Learning Scene Representation using Character-Centered Graph

Eunchong Kim[1] and Yun-Gyung Cheong[2]

[1] Department of Artificial Intelligence, Sungkyunkwan University, Suwon, South Korea, prokkec@ skku.edu
[2] Department of Artificial Intelligence, Sungkyunkwan University, Suwon, South Korea, aimecca@ skku.edu

Abstract

With the advances in Natural Language Processing techniques, there is a growing interest in story comprehension and generation. This paper presents a graph-based approach to build scene embeddings, to understand narrative in movie scripts. Our method segments a script into scenes and extracts character actions and dialogues to form a story graph. The graph is then used for learning the scene representation leveraging the Metapath2Vec algorithm. For evaluation, we design classification tasks to test whether a scene embedding can represent the scene's contextual information. The results of the evaluation demonstrate that the proposed scene embeddings are effective in predicting the presence of characters and scene orderings.

Keywords— *Story, narrative, movie script, scene embedding, graph-based approach, context awarenesss, deep learning, NLP*

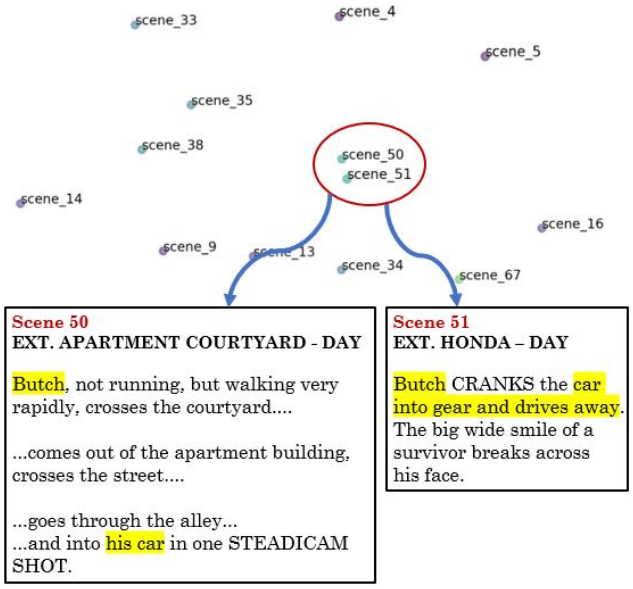

Fig. 1. An illustration of scene embeddings. The scenes close to each other in an embedding space share a context. For instance, scenes 50 and 51 contain the same character in a car, as indicated with the highlighted words.

I. Introduction

With the advances in Natural Language Processing techniques, there is a growing interest in story comprehension and generation. While stories can be conveyed in diverse formats, such as books and cartoons, movies and TV shows are the most popular media [15], with a multi-billion dollar market. Therefore, it becomes more important to understand film scripts.

Film scripts constitute a valuable corpus for investigating narrative and various NLP research topics. Prior research includes prediction of getting nominated for an award [4] and the detection of a bias in dialogue [1, 23]. In [10], screenplays are used to classify movies in terms of genre, mood, plot types, attitude, style, etc.

A screenplay consists of multiple scenes [3]. A scene can be viewed as a short story in itself and accounts for series of action and events that occur in continuous time and space [15]. Scene unit information can be useful to summarize a script by identifying importance scene chains [9, 17]. [19] present a method to identify scenes with a turning point, where the plot unfolds in a different direction.

While a number of previous works have utilized scenes for narrative generation and comprehension, little has been researched to understand them in terms of representation. To bridge the gap, this paper presents a graph-based approach to build scene embeddings, to understand narrative in movie scripts. Figure i. illustrates the scene embedding vectors with their corresponding scenes in the screenplay. To build scene embeddings, we first construct a graph which contains Scene, Character, Act, and Main Act nodes we define for this study. While the Act and Main Act nodes refer to sentences in a scene, the Scene and Character nodes represent the concepts of scenes and characters. Thus, we utilize a sentence embedding when initializing the (Main) Act node embeddings.

On completion of the story graph, the node embeddings are learned using Metapath2Vec, a 2-step representation learning algorithm [6]. The first step utilizes random walks following the predefined meta paths to generate training

data from the story graph. The second step learns an embedding vector for each node in the story graph, using the Word2Vec algorithm [16] with the training data built in the first step.

