Generating Knowledge Graphs using GPT-J for Automated Story Generation Purposes

Samihan Dani

Approved by:

Dr. Riedl, Advisor
School of Interactive Computing
Georgia Institute of Technology

Date Approved: 5/4/2022 | 12:29 PM EDT

DocuSigned by:
Mark Riedl
D69235FF00B2469...

Dr. Xu
School of Interactive Computing
Georgia Institute of Technology

Date Approved: 5/5/2022 | 11:02 AM EDT

DocuSigned by:
Wei Xu
F95935278BAA471...

Table of Contents

1. Introduction

AI storytelling is a key component of computational creativity. Storytelling is integral to being human since humans use it to entertain, share experiences, and facilitate social bonding. Therefore, the ability to tell a story may be used as a metric for determining whether a system is intelligent or not. Automated Story Generation is a recent challenge in Artificial Intelligence that aims to construct a sequence of sentences in a coherent and understandable way.

While the field of automated story generation has made much progress since its humble beginnings, there are still many issues. Recent techniques have focused on using large pre-trained language models such as GPT-3 (Brown et al., 2020). While these models are fluent and produce grammatically sound sentences, they are prone to producing errors such as repetition or lack of coherence. The main reason for this is that these language models solely produce text that sample from a learned distribution which is unlike how humans would generate text to form a story; thus, this leads to a lack of coherence.

Previous attempts have tried to use high level plot outlines or emotional arcs to dictate how a story is played out (Riedl & Young, 2010). However, this approach is more manual, but inspiration can be taken to guide neural networks in a similar way. This research seeks to use a knowledge graph to make associations between sentences that make logical sense. This will attempt to increase coherency between events in stories. It has even been shown that readers comprehend stories by tracking the relations between events such as causal consequence and character motivations (Graesser et al., 1991). Thus, this paper seeks to create a knowledge graph which can be used to guide automated story generation to find a balance between plot outlines that dictate a story and pre-trained language models that lack coherence.

2. Literature Review

2.1 Guided Story Generation

This form of story generation uses external cues to guide a language model to create a story. These language models are often different types of transformers, and more recently they are often large pre-trained models such as BERT and GPT-3. Several papers use a knowledge graph to make inferences of the logical coherence between sentences. Peng et al. (2021a) uses COMET to make inferences about character's goals and motivations for making predictions for new sentences in the generation process. To go even further, they trained the model on a reinforcement learning policy gradient to generalize the inferencing done by COMET. Like, Peng et al. (2021a), Castricato et al. (2021) uses COMET as well. However, they use COMET for getting what questions to ask a question answering model. This method is quite unique since the model generates the story backwards from a given ending state to a start state. Tambawekar et al. (2019) uses reward shaping to guide their model's story generation. They use a dense reward function to measure how close a story is to a given end goal and to make suggestions to a model for which future candidate sentence brings the story closer to that goal. The clever part of this model is that there are intermediate rewards on the way to the end state instead of just at the end state which improves coherence between sentences for the story.

My research relates this this since I aim to construct a knowledge graph that can guide a language model similar to these studies. This is in contrast to simply just letting a language model create a story in free form because it will not be as structured or keep track of important entities. Using a knowledge graph to guide the story will keep it more coherent. Peng et al. (2021a) and Tambawekar et al. (2019) have shown that guiding the story generation process

does yield more coherent stories. I predict that guiding a language model using a knowledge graph created by GPT-J will also improve coherency of stories.

2.2 Hierarchical Story Generation

Hierarchical Story Generation can be considered as a sub-domain of guided story generation in which a story is generated at a high level and then the surface level text is generated from this high level abstraction. This concept was first introduced by Fan et al. (2018). This is in contrast to the conventional story generation methods which generate text from left to right; this method is closer to how humans may write stories by creating a rough outline of a plot and then writing a story from there. Fan et al. (2019) improved their previous model by both using semantic role labeling for high level abstraction and using a much more powerful pre-trained model with GPT-2. The semantic role labeling gives the model much more specificity and logical progression to create a better story. Yao et al. (2019) also does hierarchical story generation; however, it differs from the previous two papers mentioned since their model uses a knowledge graph for its high level abstraction. It also provides evidence to show that generating the high level abstraction along with the surface text performs significantly worse than generating the abstraction and surface text disjointly. Martin et al. (2020) creates a model which transforms events to sequences. These events have more plot meaning with characters, goals, and actions. This model accomplishes this by using event tuples from WordNet to create causal relationships between these two events. This results in higher coherency between plots.

Several different components of these papers are being applied to my research. The keywords used for the story generation will also be used for my research with story generation. The exception is that the linkage between keywords will likely be done using a knowledge graph

like ConceptNet. In addition, A story can be generated at a high level like done in Martin et al. (2020). An event to event network can be created using the knowledge graph. Eventually the event to sequence will be used to create the surface text of the story. Lastly, semantic role labeling which was done in Fan et al. (2019) might be used for extracting answers out of answers generated by a question answering model used for knowledge graph expansion.

2.3 Open domain question answering

Classical Question answering systems require a context that is typically a paragraph long and a question so that it can retrieve an answer from the passage given. These systems do not necessarily perform question answering in the normal sense. They are more reading comprehension systems that know how to look up an answer from a text. These models only perform well with passages that are 5-7 sentences long. Open-domain question answering solves the problem of requiring these context passages and instead is able to provide an answer with just a question. These work by having a large document store of passages. Yang et al. (2019) presents a model called bertserini. This model has a document store of the entire wikipedia corpus. It uses tf-idf like some other models to retrieve passages from the document store. However, it selects up to a designated k documents since just because a document has a high tf-idf score, that does not mean it contains the answer to the original question. This model creates combines score from passage retrieval and question answering to find the best answer, which may come from the second or third document if it's question answering score was high. Simply copy pasting a segment of a passage may not be quite accurate for answering the question, A passage may contain the answer, but has to be phrased differently to actually answer the question. Lewis et al. (2020) creates a retrieval generative model. The retriever acts

similar to previous retrievers but instead uses a dense vector to retrieve passages. The generative model is trained with BERT to generate an answer. This model is also unique because the retriever and generator are trained together which improves efficiency for finding the correct passage for an answer.

