Assignment 3

Name: **Ong Jing Wei**

Student ID: **32909764**

Generative AI was used in this assignment

---

1. Collect a set of (machine-readable text) documents from an area of interest. For example, these could be a set of news stories, movie reviews, blogs, factual or creative writing. There is no restriction on the type of material you can choose although please avoid texts that might be offensive to people. As a guide, you should aim for the following:
   - Each document should be at least 100 words in length. Collect at least 15 documents.
   - Ideally, each document should cover one main topic and you should have at least 3 different topic areas in your collection of documents. Label each document as belonging to one of the topic areas in your corpus.
   - You can collect the documents as PDFs or as copied text from web-based articles or as text or other files.
   - Reference the source of your documents (URL or bibliographic citation (APA or Harvard style). **(2 Marks)**

For data collection, I collected documents related to the topic of Artificial Intelligence (AI). These documents were gathered from online resources such as blogs, research papers, and articles. I retrieved the documents by copying and pasting relevant paragraphs into text files, and then labelled them according to their topics. The documents are categorized into three different areas relevant to the chosen topic: machine learning, natural language processing, and robotics.

2. Create your corpus by first converting each document into a text format. The type of original material you collect will determine the way you need to do this. For some formats you can simply copy and paste the text into an empty text file. For Word documents, HTML, and PDFs etc., you may find it simpler to create the text document using “export”, or “save as” function in software.
   - Describe the process you follow for this step in your report.
   - Create your corpus using one of the methods covered in lecture videos and applied sessions. This could either be a folder of text files or a suitably formatted CSV file. Use suitable identifiers for your text file names or document IDs so that you can recognize the document in your clustering or network graphs. **(3 Marks)**

I collected material from various sources such as blogs, research papers, and articles related to Artificial Intelligence (AI). To convert these materials into a usable format, I copied and pasted a few paragraphs from each document into an empty text file. This process ensured that I could retain the relevant content while creating a text-based corpus.

I used a consistent naming convention for each file to organise the documents. Each file was named according to its topic, with a number appended to distinguish between different

documents in the same category. For instance, a document related to machine learning was named "MachineLearning1.txt", "MachineLearning2.txt", and so on. All of these text files were then stored within a single folder named "Artificial Intelligence."

```
# getting document folder
docs_folder <- "/Users/jingweiong/Downloads/Artificial Intelligence/"
setwd(docs_folder)

# check all the files inside that folder
list.files()

# creating a corpus from text
docs <- Corpus(DirSource((docs_folder)))
summary(docs)
```

I loaded the text files into a corpus which was done using the Corpus() function from the tm package in R. The DirSource() function was used to create a directory source from which the corpus was built.

```
> summary(docs)
                              Length Class             Mode
MachineLearning1.txt          2      PlainTextDocument list
MachineLearning2.txt          2      PlainTextDocument list
MachineLearning3.txt          2      PlainTextDocument list
MachineLearning4.txt          2      PlainTextDocument list
MachineLearning5.txt          2      PlainTextDocument list
NaturalLanguageProcessing1.txt 2     PlainTextDocument list
NaturalLanguageProcessing2.txt 2     PlainTextDocument list
NaturalLanguageProcessing3.txt 2     PlainTextDocument list
NaturalLanguageProcessing4.txt 2     PlainTextDocument list
NaturalLanguageProcessing5.txt 2     PlainTextDocument list
Robotics1.txt                 2      PlainTextDocument list
Robotics2.txt                 2      PlainTextDocument list
Robotics3.txt                 2      PlainTextDocument list
Robotics4.txt                 2      PlainTextDocument list
Robotics5.txt                 2      PlainTextDocument list
```

By looking at the summary of each document in the corpus, we can see that each document has the class PlainTextDocument indicates that the documents are plain text files.

3. Follow the text processing steps covered in lecture videos and applied sessions to create your Document-Term Matrix (DTM).
   - As part of this process, you may need to make particular text transformations to either preserve key words, or to remove unwanted terms, for example, characters or artefacts from the original formatting. Describe any processing of this kind in your report or state why you did not need to do so.
   - Your DTM should contain approximately 20 tokens after you have removed sparse terms. You will need to do this by trial-and-error to get the right number of tokens.
   - Include your DTM as a table in the appendix of your report. **(3 Marks)**

Before creating a Document-Term Matrix, we need to tokenize, filter and stem the documents. Hence, I first tokenized the documents by removing unwanted punctuation marks, numbers, and also converting all the characters to lowercase. Next, I filter the documents including replacing the special characters with spaces and removing all the white spaces as white space

are considered as characters in text. Words are also stemmed to reduce them to their root form, which helps in grouping together variations of words. Certain common words that do not add much meaning to the text such as ‘can’, ‘use’ and ‘make’ were also removed.

After all these, the document-term matrix was created using DocumentTermMatrix() function from tm package.

```
> inspect(dtm)
<<DocumentTermMatrix (documents: 15, terms: 878)>>
Non-/sparse entries: 1443/11727
Sparsity           : 89%
Maximal term length: 18
Weighting          : term frequency (tf)
Sample             :
                                Terms
Docs                             comput data human intellig languag learn machin model robot word
  MachineLearning1.txt                7    7     2        1       1    16     11     3     0    0
  MachineLearning2.txt                8   14     5        7       1    15     15     7     0    0
  MachineLearning4.txt                0    9     0        4       0     7      5     0     0    0
  NaturalLanguageProcessing1.txt      0    2     3        0       9     0      0    12     0    2
  NaturalLanguageProcessing2.txt      2    0     1        1      10     1      1     0     0    2
  NaturalLanguageProcessing5.txt      1    4     5        0       4     1      0     0     0    8
  Robotics2.txt                       1    0     0        0       2     0      1     0    16    2
  Robotics3.txt                       0    0     0        3       0     1      0     2     7    0
  Robotics4.txt                       0    2     0        0       0     0      0     0     6    0
  Robotics5.txt                       2    0     3        0       0     2      3     0     6    0
```