We create the scene embeddings and examine whether they represent scene-relevant information. We design three tasks to check if a scene embedding can represent the presence of main characters in the scene, scene ordering, and the scene's contextual information. For evaluation, we utilize the movie scripts from the two existing datasets: ScriptBase [9] and TRIPOD [19].

The contributions this paper makes are as follows:

- *We propose a novel embedding method to represent scenes, leveraging a story graph and a representation learning algorithm. To the best of our knowledge, this work is the first attempt to learn representations focusing on scenes in a movie script in an unsupervised manner.*
- *We design three tasks and build datasets from existing data to evaluate the effectiveness of the proposed scene embedding method.*
- *We carried out evaluations and the results indicate that our proposed embedding can effectively represent scenes compared with simple baseline models.*

The rest of this paper is organized as follows. Section ii. summarizes related works. Section iii. describes our method to create scene embeddings. Section iv. presents the experiments and the results. Section **??** discusses the findings obtained from qualitative analysis. Finally, Section v. provides conclusive remarks and future works.

II. RELATED WORK

In this section, we briefly summarize prior studies on movie script and narrative analysis using a graph structure.

A. Analysis of Movie Scripts

Several datasets provide movie scripts [9, 10]. The ScriptBase dataset [9] collects 1,276 scripts from web resources such as IMSDb[1]. In addition to the scripts, the dataset offers metadata such as keywords and summaries extracted from IMDB[2] and Wikipedia. The TRIPOD dataset contains movie scripts and synopses annotated with five turning points where the plot of the movie changes its direction [19].

Movie scripts can serve as excellent resources for NLP research. The dialogues in scripts are used to inspect bias in conversation [23, 1, 29, 22]. In [25], the BART model is used to identify seven biases, including age and gender, found in the dialogues of 35 films. [22] applies a connection frame with power and agency relationships to the narration and dialogue parts of the movie script to analyze gender bias. [4] analyzes movie scripts at the word-level to predict if the movies would get nominated for awards using tf-idf representation and the SVM classification algorithm. The classification performance was F1 score of 62.35% when using the ScriptBase dataset.

However, only a few studies analyze screenplays to understand narrative in movies. Scripts have complex formats, including action statements, character dialogues, slug lines, and camera cut scene directions [2]. Scenes can be regarded as the basic unit of a script. papalampidi-etal-2019-movie presents a neural approach to detecting turning points in the scenes, and its follow-up study summarizes a movie by extracting important scenes using the turning points [18]. [17] aligns sentences in a summary with their corresponding scenes in the script based on their lexical and semantic similarities.

As of our knowledge, [2] is the first work that presents scene embeddings. They embed the characters, the sentences of dialogues, and actions and combine them to create scene embeddings. These scene embeddings are then combined to build the script embeddings, which are used for various classification tasks such as predicting genre, mood, etc. Their embedding method leverages supervised learning algorithms which require labels. Moreover, their scene embeddings serve as an intermediate representation for building script embeddings.

Unlike previous studies, our work focuses on scenes, essential narrative units for analyzing movie scripts. Our goal is to learn scene embeddings in an unsupervised manner to understand narrative in movies.

B. Narrative Comprehension Using Graphs

Graphs have been used for narrative analysis and comprehension. Some prior studies leverage graphs to verify narrative analysis theories [13, 7, 24]. For instance, [13] conduct experiments with 19th-century English novels. They build story networks with characters as nodes and interactions between characters as edges, to find the difference between urban and rural novels. Graphs are used for capturing the flow of information found in a story [24] and character detection [27].

While the previous works make use of the graph's topological properties, recent works [9, 12] leverage graph structure for representation learning and summarization. [9] uses a graph-based model to find the optimal chain of scenes for generating summaries. They create a bipartite graph consisting of characters and scenes, and the degree of connection between the nodes is approximated using the random walk with restart strategy [26].

