HARMONIZING KNOWLEDGE GRAPHS AND DEEP LEARNING: UNRAVELING BUSINESS INSIGHTS FROM MOVIE REVIEWS

Betzalel Moskowitz
MSDS453-DL: Natural Language Processing
November 27, 2023

Introduction

This paper unfolds a dual exploration into knowledge graph (KG) construction and deep learning experiments, illuminating their collaborative potential for extracting meaningful insights from movie reviews. Beyond their intrinsic value, these endeavors hold substantial business relevance in enhancing information extraction processes. The meticulous construction of an ontology and the subsequent development of a knowledge graph lay the groundwork, emphasizing the modeling of entities and relationships. As this knowledge graph undergoes refinement, techniques like pronoun resolution and manual equivalent term resolution contribute to its evolution into a coherent and enriched structure. Simultaneously, the paper introduces deep learning methodologies for sentiment and genre classification tasks, scrutinizing the roles of dense layer sizes, layer numbers, and anti-overfitting methods on model performance. A comparative analysis to previous work (Moskowitz, 2023) reveals the nuanced performance of deep learning versus classical machine learning approaches in natural language processing (NLP) tasks, underscoring the business value of leveraging these technologies for enhanced information extraction from textual data, particularly in the domain of movie reviews.

Knowledge Graph Experiments

Methods

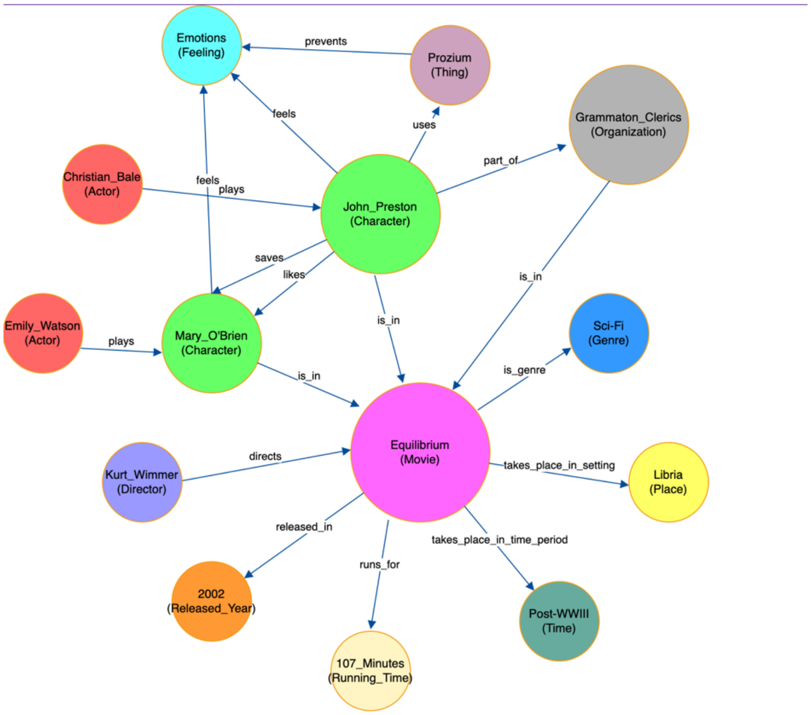
Dataset

The corpus for these experiments is comprised of ten movie reviews about the movie "*Equilibrium*". These movie reviews were sourced from *Rotten Tomatoes* and *IMDB* and included five "positive" and five "negative" reviews.

First Pass Ontology

An ontology was constructed using prior knowledge about the movie *Equilibrium*. The ontology focused on modelling simple instances – it included entities such as actors, characters, organizations, things (items, feelings), movies, and movie attributes (genres, run times, time setting, place setting). Since most of these reviews were truncated at 500 tokens where most reviews had barely finished summarizing the movie, it was deemed that it would be best to use the reviews to model what happened in the movie. In addition to critical text being limited in the movie reviews, these types of "something-is-about-something" ontologies are difficult and messy to model. For this reason, these ontologies were deemed out of scope for this experiment. *Figure 1.* displays the first pass ontology developed for the movie *Equilibrium.* It is available in a larger format in *Appendix A1*. The ontology will be compared later to the knowledge graph as a point of reference – the knowledge graph will be stronger and more complete the more it seems to resemble the ontology.

Figure 1. First Pass Ontology for Equilibrium



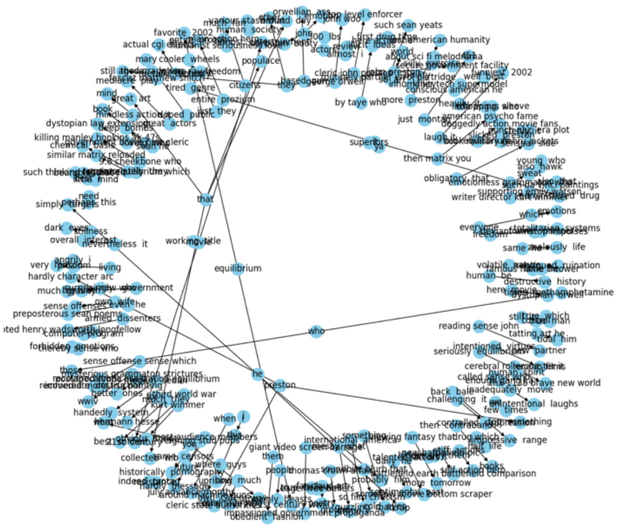
These types of ontologies are useful because they allow human beings to conceptualize and standardize the types of entities and relationships present in subsequent knowledge graphs.

Data Preparation

Constructing a knowledge graph requires triplets to be extracted from sentences in the documents. Triplets are tuples of three elements – an entity (often called the subject), a relation (often the verb or predicate of the sentence), and a second entity (often the direct object of the sentence). For instance, the triplet ("Bob", "walks", "dog") would be extracted from the sentence "Bob walks his dog in the morning and evening". One might notice that extracting this triplet requires knowledge of the entities in the sentence as well as grammatical structure. If a machine is to extract the terms correctly, it needs to know that "Bob" is the subject, "walks" is the predicate/verb, and "dog" is the object. To gain this knowledge, the documents were split into sentences using spaCy's NLP pipeline (spaCy), and every word in each sentence was tagged using the same pipeline to determine the part of speech (PoS). The part of speech for each token helped to identify entities and relations. It also enables and recognition and resolution of instances where entities and relations consisted of compound words. Using this new knowledge of tagging parts of speech in each sentence, triplets were extracted. The first entity of each triplet was referred to as the "source", the relationship as the "edge", and the second entity as the "target". All triplets were converted to lower case, while any triplets that contained an empty string for any of its elements were removed from the data.

Once the triplets were extracted from the raw text, an initial knowledge graph was constructed from the triplets, and saved in a data frame with three columns ("source", "target", "edge"). Each row in the data frame represented a triplet. The knowledge graph could then be visualized using Python's networkX module (Hagberg et al., 2008) as seen in *Figure 2*. The KG is also available in a larger format in *Appendix A2*.

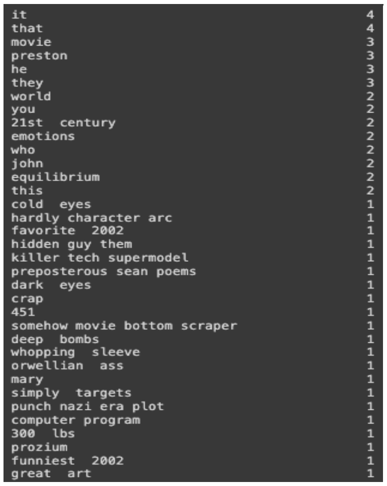
Figure 2. Initial Knowledge Graph



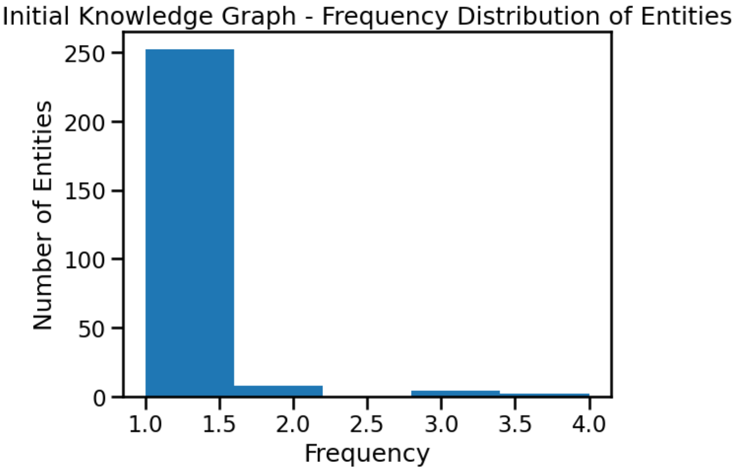
Knowledge Graph Experiments

Unfortunately, the initial knowledge graph was messy and seemed to have a substantial number of separate nodes that should have been represented as the same entity. In addition, there were many pronoun entities that made it unclear which entity was referenced. This is evident by looking at a frequency count of entities (*Figure 3*), where entity frequency counts appear sparse, which is less useful when representing knowledge about entities. In addition, *Figure 3* also shows multiple entities that require resolution – "john" and "preston" are considered equivalent classes – they both refer to the main character of the movie "John Preston" and should be merged into the same entity to collapse frequency counts. The initial knowledge graph contained 266 different entities. The goal in the experiments will be to reduce this number to build a more dense and complete KG.

