DiscoGraMS: Enhancing Movie Screen-Play Summarization using Movie Character-Aware Discourse Graph

Maitreya Prafulla Chitale[1], **Uday Bindal**[1], **Rajakrishnan Rajkumar**[1], **Rahul Mishra**[1]

[1]IIIT Hyderabad
{maitreya.chitale, uday.bindal}@research.iiit.ac.in
{raja, rahul.mishra}@iiit.ac.in

Abstract

Summarizing movie screenplays presents a unique set of challenges compared to standard document summarization. Screenplays are not only lengthy, but also feature a complex interplay of characters, dialogues, and scenes, with numerous direct and subtle relationships and contextual nuances that are difficult for machine learning models to accurately capture and comprehend. Recent attempts at screenplay summarization focus on fine-tuning transformer-based pre-trained models, but these models often fall short in capturing long-term dependencies and latent relationships, and frequently encounter the *"lost in the middle"* issue. To address these challenges, we introduce **DiscoGraMS**, a novel resource that represents movie scripts as a movie character-aware discourse graph **(CaD Graph)**. This approach is well-suited for various downstream tasks, such as summarization, question-answering, and salience detection. The model aims to preserve all salient information, offering a more comprehensive and faithful representation of the screenplay's content. We further explore a baseline method that combines the CaD Graph with the corresponding movie script through a late fusion of graph and text modalities, and we present very initial promising results. We have made our code[1] and dataset[2] publicly available.

1 Introduction

Text summarization has been extensively studied within the NLP community (Nallapati et al., 2016, 2017; Zheng and Lapata, 2019; Urlana et al., 2024). Recently, large language models (LLMs) have demonstrated human-level performance in this area (Liu et al., 2023; Zhang et al., 2024). However, summarizing long documents remains a challenge for even the most advanced LLMs, as their effectiveness can be influenced by the location of salient information within the text (Liu et al., 2024). For language models to effectively utilize information within very long input documents, their performance should exhibit minimal sensitivity to the positional placement of relevant information within the input (Liu et al., 2024). Movie script or screenplay summarization (Papalampidi et al., 2020; Saxena and Keller, 2024) is a relatively hard task compared to standard document summarization due multitude of reasons. Movie scripts are typically very long documents characterized by intricate narratives, numerous subplots, and substantial dialogue, which pose significant challenges for summarizing the content without losing the core elements of the story. Many of the movie scripts have non-linear flow of events such as flashbacks, flash-forwards, and parallel plot lines, making the summary to retain the coherence and original flow.

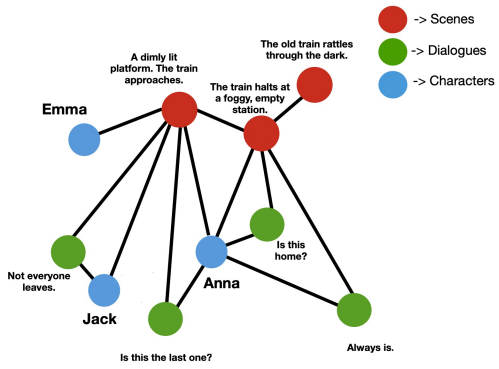

Figure 1: Example of a graph constructed from a movie script.

To address this, we present DiscoGraMS, an innovative resource that represents movie scripts as a character-aware discourse graph (CaD Graph). This graph captures the core essence of the movie plot by modeling latent relationships among key elements, including characters, the scenes they participate in, and the dialogues they deliver, thereby highlighting all possible semantically important aspects of the narrative. The CaD Graph captures intricate

[1]https://github.com/Maitreya152/DiscoGraMS
[2]https://huggingface.co/datasets/Maitreya152/CaD_Graphs

Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers), pages 954–965

nuances and the interplay between characters and scene sequences, effectively addressing challenges like flashbacks and sudden plot twists that are difficult to capture using only textual content. The main contributions of this work are as follows: 1) We introduce, for the first time to our knowledge, a movie character-aware discourse graph (**CaD Graph**) specifically designed for movie script summarization. 2) We propose a **late modality fusion model** that combines both CaD Graphs and textual content for improved movie script summarization. 3) We perform an **ablation study** to demonstrate the effectiveness of CaD Graphs in enhancing summarization.

2 Related Work

Since the origin of modern graph theory in 1736 with Euler's proof the *Seven Bridges of Königsberg* problem *i.e.* traversing a city crossing 7 bridges exactly once (Harary, 1960), graph representations have been used to model data in diverse fields like chemistry, biology and computer science. Linguistic data has also been represented as graph structures like dependency representations (Tesnière, 1959) and successfully deployed in NLP applications. The idea of representing entire texts as graphs was proposed in seminal work by Mihalcea and Tarau (2004). They created graphs comprising of nodes which keywords connected to other words located within a window of 2 to 10 words. This approach was extremely effective for the task of extractive summarization. More recently, Wang et al. (2022) show the efficacy of this technique for abstractive summarization of scientific articles. Here, entities in the text served as nodes (with coreferential entity clusters represented as a single node) connected to one another via labelled edges depicting relationships (like hyponymy) between nodes. (Kounelis et al., 2021) proposed a movie recommendation system using character graph embeddings to model relationships for movie similarity while (Papalampidi et al., 2021) propose a model for summarizing movie videos by constructing a sparse graph using only the turning point scenes from videos. In contrast, our CaD Graph method integrates scene, dialogue, and character interactions and focuses on summarizing movie text scripts, which presents a distinct set of challenges due to the long-form nature of screenplay texts. Prior work has explored character-based graphs in narratives. (Agarwal et al., 2013) introduced SINNET, a system for extracting social interaction networks from text. (Srivastava et al., 2016) focused on inferring interpersonal relationships in narrative summaries, while (Elson et al., 2010) developed methods for extracting social networks from literary fiction. (Zhao et al., 2020) propose DualEnc to bridge the structural gap in data-to-text generation by integrating graph and sequential representations. Our work builds on these approaches by constructing a CaD Graph to enhance screenplay summarization. There have been no significant efforts to employ graphs for movie script summarization. Only recently, (Saxena and Keller, 2024) adapted TextRank (Zheng and Lapata, 2019), a sentence centrality-based graph approach, for movie scripts. However, this approach was outperformed by the simpler Longformer Encoder-Decoder (LED) model (Beltagy et al., 2020) by large margin.

