Exploring the Usage of Knowledge Graphs in Identifying Human and LLM-Generated Fake Reviews

Ming Liu and Massimo Poesio
Queen Mary University of London
Mile End Road
London E1 4NS
{acw661,m.poesio}@qmul.ac.uk

Abstract

The emergence of large language models has led to an explosion of machine-generated fake reviews. Although distinguishing between human and LLM-generated fake reviews is an area of active research, progress is still needed. One aspect which makes current LLM-generated fake reviews easier to recognize is that LLMs–in particular the smaller ones–lack domain-related knowledge. The objective of this work is to investigate whether large language models can produce more realistic artificial reviews when supplemented with knowledge graph information, thus resulting in a more challenging training dataset for human and LLM-generated fake reviews detectors. We propose a method for generating fake reviews by providing knowledge graph information to a llama model, and used it to generate a large number of fake reviews which used to fine tune a state-of-the-art human and LLM-generated fake reviews detection system. Our results show that when knowledge graph information is provided as part of the input, the accuracy of the model is improved by 0.24%. When the knowledge graph is used as an embedding layer and combined with the existing input embedding layer, the accuracy of the detection model is improved by 1.279%.

1 Introduction

Large Language Models (LLMs) are now able to generate high-quality and widely used text. For example, ChatGPT (ChatGPT, 2022) can generate text of near-human level quality, including for the case of creative texts such as music lyrics (Khan, 2024). But as the text-generation capabilities of LLMs grow, so does their use to generate misinformation, plagiarized text, and fake product reviews (Sun et al., 2024). More and more LLM-generated, human-like fake reviews affect online purchasing of all types of goods, including book buying, the type of reviews we're studying. Therefore, discriminating text generated by humans from text generated by LLMs has become a major technological challenge and societal challenge (Prajapati et al., 2024).

Although distinguishing between human and LLM-generated fake reviews is an area of active research (Wu et al., 2025), progress is still needed. In particular, it is necessary to train classifiers on datasets in which the artificial reviews are not so different from real reviews as to make them easily detectable. One aspect which makes current LLM-generated fake reviews easier to recognize is that LLMs–in particular the smaller ones–lack domain-related knowledge (Feng et al., 2023) The objective of this work is to investigate whether large language models can produce more realistic artificial reviews when supplemented with knowledge graph information, thus resulting in a more challenging training dataset for human and LLM-generated fake reviews detectors.

Two knowledge graph fusion methods are explored: Prompt Embedding and Model Embedding. When generating new fake reviews, Prompt Embedding uses the knowledge graph as the input of the generation model, while Model Embedding uses the knowledge graph as part of the generation model. Our results show that that the generation model in which the knowledge graph is fused in the architecture can better generate fake reviews, thereby improving the performance of the detection model.

2 Related Work

In this Section we briefly review previous literature on detecting LLM-generated text and on using knowledge graphs in LLMs.

Proceedings of Recent Advances in Natural Language Processing,pages 674–681
Varna, Sep 8–10, 2025
https://doi.org/10.26615/978-954-452-098-4-078

2.1 LLM-Generated Text Detection

In recent years, the development of automatic detection methods to identify LLM-generated texts has become a very active area of research (Prajapati et al., 2024). Sadasivan et al. (2023) proposed a framework dealing with distinctive writing styles, clever prompt design, or text paraphrasing. At the same time, the study shows that even when a large model is protected with a watermark, the model is vulnerable to spoofing attacks (Sadasivan et al., 2023). DetectGPT (Mitchell et al., 2023) computes log-probability to detect machine-generated text, rather than training a separate classifier or even collecting a dataset. However, this method is limited to white-box environments. For models that do not provide log probabilities for calculation, DetectGPT cannot help (Mitchell et al., 2023). Watermark (Kirchenbauer et al., 2023) controls the text generation process, selecting tokens marked as 'green'. In practice, complex watermarks can degrade text quality due to the lack of flexibility of large language models (Kirchenbauer et al., 2023). GLTR (Gehrmann et al., 2019) is a tool for detecting whether text is LLM-generated. Without using pre-training, GLTR improves the human detection rate of fake text from 54% to 72% (Prajapati et al., 2024). However, if non-biased sampling is used to generate text, this method is difficult to identify (Prajapati et al., 2024).

2.2 Graph-Based Retrieval-Augmented Generation (RAG)

Knowledge graphs are machine-readable graph structures that include entities, relationships, and attributes. They store data in a structured manner in a graph database, which can reflect the physical objects and relationships in real life (Lavrinovics et al., 2025).