Figure 3. Frequency Counts and Distribution of Entities in Initial Knowledge Graph



```
it                                              4
that                                            4
movie                                           3
preston                                         3
he                                              3
they                                            3
world                                           2
you                                             2
21st  century                                   2
emotions                                        2
who                                             2
john                                            2
equilibrium                                     2
this                                            2
cold  eyes                                      1
hardly character arc                            1
favorite  2002                                  1
hidden guy them                                 1
killer tech supermodel                          1
preposterous sean poems                         1
dark  eyes                                      1
crap                                            1
451                                             1
somehow movie bottom scraper                    1
deep  bombs                                     1
whopping  sleeve                                1
orwellian  ass                                  1
mary                                            1
simply  targets                                 1
punch nazi era plot                             1
computer program                                1
300  lbs                                        1
prozium                                         1
funniest  2002                                  1
great  art                                      1
```



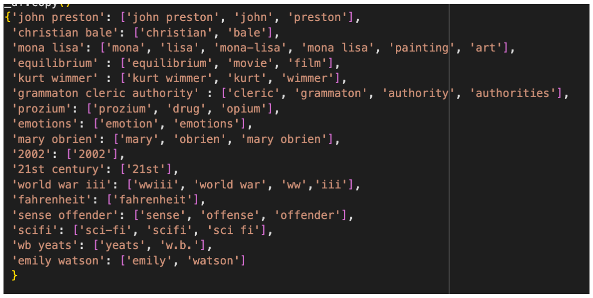
Pronoun Resolution using OpenAI Chat Completion API

One way to collapse the frequency counts of the equivalent classes is to convert each pronoun in each sentence into the noun or proper noun that it is referencing. To do this, OpenAI's Chat Completion API (OpenAI) was used to return the same text with all pronouns converted into the entities they reference. The triplets could then be re-extracted using the methods outlined in the previous step to build a new knowledge graph.

Manual Equivalent Term Resolution

Another important way to collapse frequency counts in equivalent classes is to use entity recognition to normalize the equivalent classes into a single term. This process is extremely time consuming as it requires an iterative approach to identify terms that should be normalized and finding the equivalent terms to which the resolution should be applied. Therefore, the experiments focused on normalizing terms that are important to the movie. The equivalent classes and their normalized terms were defined as they are displayed in *Figure 4*. After the pronoun resolution step, any equivalent terms were replaced with the manually determined normalized term. The knowledge graph was then reconstructed.

Figure 4. Manually Defined Normalized Terms and Equivalent Classes

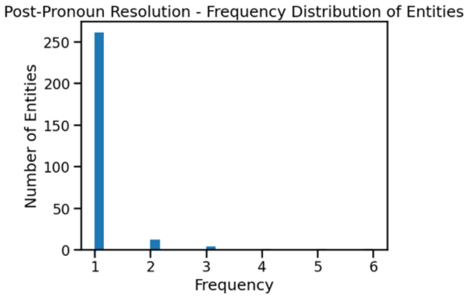


```
{'john preston': ['john preston', 'john', 'preston'],
 'christian bale': ['christian', 'bale'],
 'mona lisa': ['mona', 'lisa', 'mona-lisa', 'mona lisa', 'painting', 'art'],
 'equilibrium' : ['equilibrium', 'movie', 'film'],
 'kurt wimmer' : ['kurt wimmer', 'kurt', 'wimmer'],
 'grammaton cleric authority' : ['cleric', 'grammaton', 'authority', 'authorities'],
 'prozium': ['prozium', 'drug', 'opium'],
 'emotions': ['emotion', 'emotions'],
 'mary obrien': ['mary', 'obrien', 'mary obrien'],
 '2002': ['2002'],
 '21st century': ['21st'],
 'world war iii': ['wwiii', 'world war', 'ww','iii'],
 'fahrenheit': ['fahrenheit'],
 'sense offender': ['sense', 'offense', 'offender'],
 'scifi': ['sci-fi', 'scifi', 'sci fi'],
 'wb yeats': ['yeats', 'w.b.'],
 'emily watson': ['emily', 'watson']
 }
```

Results

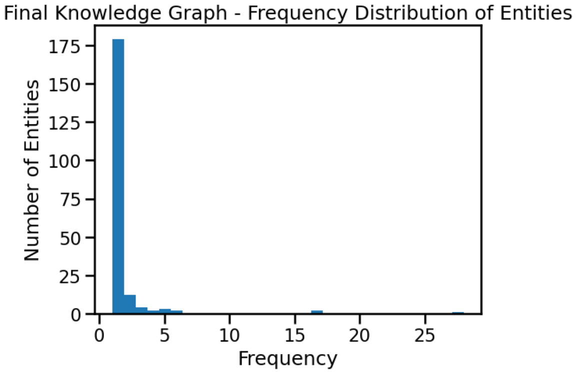
Utilizing OpenAI's ChatCompletion API to convert pronouns to entities actually increased the number of terms to 280. This was likely due to the additional descriptive entities that the API used to replace the pronouns. Although taking this step seemingly took a step backward, its role in reaching the final knowledge graph proved pivotal – without converting the pronouns it would have been difficult to perform entity resolution on these terms. This would have prevented the further resolution of equivalent terms. *Figure 5* displays a frequency count of entities at each frequency level of occurrence.

Figure 5. Frequency Distribution of Entity Occurrences Post-Pronoun Conversion



Performing the manual term resolution proved to be a very manual and iterative process but nonetheless proved successful in further reducing the frequency count to 205. The final frequency distribution of entity occurrences can be seen in *Figure 6*. After this point it became difficult to keep finding terms to resolve, marking the end of the effort to resolve terms manually. Despite there still being about 200 terms that only entities in the final knowledge graph, the frequency count has been collapsed sufficiently enough to provide a cleaner and denser knowledge graph.

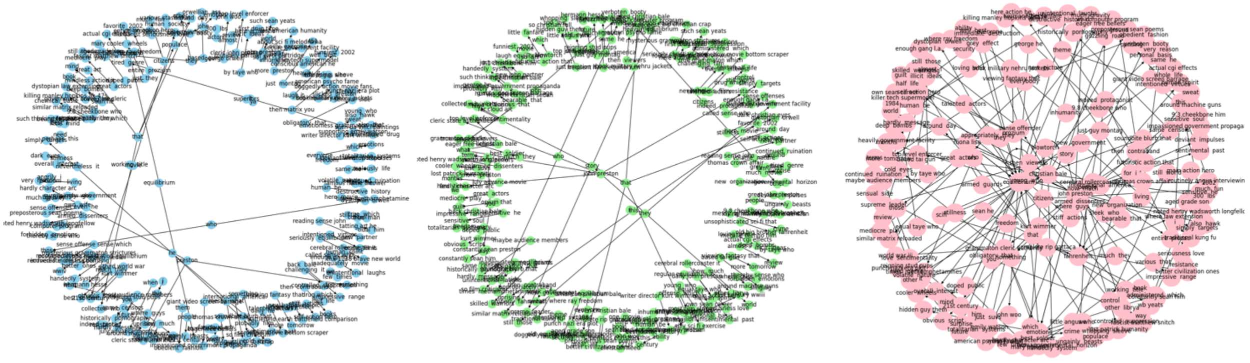
Figure 6. Frequency Distribution of Entity for Final Knowledge Graph



Analysis

Overall, using the OpenAI API to replace the pronouns in the sentences helped to disambiguate the terms, allowing for more entities to be resolved once the equivalent terms were merged into single entities. *Figure 7* displays the three knowledge graphs side by side – the initial KG (left), the KG following pronoun resolution (center), and the final KG following the entity resolution (right). One can see how the knowledge graphs grow denser after each step – this makes the knowledge graph more complete and allows different terms that represent the same entity to be represented as one entity in the graph. It also ensures that the relationships previously tied to multiple equivalent classes of the same entity can be modeled as relationships to the single underlying entity, producing a richer set of connections for each true entity.

Figure 7. A Comparison of Knowledge Graphs (From left to right: Initial KG, Pronouns Replaced KG, Term-Resolved Final KG)



Since many of these entities can be difficult to see in these KGs, several entities of interest were extracted from the graph and visualized to show how the steps taken in these experiments helped to perform entity resolution and provide a much richer knowledge representation of some of the concepts in the movie. Some of the entity subsets of the KGs seem to resemble some of the entities and relationships outlined in the first pass ontology (see *Appendix A5*). For instance, the connections between the movie "*Equilibrium*" to some of its targets ("scifi" and "2002") each directly correspond to the genre and release date of the movie defined in the ontology. Similarly, the entity "John Preston" seems to have connections to "emotions" and "grammaton cleric authority" just as the ontology does. This seems to suggest that the final knowledge graph represented the knowledge about the movie "*Equilibrium*" reasonably well.

However, there are still some fundamental challenges that were uncovered when building these knowledge graphs. First, not all of the entity relationships are useful – with movie reviews, many of these relationships and entity pairs are based on opinions and not necessarily about facts. These types of concepts are a lot more difficult to model and did not match the ontology set out at the beginning of the experiment.