The retrieval generative model seems to be the best model to use for question answering. The dense vector embedding allows better retrieval of passages than using tf-idf. In addition, the joint training of the models provides better synchronization between the generator and retriever. The generative is also more beneficial to just a simple copy paste. This model can be used for knowledge graph expansion. For instance, question templates can be formed about certain nouns to create new connections which are not there in current knowledge graphs. Using a question answering model also for more unique connections. In turn, stories can also become more unique themselves.

However, the scope of information from the previous question answering papers that can be retrieved is limited to the information in available in question-answering datasets such as SQUAD. These yield great results for factual based questions, but these models are not trained for more open-ended purposes. In contrast, large pre-trained language models such as GPT-J are trained on an extremely large corpus and have knowledge from the entire internet. With correct prompting, GPT-J may be able to be used for open domain question answering.

2.4 Knowledge Graphs and Semantic Role Labeling

Knowledge graphs are graph structures that contain associations between multiple entities. ConceptNet5 is a knowledge graph produced by Spencer et al. (2017) which contains a list of tuples for a large number of words. These tuples contain relations between two words;

for instance, one tuple might be (dog, hasA, tail). Another knowledge graph called COMET is trained with a transformer to automatically generate a knowledge graph with inferences. This model is unique since it captures implicit information from its training to generate explicit knowledge. For this paper, I'll simply refer to ConceptNet5 as ConceptNet.

Di Fabio et al. (2019) presented verbAtlas which is a lexical-semantic resource which brings all the verbs from WordNet into semantically coherent frames. These frames contain a prototypical argument structure and provide concept specific information. This also includes semantic roles. For instance, with the sentence "Jenny lived in Florida", Florida will be tagged as a location. In addition, Jenny is tagged as an agent. So for our team's purpose, we can extract a relation triple which is represented as (Jenny, LOC, Florida).

These knowledge graphs combined with the semantic role labeling from verbAtlas will can used to generate a knowledge graph with story remixing. This will be known as the preparing process. When given a starting sentence, semantic role labeling is done to add relation triples to the knowledge graph. We take entities identified by the knowledge graph and look at the associations with these entities in conceptNet. However, I seek to use a knowledge graph created by GPT-J to replace ConceptNet in this scheme.

3. Methodology

Word expansion to create a knowledge graph involves multiple components and models in order to expand a word beyond the inference given in a common knowledge graph like ConceptNet and add new related words. Different prompts are provided to append to a given word which is fed into a large pre-trained language model called GPT-J (Wang & Komatsuzaki, 2021). Currently these prompts only apply to nouns.

3.1 The Algorithm

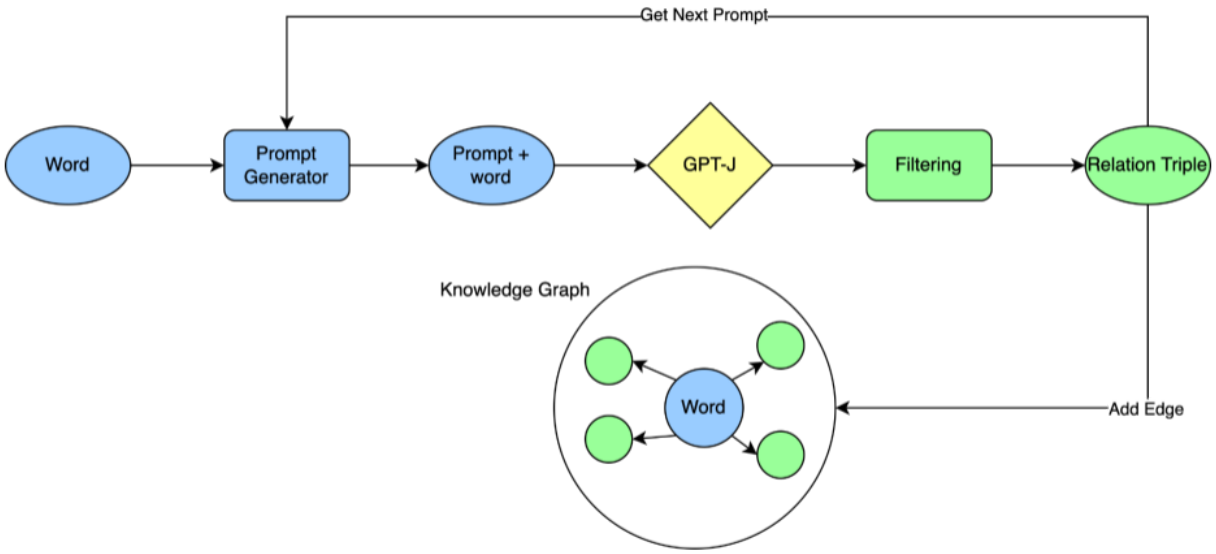


Figure 1. The overall procedure of Knowledge Graph Expansion using GPT-J

1. A given word *w* is passed into a prompt generator and is then appended to a generated prompt

2. This string is passed into GPT-J for inference in batch

3. The results are filtered to only contain the answer to the prompt

4. The remaining relation triples which contain (word, relation, word by relation) are appended as an edge to a knowledge graph for the originally passed in word.