In [12], graphs are used to detect the presence of characters in novels. They extract character-related information

[1] https://imsdb.com/, one of the most famous sites that releases movie scripts

[2] https://www.imdb.com/, online site giving information of the movies, games, and TV shows

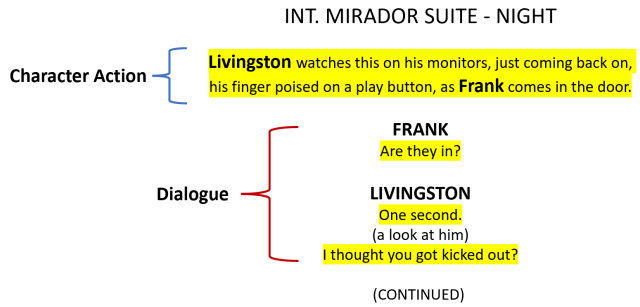

Fig. 2. An excerpt from the movie 'Ocean Eleven'. A scene contains the character's actions and dialogues, which are highlighted and used for constructing its story graph.

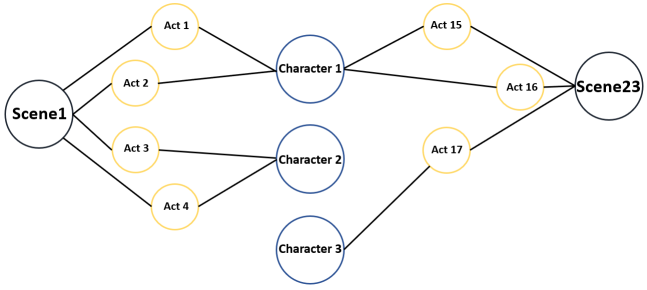

Fig. 3. A story graph which contains scene, character, and act nodes. The scene is linked to act and act is linked to character. The characters are also connected to acts in other scenes. Therefore, the character node creates a connection between different scenes.

from the sentences where characters appear. This information is used to form a character graph. Then, the DeepWalk method [20] is applied to the graph to learn the character embeddings. This method is suitable for homogeneous graphs, but not applicable to heterogeneous graphs with various node types.

In this paper, we construct a heterogeneous graph consisting of various node types representing story elements. Hence, we use the Metapath2Vec algorithm, suitable for dealing with heterogeneous graphs to learn scene embeddings.

III. METHODOLOGY

This section describes the proposed method to convert the textual description of a scene into an embedding vector via two phases: story graph construction and embedding.

A. STORY GRAPH CONSTRUCTION

First, we preprocess the script to remove text within parentheses since they typically denote acting directions. Next, we extract scenes using the headings 'INT(interior)' and 'EXT(exterior)', denoting scene transitions following [17]. Then, we extract character-related information from the scenes–actions and dialogues of characters, as shown in the highlighted parts in Figure A..

We use this character-related information to construct a story graph. Figure A. exemplifies a graph which is composed of three different nodes: scene, act, and character. A scene node represents a scene, so a graph contains as many scene nodes as the number of scenes in the script. An act node shows a character's action and dialogue. The action and dialogue parts obtained in the preprocessing stage are separated into sentences. And each sentence is represented as an act node. A character node identifies a character appearing in a scene. While a particular character can appear in multiple scenes, a story graph contains one character node for each character that is in a script. Hence, a story graph contains as many character nodes as the number of characters in the story.

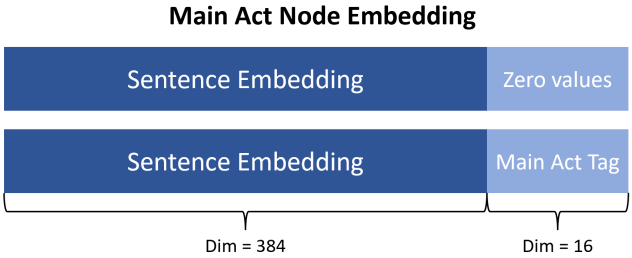

Fig. 4. A 16-dimension tag is attached only to Main Act. We use sinusoidal values to assign different tags to distinguish different scenes.

There are two types of edges: *a scene-act edge*, and an *act-character* edge. A *scene-act* edge connects a scene node and an act node, meaning that the act appears in the scene. An *act-character* edge denotes a relationship between an act and its associated character. When an act represents a dialogue (including voice over), the associated character is the speaker. When an act represents an action, the associated character is the character whose names appear in the sentence.