Second, collapsing frequency counts by resolving entities to a single standardized term was a manual process that required an iterative human-in-the-loop process. This is not a particularly scalable or easily updatable process. Performing entity resolution in the fashion described above can also result in valuable information loss. One of the original extracted triplets was ("equilibrium", "takes", "21st century wwiii"). When going through the process of resolving equivalent classes "wwiii" was an already existing entity, so the entity was resolved to "wwiii".

However, doing so meant that the knowledge that Equilibrium takes place in the 21st century was lost. It would have been advantageous to split this triplet into two – (“equilibrium”, “takes”, “wwiii”) and (“equilibrium”, “takes”, “21st century”). Similarly, there were other entities in sentences that would be useful to include in a knowledge graph but were not extracted because they were not subjects, predicates, or direct objects in the sentences, but rather in indirect objects or prepositions.

It’s also possible that triples may be derived from logical deduction from a sentence. Going back to the example “Bob walks his dog in the morning and evening”, the methods used in this experiment would only extract (“Bob”, “walks”, “dog”). However, the sentence provides a lot more information that could be inferred, and the knowledge could be modeled in triplets. In the previous example, one could also infer (“Bob”, “owns”, “dog”) (“Bob”, “walks during”, “morning”), (“Bob”, “walks during”, “night”), (“dog”, “walks during”, “morning”), and finally (“dog”, “walks during”, “night”). This type of knowledge cannot be extracted into triples using part of speech tagging directly. It requires logical deduction. Future work might experiment with using OpenAI’s Chat Completion API to pull triplets from sentences to capture more of this type of knowledge. This will enable the construction of an even richer and more complete knowledge graph.

Deep Learning Experiments

Methods

Dataset

The corpus for these experiments is comprised of twenty different movie reviews of movies across four different genres (“Action”, “Comedy”, “Horror”, “Sci-Fi”). These movie reviews were sourced from *Rotten Tomatoes* and *IMDB.* For each of the twenty movies, ten reviews were selected – five contained “positive” sentiment and the other five contained “negative” sentiment. This resulted in a corpus of 200 movie reviews which are referred to in this report as the *documents*.

Data Preparation

Each document was truncated at 500 tokens. The text from each document was normalized by removing punctuation, converting to lowercase, removing any special HTML tags, and removing special characters and digits. The documents were then tokenized, and stop words were removed in an attempt to remove noisy words that provided little semantic value to the corpus. The stop words consisted of some of the most common English words as well as several custom stop words ('movies', 'movie', 'film', 'films', 'scene') that were prevalent across all documents and seemed to add noise that made NLP tasks more difficult. All tokens were lemmatized using NLTK’s WordNetLemmatizer (NLTK). The dataset was then split into 80% train, 10% validation, and 10% test sets.

Vectorization

With the data preparation complete, the documents were vectorized into numerical representations to prepare them for Deep Learning. The documents were vectorized using Keras's TextVectorization encoder with a max vocabulary size of 5,000 tokens.

Model Architectures

The experiments were implemented in Keras and TensorFlow and focused on varying deep learning architectures using Bidirectional Long-Short Term Memory (LSTM) layers. Each model began with a Keras TextVectorization layer that fed into an embedding layer with 64 units. This was followed by a bi-directional LSTM layer with 64 units and a 30% dropout, followed by an additional bi-directional LSTM layer with 32 units and a 30% dropout. The fully connected layers of the model were manipulated to examine the effect that the number of hidden nodes and number of layers had on model performance:

The experiments examined the use of hidden units in the dense layer(s) of size 32 and 64. For each of these layer sizes, an experiment was conducted using one layer, one layer followed by a 10% dropout, two layers, and two layers with each followed by a 10% dropout as well. All of these layers used a ReLU activation function. *Figure 8* displays a table to visualize the experiments.

Figure 8. Overview of Fully Connected Layer(s) Architectures

Number of Hidden Units	Layers			
32	1 Hidden Layer	1 Hidden Layer followed by 10% Dropout Layer	2 Hidden Layers	2 Hidden Layers each followed by 10% Dropout Layer
64	1 Hidden Layer	1 Hidden Layer followed by 10% Dropout Layer	2 Hidden Layers	2 Hidden Layers each followed by 10% Dropout Layer

The same architecture was used for both sentiment classification and genre classification in movie reviews up until the output layer.

The sentiment classification task is a binary classification problem, so binary cross-entropy was used as the loss function and the final dense layer was constructed with one neuron activated by a sigmoid function. This neuron was used to get the probability that a review had a positive review. If the probability was below 0.5 a review would be deemed a negative review.

However, because the genre classification was multi-class classification problem involving four classes ("Action", "Comedy", "Horror", "Sci-Fi") sparse categorical cross entropy was selected for the loss function and a final output layer was constructed with four neurons, each using a SoftMax activation function. Each of the outputs from the four neurons represented the probability that a movie review belonged to the neuron's corresponding genre and were used to determine the most likely genre for the movie review.

All models used the Adam optimizer and were trained for up to 200 epochs with an early stopping patience of two epochs in an effort to reduce overfitting.

Following the deep learning experiments, performance metrics were computed, including precision, recall, F1 score, and accuracy achieved on the test set. Training accuracy was also computed to examine the impact of overfitting. These metrics were compiled into tables and confusion matrices were also constructed after evaluating the model on the test sets. These diagrams are provided in the *Appendices A6-A9*.

Results

Sentiment Classification

As seen in *Appendix A7*, the models with 32 hidden units per dense layer seemed to overall outperform those with 64 hidden units per dense layer. The best model for this task was the single 32-unit layer without a dropout layer. This model achieved a perfect training accuracy, a test accuracy of 72%, and the highest macro average recall at 67%. The performance metrics of this model are available below in *Figure 9*. The single 64-unit layer without a dropout layer model achieved the highest test precision at 69%. The single dense layer models without dropout layers achieved higher performance for this task than those with dropout layers. However, the double dense layer models with dropout outperformed those without dropout layers.

Figure 9. Performance Metrics of Best Genre Classification Model

	Test Precision	**Test Recall**	**Test F1-score**	**Support**
Negative	0.77	0.83	0.80	12
Positive	0.60	0.50	0.55	6
Test Accuracy	0.72			18
Test Macro Avg	0.68	0.67	0.67	18
Test Weighted Avg	0.71	0.72	0.72	18
Training Accuracy	1.0000			

Genre Classification

As seen in *Appendix A9*, the best performing model had one dense layer with 32 hidden units and a 10% dropout layer. This model led all other models in practically every metric, achieving a perfect training accuracy, 83% test accuracy, 89% test precision, 88% test recall, and 86% test F1-score. Its metrics are available below in *Figure 10*, where any highest performance across all models are highlighted.

Figure 10. Performance Metrics of Best Genre Classification Model

	Test Precision	Test Recall	Test F1-score	Support
Action	1.00	1.00	1.00	2
Comedy	1.00	0.75	0.86	4
Horror	1.00	0.75	0.86	8
Sci-Fi	0.57	1.00	0.73	4
Test Accuracy	0.83			18
Test Macro Avg	0.89	0.88	0.86	18
Test Weighted Avg	0.90	0.83	0.84	18
Training Accuracy	1.0000			

Overall, it seemed as if the models with dense layers of size 32 yielded superior results than the 64-unit ones. The dropout layers appeared to boost performance in the single hidden layer models but seemed to worsen performance in the double hidden layer model.

Analysis

It is not necessarily fair to benchmark the performance of the deep learning against the SVM, decision tree, and random forest models since the models were not evaluated on the same dataset. This was a fundamental short coming that would have been corrected if more time allowed for experimentation but nonetheless makes a comparative analysis speculative at best.

For the sentiment classification test, the best LSTM model achieved a test accuracy of 72%, whereas the best non-Deep Learning model achieved a test accuracy of 61%. If the edge in performance for the LSTM model still remains when evaluated on the same test set, it would not be surprising. This problem was already made harder due to the movie reviews being truncated at 500 words, often removing important text from the reviews that is more likely to contain praise or critique than the beginning of the review. With this critical information omitted from the data, it becomes critical to rely on phrases rather than individual words to determine sentiment. For instance, if a review had a sentence saying “the movie was not good”, a machine learning model that processes one word at a time may incorrectly predict positive sentiment since the word good is present, but it may struggle to learn that the review expressed negative sentiment since it may ignore the role of “not” and focus on the token “good” which is often associated with positive reviews. LSTM helps to boost performance in these cases by better modeling the sequential aspect of the language. With this in mind, the deep learning model may use its knowledge of sequential language processing to recognize sequential patterns that may correlate with negative versus positive sentiment in particular. This may help to explain why the performance of the deep learning architectures was superior for this task.

For the genre classification task on the other hand, one previously trained TF-IDF SVM model achieved a perfect score on the test dataset for genre classification, and several other Doc2Vec SVM models achieved near-perfect performance. This is not surprising – deep learning architectures often outperform classic machine learning algorithms like random forest and SVM, but only when the dataset is large enough (thousands of training samples). In these experiments, the entire corpus consists of 200 documents, and the number of training samples is even lower once the dataset is split into train, validation, and test sets. Thus, it is likely that the reduced performance among deep learning models suffered from the lack of available data. This can also help to explain why the smaller deep learning models (32 hidden units per dense layer) seemed to fare better than those with 64 hidden units per dense layer – the larger models introduce complexity that may be necessary to model larger datasets but may lead to overfitting in smaller datasets. Additionally, genre classification does not necessarily require sophisticated sequential processing of text – it is likely that a human could determine the genre of a movie review just by looking at a list of keywords appearing in the document.

