08 MB)
Trainable params: 545505 (2.08 MB)
Non-trainable params: 0 (0.00 Byte)

CNN MODEL (Selected Model)

Tested under different kernel configurations and stopword filtering conditions

Key Observations:

- Retaining stopwords appears to slightly improve model accuracy
- Increasing the complexity of the model does not enhance the performance
- Configuration [1, 2, 3] without stopword filtering showed the best overall performance

Performance result

CNN kernel size (input condition)	Accuracy	Precision	Recall	F1
[1, 2, 3] (input without stopword filtering)	0.543	0.524	0.514	0.514
[1, 2, 3] (input with stopword filtering)	0.524	0.503	0.491	0.493
[1, 2, 3, 4] (input without stopword filtering)	0.536	0.519	0.506	0.511

ALTERNATIVE MODELS

Developed to utilize multi – layer perception

MLP Model A:

Data preprocessing: TF-IDF

Architecture:

```
Pipeline
  TfidfVectorizer
  TfidfVectorizer(tokenizer=<function tokenize at 0x7d3a427780d0>)
MLPClassifier
MLPClassifier(activation='tanh', early_stopping=True,
              hidden_layer_sizes=(16, 16, 16), learning_rate='adaptive',
              max_iter=1000)
```

MLP Model B:

Data preprocessing: NLP (TF-IDF)

Architecture:

Model: "model_3"

Layer (type)	Output Shape	Param #	Trainable
dense1 (Dense)	(None, 64)	822784	Y
dense2 (Dense)	(None, 32)	2080	Y
dropout1 (Dropout)	(None, 32)	0	Y
batch_norm1 (BatchNormalization)	(None, 32)	128	Y
dense3 (Dense)	(None, 20)	660	Y
dropout2 (Dropout)	(None, 20)	0	Y
dense4 (Dense)	(None, 16)	336	Y
dense5 (Dense)	(None, 5)	85	Y

Total params: 826073 (3.15 MB)

Trainable params: 826009 (3.15 MB)

Non-trainable params: 64 (256.00 Byte)

MLP MODEL VALIDATION

Model A Outperformed Model B

- Use of TfidfVectorizer() substantially enhanced model performance

Model Complexity Not Correlated with Performance

- Model B had more layers and hidden units, and advanced technique

  → Simpler architecture is more effective for the sentiment analysis task

MLP Model	Accuracy	Precision	Recall	F1
Model A	0.535	0.43	0.48	0.45
Model B	0.469	0.37	0.43	0.39

CONCLUSION of Project 1

- CNN model with varying kernel sizes showed competitive validation accuracy.
- MLP Models' Insights:
  - Investigated the role of TfidfVectorizer() in enhancing MLP models' accuracy.
  - Identified that simpler MLP models outperformed more complex ones.
- Key Takeaways for Future Research:
  - The effectiveness of model simplicity and TfidfVectorizer() warrants further exploration.
  - Project findings contribute to the knowledge base for sentiment analysis advancements.

PROJECT 2
Social Network Mining Report

DeepWalk Model: Setup

- DeepWalk model from iterations 1 to 10

```
node dim: 5,    num_walks: 20,  walk_length: 20 building a DeepWalk model...     number of walks: 1455900        average walk le
ngth: 20.0000   training time: 286.5846
valid auc: 0.5183


node dim: 15,   num_walks: 20,  walk_length: 20 building a DeepWalk model...     number of walks: 1455900        average walk le
ngth: 20.0000   training time: 286.3902
valid auc: 0.6086


node dim: 5,    num_walks: 30,  walk_length: 15 building a DeepWalk model...     number of walks: 2183850        average walk le
ngth: 15.0000   training time: 326.7124
valid auc: 0.5138


node dim: 15,   num_walks: 25,  walk_length: 10 building a DeepWalk model...     number of walks: 1819875        average walk le
ngth: 10.0000   training time: 199.4453
valid auc: 0.6215


node dim: 10,   num_walks: 15,  walk_length: 15 building a DeepWalk model...     number of walks: 1091925        average walk le
ngth: 15.0000   training time: 165.1049
valid auc: 0.5701
```

```
node dim: 10,   num_walks: 5,   walk_length: 5  building a DeepWalk model...     number of walks: 363975 average walk length: 5.
0000    training time: 24.8419
valid auc: 0.5976


node dim: 15,   num_walks: 10,  walk_length: 20 building a DeepWalk model...     number of walks: 727950 average walk length: 2
0.0000  training time: 145.0097
valid auc: 0.6248


node dim: 5,    num_walks: 25,  walk_length: 25 building a DeepWalk model...     number of walks: 1819875        average walk le
ngth: 25.0000   training time: 424.9159
valid auc: 0.5156


node dim: 25,   num_walks: 20,  walk_length: 25 building a DeepWalk model...     number of walks: 1455900        average walk le
ngth: 25.0000   training time: 357.8886
valid auc: 0.6845


node dim: 5,    num_walks: 25,  walk_length: 10 building a DeepWalk model...     number of walks: 1819875        average walk le
ngth: 10.0000   training time: 195.2348
valid auc: 0.5195
```