B. GRAPH TO EMBEDDING

We create scene embeddings from the story graph using the Metapath2Vec method [6]. The Metapath2Vec method samples paths on a graph. These paths are used as input samples for training the Word2Vec [16] Skip-gram model to create node embeddings. The process of sampling the paths is performed randomly, with setting the node types of the path. This path is called a meta path. For example, when the path type 'Act - Character - Act' is set, the sentences of the nodes constituting the path 'act1-character2-act3' can be sampled for training the model.

mckee1997story maintains that changes in characters lead to the development of a story, emphasizing the role of characters in narrative. Therefore, all the meta paths contain the Character node type. When training the Word2Vec model, we leverage the sentence embeddings using Sentence Transformer [21] to initialize the embedding of the Act nodes, which are represented as sentences. We believe that using this sentence embedding helps capture the contextual information among actions and dialogues. The ini-

Table 1. The number of movies, scenes and average of scenes of the datasets.

	# Movies	# Scenes	Avg Scenes
ScriptBase	1,200	166,611	138.8
TRIPOD	15	2075	138.3

tial embeddings of the Scene and the Character nodes are randomly set.

Finally, we define 'Main Act node' to mark important Act nodes that have high similarities with other Act nodes. We compute the importance of a node based on similarity as follows. When a scene contains N sentences, we calculate the cosine similarity of the sentence embeddings between each of these sentences and one of the other $N-1$ sentences. When the similarity between two sentences exceeds a predefined threshold value, we change their corresponding node types from 'Act' to 'Main Act'.

By adding these meta paths, we expect that the Main Act can contribute learning the embeddings of the Character nodes and Scene nodes of importance. We add an extra tag to the Main Act node embedding (see Figure A.) to distinguish them from non-Main Act nodes. The tag consists of 16 dimensions of sinusoidal values [28], so that the Main Acts from different scenes can have distinct values.

IV. EXPERIMENTS

A. EXPERIMENTAL SETTINGS

*1) DATASET*For evaluation, we use two datasets: the ScriptBase dataset [9] for creating the scene embedding and the TRIPOD dataset [19] for evaluating our method for classification tasks. The ScriptBase dataset contains 1,276 movie scripts with meta data such as summaries, genres, and loglines. We selected 1,200 movies that have clearly divided scenes (see table 1. The TRIPOD dataset [19] contains 99 screenplays and synopses; 84 screenplays are the train set and 15 are the test set. Five turning points are used to separate six stages: setup, new situation, progress, complications and higher stakes, the final push, and aftermath [11]. While the test set has both the screenplays and synopses annotated with these turning points, the training set has only the synopses annotated with the turning points. Hence, we selected 15 scripts from the test set for our evaluation. Each scene is annotated with one of the six stages. Table 2 displays the number of scenes for individual stages.

*2) MODEL SETTINGS*We set the hyperparameters for training the Metapath2Vec model. The walk length denotes the maximum length of a random walk. We set the walk length to 10 for the meta paths containing both the Character nodes and the Scene nodes. Otherwise, the walk length is set to 5. And in the random walks step, we can set the random walks per root node value, and we use 1 as the default. Exceptionally, considering that the number of character nodes is fewer than that of other nodes, it is set to 30 only in the path starting with a Character node. Word2Vec [16], used to learn the path, has 384 dimensions, and the window size is 3.

Sentence Transformer is used in the process of embedding action. We use the sentence-transformers/multi-qa-MiniLM-L6-cos-v1 pre-trained model[3], which has 384 dimensions. The model was learned for semantic search.

We conducted experiments to investigate the effectiveness of scene embeddings using classification tasks. For building a classification model, we leverage the BERT model [5] which has 384 dimensions and consists of 6 layers. We give scene embeddings as input to the model and use Tanh as the activation function of the last layer.

B. BASELINE MODELS

As of our knowledge, no attempts have been made to create scene embeddings in an unsupervised manner. An alternative baseline is to employ the scene embedding method using supervised algorithms [2], but their code is not accessible. For comparison, we create three baseline models as described below.

*1) DOC2VEC*First, we use doc2vec model [14] as a baseline. Doc2vec can obtain document embedding similar to word2vec, recognizing a scene as a document. For each scene, its text is given to the doc2vec model to obtain the output as the scene embedding vector.

*2) AVERAGING EMBEDDING*When we create a story graph, the embedding of individual Act node is obtained using the Sentence Transformer. We use the average of the embedding vectors of the Act nodes that make up a scene as its scene embedding.