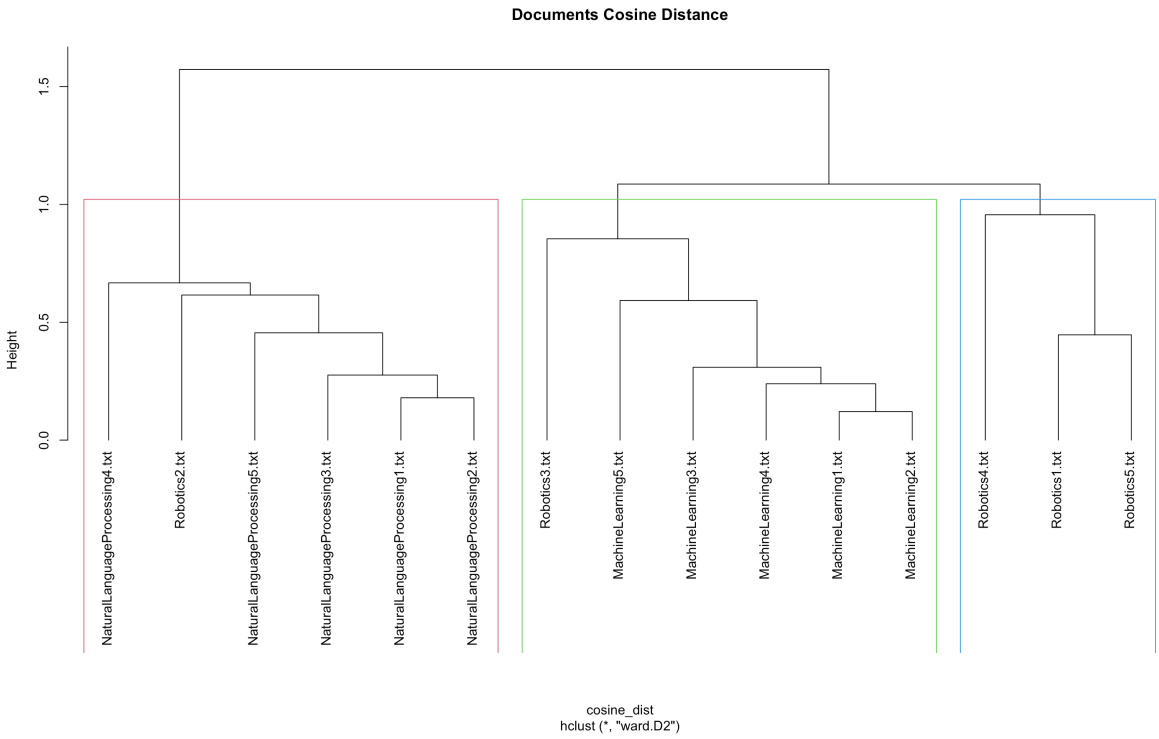
The Document-Term Matrix (DTM) consists of 15 documents and 878 terms. The sparsity of the matrix is 89%, indicating that many terms occur infrequently across the corpus. To reduce the sparsity, the sparse terms were removed to improve the efficiency and interpretability of the DTM. Terms with a sparsity of less than 0.65 were removed using removeSparseTerms() function to ensure it contains approximately 20 tokens.

```
> inspect(dtms)
<<DocumentTermMatrix (documents: 15, terms: 26)>>
Non-/sparse entries: 178/212
Sparsity           : 54%
Maximal term length: 10
Weighting          : term frequency (tf)
Sample             :
                                Terms
Docs                             algorithm comput data human intellig languag learn machin natur task
  MachineLearning1.txt                   7      7    7     2        1       1    16     11     2    0
  MachineLearning2.txt                   3      8   14     5        7       1    15     15     1    2
  MachineLearning3.txt                   1      0    2     0        0       0     6      1     0    0
  MachineLearning4.txt                   2      0    9     0        4       0     7      5     0    0
  NaturalLanguageProcessing1.txt         0      0    2     3        0       9     0      0     4    3
  NaturalLanguageProcessing2.txt         0      2    0     1        1      10     1      1     5    1
  NaturalLanguageProcessing3.txt         0      1    0     1        0       3     0      1     2    0
  NaturalLanguageProcessing5.txt         0      1    4     5        0       4     1      0     2    5
  Robotics4.txt                          0      0    2     0        0       0     0      0     0    0
  Robotics5.txt                          0      2    0     3        0       0     2      3     0    1
```

After removing the sparse terms, the resulting DTM consists of 15 documents and 26 tokens. The sparsity is lowered down to 54%. Since sparsity of 0.60 only giving 13 tokens, hence by trial and error, I decided to use terms with a sparsity of less than 0.65. The DTM in CSV file is included in the appendix.

4. Create a hierarchical clustering of your corpus and show this as a dendrogram.
   - Use the cosine distance between each document for clustering.
   - Identify which cluster each document belongs to.
   - Calculate the accuracy with which the clustering groups documents by topic.
   - Give a qualitative description of the quality of the clustering. **(4 Marks)**

Using the cosine distance matrix, hierarchical clustering was performed using the 'ward.D2' method. The resulting dendrogram below provided a visual representation of the hierarchical clustering of documents based on their cosine distances.



From the dendrogram above, we can observe that the clustering process effectively classified the majority of the documents correctly. However, it is worth noting that two of the Robotics text files were incorrectly grouped with the Machine Learning and Natural Language Processing clusters, indicating a slight misclassification.

To further evaluate the accuracy of the clustering, the documents were labelled with their corresponding topics: Machine Learning, Natural Language Processing, and Robotics. This allowed for a comparison between the assigned clusters and the true labels and help us to calculate for the accuracy.

```
> # confusion matrix
> confusion_matrix <- table(docs_clusters$Cluster, docs_clusters$TrueLabel)
> confusion_matrix

  MachineLearning NaturalLanguageProcessing Robotics
1               5                        0        1
2               0                        5        1
3               0                        0        3
> # Calculate accuracy
> accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
> print(paste("Clustering accuracy:", accuracy))
[1] "Clustering accuracy: 0.866666666666667"
```

The confusion matrix indicated that the clustering accuracy was approximately 86.67%. This suggests that the clustering effectively grouped the documents by their respective topics. Each