- The output of this model:

Best valid AUC achieved	0.68452011
Corresponding node_dim	25
Corresponding num_walks	20
Corresponding walk_length	25

DEEPWALK Model

Parameter Settings Explored

Explored 10 different parameter settings:

- Node Dimension (node_dim_combination): [5, 10, 15, 20, 25]
- Number of Walks (num_walks_combination): [5, 10, 15, 20, 25, 30]
- Walk Length (walk_length_combination): [5, 10, 15, 20, 25, 30, 35]

Validation AUC Observations

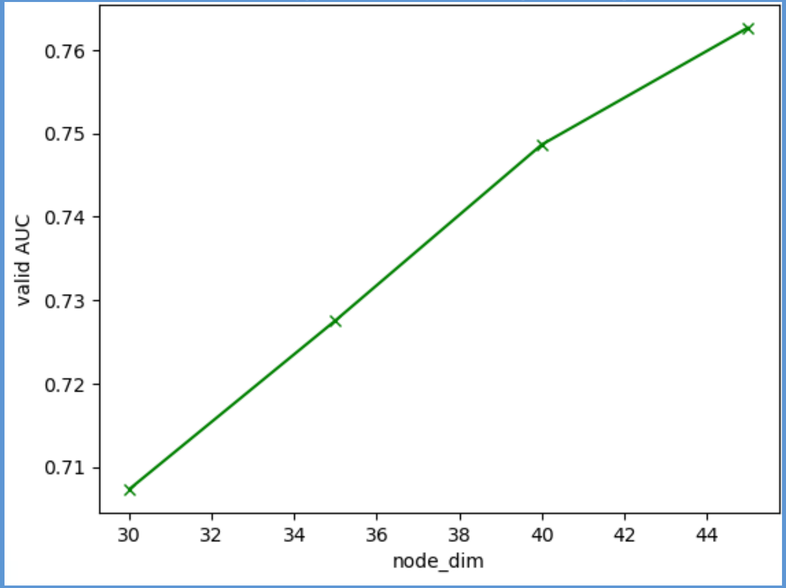
- Parameters to boost the Node2Vec model's validation AUC.

Parameter	Details	Observations of Validation AUC
Node Dimension	- 1st : 25 (Validation AUC = 0.6845) 2nd : 15 (Validation AUC = 0.6215)	Increasing node dimension by 10 raised validation AUC by 0.06.
Number of Walks and Walk Length	Both parameters should be at least 20	Using similar parameter values increases learning and AUC.
Range of Number of Walks	Range: 1450000 to 1820000	Maintain within the walk range yields a validation AUC of 0.65 to 0.70.

FURTHER INVESTIGATION

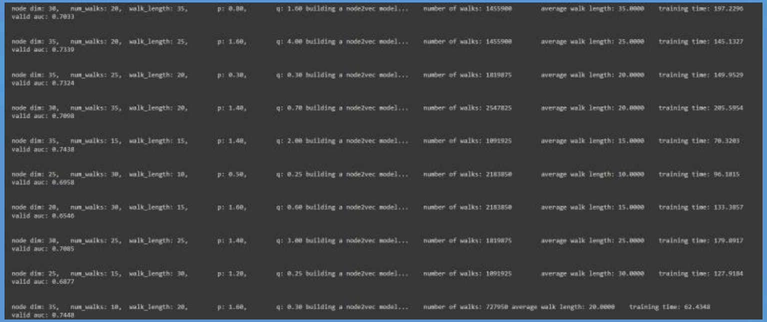
- Increasing node dimension improves validation AUC in the DeepWalk model.
- Higher dimensions capture more complex network relationships, enhancing model understanding and performance.
- A graph shows rising validation AUC with larger node dimensions, aiding tasks like link prediction and node classification.