*3) SUMMARIZATION*Summarizing a scene would generate a concise text that contains the most important information from the scene. We obtain a summary using an abstractive summarization model, since a scene includes dialogues and actions. DialogLM [30] is suitable for summarizing a long dialogue, and we use the DialogLED-large-5120[4] pre-trained model. We set the minimum length of the output statement to 20, the number of beam value to 3, and the no repeat ngram size to 2. Any symbols or numbers in the generated output sentences are removed. These summary sentences are given to the Sentence Transformer to generate its scene embedding.

C. EVALUATION TASKS

This section describes the tests we design to evaluate the effectiveness of the scene representation.

[3] Available at https://huggingface.co/sentence-transformers/multi-qa-MiniLM-L6-cos-v1

[4] Available at https://huggingface.co/MingZhong/DialogLED-large-5120

Table 2. The total number of scenes in each stage of the TRIPOD screenplays.

	# of scenes
Stage 1 (Setup)	293
Stage 2 (New situation)	462
Stage 3 (Progress)	562
Stage 4 (Complications and higher stakes)	542
Stage 5 (Final push)	284
Stage 6 (Aftermath)	90

Table 3. The number of samples in the train, validation, and test sets for the character identification and the scene ordering tasks. Thes Main Character identification task predicts whether a particular character appears in a given scene. Each dataset has a one-to-one ratio of appearance/non-appearance scenes. The Scene Ordering task predicts if a sequence of scenes is in forward or backward order. Each dataset consists of a one-to-one ratio of forward/backward labeled scene sequences

	Train	Validation	Test
1st Main Character	40,000	8,000	8,000
2nd Main Character	40,000	8,000	8,000
3rd Main Character	40,000	8,000	8,000
Scene Ordering	2,000	200	200

*1) MAIN CHARACTER IDENTIFICATION*The first task checks if the embedding can determine whether the main characters are included in the scene or not. We select three characters that appear most frequently in the script as the main characters. We run a classification task for each character to determine whether the particular character is included in a given scene or not. We build the evaluation dataset using the ScriptBase dataset, We randomly extract 56,000 scenes from the 1,200 scripts divided into train, validation, and test sets as shown in Table 3.

*2) SCENE ORDERING*The second task aims to test if the scene embedding can represent the story progression. The task is to classify if the given scenes are in forward or backward order. We build the evaluation dataset using the 1,200 scripts of the ScriptBase dataset. The data include 1,200 samples that arrange scenes in forward direction (as in the original script) and 1,200 samples in backward direction that we prepared. Table 3 shows the count of data samples for the train, valid, and test sets.

*3) SCENE COHESION*We design a scene cohesion test that checks if the spatial coordinates of two different scene embeddings are close to each other when they have similar contexts. We expect that the two scenes belonging to an identical stage would have similar contexts. In that case, their scene representation vectors would be located closely on the embedding space. When a scene is given, we look for its closest scene based on their embedding vectors. Then, a classification model determines if the given scene and its closest scene are in the same stage. We build evaluation data for this task using the 15 screenplays of the TRIPOD test set. We segment individual scripts into six stages using the five turning points annotated in the original data. Table 2 shows the number of scenes for each stage.

4) SCENE COHESION

D. RESULTS

This section reports and discusses the results of our evaluation.

*1) CHARACTER IDENTIFICATION*Table 4 shows the results of evaluating our scene embedding models for character identification and scene ordering tasks. Our proposed embedding model outperforms across all the tasks compared with the three baselines that employ Doc2vec, Averaging embedding, and summarization strategies. This indicates that our proposed embedding representation learned using the story graph and the Metapath2Vec learning method can effectively capture information of a scene.

We also experimented with the edge2vec method [8] which learns the scene representation using the same story graph and a different random walk strategy, and the result shows that its performance is lower than that of the Metapath2Vec method. The edge2vec method forms a transition matrix between different edge types and performs random walks based on it, which increases the probability of selecting the same node type as the next step. This strategy leads Edge2vec to set the path to visit the same node type, we believe that the data samples to learn Scene nodes can be relatively insufficient compared with Metapath2Vec where we can set the meta path to focus on learning the Scene nodes intensively.