Conclusion

In conclusion, the paper presented a comprehensive exploration of both knowledge graph construction and deep learning experiments using movie reviews as a corpus. The knowledge graph construction process involved the development of an ontology and the subsequent creation of a knowledge graph, which was refined through steps such as pronoun resolution and manual equivalent term resolution. The resulting knowledge graph showcased the evolution of representation from an initial messy state to a more refined and denser structure.

The knowledge graph experiments demonstrated the effectiveness of techniques such as pronoun resolution using OpenAI's ChatCompletion API and manual equivalent term resolution in improving the quality and coherence of the graph. The final knowledge graph, despite its limitations and challenges, appeared to capture essential relationships and entities related to the movie "Equilibrium," aligning reasonably well with the initial ontology.

However, challenges and limitations were identified in both knowledge graph construction and deep learning experiments. The manual nature of equivalent term resolution in knowledge graphs posed scalability issues, and certain relationships in movie reviews, driven by opinions rather than facts, proved challenging to model accurately

The deep learning experiments uncovered several key insights. Deep learning architectures such as LSTMs excel at modeling sequential data such as text, which may be useful in tasks where the order of words matter. However, deep learning approaches require a large amount of data to truly shine, and in instances when the order of words matter less and access to data is limited, classic machine learning algorithms may perform better than deep learning approaches. With this in mind, it is best for NLP practitioners to gain a good understanding of their datasets and the requirements for the task. Running a large number of experiments is also key – this must be done to gain an understanding of how design decision helps or harms performance and whether or not alternative approaches may pose a better alternative. In this experiment, running multiple experiments highlighted that a larger number of hidden layers and hidden units per layer seemed to harm performance rather than improving it.

References

Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, “Exploring network structure, dynamics, and function using NetworkX”, in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008

Moskowitz, B. Y. (2023). (rep.). *NLP CLUSTERING, CLASSIFICATION, AND TOPIC MODELING OF MOVIE REVIEWS* (pp. 17–19). Washington, District of Columbia.

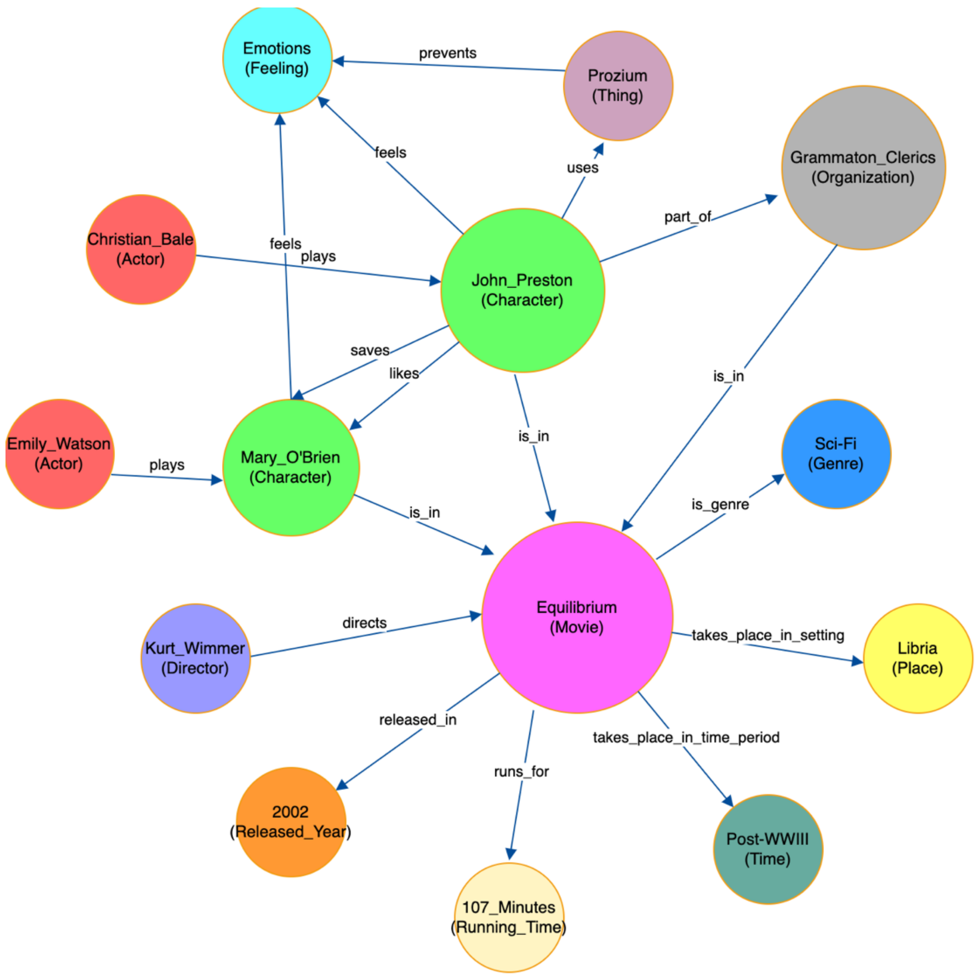
nltk.stem.wordnet module. NLTK. (n.d.). https://www.nltk.org/api/nltk.stem.wordnet.html

OpenAI. (n.d.). *API reference - openai API*. API Reference - OpenAI API. https://platform.openai.com/docs/api-reference

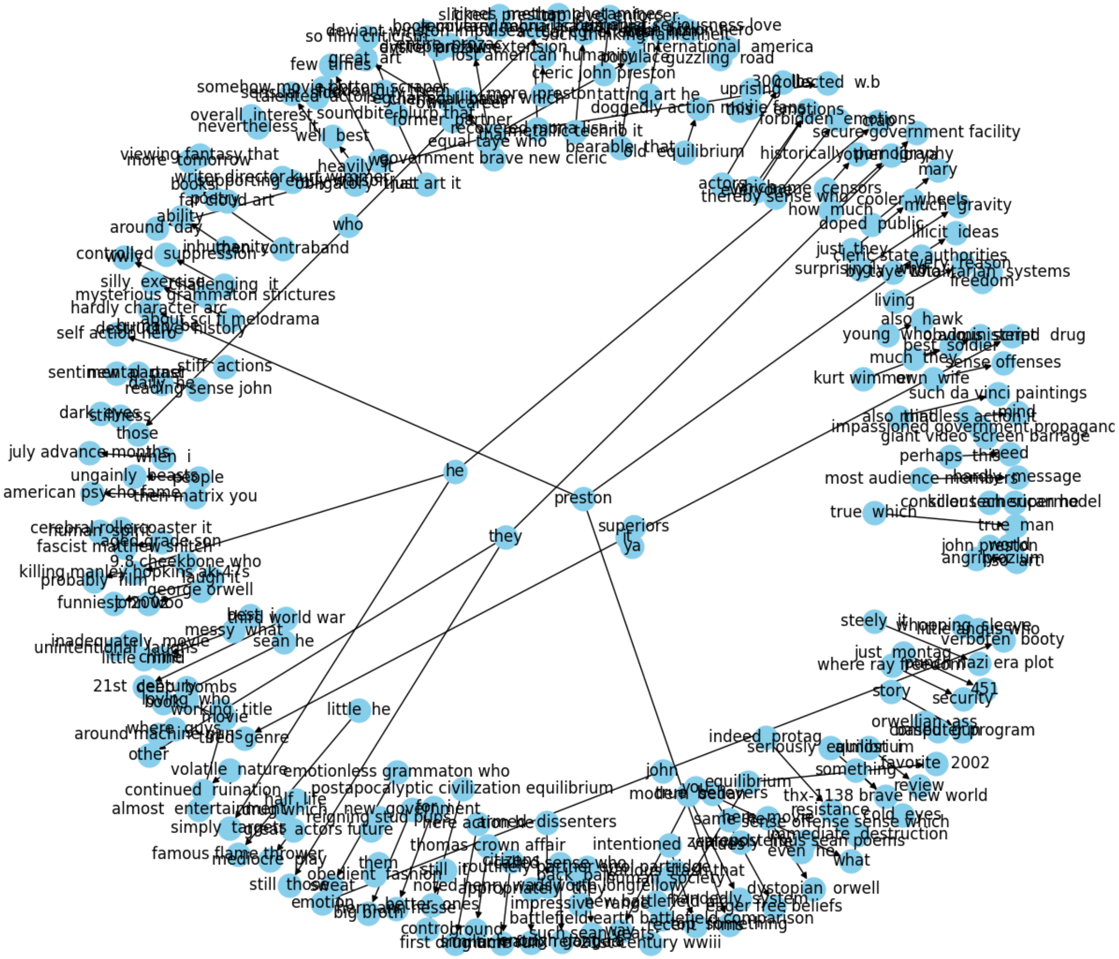
spaCy. (n.d.). *Language Processing Pipelines · spacy usage documentation*. Language Processing Pipelines. https://spacy.io/usage/processing-pipelines