5. Steps 1-4 are repeated until all prompts for a given word type have been used.

3.2 Prompt Generation

The prompt generation system accepts a word that is then classified as a certain noun type. The word gets classified as a certain noun type (person, activity, abstract noun, or physical object) based on if their heteronyms contain certain words. These heteronyms are generated by WordNet (Miller, 1995). The words required for different noun types are shown in Appendix A. Based on the noun type, the words are then added to certain questions best suited to its noun type. These questions are shown in Appendix B.

3.3 GPT-J Inference

Each prompt with the original word included is passed into GPT-J. Each prompt is batched together with identical prompts 7 times. This is done in order to gain multiple answers for each prompt. The temperature for inference was 1.

3.4 Filtering

3.4.1 Answering Shortening

GPT-J is still unpredictable and inaccurate in some of the responses given to the prompts. Thus responses need to be shortened and filtered to try and remove as many poor answers as possible. First, only the first sentence of the response is considered. Next, KeyBERT is used to only consider the 2 most important words. Certain words that are deemed to contain information not relevant to answering the prompt are removed. In addition, certain words are allowed at the beginning of responses but not at the end. Appendix C contains both lists of these words.

3.4.2 Scoring

Finally, the answer along with the original prompt is embedded by a sentence transformer which contains their semantic meaning. The dot product of these two vectors is

calculated and only dot products containing a value above 0.2 is made to become a relation triple to be added to the Knowledge graph.

4. Experiments

4.1 Obtaining coverage data

40 random words were generated from the Random-Word pypackage. Only words with a minimum of 50,000 instances in their corpora were considered to be chosen. For every single word, relations were generated using the GPT-J system. Relations were also retrieved from the ConceptNet API, but certain relation types such as SymbolOF and Synonyms were excluded because many contained words or characters outside the English language and thus would distort results if included. Then three different coverages—regular, substring, and similarity—were calculated. Regular coverage checks for an exact string match between the ConceptNet and GPT-J produced words from edges. Substring checks if a ConceptNet5 word is included as a substring in a GPT-J produced word. Lastly, due to limitations in substring coverage (if a word is a substring of another yet they are not highly related), similarity coverage uses the SequenceMatcher from the difflib pypackage with a threshold of .7 to consider a match.

4.2 GPT-J Relation Accuracy

The accuracy of the relations is measured by human participants. 20 respondents measured how much logical sense relations made from a randomly selected subset of 10 words. They could respond with either Strongly Disagree, Slightly Disagree, Slightly Agree, and Strongly Agree. Both total accuracy and accuracy with the words not covered with ConceptNet5 were calculated for analysis. It was found that while the prompts for uncountable nouns proved to be effective with above average coverage which will be discussed in section 5.1, the prompts were

either awkwardly phrased or not suitable for the given word. Because of this, respondents would have an unfair bias against these prompts despite them producing results that matched well with ConceptNet5, so they were not included in the survey or data in section 5.2.

5. Results and Discussion

5.1 Coverage of ConceptNet

There were three coverages: regular, substring, and similar. While regular coverage seems the most logical, there are often slight differences in words from GPT-J and ConceptNet. For instance, the difference between “to help” and “help” are quite similar, yet with regular coverage, this pair would not count. Similar coverage finds the correct balance between finding strings that are similar enough to be close in meaning, but not have the far reaching effect of substring coverage. For instance, substring coverage would match the word “rock” with “hard rock” when one refers to a physical object and the other refers to a store.

Coverage can be interpreted in many different ways. A high coverage may be a good sign since the model is validated to perform well compared to an already established knowledge graph. Although, ConceptNet does have some limitations lacking knowledge with nouns that are not common. Thus, a low coverage may also be favorable since it shows that GPT-J is forming more unique, better relations than those in ConceptNet; however, a low coverage may also mean that GPT-J is producing poor relations that do not make logical sense.

The overall results for 98 words showed that the regular coverage was 0.10, substring coverage was 0.27, and similarity coverage was .23. However, without further context, this data does not show much. In this section, I’ll break down several major factors which can provide insights. Trends in coverage can be seen by differentiating between types of nouns, number of

words produced by GPT-J, number of words produced by ConceptNet, and the ratio between words produced by GPT-J and words produced by ConceptNet.

5.1.1 Coverage by Noun Type

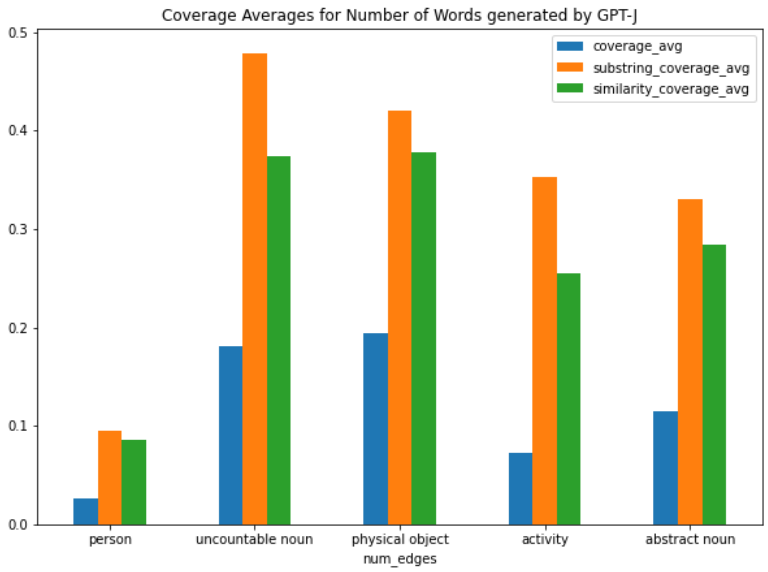


Figure 2. Coverages by Noun Type

5 different categories of nouns were chosen for this study which are physical objects, activities, abstract nouns, uncountable nouns, and people. The people nouns for this study do not include any proper nouns. Shown in Figure 1, every noun category follows the trend of having the substring coverage being the highest, followed by the similarity coverage, and finally followed with the steep drop in the regular coverage. The types of nouns can be grouped into 3 different categories with the physical objects and uncountable nouns performing the best, the