Studies have shown that adding relevant domain information when generating text in knowledge-intensive tasks can effectively improve the context-awareness of LLMs. At the same time, hallucinations are reduced while keeping the original model structure unchanged (Lewis et al., 2020; Ram et al., 2023; Agrawal et al., 2024). Therefore, using structured external information such as knowledge graphs can make LLM's output more consistent, and provide more accurate answers (Pan et al., 2023, 2024).

StructGPT (Jiang et al., 2023) uses knowledge graphs and tabular data to enhance GPT's information extraction capabilities. Wu et al. (2023) team textualised the content of the knowledge graph to enhance the model performance when performing question-answering tasks. Baek et al. (2023) team proposed the KAPING model. This model also uses the knowledge graph and extracts the corresponding triples from it, and is used for the zero-shot question answering task.

3 Methods

In this section, we explain our architecture, and introduce the two different ways of applying knowledge graphs that we developed.

3.1 Overall Architecture and Baselines

The task can be divided into three components: knowledge graph creation, generation of fake reviews to be added to the training set, and human and LLM-generated reviews detection. First, in the knowledge graph creation process, we generate the corresponding knowledge graph around the existing dataset. Then, in the generation process, we generate the corresponding dataset based on the existing dataset and merge the generated and old datasets into a new dataset. Finally, in the detection process, we use the existing efficient detection model to detect our enhanced dataset.

Although it is possible to evaluate the quality of fake reviews generation through text-quality metrics (Mitchell et al., 2023; Chim et al., 2025), ultimately, the only truly telling way to evaluate our approach for generating fake reviews is through the impact of these reviews on the overall detection task. This will be our main metric.

As a detection baseline, we use the existing state-of-the-art detection model LLM2S3 (Spiegel and Macko, 2024a). Because LLM2S3 only provides the architecture but not the trained model, we fine-tuned Falcon-7B (Almazrouei et al., 2023) and Mistral-7B (Chaplot, 2023). (Please refer to the section 3.4 for more information about the model.)

An existing dataset of fake reviews is used as starting point, and the detection results obtained after training the LLM2S3 model on this dataset are used as our baseline.

3.2 Knowledge Graph Creation

The knowledge graph is extracted out of the datasets used in this study, the Amazon dataset and DeRev 2018 (see Section 4.1).

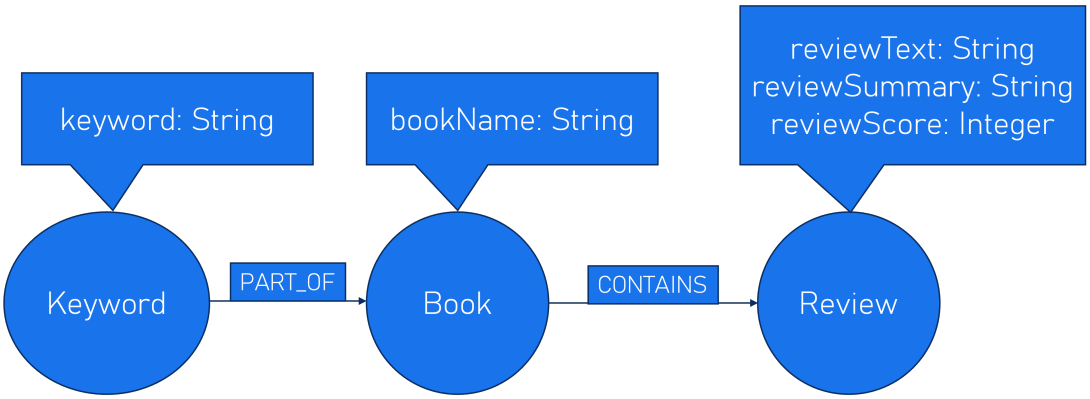

Figure 1: The Structure of Data in Knowledge Graph.

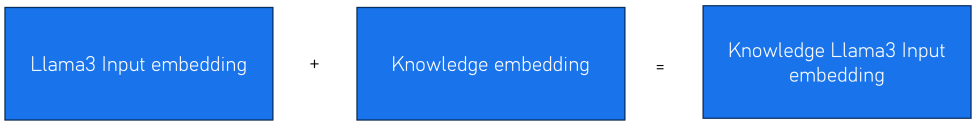

Figure 2: Input Integration of Llama Model.

Preprocessing We first run two preprocessing steps. First, the Amazon dataset and DeRev 2018 were merged. Second, we obtained query keywords by keyword extraction from the reviews in the dataset. Because there are multiple keywords, the query results will be concatenated as the final result.