Appendix

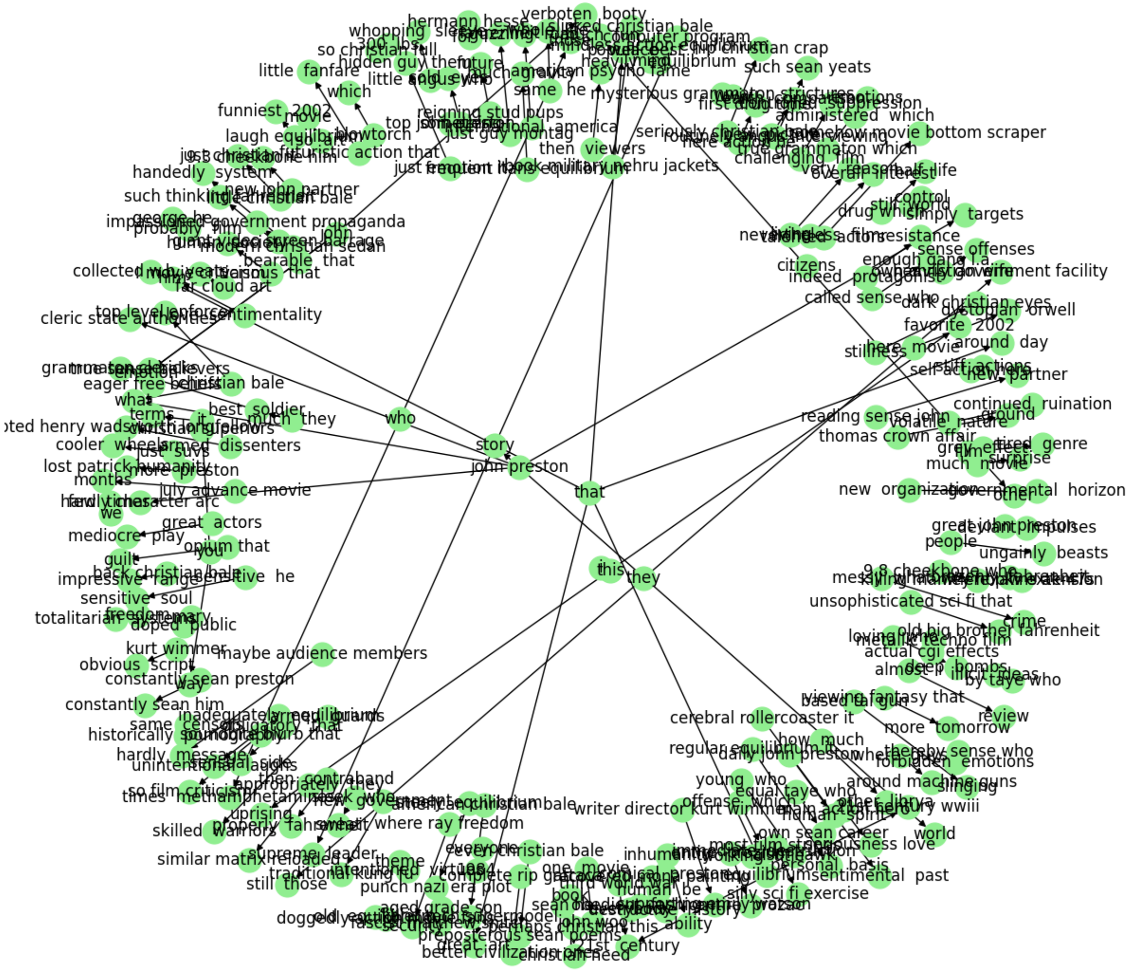
A1. Initial Ontology



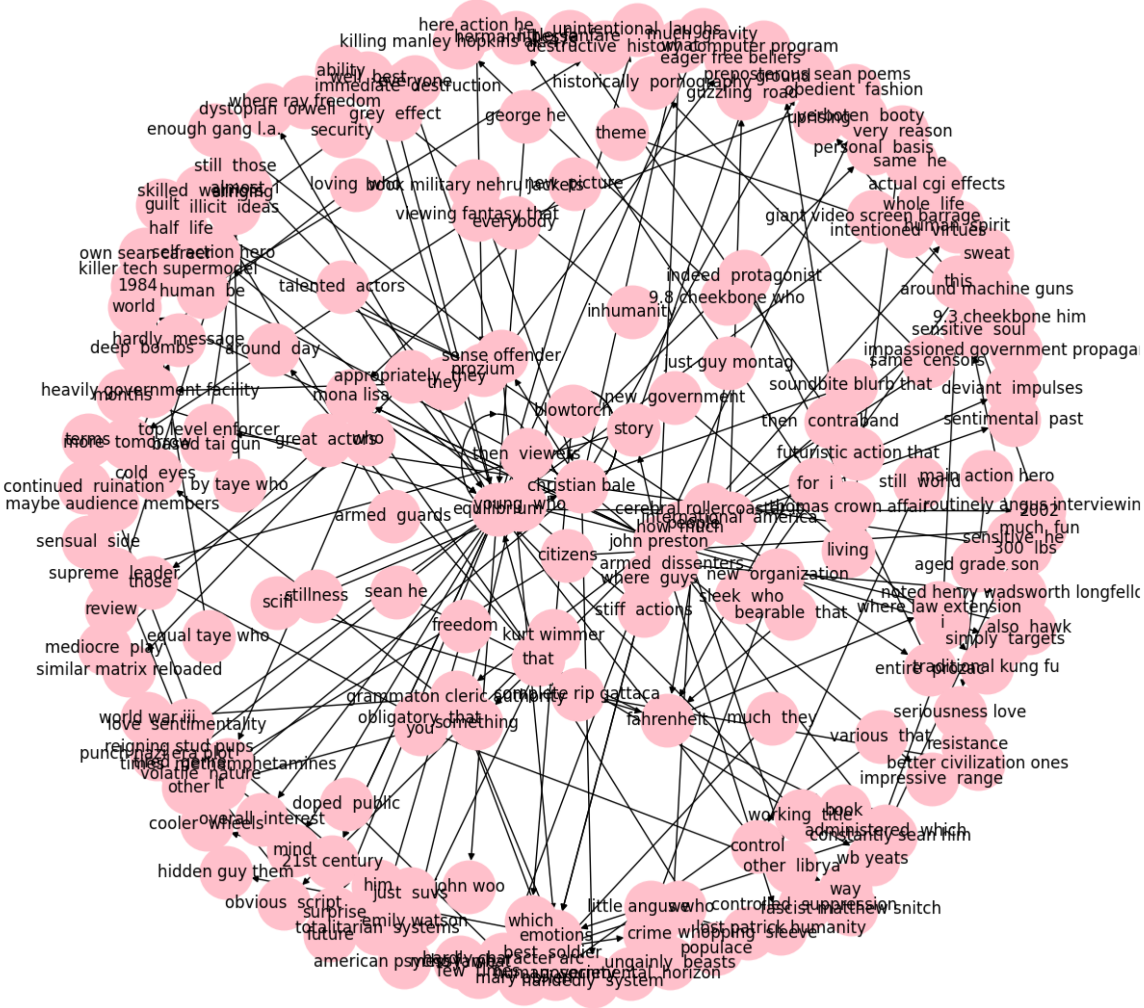
A2. Initial Knowledge Graph



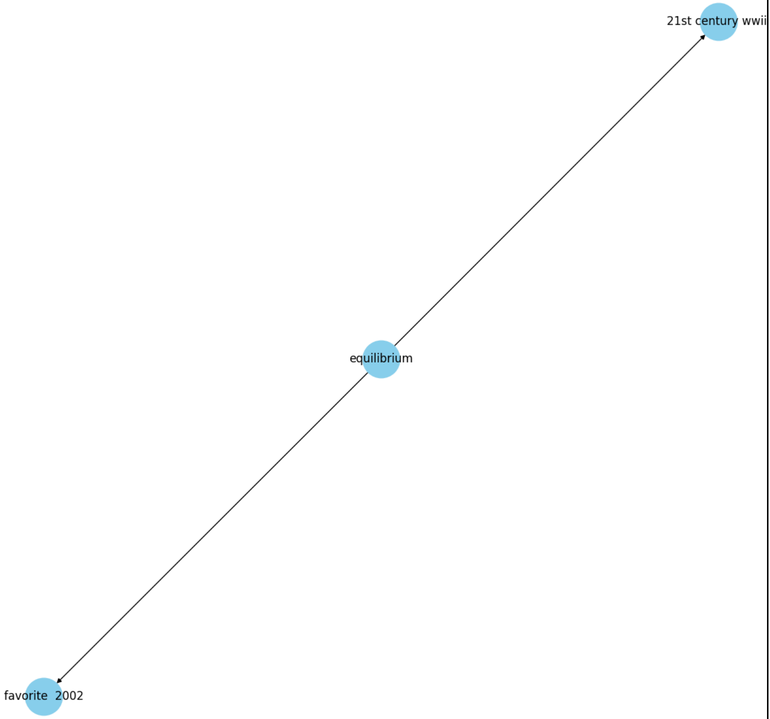
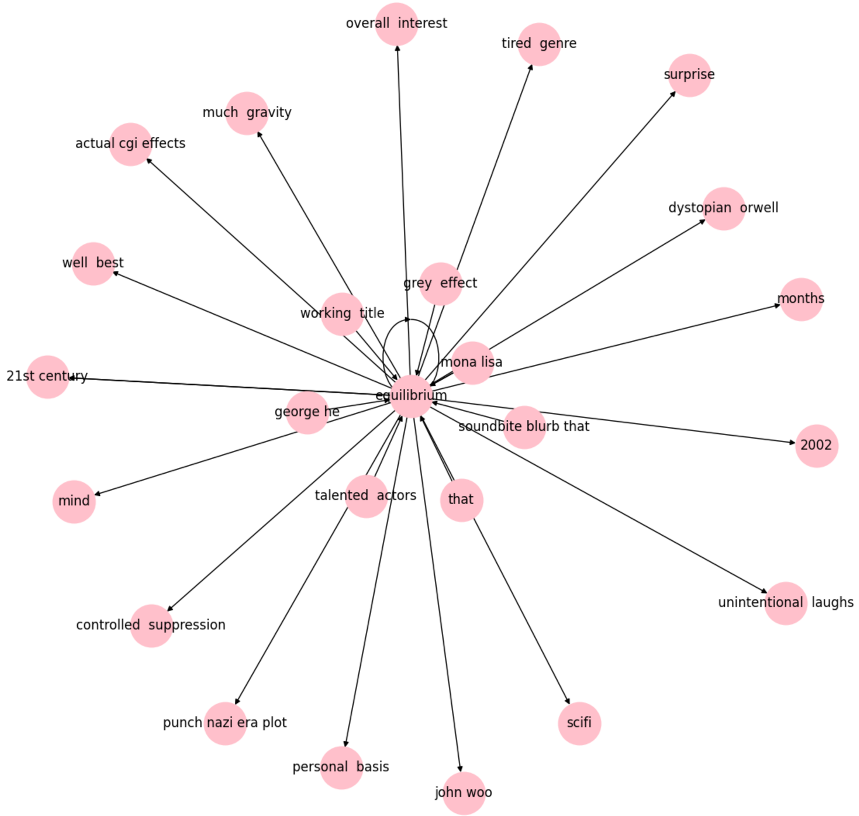
A3. Pronouns Replaced Knowledge Graph