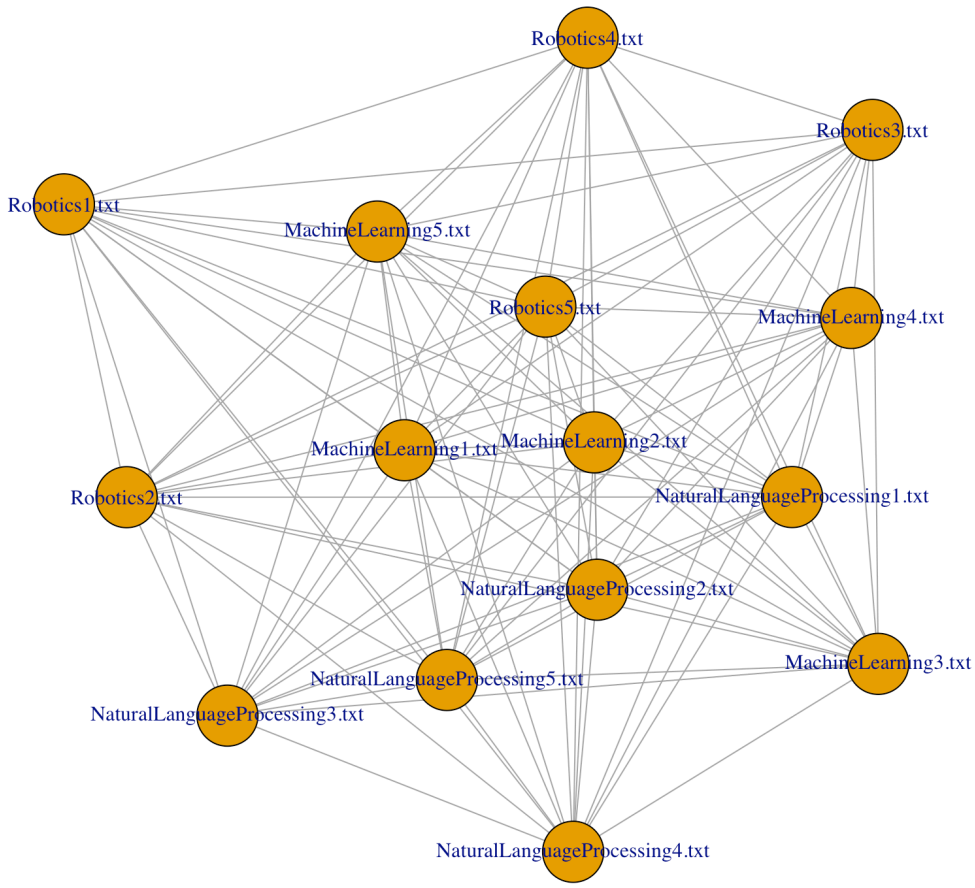
cluster predominantly contained documents from the corresponding topic, indicating a high level of accuracy in identifying the main topics present in the corpus.

5 Create a single-mode network showing the connections between the documents based on the number of shared terms.

- To do this you will need to first calculate the connections between each document using the method shown in Week 12, or another method of your choice.
- What does this graph tell you about the relationship between the documents? Are there any groups in the data you can clearly identify? What are the most important (central) documents in the network?
- Improve your graph over the basic example given in Week 12 to highlight two or more interesting features of your data, such as the strength of connections, the relative importance of nodes, communities in the network. **(4 Marks)**

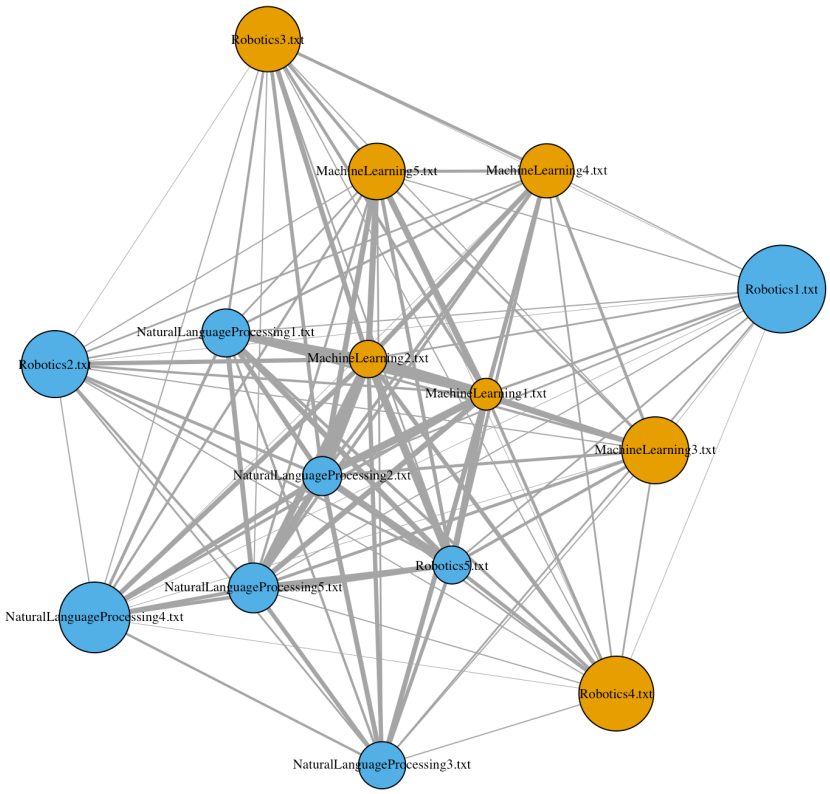
From the network graph based on shared terms among the documents, we can observe several interesting insights. The graph reveals the connections between different documents based on the number of shared terms. It provides a visual representation of the relationships between the documents. Screenshot below is the network graph:

Document Network based on Shared Terms



However, the graph above does not provide very useful information, hence to improve the graph, I adjusted the size of each node based on its closeness centrality, highlighting the most central documents in the network where bigger node size has higher importance. Next, I labelled each node with its document ID for improved readability and easier identification. Then, I assigned colours to nodes based on the detected communities, making it easier to distinguish between different groups of documents. Lastly, I scaled the width of each edge based on the strength of the connection which is the number of shared terms, emphasizing the relationships between documents where the greater the width, the stronger the relationship between the documents.

Improved Documents Network based on Shared Terms




```
> communities
IGRAPH clustering walktrap, groups: 2, mod: 0.0068
+ groups:
  $`1`
  [1] "MachineLearning1.txt" "MachineLearning2.txt" "MachineLearning3.txt" "MachineLearning4.txt"
  [5] "MachineLearning5.txt" "Robotics3.txt"        "Robotics4.txt"

  $`2`
  [1] "NaturalLanguageProcessing1.txt" "NaturalLanguageProcessing2.txt" "NaturalLanguageProcessing3.txt"
  [4] "NaturalLanguageProcessing4.txt" "NaturalLanguageProcessing5.txt" "Robotics1.txt"
  [7] "Robotics2.txt"                  "Robotics5.txt"
```