Performance of DeepWalk of valid AUC against node_dim (with num_walk = 25 and walk_length = 25 being fixed)

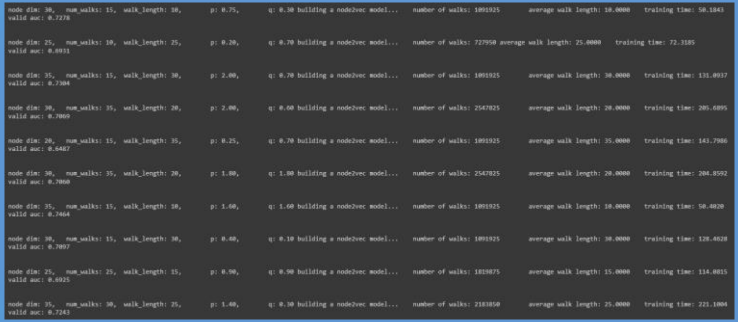


Node2Vec Model: Setup

Node2Vec Model from iteration 1 to 20



```
node dim: 30,   num_walks: 20,  walk_length: 35,        p: 0.80,        q: 1.60 building a node2vec model...    number of walks: 1455900        average walk length: 35.0000    training time: 197.2296
valid auc: 0.7033

node dim: 35,   num_walks: 20,  walk_length: 25,        p: 1.60,        q: 4.00 building a node2vec model...    number of walks: 1455900        average walk length: 25.0000    training time: 145.1327
valid auc: 0.7339

node dim: 35,   num_walks: 25,  walk_length: 20,        p: 0.30,        q: 0.30 building a node2vec model...    number of walks: 1819875        average walk length: 20.0000    training time: 149.9529
valid auc: 0.7324

node dim: 30,   num_walks: 35,  walk_length: 20,        p: 1.40,        q: 0.70 building a node2vec model...    number of walks: 2547825        average walk length: 20.0000    training time: 205.5954
valid auc: 0.7098

node dim: 35,   num_walks: 15,  walk_length: 15,        p: 1.40,        q: 2.00 building a node2vec model...    number of walks: 1091925        average walk length: 15.0000    training time: 70.3203
valid auc: 0.7438

node dim: 25,   num_walks: 30,  walk_length: 10,        p: 0.50,        q: 0.25 building a node2vec model...    number of walks: 2183850        average walk length: 10.0000    training time: 96.1815
valid auc: 0.6958

node dim: 20,   num_walks: 30,  walk_length: 15,        p: 1.60,        q: 0.60 building a node2vec model...    number of walks: 2183850        average walk length: 15.0000    training time: 133.3857
valid auc: 0.6546

node dim: 30,   num_walks: 25,  walk_length: 25,        p: 1.40,        q: 3.00 building a node2vec model...    number of walks: 1819875        average walk length: 25.0000    training time: 179.8917
valid auc: 0.7085

node dim: 25,   num_walks: 15,  walk_length: 30,        p: 1.20,        q: 0.25 building a node2vec model...    number of walks: 1091925        average walk length: 30.0000    training time: 127.9184
valid auc: 0.6877

node dim: 35,   num_walks: 10,  walk_length: 20,        p: 1.60,        q: 0.30 building a node2vec model...    number of walks: 727950 average walk length: 20.0000    training time: 62.4348
valid auc: 0.7448
```



```
node dim: 30,   num_walks: 15,  walk_length: 10,        p: 0.75,        q: 0.30 building a node2vec model...    number of walks: 1091925        average walk length: 10.0000    training time: 50.1843
valid auc: 0.7278

node dim: 25,   num_walks: 10,  walk_length: 25,        p: 0.20,        q: 0.70 building a node2vec model...    number of walks: 727950 average walk length: 25.0000    training time: 72.3185
valid auc: 0.6931

node dim: 35,   num_walks: 15,  walk_length: 30,        p: 2.00,        q: 0.70 building a node2vec model...    number of walks: 1091925        average walk length: 30.0000    training time: 131.0937
valid auc: 0.7304

node dim: 30,   num_walks: 35,  walk_length: 20,        p: 2.00,        q: 0.60 building a node2vec model...    number of walks: 2547825        average walk length: 20.0000    training time: 205.6895
valid auc: 0.7069

node dim: 20,   num_walks: 15,  walk_length: 35,        p: 0.25,        q: 0.70 building a node2vec model...    number of walks: 1091925        average walk length: 35.0000    training time: 143.7986
valid auc: 0.6487

node dim: 30,   num_walks: 35,  walk_length: 20,        p: 1.80,        q: 1.00 building a node2vec model...    number of walks: 2547825        average walk length: 20.0000    training time: 204.8592
valid auc: 0.7060

node dim: 35,   num_walks: 15,  walk_length: 10,        p: 1.60,        q: 1.60 building a node2vec model...    number of walks: 1091925        average walk length: 10.0000    training time: 50.4020
valid auc: 0.7464

node dim: 30,   num_walks: 15,  walk_length: 30,        p: 0.40,        q: 0.10 building a node2vec model...    number of walks: 1091925        average walk length: 30.0000    training time: 128.4028
valid auc: 0.7097

node dim: 25,   num_walks: 25,  walk_length: 15,        p: 0.90,        q: 0.90 building a node2vec model...    number of walks: 1819875        average walk length: 15.0000    training time: 114.0815
valid auc: 0.6925

node dim: 35,   num_walks: 30,  walk_length: 25,        p: 1.40,        q: 0.30 building a node2vec model...    number of walks: 2183850        average walk length: 25.0000    training time: 221.1004
valid auc: 0.7243
```