We ran an ablation study to investigate the effect of using the Main Act node type. In Table 4, Metapath2Vec denotes the proposed embedding method using the Main Act nodes with a tag attached to their node representations (see Figure A.). We tested two cases: when tags are not used for the Main Act node representation (denoted as 'Metapath2Vec - Tag') and when the Main Act node types are not used (denoted as 'Metapath2Vec - Main Act - Tag' in the table). The latter case means that the Main Act nodes are regarded as Act nodes, and accordingly the four meta paths that contain Main Act nodes are not used. Although these cases score the highest for the 2nd and 3rd main character detection tasks, averaging the performance scores of detecting the three main characters shows that Metapath2Vec using the Main Act node types and tagging performs the best, however, overall, the difference is not significant.

*2) SCENE ORDERING*The result of classifying scene ordering shows that Metapath2Vec using the Main Act node types and tagging performs the best, reaching 0.936 in AuC and 0.845 in accuracy (see Table 4). The ablation study shows that the accuracy decreases by 0.085 when the tag is not used for the Main Act node representation. We believe that the tag can serve as positional encoding since it employs a sinusoidal value. For this task, the averaging embedding baseline performs as well as Metapath2Vec

Table 4. Experimental result of the the character identification and the scene ordering test. 1st, 2nd, and 3rd are in the order of the main characters that frequently appear in the script. Avg means the average of the performance scores for predicting the 1st, 2nd and 3rd characters. AUC refers to the bottom area of the ROC curve. The ROC curve is drawn by changing the threshold the final sigmoid value to determining the labels.

	1st main character		2nd main character		3rd main character		Avg		Scene Ordering	
	AUC	accuracy	AUC	accuracy	AUC	accuracy	AUC	accuracy	AUC	accuracy
Doc2vec	0.4649	0.4990	0.4617	0.4998	0.4472	0.4999	0.4579	0.4996	0.4998	0.5000
Averaging Embedding	0.6517	0.6060	0.6787	0.6054	0.6129	0.5777	0.6478	0.5964	0.8228	0.7550
Summarization	0.6006	0.5719	0.6505	0.5811	0.5990	0.5674	0.6167	0.5735	0.6711	0.6250
Metapath2Vec	**0.6958**	**0.6492**	0.6898	0.6396	0.615	0.5844	**0.6669**	**0.6234**	**0.9362**	**0.8450**
Edge2vec	0.6522	0.6004	0.6721	0.6276	0.6016	0.5746	0.6420	0.6009	0.8575	0.7850
Metapath2Vec - Tag	0.6738	0.6189	**0.7006**	**0.6430**	0.6203	0.5882	0.6649	0.6167	0.8364	0.7600
Metapath2Vec - Main Act - Tag	0.6758	0.6223	0.6852	0.6263	**0.6375**	**0.6020**	0.6662	0.6169	0.8107	0.7150

Table 5. Experimental result of scene cohesion test. From top to bottom, the number of neighbors is 1, 3, and 5.

# of neighbor = 1	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Average
Random	0.1322	0.2089	0.2593	0.2466	0.1164	0.0361	0.1667
Doc2vec	0.3897	0.4312	0.5065	0.5031	0.3493	0.1784	0.4380
Averaging Embedding	0.3808	0.4612	0.4961	0.5423	0.3596	0.2394	0.4524
Summarization	0.6247	0.4152	0.3881	0.4679	0.2471	0.0364	0.4507
Metapath2Vec	04269	0.4490	0.5686	0.5258	0.3362	0.3521	**0.4729**
Edge2vec	0.4000	0.4320	0.5509	0.5052	0.3100	0.1549	0.4435
Metapath2Vec - Tag	0.4154	0.4660	0.5137	0.5505	0.3188	0.1549	0.4579
Metapath2Vec - Main Act - Tag	0.3654	0.4272	0.5157	0.5031	0.3712	0.2676	0.4385

# of neighbor = 3	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Average
Random	0.3464	0.5059	0.5936	0.5723	0.3102	0.1044	0.4213
Doc2vec	0.6731	0.7427	0.8255	0.7629	0.6681	0.4085	0.7342
Averaging Embedding	0.5740	0.7692	0.8488	0.8085	0.5972	0.4918	0.7320
Summarization	0.5132	0.7008	0.7676	0.8421	0.6505	0.2333	0.7082
Metapath2Vec	0.7154	0.7524	0.7980	0.7876	0.6681	0.5634	**0.7442**
Edge2vec	0.7154	0.7500	0.7667	0.7649	0.6900	0.4789	0.7276
Metapath2Vec - Tag	0.6769	0.7718	0.7745	0.8309	0.6114	0.4085	0.7348
Metapath2Vec - Main Act - Tag	0.7154	0.7209	0.7843	0.7794	0.6507	0.4788	0.7292