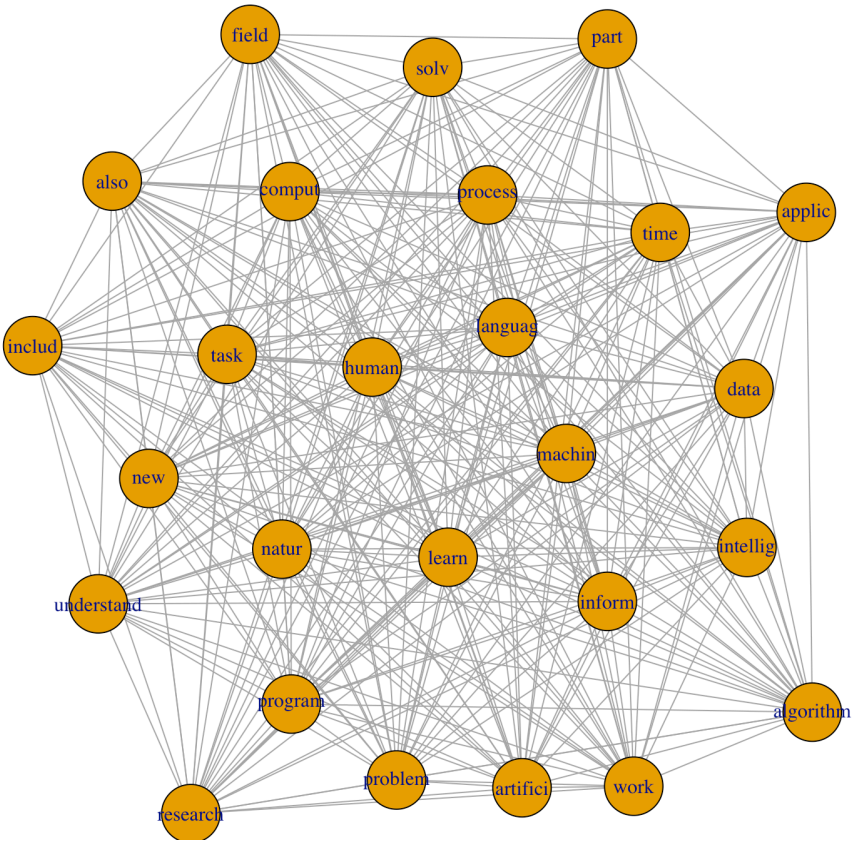
```
> sort(formatted_closeness)
          MachineLearning1.txt           MachineLearning2.txt                   Robotics5.txt
                       "0.17"                         "0.21"                         "0.21"
NaturalLanguageProcessing2.txt NaturalLanguageProcessing1.txt NaturalLanguageProcessing3.txt
                       "0.22"                         "0.26"                         "0.26"
NaturalLanguageProcessing5.txt           MachineLearning4.txt           MachineLearning5.txt
                       "0.27"                         "0.30"                         "0.31"
                 Robotics3.txt           MachineLearning3.txt                   Robotics2.txt
                       "0.36"                         "0.37"                         "0.37"
NaturalLanguageProcessing4.txt                  Robotics4.txt                   Robotics1.txt
                       "0.39"                         "0.41"                         "0.48"
```

By analysing the centrality measures and the node size from the improved graph, we find that the most central documents in the network are Robotics1.txt and Robotics4.txt, with closeness centrality scores of 0.48 and 0.41, respectively. These documents are crucial in the network as they are closely connected to other documents within their respective groups. Next, we can also identify two distinct groups from the graph network by the colour of the nodes as the first group consists of documents related to machine learning, while the second group comprises documents related to natural language processing and robotics.

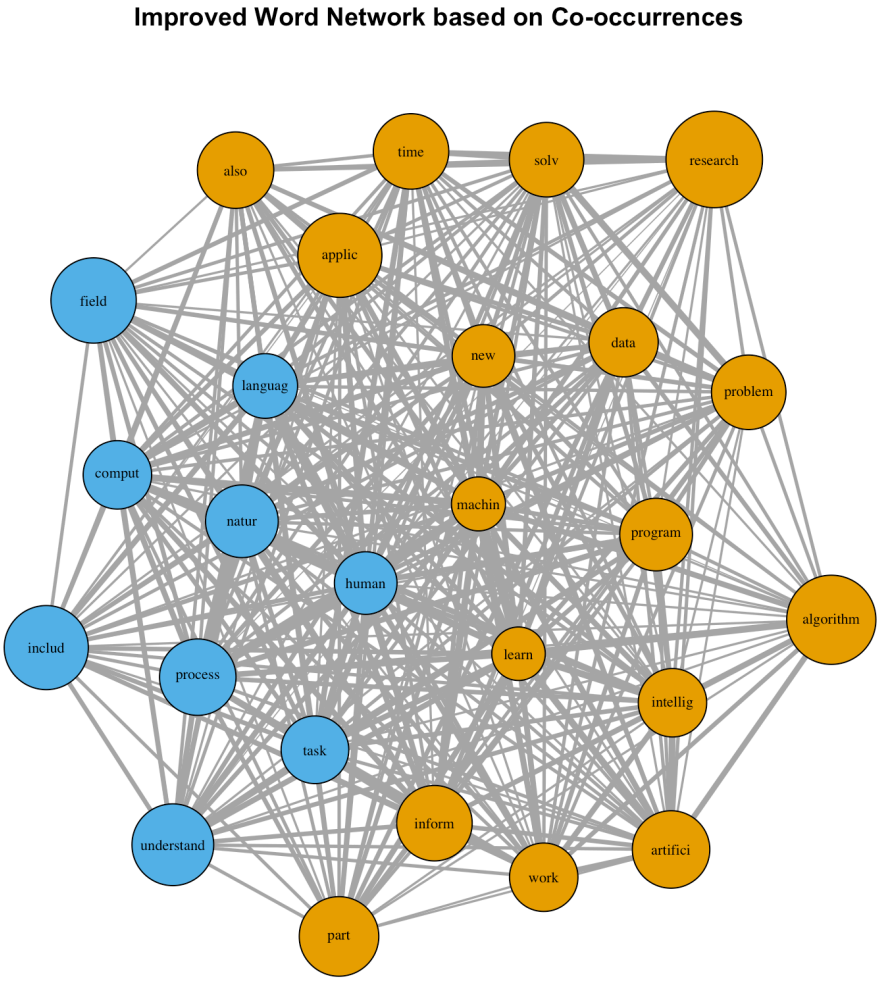
6 Repeat all the activities in Question 5, but now looking at the words (tokens). **(4 Marks)**

To create the word network based on co-occurrences, I calculated the connections between each word (token) using the shared terms method. I then constructed a single-mode network graph from the adjacency matrix, highlighting relationships between tokens.