activity and abstract nouns performing slightly less well, and the person nouns performing the worst. These results show how ConceptNet performs great with basic nouns such as water (uncountable) and table (physical object), yet we also often use gerunds such as swimming (activity) and words like evil and love that are abstract, and these are not represented well in conceptNet. Person nouns such as “teacher” and “priest” are heavily lacking in conceptNet, yet knowledge graphs are widely used for storytelling which is often character-driven.

5.1.2 Coverage by Number of words produced by GPT-J

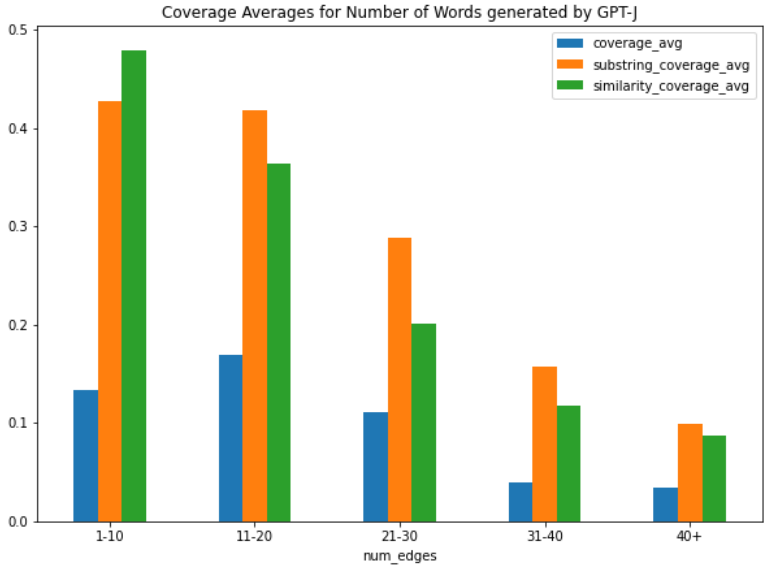
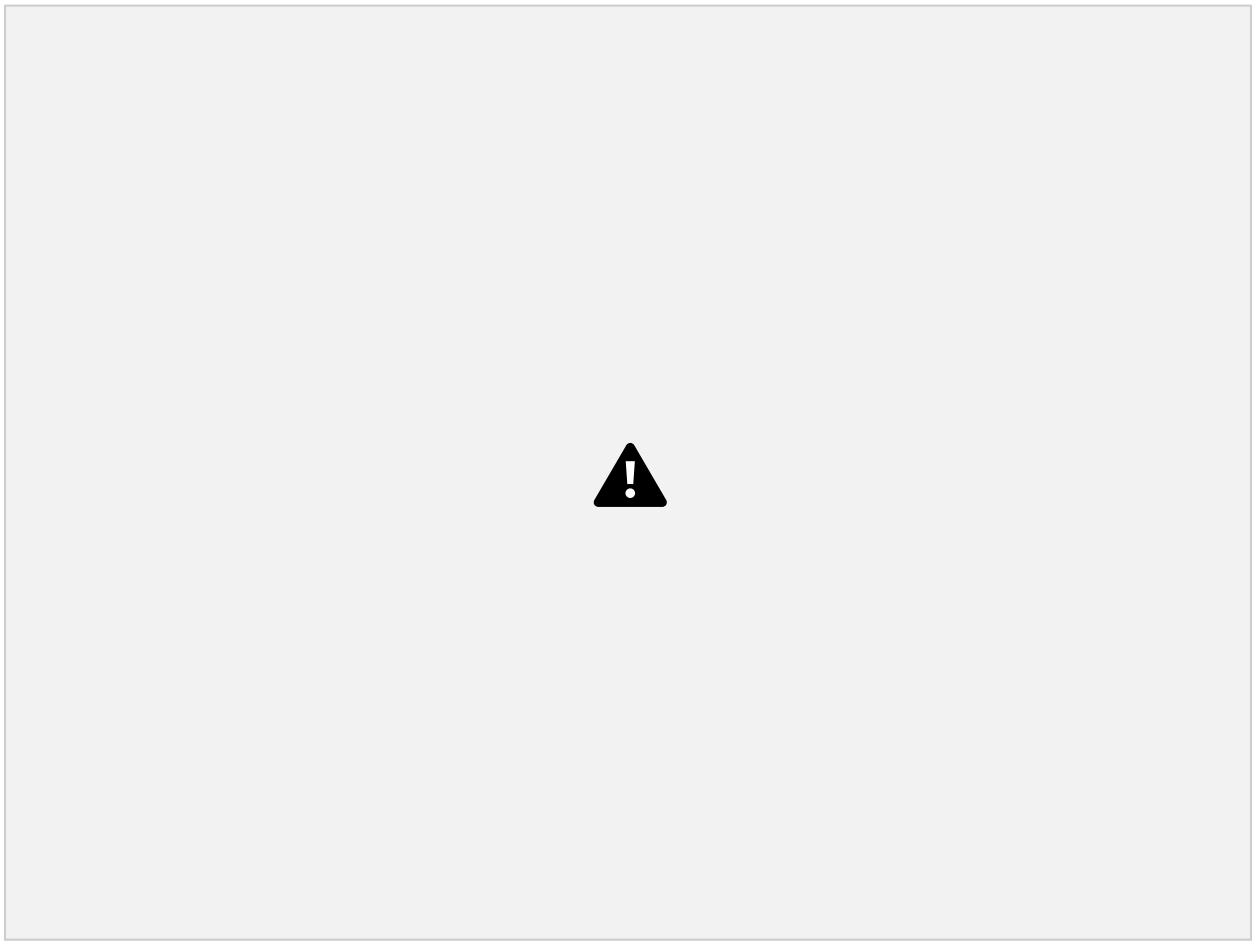