# of neighbor = 5	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Average
Random	0.5078	0.6912	0.7770	0.7572	0.4614	0.1679	0.5981
Doc2vec	0.7923	0.8447	0.9059	0.8784	0.8079	0.5070	0.8416
Averaging Embedding	0.7911	0.8636	0.9298	0.8845	0.7029	0.6230	0.8422
Summarization	0.6520	0.8540	0.8743	0.8975	0.7866	0.4521	0.8151
Metapath2Vec	0.8192	0.8471	0.8471	0.9113	0.8122	0.6338	**0.8577**
Edge2vec	0.8308	0.8883	0.8745	0.8598	0.7991	0.5634	0.8439
Metapath2Vec - Tag	0.8000	0.8592	0.8686	0.9113	0.7686	0.5493	0.8416
Metapath2Vec - Main Act - Tag	0.8346	0.8422	0.8824	0.8927	0.7860	0.5493	0.8444

without tagging. We also observe that the accuracy drops by 0.13 when the Main Act node type is not used. This suggests that the use of Main Act node type enhances the performance to capture the context between scenes.

Table 5 shows our results for the scene cohesion test, setting the number of scene neighbors as 1, 3, and 5, chosen by the Mahalanobis distance. We use the hit rate to measure the performance, the probability that the given scene and one of its neighbors are in the same stage. For the baselines, we add the 'Random' scheme which distributes embeddings randomly. The results show that the Metapath2Vec model using the Main Act node type outperforms baselines in all sectors. We also observe that the embeddings trained by the edge2vec method performs poor, scoring lower than some baselines. This indicates that the Metapath2Vec method is effective in learning the scene embeddings.

The ablation study also shows that the performance

tends to increase when tagging and Main Act node are used, although the difference is not significant. Therefore, we can conclude that the use of Main Act node type helps capture the scene's contextual information.

V. CONCLUSION

In this paper, we present a scene embedding representation using a story graph and Metapath2Vec, a 2-step representation learning algorithm. The story graph is heterogeneous, consisting of Scene, Character, and (Main) Act nodes. The Metapath2Vec algorithm creates a corpus via random walks on the graph. Then, the corpus is used to learn the node embeddings.

For evaluation, we design three tasks and build data from existing datasets to test if the embeddings can represent scenes for understanding a narrative. The results of our experiments show that our embedding method outperforms the baseline models, demonstrating its effectiveness in character identification and scene ordering detection.

As of our knowledge, this work is the first attempt to learn scene embeddings in an unsupervised manner to understand narrative in movies. In the future, we will work on enhancing the embedding method to learn essential narrative elements such as conflict.

ACKNOWLEDGMENTS

This research was partly supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Korea government (MEST) (No. 2019R1A2C1006316) and Institute of Information communications Technology Planning Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2019-0-00421, AI Graduate School Support Program), and the MSIT(Ministry of Science and ICT), Korea.

REFERENCES

[1] Amanda Bertsch, Ashley Oh, Sanika Natu, Swetha Gangu, Alan W Black, and Emma Strubell. Evaluating gender bias transfer from film data. In *Proceedings of the 4th Workshop on Gender Bias in Natural Language Processing (GeBNLP)*, pages 235–243, 2022.

[2] Gayatri Bhat, Avneesh Saluja, Melody Dye, and Jan Florjanczyk. Hierarchical encoders for modeling and interpreting screenplays. In *Proceedings of the Third Workshop on Narrative Understanding*, pages 1–12, Virtual, June 2021. Association for Computational Linguistics.

[3] Larry Brooks. *Story engineering*. Penguin, 2011.

[4] Ming-Chang Chiu, Tiantian Feng, Xiang Ren, and Shrikanth Narayanan. Screenplay quality assessment: Can we predict who gets nominated? In *Proceedings of the First Joint Workshop on Narrative Understanding, Storylines, and Events*, pages 11–16, Online, July 2020. Association for Computational Linguistics.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[6] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 135–144, 2017.