Document Network based on Tokens



Enhancements were made to improve the clarity and interpretability of the graph just like question 5. First, I adjusted the size of each node based on its closeness centrality, emphasizing the importance of each word in the network. Next, I labelled each node with its corresponding token for easier identification and interpretation. Then, I assigned colours to nodes based on the detected communities, making it easier to distinguish between different groups of words. Finally, I scaled the width of each edge based on the strength of the connection, providing insights into the relationships and associations between words. The resulting graph offers a comprehensive view of the word network, revealing distinct communities and highlighting the most important tokens within the dataset.



```
> sort(token_importance)
  research  algorithm     applic     includ       part       also      field       time    problem
    "0.54"     "0.63"     "0.64"     "0.65"     "0.65"     "0.67"     "0.67"     "0.69"     "0.70"
      solv   artifici    program understand       data       work   intellig      natur        new
    "0.70"     "0.71"     "0.74"     "0.75"     "0.76"     "0.77"     "0.78"     "0.79"     "0.81"
      task     inform     comput    process    languag     machin      human      learn
    "0.82"     "0.83"     "0.85"     "0.88"     "0.90"     "0.98"     "1.00"     "1.00"
> sort(formatted_closeness)
     learn     machin      human    languag        new     comput       data   intellig       work
    "0.23"     "0.23"     "0.27"     "0.27"     "0.27"     "0.29"     "0.29"     "0.29"     "0.29"
      task      natur    program     inform    problem    process       solv       also       time
    "0.29"     "0.31"     "0.31"     "0.32"     "0.32"     "0.32"     "0.32"     "0.32"     "0.32"
  artifici       part understand     applic      field     includ  algorithm   research
    "0.33"     "0.34"     "0.35"     "0.36"     "0.36"     "0.36"     "0.38"     "0.41"
> communities
IGRAPH clustering walktrap, groups: 2, mod: 0.024
+ groups:
  $`1`
   [1] "algorithm" "applic"    "artifici"  "data"      "inform"    "intellig"  "learn"     "machin"
   [9] "new"       "problem"   "program"   "solv"      "work"      "also"      "research"  "time"
  [17] "part"

  $`2`
  [1] "comput"     "human"      "languag"    "natur"      "process"    "field"      "includ"
  [8] "task"       "understand"
```

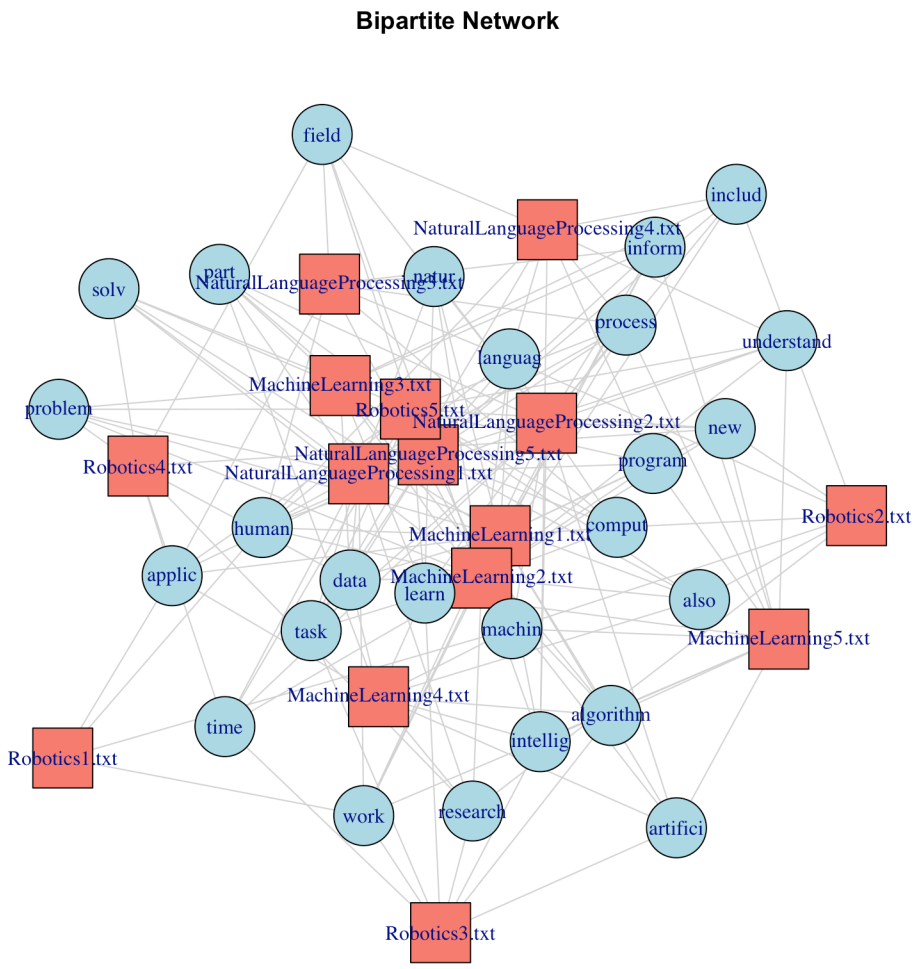
By looking at the improved graph, we can see that 'machin', 'human', 'learn' tokens are in the middle of the network graph. It indicates that these tokens have the highest closeness centrality. They played crucial roles in connecting various parts of the network. These central tokens are likely to represent key concepts or frequently discussed topics within the corpus. Understanding these central tokens can provide valuable insights into the main themes and subjects covered in the documents. We can also see their closeness centrality scores from the screenshot above, where 'machin', 'human', 'learn' has 0.98, 1.00 and 1.00 score respectively.

The graph also showing two distinct communities within the word network. These communities group together tokens with similar patterns of co-occurrence. Next, the improved graph provides a clear visualization of the connections between tokens based on co-occurrences. The width of each edge reflects the strength of the connection between two tokens. Stronger connections, indicated by wider edges, suggest frequent co-occurrence of those tokens in the documents.

7 Create a bipartite (two-mode) network of your corpus, with document ID as one type of node and tokens as the other type of node.

- To do this you will need to transform your data into a suitable format.
- What does this graph tell you about the relationship between words and documents? Are there any groups in the data you can clearly identify?
- Improve your graph over the basic example given in Week 12 to highlight two or more interesting features of your data, such as the strength of connections, the relative importance of nodes, communities in the network. **(4 Marks)**

The bipartite graph clearly illustrates the association between tokens and documents. Each document node is connected to the tokens that appear within it. Likewise, each token node is linked to the documents in which it occurs. The graph effectively captures the co-occurrence patterns between words and documents, demonstrating which words are prevalent in specific documents and vice versa.