Figure 3. Coverages by the number of edges produced by GPT-J

The data shown in Figure 2 follows the logical explanation that as the number of edges increases the coverage in general decreases. This is because as there are more entities

generated by GPT-J there are more entities that need to be matched. The one outlier is the 11-20 group which increased its regular coverage from the 1-10 group; however, both the substring and similarity coverages decrease which was previously discussed to be the more accurate measurement.

5.1.3 Coverage by Number of Words produced by ConceptNet



The data shown in Figure 4 also follows a logical general trend with coverage increasing as the number of words produced by conceptNet increases. As the number of words in ConceptNet increases there are more opportunities for words produced by GPT-J to match for coverage. These results also point out holes in ConceptNet since the groups with the highest

counts are more often found with a lower amount of relations. Given this context, it can be seen that the majority of the words are coming from the left side of the graph.

5.1.4 Coverage by ratio of GPT-J Produced Words to ConceptNet Produced Words

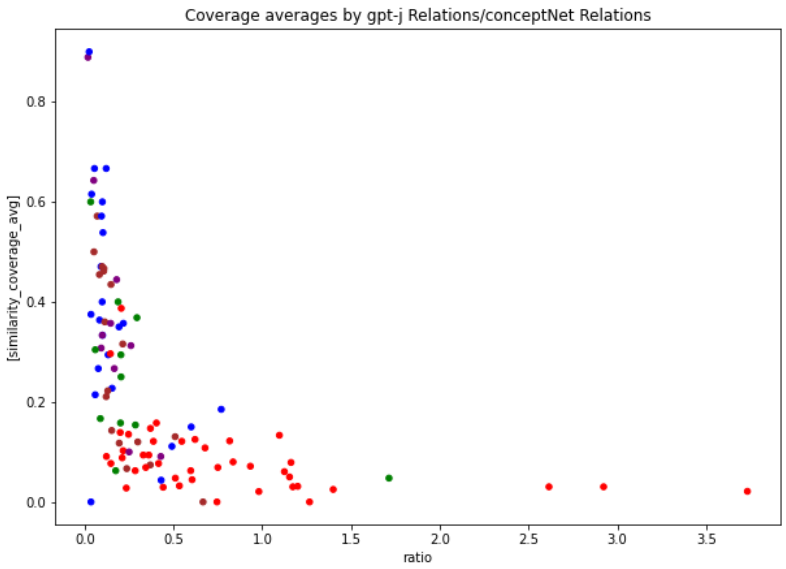


Figure 5. Similarity Coverage by Ratio of GPT-J edges to ConceptNet edges

The data shown in figure 4 shows how the relationship between the ratio and similarity coverage is non-linear with the similarity coverage generally decreasing as ratio increases. However, when looking at subsets of the ratio, it can be seen that there is a somewhat normal distribution of similarity coverages in the subset of ratios from 0 to 0.25. A possible explanation for this is that the number of relations of ConceptNet is high enough that it does not prove to be much of a factor. Another possible insight from this is that data points with both a low

similarity coverage and low ratio may contain poorly generated relations from GPT-J. Another subset has the opposite behavior as the range of ratios from .25 to 1.5 have general consistent coverages with just a slight decrease. This indicates that past the ratio of .25, ConceptNet does not have enough words to cover the relations produced by GPT-J.

5.2 Word Relation Evaluation

A number of 1, 2, 3, and 4 was given to responses of Strongly Disagree, Slightly Disagree, Slightly Agree, and Strongly Agree. For the results, what constitutes a good relation is one with an average above 3.00. A relation was labeled as bad if it was below 2.00. Making the threshold simply greater than or less than ensures that at least one respondent gave a 4 for a good response and at least one respondent gave a 1 for a bad response. Numbers within the range of 2 and 3 including these 2 numbers were labeled as neutral responses. In addition, the survey responses for words that were not found to fall in the similarity coverage of ConceptNet5 were analyzed as well.