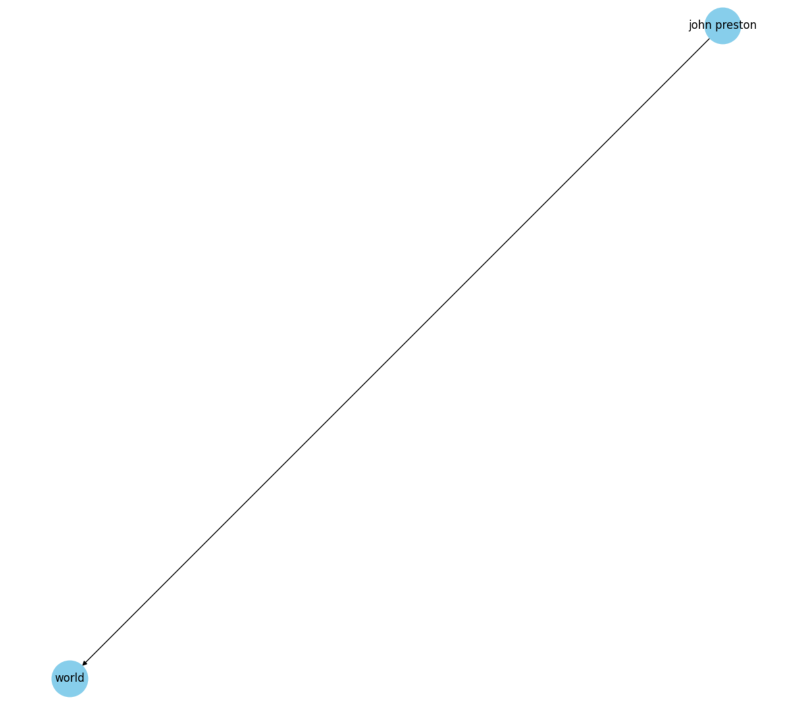
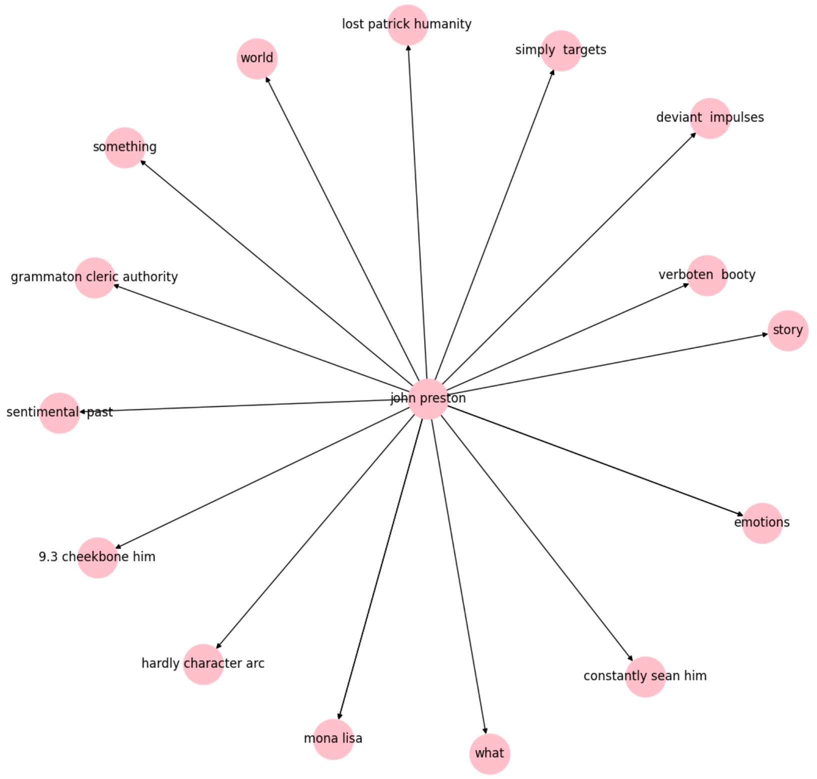


A4. Final Knowledge Graph (Manual Entity Resolution)

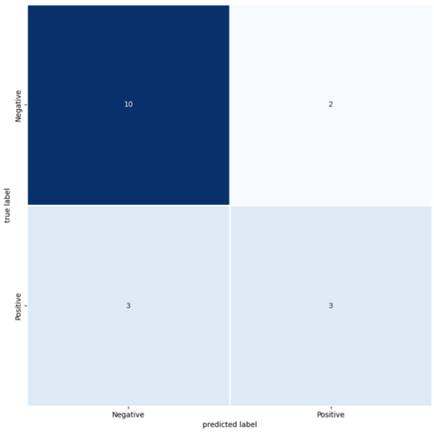
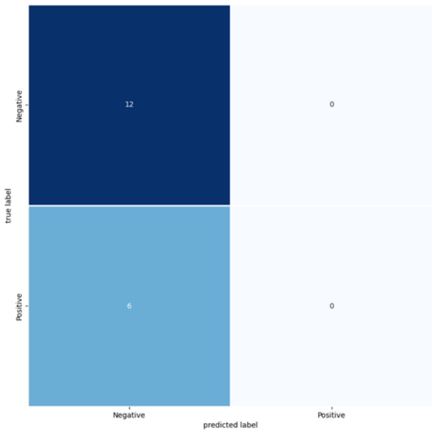
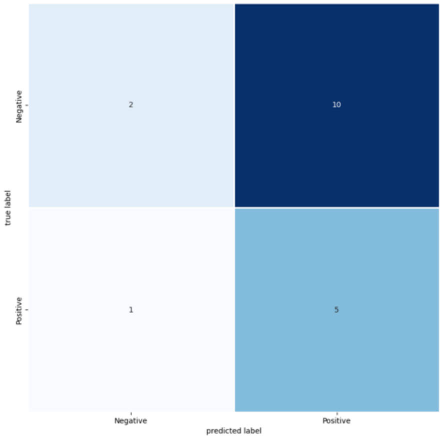
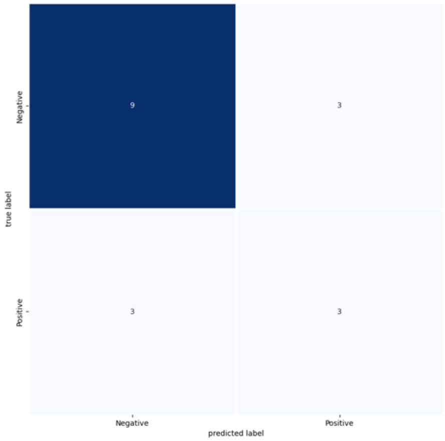
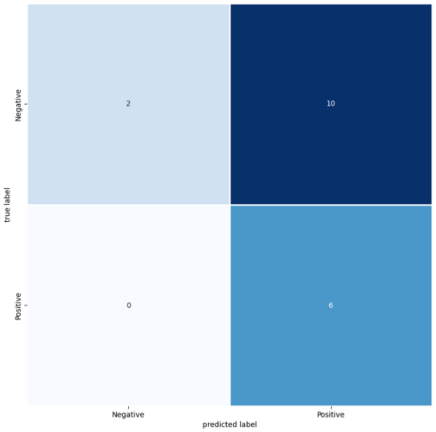
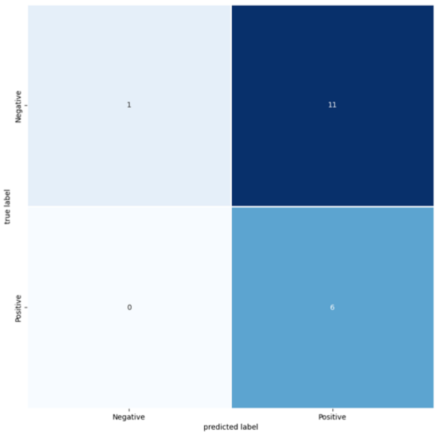
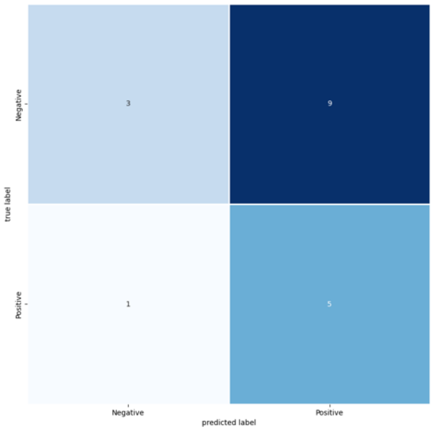
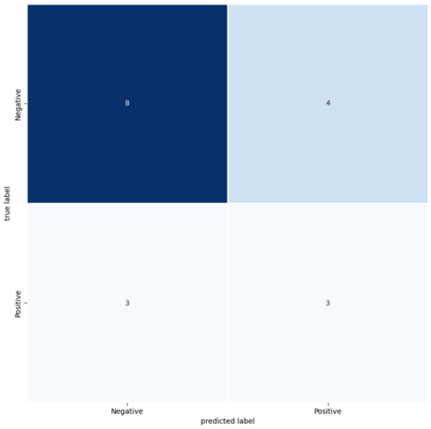