3 Dataset

We use the MovieSum (Saxena and Keller, 2024) dataset, a comprehensive resource for movie summarization, containing 2,200 movie screenplays along with metadata and plot summaries, including movies up to 2023. The plot summaries are sourced from IMDb and Wikipedia, ensuring a diverse range of writing styles and perspectives. The summaries were generated through a combination of automatic extraction and manual curation by trained annotators. The scripts are in XML format, preserving key elements such as scene descriptions, dialogues, and character names for efficient analysis. The dataset is split into training (1,800 movies), validation (200 movies), and test (200 movies) sets, with average screenplay lengths of 29k words and summaries of 717 words. The summaries, sourced from IMDb and Wikipedia, blend automatic extraction and manual curation. Analysis reveals a high level of abstractiveness in the summaries, indicated by novel 3-grams and 4-grams not found in the original scripts.

% Novel n-grams in Summary			
1-grams	2-grams	3-grams	4-grams
31.69	68.88	93.12	98.6

Table 1: Percentage of novel n-grams in summary. (Saxena and Keller, 2024)

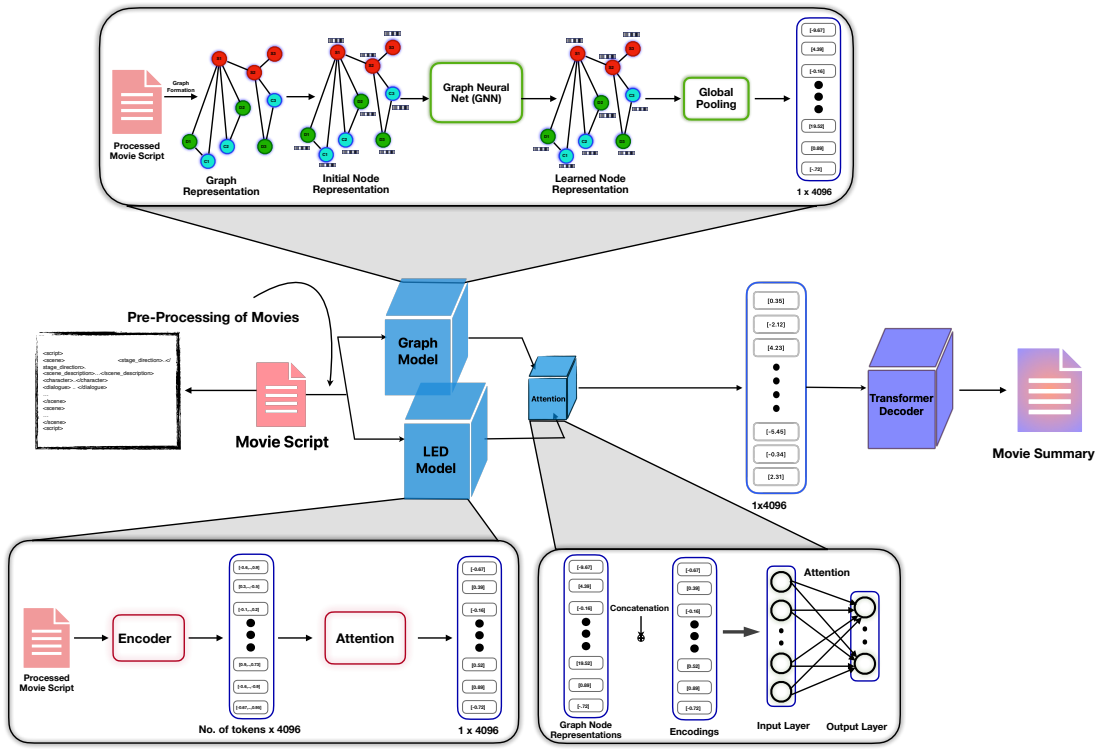

Figure 2: Architecture Diagram for the proposed model LGAT.

4 Methodology

In this section, we describe the process of constructing the character-aware discourse graph (CaD Graph) from movie scripts. We then present a baseline method that leverages both the CaD Graph and the textual content of the scripts, using a late modality fusion approach to generate movie script summaries.

4.1 Graph Construction and Encoding:

The first step involves constructing a graph representation of the movie script. In this representation, nodes are created for key elements, which are scenes, characters, and their dialogues.