5.2.1 Overall Evaluation

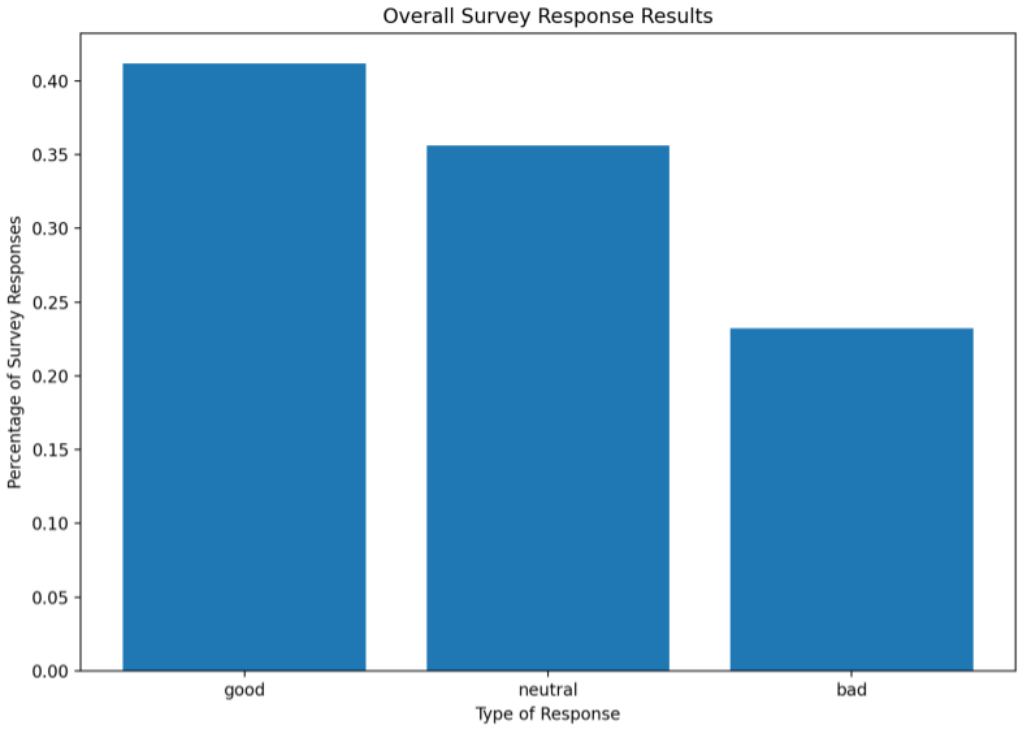


Figure 6. Overall Average Percentage of Different Survey Responses

Figure 6 shows the breakdown of the total responses from the survey. The good responses make up a plurality of all the responses produced by GPT-J with 41.1% of the total response. This is followed by 35.6% of responses being neutral, and 23.2% of responses being bad. Bad, neutral, and good responses have averages below 2, between 2 and 3, and above 3 respectively. This graph does not take the average of all relations, but rather the average of the percentage of good relations that each word has. This standardizes against words which have a noun type that produces a lot of relations such as the person noun type. Although the majority of responses are not good responses, these results are promising since a large portion of results are neutral. This means the system produces a majority of responses that are not failures with a

combined percentage of 76.1% of responses not being bad responses. Although respondents were not asked for reasons behind their answers, possible reasons for relations being neutral are poor grammar and syntax produced by GPT-J. This hints to the possibility that with further improvements, the neutral responses can perhaps become good responses.

5.2.2 Evaluation by Noun Type

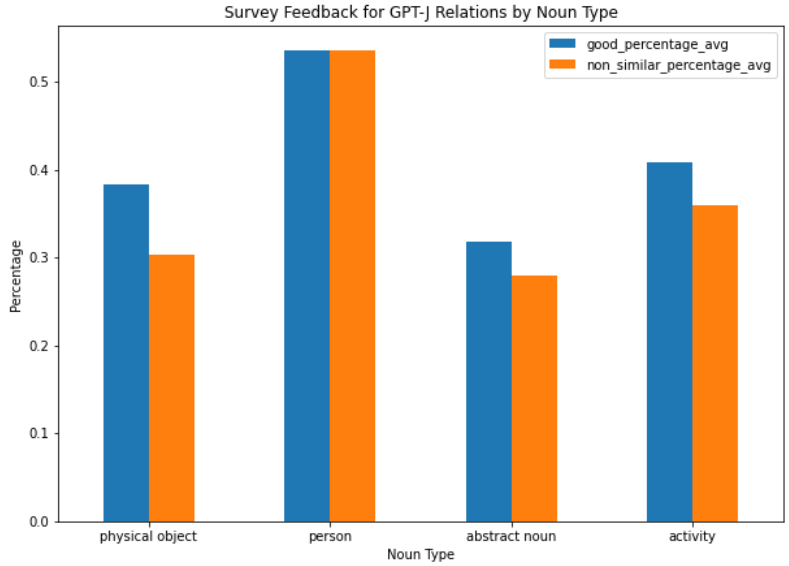


Figure 7. Positive Survey Response Percentages by Noun Type

The data shown in Figure 7 shows both the average percentage of relations and average percentage of relations not similar to those in ConceptNet5 grouped by the noun type. The person noun type has both the highest percentage of total relations and relations not similar to ConceptNet5. In contrast abstract noun has the lowest percentages. The person noun type has

very similar percentages because the vast majority of relations are not included in ConceptNet. Yet it can be seen that a little over 50% of the relations not included in ConceptNet5 are once with positive feedback. With the assumption that the ConceptNet5 relations are good relations, it can even be seen for the other noun types that ConceptNet only covers a small fraction of the total good responses of GPT-J. This points to the fact that GPT-J produces many unique relations that are both good and not in conceptNet.

Although these results do point to potential holes within ConceptNet5, this system still needs major refinements and filtering to ensure that only good responses are kept. A knowledge graph with a low percentage of positive responses may harm story generation.