The output of this model:

Best valid AUC achieved	0.7464476625
Corresponding node_dim	35
Corresponding num_walks	15
Corresponding walk_length	10
Corresponding p	1.6
Corresponding q	1.6

PARAMETER SETTINGS

Explored 20 different parameter for the Node2Vec model:

Parameter	Values
Node Dimension	20, 25, 30, 35
Number of Walks	10, 15, 20, 25, 30, 35
Walk Length	10, 15, 20, 25, 30, 35
p (p_combination)	0.1, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.75, 0.8, 0.9, 1.2, 1.4, 1.6, 1.8, 2, 3, 4
q (q_combination)	0.1, 0.2, 0.25, 0.3, 0.4, 0.5, 0.6, 0.7, 0.75, 0.8, 0.9, 1.2, 1.4, 1.6, 1.8, 2, 3, 4

VALIDATION AUC Observations

Observed the following major phenomena:

- After evaluating various settings, we selected the best-performing Node2Vec model based on key observations from the training results.

Parameter	Observation and Impact on Validation AUC	Selected Value
Node Dimension	Highest validation AUC achieved with a node dimension of 35, indicating that increasing the node dimension increases AUC.	35
Number of Walks	Approximately 1100000 walks needed for a validation AUC of ~0.74, fewer than required in the DeepWalk model for similar AUC.	1,100,000
Walk Length	Effective walk length for higher validation AUC in the Node2Vec model is around 35, similar to the DeepWalk model.	35
p and q Parameters	No consistent pattern observed in the impact of p and q on validation AUC; their effectiveness depends on network specifics.	p: 0.5, q: 0.5

FURTHER INVESTIGATION

- Node Dimension: Best at 250; higher dimensions don't improve AUC.
- Parameters p and q: Optimal below 1 to balance exploration-exploitation.
- Walks and Length: Moderate values recommended; high values cause overfitting.

Outputs of different models

node_dim	num_walks	walk_length	p	q	valid auc acheived
50	15	15	0.5	0.5	0.78810
100	15	15	0.5	0.5	0.83343
150	15	15	0.5	0.5	0.84520
200	15	15	0.5	0.5	0.84581
200	100	100	0.5	0.5	0.58102
250	**15**	**15**	**0.5**	**0.5**	**0.84691**
300	15	15	0.5	0.5	0.84548
500	15	15	0.5	0.5	0.84510

CONCLUSION of Project 2

- Key Parameters:
  - Node Dimension: Increasing dimension improves validation AUC.
  - Number of Walks and Walk Length: Adequate numbers and lengths are crucial for effective model learning.
- Optimal Node2Vec Settings:
  - Node Dimension: 250;
  - Walks: 1,091,925
  - Walk Length: 15;
  - p and q: 0.5 each
- The best settings may vary based on the network's unique characteristics
- Future Steps:
  - Test these models on different social networks.
  - Explore advanced techniques like GCNs and GraphSAGE for improved performance.

THANKS!

Q&A