The constructed graph can be described as a heterogeneous graph $G = (V, E)$, where V is the set of nodes, and E is the set of edges. There are three types of nodes, scenes (V_s), dialogues (V_d), and characters (V_c). The edges represent different relationships, $E_{ss} \subseteq V_s \times V_s$: Edges between consecutive scenes, $E_{sd} \subseteq V_s \times V_d$: Edges between scenes and dialogues occurring in those scenes, $E_{sc} \subseteq V_s \times V_c$: Edges between scenes and characters appearing in those scenes, $E_{cd} \subseteq V_c \times V_d$: Edges between characters and dialogues spoken by those characters. Formally, the graph construction is written as follows:

$G = (V_s \cup V_d \cup V_c, E_{ss} \cup E_{sd} \cup E_{sc} \cup E_{cd})$

Scene Nodes: $V_s = \{s_i \mid s_i \text{ is a scene}\}$ Each scene node s_i has an associated embedding $\mathbf{e}(s_i)$ representing the scene description text, derived from the sentence embedding model (SE) (Reimers and Gurevych, 2019): $\mathbf{e}(s_i) = \text{SE}(\text{Scene Description}(s_i))$ The scenes list is ordered according to the order in which the scenes occur in the movie.

Dialogue Nodes: $V_d = \{d_j \mid d_j \text{ is a dialogue}\}$ Each dialogue node d_j has an associated embedding $\mathbf{e}(d_j)$, representing the dialogue text: $\mathbf{e}(d_j) = \text{SE}(\text{Dialogue Text}(d_j))$

Character Nodes: $V_c = \{c_k \mid c_k \text{ is a character}\}$ The characters are initialised with zero embedding whose dimension matches with the embedding dimension of the sentence encoder.

Edges: The edges between the scenes and other entities are defined as follows: the scene-to-scene edges are given by $E_{ss} = ((s_i, s_{i+1}) \mid s_i, s_{i+1} \in V_s)$ the scene-to-dialogue edges are defined as $E_{sd} = ((s_i, d_j) \mid d_j \in V_d,\ s_i \in V_s,\ d_j \textit{ occurs in scene } s_i)$ the scene-to-character edges are defined as $E_{sc} = ((s_i, c_k) \mid c_k \in V_c,\ s_i \in V_s,\ c_k \textit{ occurs in scene } s_i)$ and finally, the character-to-dialogue edges are given by $E_{cd} = ((c_k, d_j) \mid c_k \in V_c,\ d_j \in V_d,\ d_j \textit{ is spoken by } c_k)$.

A movie's CaD Graph consists of intricate connections that represent the three-way relationships between scenes, characters, and dialogues, as illustrated in Figure 1. Adding sequential links between scenes helps the model capture the movie's overall flow. The connections from scenes to characters and dialogues to characters enable the model to differentiate between characters and understand their roles. We hypothesize that this structure also helps the model infer a character's significance within the movie, making our graphs, DiscoGraMS, character-aware.

4.2 The Proposed Model LGAT

We propose a novel late fusion-based model, LGAT, which integrates the CaD Graph and the textual content of movie scripts through a Graph Neural Network (GNN) using graph attention with convolutions and a Longformer Encoder-Decoder (LED) (Beltagy et al., 2020) text encoder, as illustrated in Fig 2. This combination generates the script's encoding, followed by a decoder that produces the summary. A detailed explanation of the model's internals is provided in Appendix Sec A due to space limitations.

5 Results

We select the models **LongT5** (Guo et al., 2022), **PEGASUS-X** (Phang et al., 2023), and the Longformer Encoder-Decoder (**LED**) model (Beltagy et al., 2020), (See Table 2) as the baselines (inspiration for baselines are drawm from (Saxena and Keller, 2024)) to compare with our proposed model.

Model	R-1 ↑	R-2 ↑	R-L ↑	BS_p ↑	BS_r ↑	BS_{f1} ↑
Baseline Models						
Pegasus-X 16K	42.42	8.16	40.63	58.81	56.06	54.36
LongT5 16K	41.49	8.39	39.78	56.09	55.60	55.68
Longformer (LED) 16K	44.85	9.83	**43.12**	59.11	58.43	58.73
Proposed Model						
LGAT (Ours)	**49.25**	**13.12**	34.61	**80.68**	**82.36**	**81.51**

Table 2: Comparison of Baseline Models and Proposed LGAT Model on the test set. The results of the baselines are referred to from (Saxena and Keller, 2024). Best scores are **bold**. Second Best scores are underlined. ↑ Indicates higher values are better.

The proposed model has the following configuration: LongFormer Encoder (LE) 4K + GATConv (**LGAT**), Where **LE** (Beltagy et al., 2020), is the longformer encoder. We use 4K context window for LED only compared to 16K used in MovieSum (Saxena and Keller, 2024) due to limited compute resources (Appendix C) availability, The results for this experiment can be obtained in Table 2.

As presented in Table 2, our proposed model, LGAT, significantly outperforms all baseline models on both ROUGE and BERT score metrics. This improvement can be attributed to the cues and patterns provided by the CaD Graph, which capture the overall essence of the movie plot. However, we observe that for the ROUGE-L metric, LGAT does not surpass the LED baseline, likely due to the smaller context window used in our encoder (4K vs. 16K).