5.2.3 Evaluation by ConceptNet relations and Ratio of GPT-J to ConceptNet relations

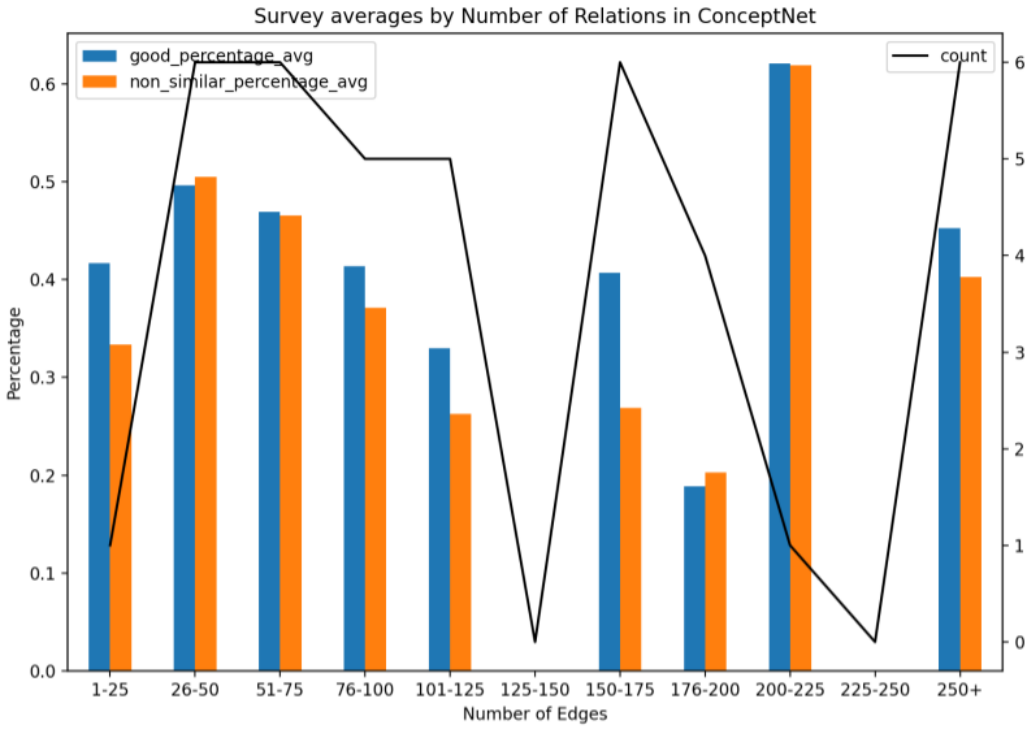


Figure 8. Positive Survey Response Percentages by Number of ConceptNet Edges

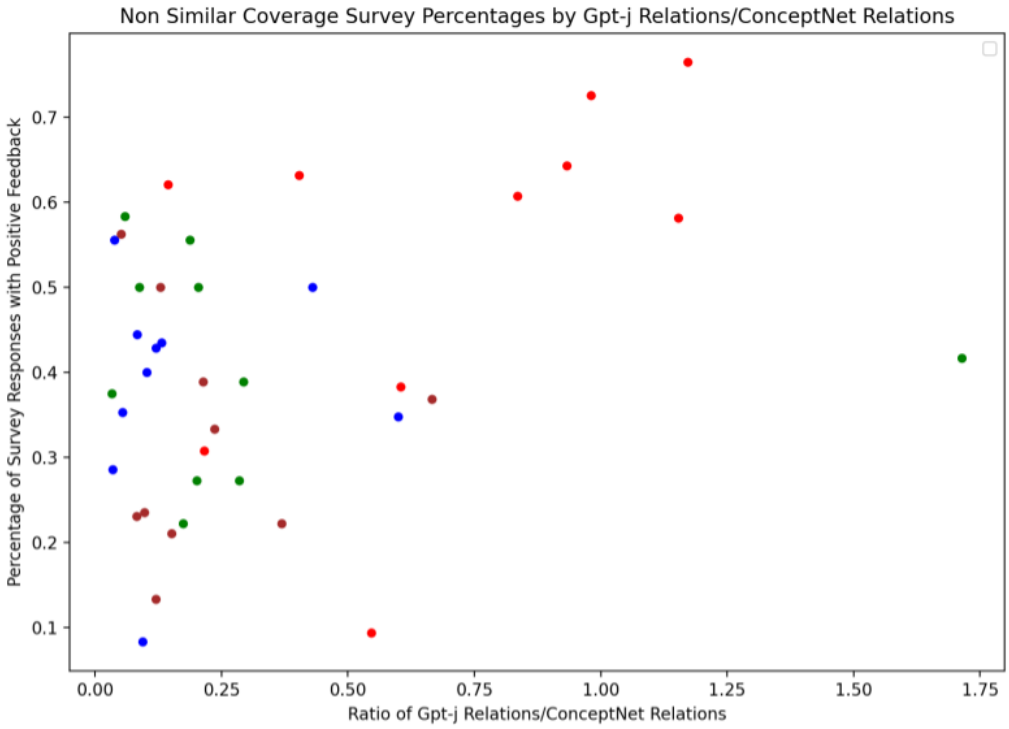


Figure 9. Positive Non-Similar Survey Response Percentage by Ratio of GPT-J Edges to ConceptNet5 Edges

Figure 8 shows both average percentage of positive relations from all relations and those not similar to ConceptNet and groups them by how many relations are in ConceptNet. The black line refers to the count of the words per bin shown on the right y-axis. Figure 9 shows just the percentage for positive relations not similar to ConceptNet and their ratios of GPT-J relations to ConceptNet5 relations while grouping the data points by noun type. The first Figure shows that the percentages of good responses are higher for a low amount of relations, low for a medium amount of relations, and then are high for a high amount of relations. The bin with 200-225 relations can be seen as an outlier since there is only 1 word that makes up the bin.

For figure 9, ideal data points would be in the top right with a high percentage and high ratio. This is because many GPT-J relations can be used to enhance a ConceptNet5 word that has a relatively low amount of relations. Overall, most data points have a low ratio while having a relatively even distribution between high and low percentages in that subset of ratios. A reason for this may be that a low amount of prompts were given for these words, thus less relations were formed. In contrast, the person noun type had the most amount of prompts and performed great with a high percentage and high ratio. This once again, emphasizes ConceptNet5's weakness of having a low amount of relations for person noun types which are common in storytelling.