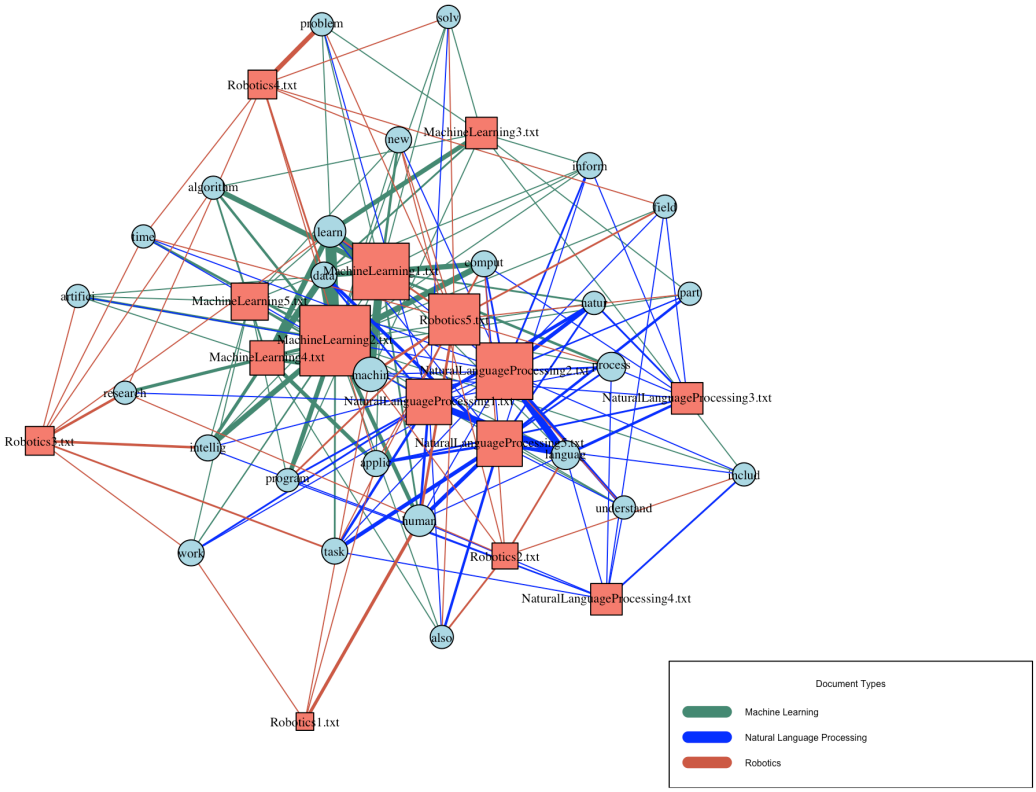


However, the original graph does not provide much information about their relationship. Hence, to enhance the visualization and analysis of the bipartite network, the size of the nodes is scaled based on their degree, providing insights into the relative importance of each node in the network. Next, the thickness of the edges between tokens and documents is scaled based on their weights, indicating the strength of the association. The edges are also coloured based on the type of document, allowing for a quick visual identification of different document categories. The improved graph is shown below.

From the improved graph, we can see that the graph highlights strong connections between specific words and documents through thicker edges. These connections suggest that certain words are highly prevalent in specific documents. For instance, words like ‘algorithm’, ‘data’, and ‘learn’ have thick edges connecting them to multiple Machine Learning documents, indicating their importance and frequent usage within this category. Next, the documents labelled ‘MachineLearning5.txt’ and ‘NaturalLanguageProcessing3.txt’ are significantly larger, suggesting they contain a higher frequency of words or cover more comprehensive content in their categories.

The color-coded edges also reveal distinct clusters or communities within the network. Documents and words related to Machine Learning are predominantly connected by green edges, those related to Natural Language Processing by blue edges, and Robotics by red edges. This color-coding helps to visually separate and identify the different thematic areas covered in the dataset.

Improved Bipartite Network with Weighted Edges, Scaled Node Sizes, and Colored Edges



8 Write a brief report (suggested length 8 – 10 pages).

- Briefly summarise your results identifying important documents, tokens and groups within the corpus. Comment on the relative effectiveness of clustering versus social network analysis to identify important groups and relationships in the data.
- Can you suggest improvements to text processing used in this assignment to better discriminate between the documents studied? Describe briefly how and why these methods work. To do this you may want to refer to recent references on Natural Language Processing. You are not required to implement these improvements.
- Include your R script as an appendix. Use commenting in your R script, where appropriate, to help a reader understand your code. Alternatively combine working, comments and reporting in R Markdown. **(8 Marks)**

Relative effectiveness of clustering versus social network analysis

Clustering techniques group documents based on how similar their content is. This helps in identifying cohesive groups within the dataset, making it easier to segment documents into distinct thematic areas. Clustering is useful for understanding the overall structure of the dataset and for identifying major topics.

On the other hand, Social Network Analysis (SNA) focuses on the relationships and interactions between documents and words. It highlights which documents and words are most

important and how often they appear together. SNA is particularly useful for finding key documents and words that connect multiple themes.

Both clustering and SNA provide valuable insights, but they serve different purposes. Clustering is effective for identifying broad thematic groups, while SNA offers a detailed view of the interactions and importance within those groups. Using both methods together provides a comprehensive understanding of the dataset, showing both the overall structure and the detailed relationships within the data.

Suggested Improvements to Text Processing

We can employ topic modelling techniques like Latent Dirichlet Allocation (LDA) to discover latent topics within the corpus. This method can help identify key themes and improve the differentiation between documents. Next, we can also use word embeddings such as Word2Vec or GloVe to capture semantic relationships between words. These embeddings can represent words in a continuous vector space, preserving semantic similarity and improving the discriminatory power of the text representation.

Appendix

	algorithm	applic	artifici	comput	data	human	inform	intellig	languag	learn	machin	natur	new
MachineLear	7	1	1	7	7	2	1	1	1	16	11	2	3
MachineLear	3	0	3	8	14	5	1	7	1	15	15	1	1
MachineLear	1	0	0	0	2	0	1	0	0	6	1	0	0
MachineLear	2	5	1	0	9	0	0	4	0	7	5	0	0
MachineLear	1	0	1	0	0	0	1	1	0	1	2	0	1
NaturalLangu	0	0	0	0	2	3	0	0	9	0	0	4	0
NaturalLangu	0	0	1	2	0	1	2	1	10	1	1	5	1
NaturalLangu	0	2	0	1	0	1	1	0	3	0	1	2	0
NaturalLangu	0	0	0	0	0	1	0	1	1	0	0	0	0
NaturalLangu	0	3	0	1	4	5	1	0	4	1	0	2	0
Robotics1.txt	0	1	0	0	0	4	0	0	0	0	1	0	0
Robotics2.txt	0	0	0	1	0	0	0	0	2	0	1	0	1
Robotics3.txt	1	0	1	0	0	0	0	3	0	1	0	0	0
Robotics4.txt	0	1	0	0	2	0	0	0	0	0	0	0	1
Robotics5.txt	0	1	0	2	0	3	0	0	0	2	3	0	1

problem	process	program	solv	work	also	field	includ	research	task	time	understand	part
1	3	1	1	1	0	0	0	0	0	0	0	0
1	0	6	1	1	1	1	1	4	2	3	1	0
1	0	0	1	0	0	0	1	0	0	0	0	1
0	0	0	0	0	1	0	0	1	0	0	0	1
0	1	1	0	1	0	0	0	0	0	0	1	0
1	1	2	1	2	1	0	0	1	3	1	0	1
0	2	0	0	1	0	1	1	0	1	1	4	1
0	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	0	0	0	1	2	0	1	0	1	0
0	2	0	0	0	3	0	1	0	5	0	1	3
0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	2	0	1	1	0	0	0	0
0	0	0	0	1	0	0	0	3	2	1	0	0
6	0	0	1	0	0	1	0	1	0	1	0	0
1	1	2	1	0	1	2	0	0	1	1	1	1