5.1 Ablation Studies

The **LE** architecture, along with **GATConv**, has proven to be suited for processing long sequences. Following this, we run ablation studies on **LGAT** to prove the effectiveness of our proposed architecture of combining **GATConv** and **LE**. Specifically, we train both the encoders decoupled and test them on the test set. We compare the results against the full model (LGAT) to prove the effectiveness of the individual parts of the architecture, and hence show how they individually contribute towards the final result. To further strengthen our hypothesis regarding the importance of incorporating character information in the graph, we perform an additional ablation study. Specifically, we remove all character-related nodes and edges from the graph and evaluate the performance of the model in this modified setup. This ablation isolates the impact of character awareness in the graph structure and provides insight into the contribution of character-related information to the model's effectiveness. The results for this ablation study can be found in Table 3. We observe that GNN-based CAD graph encoding is very useful andcontributing more than LED-based textual encoder. Moreover, it is proved that character-awareness has a positive impact towards the performance of the model.

Model ↑	R-1	R-2	R-L	BS_{f1}
LE	16.16	1.63	13.20	71.95
GATConv	43.60	8.91	28.70	79.07
LGAT (Without Characters)	45.99	10.78	30.61	80.31
LGAT (Full)	**49.25**	**13.12**	**34.61**	**81.51**

Table 3: Results of Ablation Studies in comparison to our full model. Best Scores are **bold**. Second Best Scores are underlined. ↑ Indicates Higher The Better for all scores.

6 Discussion

Our experiments on abstractive summarization of movie screenplays (*i.e.*, the process of generating a plot summary given a screenplay) show that representing screenplays as graphs consisting of scenes, dialogues, and characters holds a lot of promise for movie summarization. To show how our character-aware graphs capture the roles of different characters in the graph and represent them, we plot the extracted node embeddings from our model in Figure 3. First, the node embeddings of all the nodes in the graph are extracted by passing the required movie graph over the GNN part of the final trained model. Next, the acquired node embeddings are filtered so that they only contain the node embeddings of the movie characters, and other node embeddings such as those of scenes and dialogues are discarded. Once the character node embeddings are extracted, they are analyzed using Principal Component Analysis (PCA) to reduce their dimensionality while preserving essential variance. We employ PCA to project the high-dimensional embeddings into a three-dimensional space, allowing for better visualization and interpretability. The transformed embeddings are then clustered using the K-Means algorithm, which groups characters into distinct clusters based on their learned representations.

To further illustrate the relationships and roles of different characters, we visualize the clusters in a three-dimensional scatter plot, where each point represents a character, and the color corresponds to the assigned cluster through K-Means clustering. This visualization enables us to observe meaningful patterns in the character representations. Characters who frequently interact or share similar narrative functions often appear closer together, whereas those with distinct roles are more clearly separated. The clustering also helps to reveal latent groupings, such as protagonists, antagonists, and supporting characters, as also depicted in Figure 3. This demonstrates how our approach successfully captures narrative structures through graph-based representation learning.

The effectiveness of our method in clustering and analyzing character embeddings suggests that our GNN-based approach learns informative representations that reflect underlying narrative and character dynamics.

Figure 3: Character Embeddings from the Movie: A Nightmare on Elm Street 3: Dream Warriors. Sections are annotated with the class of characters that is a majority within them.

7 Conclusion

Our approach outperforms quantitative results (except R-L) reported in prior work on movie summarization using the same dataset (Saxena and Keller, 2024). We attribute the better performance of our system to the presence of richer graphs, and encoding schemes. Specifically, we attribute the phenomenal improvement in BERT Score to the introduction of an attention layer to combine the encodings of the chunks as discussed in Section A.1 and the novel **CaD Graph** which enables the model to easily retain salient information which is validated by the high BERT Scores. We suspect that the low scores obtained in R-L are mainly due to the lower context size model (LED 4K) due to a restriction on the available compute resources. The model's (LED) low performance in isolation validates our believes. Our results indicate that knowledge-based representations of the text and plot structure help deep learning algorithms.

We expect our approach to have implications for other NLP problems like Question-Answering, Genre Identification, and Saliency Detection. (Xu et al., 2024) propose a system to represent narrative text consisting of passages as nodes connected by edges encoding cognitive relations between them. In addition to mainstream engineering applications, our graph representations can be deployed in scientific studies of inferencing processes in narrative comprehension by humans.

Limitations

Our graphs are devoid of co-reference resolution strategies which can take insights from the referred

characters and add crucial information about the movie plot. In addition to this, we were inhibited by our lack of compute resources, due to which we were not able to load the LED 16K model to encode movie scripts. This lack of compute resources also limited our choice of *architecture_dim* which is capped at 4K. This constraint potentially impacts the Rouge-L scores, resulting in lower performance. We were unable to conduct graph ablations (specifically, the removal of character and dialogue nodes) to evaluate their individual contributions to the model's performance. In future work, we plan to address these.

Ethics Statement

Dataset: Even though metadata and summaries of each movie are sourced from public domains (wikipedia, imdb), privacy and copyright considerations have been respected. Care has been taken so no sensitive or personally identifiable information is included. The movie scripts may reflect bias to particular genres or cultural context which may affect the behavior of the model.

Language Models: The paper includes the usage of pre-trained language models for the task of generating embeddings (section 4). These models are susceptible to biases inherent in their training data . As a result, any summaries produced from our model should be subject to manual review before being released.

References

Apoorv Agarwal, Anup Kotalwar, Jiehan Zheng, and Owen Rambow. 2013. Sinnet: Social interaction network extractor from text. In *The Companion Volume of the Proceedings of IJCNLP 2013: System Demonstrations*, pages 33–36.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *Preprint*, arXiv:2004.05150.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

David K Elson, Kathleen McKeown, and Nicholas J Dames. 2010. Extracting social networks from literary fiction.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736, Seattle, United States. Association for Computational Linguistics.