A5. KG Comparison of Entities of Interest

Term	Initial KG	Final KG
“equilibrium”		 overall interest tired genre surprise much gravity actual cgi effects dystopian orwell well best grey effect working title months mona lisa 21st century equilibrium george he soundbite blurb that 2002 mind talented actors that unintentional laughs controlled suppression punch nazi era plot scifi personal basis john woo

“john preston”		 lost patrick humanity world simply targets deviant impulses something grammaton cleric authority verboten booty story john preston sentimental past emotions 9.3 cheekbone him hardly character arc constantly sean him mona lisa what

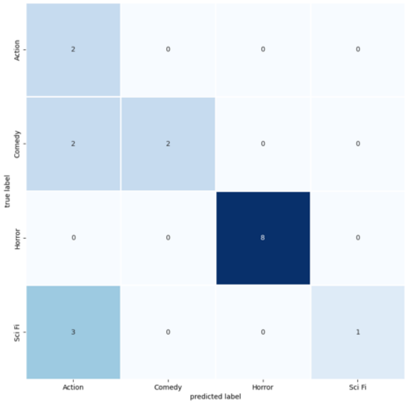
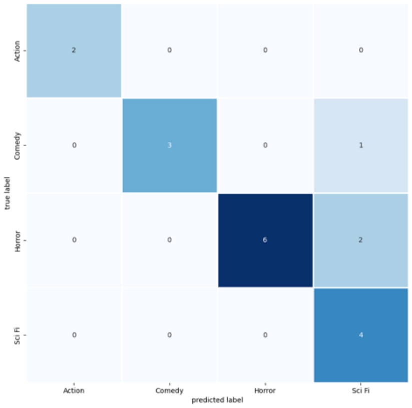
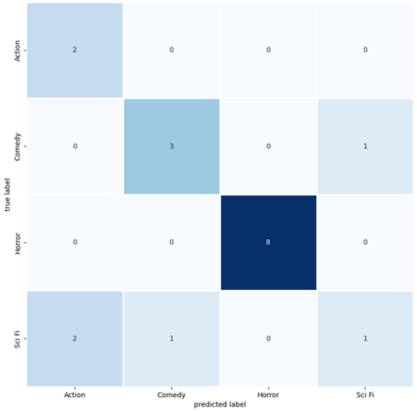
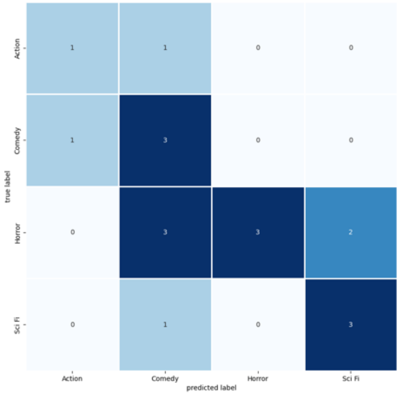
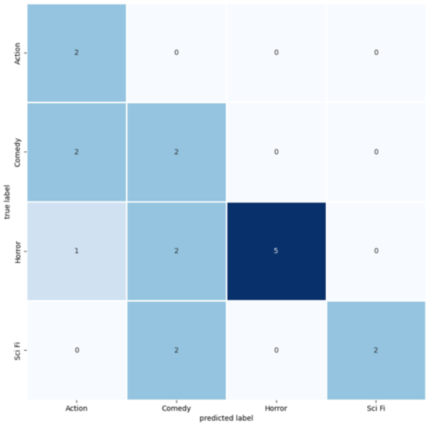
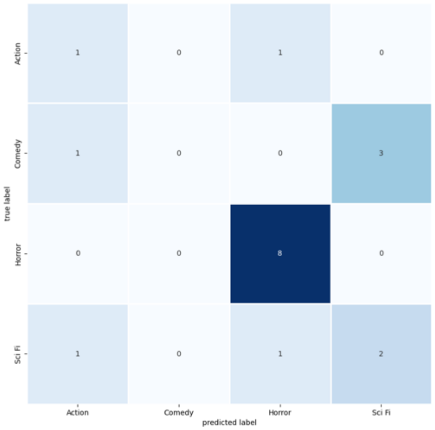
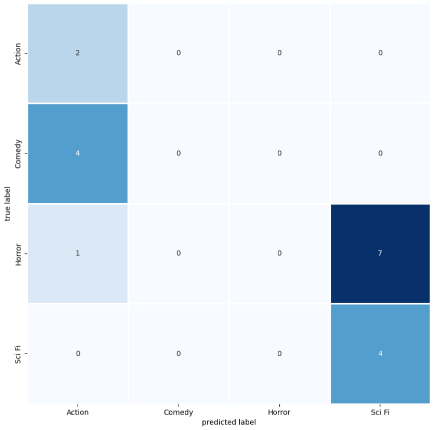
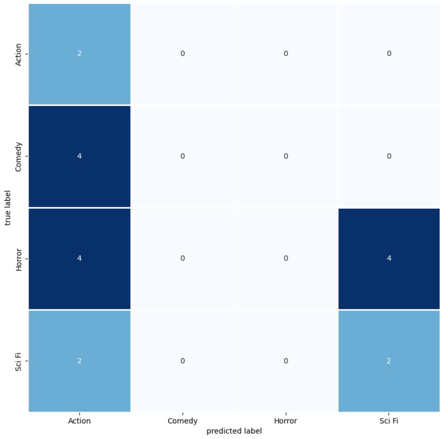
A6. Sentiment Analysis Confusion Matrices – Deep Learning Approaches

Hidden Nodes Per Layer	Number of Layers			
	1 Hidden Layer	1 Hidden Layer with Dropout	2 Hidden Layers	2 Hidden Layers with Dropout
32				
64				

A7. Review Sentiment Classification Performance Metrics – Deep Learning Approaches

Hidden Nodes Per Layer	Number of Layers		Test Precision	Test Recall	Test F1-score	Support
32	1 Hidden Layers	Negative	0.77	0.83	0.80	12
		Positive	0.60	0.50	0.55	6
		Test Accuracy	0.72			18
		Test Macro Avg	0.68	0.67	0.67	18
		Test Weighted Avg	0.71	0.72	0.72	18
		Training Accuracy	1.0000			
	1 Hidden Layer with Dropout	Negative	0.67	1.00	0.80	12
		Positive	0.00	0.00	0.00	6
		Test Accuracy	0.67			18
		Test Macro Avg	0.33	0.50	0.40	18
		Test Weighted Avg	0.44	0.67	0.53	18
		Training Accuracy	0.4969			
	2 Hidden Layers	Negative	0.67	0.17	0.27	12
		Positive	0.33	0.83	0.48	6
		Test Accuracy	0.39			18
		Test Macro Avg	0.50	0.50	0.37	18
		Test Weighted Avg	0.56	0.39	0.34	18
		Training Accuracy	0.9686			
	2 Hidden Layers with Dropout	Negative	0.75	0.75	0.75	12
		Positive	0.50	0.50	0.50	6
		Test Accuracy	0.67			18
		Test Macro Avg	0.62	0.62	0.62	18
		Test Weighted Avg	0.67	0.67	0.67	18
		Training Accuracy	1.0000			
64	1 Hidden Layers	Negative	1.00	0.17	0.29	12
		Positive	0.38	1.00	0.55	6
		Test Accuracy	0.44			18
		Test Macro Avg	0.69	0.58	0.42	18
		Test Weighted Avg	0.79	0.44	0.37	18
		Training Accuracy	0.9686			
	1 Hidden Layer with Dropout	Negative	1.00	0.08	0.15	12
		Positive	0.35	1.00	0.52	6
		Test Accuracy	0.39			18
		Test Macro Avg	0.68	0.54	0.34	18
		Test Weighted Avg	0.78	0.39	0.28	18
		Training Accuracy	0.9182			
	2 Hidden Layers	Negative	0.75	0.25	0.38	12
		Positive	0.36	0.83	0.50	6
		Test Accuracy	0.44			18
		Test Macro Avg	0.55	0.54	0.44	18
		Test Weighted Avg	0.62	0.44	0.42	18
		Training Accuracy	0.8428			
	2 Hidden Layers with Dropout	Negative	0.73	0.67	0.70	12
		Positive	0.43	0.50	0.46	6
		Test Accuracy	0.61			18
		Test Macro Avg	0.58	0.58	0.58	18
		Test Weighted Avg	0.63	0.61	0.62	18
		Training Accuracy	1.0000			