6 Conclusion and Future Work

GPT-J has shown to generate many relations to words that are not included in ConceptNet5. Many of these relations have been shown to not only be unique but also logically correct. The biggest weakness that this system fixed was providing relations for words in ConceptNet5 that had a low amount of relations. Additionally, this paper identified key noun types in which ConceptNet5 performed poorly, most notably the person type which is a key element in storytelling. However, there are still some serious limitations with this system. The amount of answers that were deemed good responses only constituted 40%. While the majority of words not included in this were neutral, this system still needs to be improved for obtaining only good responses.

There are a number of future directions to take with applying this knowledge graph to automated storytelling. The words used for knowledge graph generation were simply randomly generated. In contrast, knowledge graphs for entities contained in ROCStories can be generated

to be more suited for the purposes of storytelling. Furthermore, this system can replace ConceptNet5 in the automated story generation system presented in Peng et al. (2021b). Using this system in contrast to ConceptNet5 may yield stories that are more unique and detailed. Concerning the knowledge graph generation itself, the randomly generated words used had a rather high threshold for how many times they appeared in the random word generator corpus. More experiments can be tried with less common words.

References

Bosselut, A., Rashkin, H., Sap, M., Malaviya, C., Celikyilmaz, A., & Choi, Y. (2019). Comet: Commonsense transformers for automatic knowledge graph construction. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Retrieved from https://arxiv.org/abs/1906.05317.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Castricato, L., Frazier, S., Balloch, J., & Riedl, M. (2021, January). Tell Me A Story Like I'm Five: Story Generation via Question Answering. In *Proceedings of the 3rd Workshop on Narrative Understanding*.

Di Fabio, A., Conia, S., & Navigli, R. (2019). Verbatlas: A novel large-scale verbal semantic resource and its application to semantic role labeling. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Retrieved from https://aclanthology.org/D19-1058/.

Fan, A., Lewis, M., & Dauphin, Y. (2018). Hierarchical neural story generation. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Retrieved from https://arxiv.org/abs/1805.04833.

Fan, A., Lewis, M., & Dauphin, Y. (2019). Strategies for structuring story generation. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Retrieved from https://arxiv.org/abs/1902.01109.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Martin, L., Ammanabrolu, P., Wang, X., Hancock, W., Singh, S., Harrison, B., & Riedl, M. (2020, April). Event representations for automated story generation with deep neural nets. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1).

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39-41.

Peng, X., Li, S., Wiegreffe, S., & Riedl, M. (2021). Inferring the Reader: Guiding Automated Story Generation with Commonsense Reasoning. *arXiv preprint arXiv:2105.01311*.

Peng, X., Xie, K., Alabdulkarim, A., Kayam, H., Dani, S., & Riedl, M. O. (2021). Guiding Neural Story Generation with Reader Models. *arXiv preprint arXiv:2112.08596*.

Speer, R., Chin, J., & Havasi, C. (2017, February). Conceptnet 5.5: An open multilingual graph of general knowledge. In Thirty-first AAAI conference on artificial intelligence.

Tambwekar, P., Dhuliawala, M., Martin, L. J., Mehta, A., Harrison, B., & Riedl, M. O. (2019). Controllable neural story plot generation via reward shaping. *Proceedings of the Twenty-Eighth*

International Joint Conference on Artificial Intelligence. Retrieved from https://arxiv.org/abs/1809.10736.

Wang, B., & Komatsuzaki, A. (2021). GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model.

Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M., & Lin, J. (2019). End-to-end open-domain question answering with bertserini. *Proceedings of the 2019 Conference of the North*. Retrieved from https://arxiv.org/abs/1902.01718.

Appendix

A. List of WordNet synsets for each noun type

Abstract Noun: attribute.n.02

Physical Object: object.n.01

Person: person.n.01

Activity: activity.n.01 OR act.n.01

Uncountable Nouns: substance.n.01 OR matter.n.03 OR measure.n.02

B. Questions for Each Noun Type

Questions shown in format: Relation: Question. "{}" is the placeholder for the word.

Questions for Person

LIVE: Where does the {} live?,

KNOWN: What is the {} known for?,

NAME: What is {}s name?,

LIKE: What does the {} like to do?,

DISLIKE: What does the {} not like to do?,

FROM: Where is the {} from?,

GOING: Where is the {} going?,

DESIRE: What does the {} desire?,

MOTIVATED: What is {} motivated by?,

DOING: What is {} doing right now?,

UNIQUE: What is unique about the {}?,

DESCRIBE: How is the {} described?

Questions for Activity

DEF: What is {}?,

PURP: What is the purpose of {}?,

POS: What do people like about {}?,

NEG: What do people dislike about {}?,

LOC: Where does {} take place?,

UNIQUE: Whats unique about {}?,

Questions for Abstract Nouns

DEF: What is {}?,

MEAN: What does {} mean?,

IMPORTANT: What is important about {}?,

UNIQUE: What is unique about {}?,

SIM: What is {} similar to?,

POS: What is good about {}?,

NEG: What is bad about {}?,

OPP: What is the opposite of {}?,

Questions for Physical Object

DEF: What is a {}?,

LOC: Where is a {} found?,

MADE: What is a {} made of?,

UNIQUE: What is unique about a {}?,

IMPORTANT: What is important about a {}?,

PROP: What are the properties of a {}?,

SIMILAR: What is a {} similar to?,

USED: What is a {} used for?

Questions for Uncountable Noun

DEF: What is {}?,

LOC: Where is {} located?,

MADE: What is {} made of?,

UNIQUE: What makes {} unique?,

IMPORTANT: What is important about {}?,

PROP: What are the properties of {}?,

SIMILAR: What is {} similar to?,

USED: What is {} used for?

C. Words excluded in post-processing

Words to be excluded: the, unique, question, is, a, at, so, is, are, for, an, in, of, and, or, on, he, she, his, he, but, by, be, who, when, where, why, what, from

Words to only be excluded at the end: has, that, with, it, to