Frank Harary. 1960. Some historical and intuitive aspects of graph theory. *SIAM Rev.*, 2(2):123–131.

Agisilaos Kounelis, Pantelis Vikatos, and Christos Makris. 2021. Movie recommendation system based on character graph embeddings. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 418–430. Springer.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yixin Liu, Alex Fabbri, Pengfei Liu, Yilun Zhao, Linyong Nan, Ruilin Han, Simeng Han, Shafiq Joty, Chien-Sheng Wu, Caiming Xiong, and Dragomir Radev. 2023. Revisiting the gold standard: Grounding summarization evaluation with robust human evaluation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4140–4170, Toronto, Canada. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 3075–3081. AAAI Press.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Gulçehre Çağlar, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Pinelopi Papalampidi, Frank Keller, Lea Frermann, and Mirella Lapata. 2020. Screenplay summarization using latent narrative structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1920–1933, Online. Association for Computational Linguistics.

Pinelopi Papalampidi, Frank Keller, and Mirella Lapata. 2021. Movie summarization via sparse graph construction. In *Proceedings of the AAAI Conference*

on Artificial Intelligence, volume 35, pages 13631–13639.

Jason Phang, Yao Zhao, and Peter Liu. 2023. Investigating efficiently extending transformers for long input summarization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3946–3961, Singapore. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Rohit Saxena and Frank Keller. 2024. Moviesum: An abstractive summarization dataset for movie screenplays. In *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand. Association for Computational Linguistics.

Shashank Srivastava, Snigdha Chaturvedi, and Tom Mitchell. 2016. Inferring interpersonal relations in narrative summaries. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Lucien Tesnière. 1959. *Eléments de Syntaxe Structurale*. Klincksieck, Paris.

Ashok Urlana, Pruthwik Mishra, Tathagato Roy, and Rahul Mishra. 2024. Controllable text summarization: Unraveling challenges, approaches, and prospects - a survey. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 1603–1623, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. *Preprint*, arXiv:1710.10903.

Pancheng Wang, Shasha Li, Kunyuan Pang, Liangliang He, Dong Li, Jintao Tang, and Ting Wang. 2022. Multi-document scientific summarization from a knowledge graph-centric view. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 6222–6233, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Liyan Xu, Jiangnan Li, Mo Yu, and Jie Zhou. 2024. Fine-grained modeling of narrative context: A coherence perspective via retrospective questions. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5822–5838, Bangkok, Thailand. Association for Computational Linguistics.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2024. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 2481–2491.

Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6236–6247, Florence, Italy. Association for Computational Linguistics.

A Details of the Proposed Model

The constructed CAD graph is subsequently encoded using a Graph Attention Network (GATConv in PyTorch Geometric [3]) (Veličković et al., 2018). This encoding process helps in capturing complex relationships and contextual information inherent in the graph structure. The resulting graph embeddings provide a rich representation of not only the interconnections among scenes, characters, and dialogues, but also the information contained within the scenes, and dialogues.

The choice of a GATConv was made by keeping in mind that not all scenes, dialogues, or characters, are equally important and should be included in the summary. Thus, a convolution method which attends differently to different nodes was an ideal choice for this.

A.1 Movie Script Encoding:

We employ the longformer encoder to generate embeddings for the textual content of the movie script.

First, the entire script is divided into chunks, with each chunk sized according to the maximum input length the encoder can process.

Each chunk is then passed through the encoder, producing an encoding of shape $[chunk\_size, max\_tokens, encoding\_dim]$, where *encoding_dim* refers to the dimensionality of the

[3] https://pytorch-geometric.readthedocs.io/en/latest/generated/torch_geometric.nn.conv.GATConv.html

encoder.
Finally, these embeddings are transformed into a single embedding of shape [1, *architecture_dim*] via a **multi-headed self-attention layer** (Vaswani et al., 2023). Here, *architecture_dim* is a hyper-parameter, as described in Appendix C, it also represents the final embedding dimension for the movie.
We hypothesize that by applying multi-headed self-attention, the resulting compressed embedding will effectively capture the most relevant parts of the movie for the purpose of summarization.

A.2 Encoding Integration:

After obtaining the encodings from both the Graph Encoder Model and the Text Encoder Model, we perform a **concatenation** of these representations and then pass it through another **multi-headed self-attention layer**. This integration facilitates an effective combination of features and relations derived from the graph as well as the raw text, resulting in a representation that contains both structural and linguistic information. This also allows our model to give preference to certain features and relations in specific cases. The combined encodings are then passed through a **feed-forward neural network**. The aim here is to collapse the dimension of the model from $2 * architecture\_dim$ (obtained after concatenation), back to *architecture_dim*. While doing this, we also hypothesise that the model prunes all the values with low importance after the concatenation, and only keeps the features and relations of high importance for the decoding part.

A.3 Decoding

We use the standard Transformer Decoder architecture described in (Vaswani et al., 2023) as the decoding architecture to facilitate the generation of movie summaries from the learned embeddings. The details of implementation of this decoder can be found in the Appendix C.

B Results and Findings

In this section, we provide the detailed results obtained during our experiments with **DiscoGraMS**.