Figure 1: DTM

References

Brown, S. (2021, April 21). *Machine Learning, explained*. MIT Sloan. https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained

Gibney, E. (2024, May 28). *The ai revolution is coming to robots: How will it change them?*. Nature News. https://www.nature.com/articles/d41586-024-01442-5

Helfrich, T. (2023, October 5). *Council post: Why robotics and Artificial Intelligence are the future of Mankind*. Forbes. https://www.forbes.com/sites/forbestechcouncil/2022/05/31/why-robotics-and-artificial-intelligence-are-the-future-of-mankind/?sh=29f6870f1689

Jonathan Rossiter University of Bristol, Rossiter, J., & Bristol, U. of. (n.d.). *Robotics, smart materials, and their future impact for humans*. OpenMind. https://www.bbvaopenmind.com/en/articles/robotics-smart-materials-and-their-future-impact-for-humans/

Journal of Robotics | Hindawi. (n.d.-a). https://www.hindawi.com/journals/jr/2023/6630038/

Khurana, D., Koli, A., Khatter, K., & Singh, S. (2022, July 25). *Natural language processing: State of the art, current trends and challenges - multimedia tools and applications*. SpringerLink. https://link.springer.com/article/10.1007/s11042-022-13428-4

Machine learning. Machine Learning - an overview | ScienceDirect Topics. (n.d.). https://www.sciencedirect.com/topics/computer-science/machine-learning

Natural Language Processing: State of the art, current trends and challenges. (n.d.-b). https://www.researchgate.net/publication/319164243_Natural_Language_Processing_State_of_The_Art_Current_Trends_and_Challenges

The power of Natural Language Processing. Harvard Business Review. (2022, April 22). https://hbr.org/2022/04/the-power-of-natural-language-processing

Sarker, I. H. (2021a, March 22). *Machine learning: Algorithms, real-world applications and research directions - SN computer science*. SpringerLink. https://link.springer.com/article/10.1007/s42979-021-00592-x

Sarker, I. H. (2021b, March 22). *Machine learning: Algorithms, real-world applications and research directions - SN computer science*. SpringerLink. https://link.springer.com/article/10.1007/s42979-021-00592-x

Spiceworks.com. (n.d.). https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/

Terra, J. (2024, February 8). *The Future of Robotics: Everything you need to know in 2024: Simplilearn*. Simplilearn.com. https://www.simplilearn.com/future-of-robotics-article

What is natural language processing? definition and examples. Coursera. (n.d.). https://www.coursera.org/articles/natural-language-processing

What is NLP (Natural Language Processing)?. IBM. (2021, September 23). https://www.ibm.com/topics/natural-language-processing

When machine learning goes off the rails. Harvard Business Review. (2023, November 13). https://hbr.org/2021/01/when-machine-learning-goes-off-the-rails