A8. Genre Classification Confusion Matrices – Deep Learning Approaches

Hidden Nodes Per Layer	Number of Layers			
	1 Hidden Layer	1 Hidden Layer with Dropout	2 Hidden Layers	2 Hidden Layers with Dropout
32				
64				

A9. Genre Classification Performance Metrics – Deep Learning Approaches

Hidden Nodes Per Layer		Number of Layers															
		1 Hidden Layer				1 Hidden Layer with Dropout				2 Hidden Layers				2 Hidden Layers with Dropout			
		Test Precision	Test Recall	Test F1-score	Support	Test Precision	Test Recall	Test F1-score	Support	Test Precision	Test Recall	Test F1-score	Support	Test Precision	Test Recall	Test F1-score	Support
32	Action	0.29	1.00	0.44	2	1.00	1.00	1.00	2	0.50	1.00	0.67	2	0.50	0.50	0.50	2
	Comedy	1.00	0.50	0.67	4	1.00	0.75	0.86	4	0.75	0.75	0.75	4	0.38	0.75	0.50	4
	Horror	1.00	1.00	1.00	8	1.00	0.75	0.86	8	1.00	1.00	1.00	8	1.00	0.38	0.55	8
	Sci-Fi	1.00	0.25	0.40	4	0.57	1.00	0.73	4	0.50	0.25	0.33	4	0.60	0.75	0.67	4
	Test Accuracy	0.72			18	0.83			18	0.78			18	0.56			18
	Test Macro Avg	0.82	0.69	0.63	18	0.89	0.88	0.86	18	0.69	0.75	0.69	18	0.62	0.59	0.55	18
	Test Weighted Avg	0.92	0.72	0.73	18	0.90	0.83	0.84	18	0.78	0.78	0.76	18	0.72	0.56	0.56	18
	Training Accuracy	1.0000				1.0000				1.0000				0.8050			
64	Action	0.40	1.00	0.57	2	0.33	0.50	0.40	2	0.29	1.00	0.44	2	0.17	1.00	0.29	2
	Comedy	0.33	0.50	0.40	4	0.00	0.00	0.00	4	0.00	0.00	0.00	4	0.00	0.00	0.00	4
	Horror	1.00	0.62	0.77	8	0.80	1.00	0.89	8	0.00	0.00	0.00	8	0.00	0.00	0.00	8
	Sci-Fi	1.00	0.5	0.67	4	0.40	0.50	0.44	4	0.36	1.00	0.53	4	0.33	0.50	0.40	4
	Test Accuracy	0.61			18	0.61			18	0.33			18	0.22			18
	Test Macro Avg	0.68	0.66	0.60	18	0.38	0.50	0.43	18	0.16	0.50	0.24	18	0.12	0.38	0.17	18
	Test Weighted Avg	0.79	0.61	0.64	18	0.48	0.61	0.54	18	0.11	0.33	0.17	18	0.09	0.22	0.12	18
	Training Accuracy	1.0000				0.9937				0.4717				0.4717			

A10. Sentiment Classification Performance Metrics - Classical Machine Learning Approaches (Moskowitz, 2023)

Model		TF-IDF				Doc2Vec Size 100				Doc2Vec Size 500				Doc2Vec Size 1000			
		Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Support Vector Machine	Negative	0.53	0.82	0.64	33	0.53	0.55	0.54	33	0.52	0.42	0.47	33	0.50	0.50	0.49	33
	Positive	0.60	0.27	0.37	33	0.53	0.52	0.52	33	0.51	0.61	0.56	33	0.50	0.52	0.51	33
	Accuracy			0.55	66			0.53	66			0.52	66			0.50	66
	Macro Avg	0.56	0.55	0.51	66	0.53	0.53	0.53	66	0.52	0.52	0..51	66	0.50	0.50	0.50	66
	Weighted Avg	0.56	0.55	0.51	66	0.53	0.53	0.53	66	0.52	0.52	0.51	66	0.50	0.50	0.50	66
Decision Tree	Negative	0.53	0.55	0.54	33	0.49	0.61	0.54	33	0.46	0.36	0.41	33	0.57	0.61	0.59	33
	Positive	0.53	0.52	0.52	33	0.48	0.36	0.41	33	0.47	0.58	0.52	33	0.58	0.55	0.56	33
	Accuracy			0.53	66			0.48	66			0.47	66			0.58	66
	Macro Avg	0.53	0.53	0.53	66	0.48	0.48	0.48	66	0.47	0.47	0.46	66	0.58	0.58	0.58	66
	Weighted Avg	0.53	0.53	0.53	66	0.48	0.48	0.48	66	0.47	0.47	0.46	66	0.58	0.58	0.58	66
Random Forest	Negative	0.57	0..61	0.59	33	0.60	0.64	0.62	33	0.49	0.52	0.50	33	0.57	0.61	0.59	33
	Positive	0.58	0.58	0.58	33	0.61	0.58	0.59	33	0.48	0.45	0.47	33	0.58	0.55	0.56	33
	Accuracy			0.58	66			0.61	66			0.48	66			0.58	66
	Macro Avg	0.58	0.58	0.58	66	0.61	0.61	0.61	66	0.48	0.48	0.48	66	0.58	0.58	0.58	66
	Weighted Avg	0.58	0.58	0.58	66	0.61	0.61	0.61	66	0.48	0.48	0.48	66	0.58	0.58	0.58	66

A11. Genre Classification Performance Metrics – Classical Machine Learning Approaches (Moskowitz, 2023)

Model		TF-IDF				Doc2Vec Size 100				Doc2Vec Size 500				Doc2Vec Size 1000			
		Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support	Precision	Recall	F1-score	Support
Support Vector Machine	Action	1.00	1.00	1.00	17	0.93	0.82	0.87	17	1.00	1.00	1.00	17	1.00	1.00	1.00	17
	Comedy	1.00	1.00	1.00	16	0.93	0.88	0.90	16	0.94	1.00	0.97	16	0.94	1.00	0.97	16
	Horror	1.00	1.00	1.00	17	0.94	0.94	0.94	17	1.00	1.00	1.00	17	1.00	1.00	1.00	17
	Sci-Fi	1.00	1.00	1.00	16	0.74	0.88	0.80	16	1.00	0.94	0.97	16	1.00	0.94	0.97	16
	Accuracy			1.00	66			0.88	66			0.98	66			0.98	66
	Macro Avg	1.00	1.00	1.00	66	0.89	0.88	0.88	66	0.99	0.98	0.98	66	0.99	0.98	0.98	66
	Weighted Avg	1.00	1.00	1.00	66	0.89	0.88	0.88	66	0.99	0.98	0.98	66	0.99	0.98	0.98	66
Decision Tree	Action	0.60	0.35	0.44	17	0.73	0.47	0.57	17	0.58	0.41	0.48	17	0.19	0.18	0.18	17
	Comedy	0.71	0.62	0.67	16	0.43	0.38	0.40	16	0.61	0.69	0.65	16	0.26	0.31	0.29	16
	Horror	0..76	0.76	0.76	17	0.47	0.53	0.50	17	0.63	0.71	0.67	17	0.54	0.41	0.47	17
	Sci-Fi	0.44	0.69	0.54	16	0.36	0.50	0.42	16	0.53	0.56	0.55	16	0.50	0.56	0.53	16
	Accuracy			0.61	66			0.47	66			0.59	66			0.36	66
	Macro Avg	0.63	0.61	0.60	66	0.50	0.47	0.47	66	0.59	0.59	0.59	66	0.37	0.37	0.37	66
	Weighted Avg	0.63	0.61	0.60	66	0.50	0.47	0.48	66	0.59	0.59	0.59	66	0.37	0.36	0.36	66
Random Forest	Action	0.85	1.00	0.92	17	0.50	0.53	0.51	17	0.60	0.53	0.56	17	0.65	0.65	0.65	17
	Comedy	0.94	1.00	0.97	16	0.71	0.62	0.67	16	0.67	0.38	0.48	16	0.83	0.62	0.71	16
	Horror	1.00	0.88	0.94	17	0.79	0.65	0.71	17	0.74	0.82	0.78	17	0.71	0.59	0.65	17
	Sci-Fi	1.00	0.88	0.93	16	0.50	0.62	0.56	16	0.35	0.50	0.41	16	0.48	0.69	0.56	16
	Accuracy			0.94	66			0.61	66			0.56	66			0.64	66
	Macro Avg	0.95	0.94	0.94	66	0.62	0.61	0.61	66	0.59	0.56	0.56	66	0.67	0.64	0.64	66
	Weighted Avg	0.95	0.94	0.94	66	0.63	0.61	0.61	66	0.59	0.56	0.56	66	0.67	0.64	0.64	66