B.1 Evaluation Metrics

To assess the performance of our proposed models in generating summaries, we employ two widely recognized evaluation metrics: **ROUGE** and **BERT Scores**. These metrics provide valuable insights into the quality and effectiveness of the generated summaries in comparison to the reference (gold) summaries. More details about the evaluation metrics can be found in Appendix E

C Implementation Details

We used a single NVIDIA RTX 6000 with 50 GB VRAM to train and test our model. The VRAM of the GPU was not enough to load models with a higher context size than 4K. 20 Epochs on the train set take 42 hours to complete, while testing on all 20 epochs takes another 4 hours. The hyper-parameters used while training are as follows:

- Number of Epochs: 20
- Learning Rate: 0.00001
- Architecture Dimension: 4096
- Sentence Encoder (SE) Dimension: 768
- Longformer Encoder (LE) Dimension: 1024
- Dropout in Attention Layer of Encoder: 0.15
- Number of heads in Encoder side Attention: 8
- Dropout in Attention of Encoding Integration: 0.15
- Number of heads in Attention of Encoding Integration: 8
- Decoder Number of Heads: 8
- Decoder Heads: 6
- Internal Dimension of Decoder: 8192
- Max Sequence Length of the Decoder: 2284

D Example of a CaD Graphs from the Dataset.

In this section, we provide real graphs that we obtain from the dataset used. We visualise these graphs with the help of gephi [4]. Through these examples, we aim to demonstrate our effective character-aware graph construction method and how it helps the model identify the salient characters in the network and the roles that they play. This can be observed by the high density of edges around pivotal characters in the movie. Naturally (or by design), the model will tend to give more importance to these nodes and their connected nodes, deeming them to salient.

- Example graph of the movie *8MM* from 1999 can be seen in Figure 4
- Example graph of the movie *The Iron Lady* from 2011 can be seen in Figure 5

[4] https://gephi.org/

- Example graph of the movie *Adventureland* from 2009 can be seen in Figure 6

E Evaluation Metrics

E.1 ROUGE Scores

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) Scores (Lin, 2004) are a set of metrics used to evaluate automatic summarization and machine translation by comparing the **overlap of n-grams** between the generated summaries and the reference summaries. We utilize three variants of ROUGE scores:

- **ROUGE-N**: This measures the overlap of n-grams (where n can be 1, 2, or higher) between the generated summary and the reference summaries. Specifically, ROUGE-1 (Referred to as **R-1** Later) calculates the overlap of uni-grams, while ROUGE-2 (Referred to as **R-2** Later) evaluates the overlap of bi-grams.

- **ROUGE-L**: This metric assesses the longest common sub-sequence between the generated and reference summaries. It captures the fluency of the summary and provides insights into its coherence by considering the order of the words. (This is Referred to as **R-L** Later)

Higher ROUGE scores indicate better alignment with the reference summaries.

E.2 BERT Scores

BERT Scores (Zhang* et al., 2020) leverage contextual embeddings derived from the BERT model (Devlin et al., 2019) to evaluate the quality of generated summaries. Unlike traditional n-gram-based methods, BERT scores take into account the **semantic similarity** between the generated and reference summaries. BERT Scores are usually reported as:

- BERT Score Precision ($\mathbf{BS}_p$): It focuses on the accuracy of the generated content.

- BERT Score Recall ($\mathbf{BS}_r$): It emphasizes completeness in capturing relevant content.

- BERT Score F1 Score ($\mathbf{BS}_{f1}$): It combines both metrics to provide a balanced assessment of summary quality

By utilizing both ROUGE and BERT scores, we can gain a well-rounded understanding of how our proposed models perform in terms of both surface-level text overlap and deeper semantic alignment with gold summaries. This dual approach allows for a more robust evaluation of the generated summaries, ensuring that they not only contain relevant information but also maintain coherence and fluency.