R code

```
library(slam)
library(tm)
library(SnowballC)
library(igraph)

rm(list = ls())
#_____________________________________________________________________#
# Question 2

# getting document folder
docs_folder <- "/Users/jingweiong/Downloads/Artificial Intelligence/"
setwd(docs_folder)

# check all the files inside that folder
list.files()

# creating a corpus from text
docs <- Corpus(DirSource((docs_folder)))
summary(docs)

#_____________________________________________________________________#
# Question 3

toSpace <- content_transformer(function(x, pattern)gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "–")
docs <- tm_map(docs, toSpace, "“")
docs <- tm_map(docs, toSpace, "”")
docs <- tm_map(docs, toSpace, "’")
docs <- tm_map(docs, toSpace, "‘")
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, removeWords, stopwords("english"))
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs, stemDocument, language = "english")
removeWord <- content_transformer(function(x, pattern)gsub(pattern, "", x))
docs <- tm_map(docs, removeWord, "can")
docs <- tm_map(docs, removeWord, "use")
docs <- tm_map(docs, removeWord, "make")

# for checking purpose
writeLines(as.character(docs[[15]]))

# creating document term matrix
dtm <- DocumentTermMatrix(docs)
inspect(dtm)
dim(dtm)
```

```
# removing sparse terms
dtms <- removeSparseTerms(dtm, 0.65)
inspect(dtms)
dim(dtms)

freq <- colSums(as.matrix(dtms))
ord <- order(freq)
freq[tail(ord)]
freq[head(ord)]

#export document term matrix as csv
dtms = as.data.frame(as.matrix(dtms))
write.csv(dtms, "dtms.csv")
dtms_df <- read.csv("dtms.csv")
dtms_df

#_____________________________________________________________________#
# Question 4


# Calculate the cosine distance between documents
dtms_matrix <- as.matrix(dtms)
cosine_dist <- proxy::dist(dtms_matrix, method = "cosine")

# Perform hierarchical clustering using the cosine distance matrix
hc <- hclust(cosine_dist, method = "ward.D2")

# Plot the dendrogram
plot(hc, hang = -1, main = "Documents Cosine Distance")

# Assign documents to clusters
clusters <- cutree(hc, k = 3) # 3 topics
rect.hclust(hc, k=3, border = 2:5)
table(clusters)

# Create a data frame with document IDs and their assigned clusters
docs_clusters <- data.frame(DocumentID = rownames(dtms_matrix), Cluster = clusters)
docs_clusters

topics <- c(
  "MachineLearning", "MachineLearning", "MachineLearning", "MachineLearning",
"MachineLearning",
  "NaturalLanguageProcessing", "NaturalLanguageProcessing",
"NaturalLanguageProcessing",
  "NaturalLanguageProcessing", "NaturalLanguageProcessing","Robotics", "Robotics",
"Robotics",
  "Robotics", "Robotics"
)

table(GroupNames=topics, Clusters=clusters)
```

```
# Add true labels to the docs_clusters data frame
docs_clusters$TrueLabel <- topics
docs_clusters

# confusion matrix
confusion_matrix <- table(docs_clusters$Cluster, docs_clusters$TrueLabel)
confusion_matrix

# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
print(paste("Clustering accuracy:", accuracy))

#_____________________________________________________________________#
# Question 5

dtm_matrix <- as.matrix(dtms)

#convert to binary matrix
dtm_matrix = as.matrix((dtm_matrix > 0) + 0)

#multiple binary matrix by its transpose
abs_matrix = dtm_matrix %*% t(dtm_matrix)

#make leading diagonal zero
diag(abs_matrix) = 0

graph <- graph_from_adjacency_matrix(abs_matrix, mode = "undirected", weighted =
TRUE, diag = FALSE)
plot(graph, main = "Document Network based on Shared Terms")

# to improve graph
# Calculate closeness centrality
closeness_centrality <- closeness(graph, normalized = TRUE)
formatted_closeness <- format(closeness_centrality, digits = 2)

# Set vertex size based on closeness centrality
V(graph)$size <- closeness_centrality * 50  # Scale for better visualization

# Assign labels for better readability
V(graph)$label <- V(graph)$name
V(graph)$label.cex <- 0.7
V(graph)$label.color <- "black"

# Set vertex color based on communities
communities <- cluster_walktrap(graph)
V(graph)$color <- communities$membership

# Set edge width based on weights
E(graph)$width <- E(graph)$weight / max(E(graph)$weight) * 8  # Scale edge widths
```

```
# Plot the graph
plot(graph,
     vertex.size = V(graph)$size,
     vertex.label.cex = V(graph)$label.cex,
     vertex.label.color = V(graph)$label.color,
     vertex.color = V(graph)$color,
     edge.width = E(graph)$width,
     main = "Improved Documents Network based on Shared Terms")

# Analyse the network
communities
sort(formatted_closeness)

#_____________________________________________________________________#
# Question 6

term_matrix <- t(dtm_matrix) %*% dtm_matrix
diag(term_matrix) <- 0
term_matrix <- as.matrix(term_matrix)
word_graph <- graph_from_adjacency_matrix(term_matrix, mode = "undirected", weighted
= TRUE, diag = FALSE)
plot(word_graph, main = "Document Network based on Tokens")

# Calculate closeness centrality
closeness_centrality <- closeness(word_graph, normalized = TRUE)
formatted_closeness <- format(closeness_centrality, digits = 2)

# Set vertex size based on closeness centrality
V(word_graph)$size <- closeness_centrality * 60  # Scale for better visualization

# Assign labels for better readability
V(word_graph)$label <- V(word_graph)$name
V(word_graph)$label.cex <- 0.7
V(word_graph)$label.color <- "black"

# Set vertex color based on communities
communities <- cluster_walktrap(word_graph)
V(word_graph)$color <- communities$membership

# Set edge width based on weights
E(word_graph)$width <- E(word_graph)$weight / max(E(word_graph)$weight) * 6  # Scale
edge widths

# Plot the graph
plot(word_graph,
     vertex.size = V(word_graph)$size,
     vertex.label.cex = V(word_graph)$label.cex,
     vertex.label.color = V(word_graph)$label.color,
     vertex.color = V(word_graph)$color,
```

```
    edge.width = E(word_graph)$width,
    main = "Improved Word Network based on Co-occurrences")

token_importance <- format(eigen_centrality(word_graph)$vector, digits = 2)
sort(token_importance)
sort(formatted_closeness)
communities

#_____________________________________________________________________________#
# Question 7

dtm_matrix <- as.matrix(dtms)
# clone dtms
dtmsa = as.data.frame(dtm_matrix)
dtmsa$ABS = rownames(dtmsa)
dtmsb = data.frame()
for (i in 1:nrow(dtmsa)){
  for (j in 1:(ncol(dtmsa)-1)){
    touse = cbind(dtmsa[i,j], dtmsa[i,ncol(dtmsa)],colnames(dtmsa[j]))
    dtmsb = rbind(dtmsb, touse)
  }
}
colnames(dtmsb) = c("weight", "Docs", "token")
dtmsc = dtmsb[dtmsb$weight != 0, ]
dtmsc = dtmsc[, c(2, 3, 1)]

# Create a bipartite graph
bipartite_graph <- graph.data.frame(dtmsc, directed = FALSE)
bipartite.mapping(bipartite_graph)
V(bipartite_graph)$type <- bipartite_mapping(bipartite_graph)$type
V(bipartite_graph)$color <- ifelse(V(bipartite_graph)$type, "lightblue", "salmon")
V(bipartite_graph)$shape <- ifelse(V(bipartite_graph)$type, "circle", "square")
E(bipartite_graph)$color <- "lightgray"
plot(bipartite_graph, main = "Bipartite Network")


# to improve the graph
# Function to scale edge widths based on weights
scale_edge_width <- function(weights, min_width = 1, max_width = 10) {
  scaled_weights <- (weights - min(weights)) / (max(weights) - min(weights)) * (max_width -
min_width) + min_width
  return(scaled_weights)
}

# Function to scale node sizes based on degree
scale_node_size <- function(degrees, min_size = 5, max_size = 20) {
  scaled_sizes <- (degrees - min(degrees)) / (max(degrees) - min(degrees)) * (max_size -
min_size) + min_size
  return(scaled_sizes)
}
```

```
# Create the bipartite graph
bipartite_graph <- graph.data.frame(dtmsc, directed = FALSE)

# Ensure the bipartite attribute is set correctly
V(bipartite_graph)$type <- bipartite_mapping(bipartite_graph)$type

# Set node colors and shapes
V(bipartite_graph)$color <- ifelse(V(bipartite_graph)$type, "lightblue", "salmon")
V(bipartite_graph)$shape <- ifelse(V(bipartite_graph)$type, "circle", "square")

# Set edge widths based on weights
E(bipartite_graph)$weight <- as.numeric(E(bipartite_graph)$weight)
E(bipartite_graph)$width <- scale_edge_width(E(bipartite_graph)$weight)

# Set node sizes based on their degree
V(bipartite_graph)$size <- scale_node_size(degree(bipartite_graph))

# Define a function to assign edge colors based on document names
get_edge_color <- function(doc_name) {
  if (grepl("MachineLearning", doc_name, ignore.case = TRUE)) {
    return("aquamarine4")
  } else if (grepl("NaturalLanguageProcessing", doc_name, ignore.case = TRUE)) {
    return("blue")
  } else if (grepl("Robotics", doc_name, ignore.case = TRUE)) {
    return("coral3")
  } else {
    return("gray") # Default color for other cases
  }
}

# Extract document names from edges
edge_colors <- sapply(dtmsc$Docs, get_edge_color)
E(bipartite_graph)$color <- edge_colors

# Plot the improved bipartite graph
plot(bipartite_graph,
     vertex.size = V(bipartite_graph)$size,
     vertex.label.cex = 0.7,
     vertex.label.color = "black",
     edge.color = E(bipartite_graph)$color,
     edge.width = E(bipartite_graph)$width,
     main = "Improved Bipartite Network with Weighted Edges, Scaled Node Sizes, and
Colored Edges")

# Display legend for clusters
legend("bottomright", legend = c("Machine Learning", "Natural Language Processing",
"Robotics"),
       col = c("aquamarine4", "blue", "coral3"),
       lwd = 10, title = "Document Types", cex = 0.5)
```