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

[8] Zheng Gao, Gang Fu, Chunping Ouyang, Satoshi Tsutsui, Xiaozhong Liu, Jeremy Yang, Christopher Gessner, Brian Foote, David Wild, Ying Ding, et al. edge2vec: Representation learning using edge semantics for biomedical knowledge discovery. *BMC bioinformatics*, 20(1):1–15, 2019.

[9] Philip Gorinski and Mirella Lapata. Movie script summarization as graph-based scene extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1066–1076, 2015.

[10] Philip Gorinski and Mirella Lapata. What's this movie about? a joint neural network architecture for movie content analysis. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1770–1781, 2018.

[11] Michael Hauge. *Storytelling Made Easy: Persuade and Transform Your Audiences, Buyers, And Clients-Simply, Quickly, and Profitably*. BookBaby, 2017.

[12] Naoya Inoue, Charuta Pethe, Allen Kim, and Steven Skiena. Learning and evaluating character representations in novels. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1008–1019, 2022.

[13] Prashant Jayannavar, Apoorv Agarwal, Melody Ju, and Owen Rambow. Validating literary theories using automatic social network extraction. In *Proceedings of the fourth workshop on computational linguistics for literature*, pages 32–41, 2015.

[14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR, 2014.

[15] Robert McKee. *Story: style, structure, substance, and the principles of screenwriting*. Harper Collins, 1997.

[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[17] Paramita Mirza, Mostafa Abouhamra, and Gerhard Weikum. Alignarr: Aligning narratives on movies. In *The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 427–433. ACL, 2021.

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

[19] Pinelopi Papalampidi, Frank Keller, and Mirella Lapata. Movie plot analysis via turning point identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1707–1717, Hong Kong, China, November 2019. Association for Computational Linguistics.

[20] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[21] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics.

[22] Maarten Sap, Marcella Cindy Prasettio, Ari Holtzman, Hannah Rashkin, and Yejin Choi. Connotation frames of power and agency in modern films. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 2329–2334, 2017.

[23] Alexandra Schofield and Leo Mehr. Gender-distinguishing features in film dialogue. In *Proceedings of the Fifth Workshop on Computational Linguistics for Literature*, pages 32–39, 2016.

[24] Matthew Sims and David Bamman. Measuring information propagation in literary social networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 642–652, Online, November 2020. Association for Computational Linguistics.

[25] Sandhya Singh, Prapti Roy, Nihar Sahoo, Niteesh Mallela, Himanshu Gupta, Pushpak Bhattacharyya, Milind Savagaonkar, Nidhi Sultan, Roshni Ramnani, Anutosh Maitra, and Shubhashis Sengupta. Hollywood identity bias dataset: A context oriented bias analysis of movie dialogues. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 5274–5285, Marseille, France, June 2022. European Language Resources Association.

[26] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Sixth international conference on data mining (ICDM'06)*, pages 613–622. IEEE, 2006.

[27] Hardik Vala, David Jurgens, Andrew Piper, and Derek Ruths. Mr. bennet, his coachman, and the archbishop walk into a bar but only one of them gets recognized: On the difficulty of detecting characters in literary texts. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 769–774, 2015.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[29] Yigeng Zhang, Mahsa Shafaei, Fabio Gonzalez, and Thamar Solorio. From none to severe: Predicting severity in movie scripts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3951–3956, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[30] Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. Dialoglm: Pre-trained model for long dialogue understanding and summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11765–11773, 2022.

SUMMARY OF THIS PAPER

A. *Problem Setup*

Using the unsupervised method, we created a scene embedding that exists in the movie script for analyze movie script.

B. *Novelty*

We propose a novel embedding method to represent scenes, leveraging a story graph and a representation learning algorithm. To the best of our knowledge, this work is the first attempt to learn representations focusing on scenes in a movie script in an unsupervised manner.

C. *Algorithms*

Our method segments a script into scenes and extracts character actions and dialogues to form a story graph. The graph is then used for learning the scene representation leveraging the Metapath2Vec algorithm.

D. *Experiments*

We design three tasks to check if a scene embedding can represent the presence of main characters in the scene, scene ordering, and the scene's contextual information.