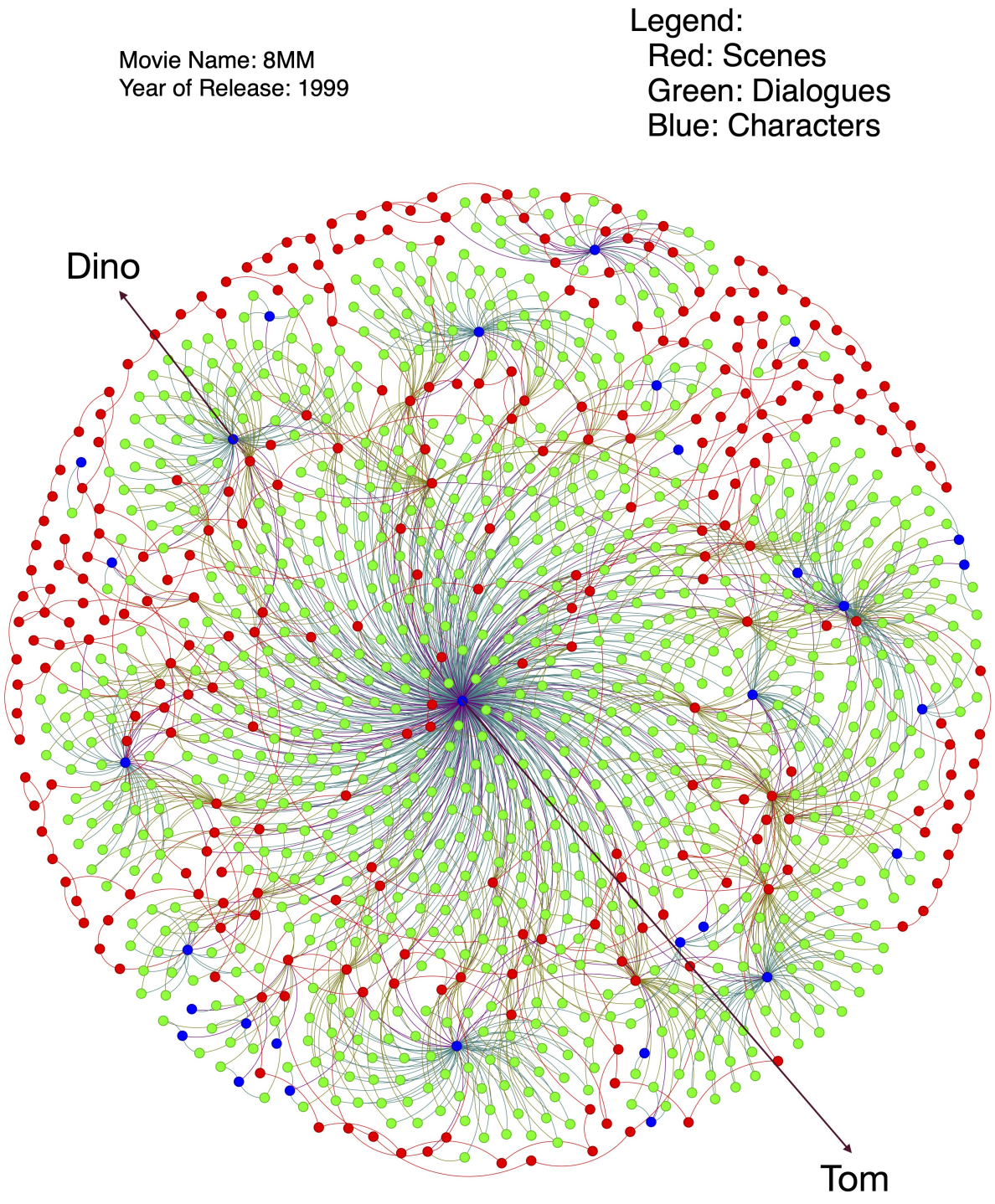

Figure 4: Tom being the Main Protagonist of the movie, naturally has the highest density of edges and is one of the central figures in the graph. This is expected as most of the movie revolves around him. Additionally, a high density can also be observed around the villains such as Dino.

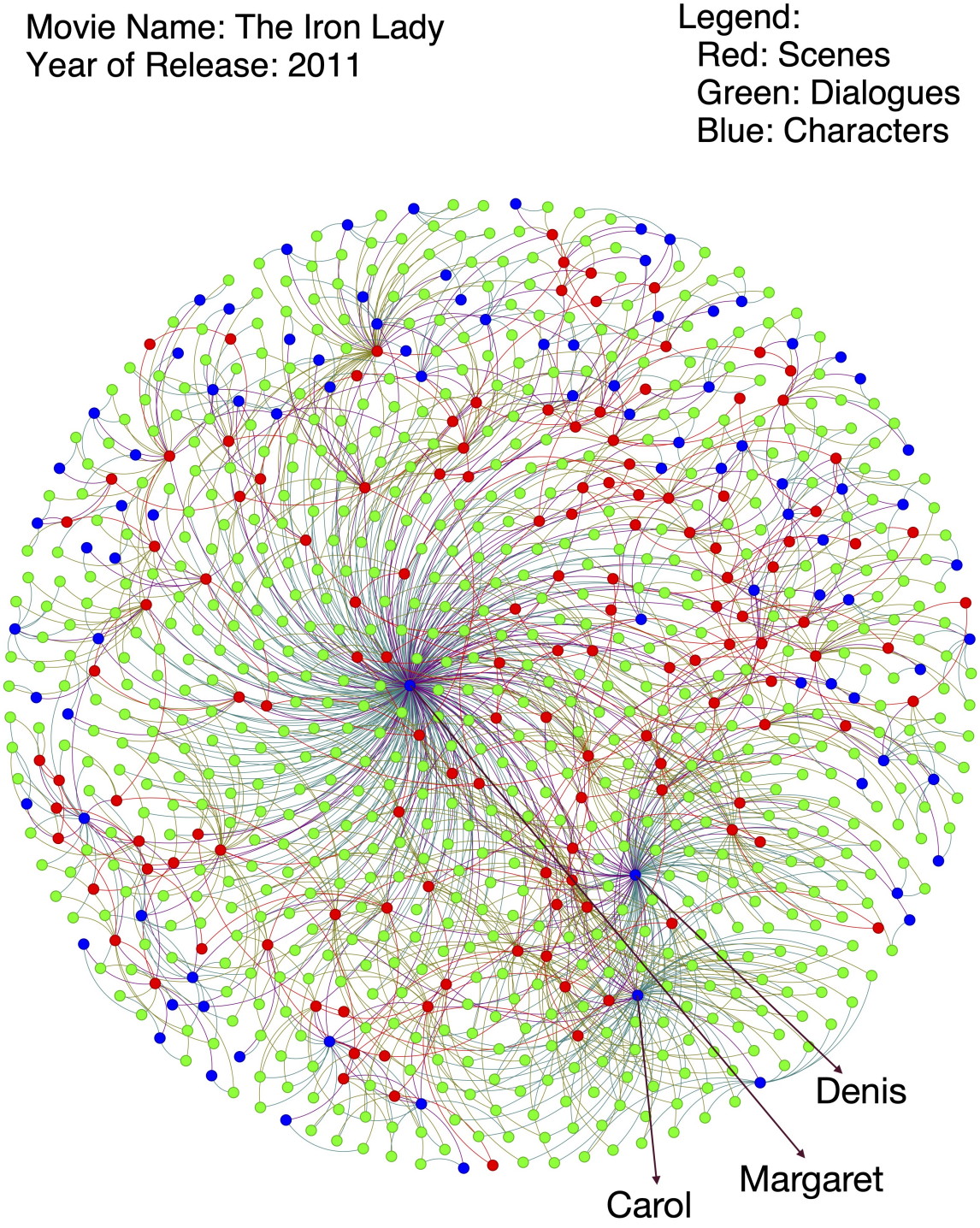

Figure 5: Margaret is the main protagonist of this movie and thus naturally has the highest concentration of edges around her. Additionally, Denis and Carol, her husband and daughter seem to be decently dense as well as they are the immediate family of the main protagonist and they too play an important role in the movie. Owing to the nature of the movie, there is no clear antagonist, and thus, no other major concentration region as well.

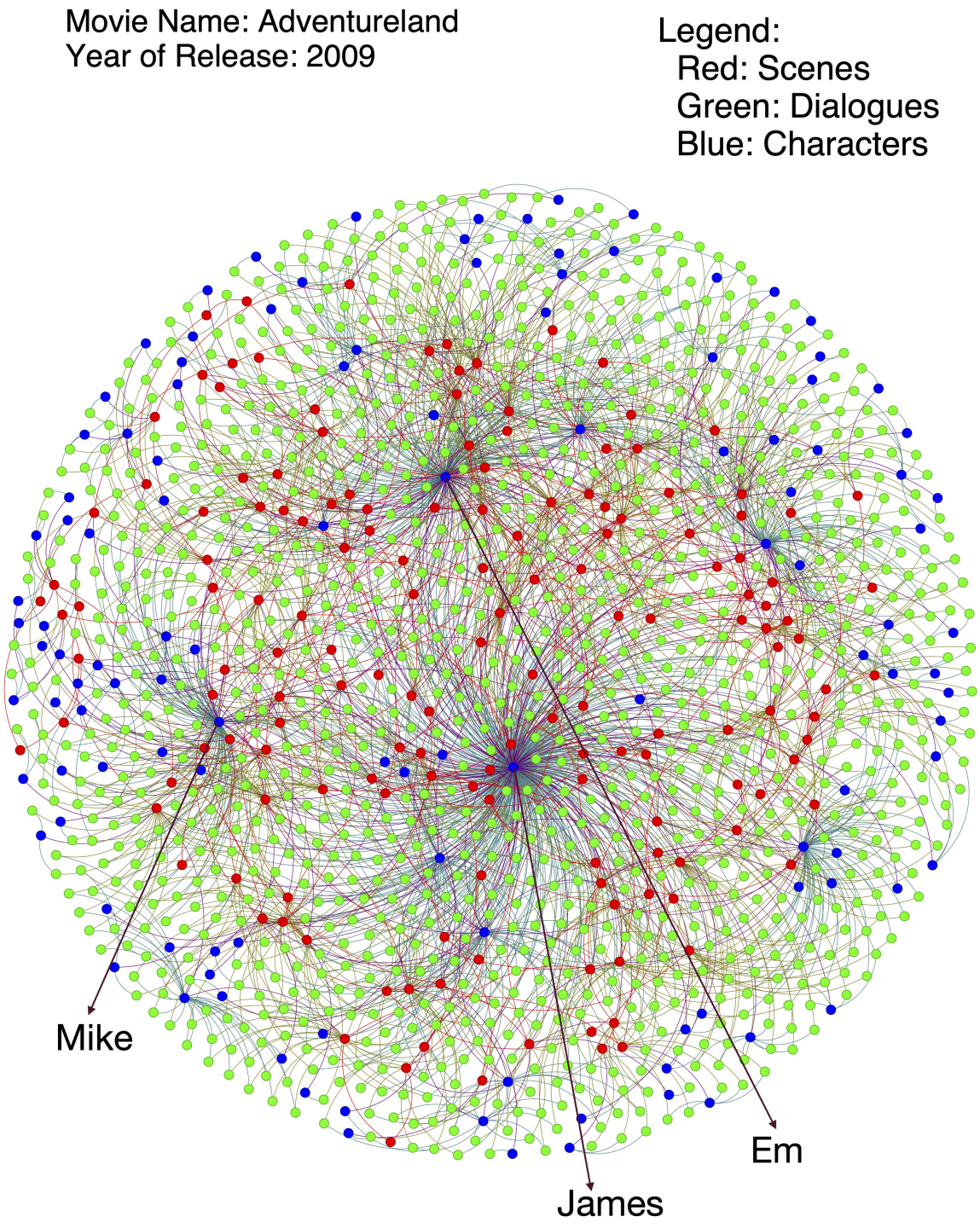

Figure 6: James and Em are the Main Protagonists in the movie, who have a relationship that has bloomed as their summer jobs started at the amusement park Adventureland. Mike is not a traditional villain, but complicates the protagonists relationship as he has an affair with Em. Thus, all three of them have high density edge connections as they contribute to the main density of the movie.