Neo4j We use Neo4j (Neo4j, 2012) as the storage tool for the knowledge graph. It is an ideal graph data storage and query solution that can be applied to a variety of fields, such as social network analysis, knowledge graph modelling, and healthcare.

Data Modelling The structure of the data model can be found in Figure 1. It contains three nodes: ”Keyword”, ”Book”, and ”Review”. The ”Keyword” node contains a string attribute ”keyword”, whose content is the keyword itself. The ”Book” node contains a string attribute ”bookName”, whose content is the name of the book. The ”Review” node contains three attributes: the string attribute ’reviewText’, whose content is the review body; the string attribute ’reviewSummary’, which contains the review summary; the integer attribute ’reviewScore’, which contains the user rating of the purchased book.

The relationship between these nodes is as follows. 1. ”Book” contains multiple ”Review” from different users. 2. Each ”Review” can extract a number of ”Keyword”. However, in the actual model creation, in order to make retrieval more convenient, we changed the association relationship among three nodes: 1. ”Book” contain multiple ”Review” from different users (CONTAIN). 2. ”Keyword” is part of ”Book” (PART_OF). Here we remove the relationship between ”Review” and ”Keyword”, and transfer this relationship to ”Book” and ”Keyword”.

Data Import During the data import process, we import the Amazon Customer Reviews Dataset (Amazon, 2018) and DeRev 2018 (Fornaciari and Poesio, 2014; Fornaciari et al., 2020) datasets into the created data model. It should be noted that there are certain differences in the content of the two datasets, which affect the merging of the data. Among them, the book names in the two datasets are obviously different. To create a better knowledge graph, we uniformly modify the book name format of the Amazon dataset to the same Upper Camel Case as DeRev 2018. Refer to Section 4.1 for more information about the two datasets.

3.3 Using the KG for Generation

During the generation process, we mainly use two methods: Prompt Embedding and Model Embedding. Both methods are implemented using llama3.1-8B (Grattafiori et al., 2024).

Prompt Embedding This method involves modifying the input of the training model by adding the corresponding knowledge graph content in the Prompt. The following is an example input:

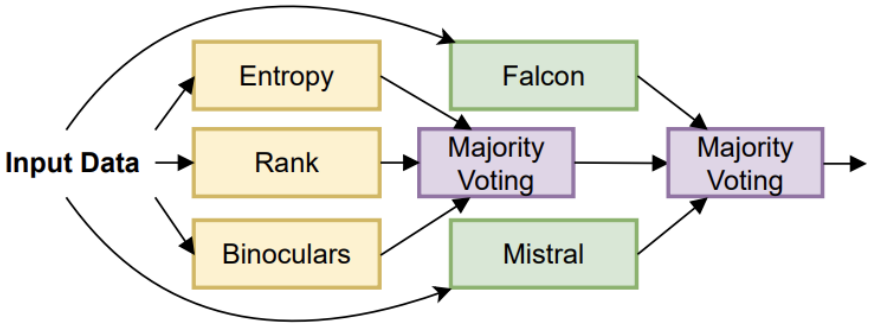

Figure 3: The Model of LLM2S3 created by KInIT team in the SemEval-2024 competition.

```
<review ID="1" title="ADubiousPlan" writer="GeraldKubicki" author="SandraParker" date="July 20, 2012
        serialdate="735084" stars="5" found="1" fold2014="1" fold2016="1" gold2014="0" gold2016="0"
        silverMaj4="0" silverMaj3="0" silverRay4byMaj4="0" silverRay4byMaj3="0" silverRay4byRand="1"
        silverRay3byMaj4="1" silverRay3byMaj3="0" silverRay3byRand="1" silverWhi4="0" silverWhi3="0"
        clueTot="3"  clueSB="1" clueCL="0" clueNN="1" clueUP="1">
    <object>A Dubious Game-Another Kubicki Masterpiece</object>
    <body>Gerald Kubicki has done it again. A Dubious Plan, the fifth installment of the Colton Bany
        delivers on it's promise of adventure, mystery and plenty of sex in yet another engaging and
```

Figure 4: Example of DeRev 2018. Each comment is in XML document format, which contains the title, author, time and content of the comment. It also contains tokens generated by comments.

{ “content”: ” Generate text by giving keywords and knowledges: keywords(keywords1, keywords2, ...), knowledge(concatenate(knowledge1, knowledge2, ...))”,“role”: “user” }, { “content”: ” original text”, ”role”: ”assistant” }

This example expresses the generation of a comment through keywords keyword1, keyword2, etc. At the same time, it is necessary to refer to the output of the knowledge graph. It is obvious here that the knowledge graph is part of the model input. Therefore, we call it Prompt Embedding. In this way, we achieve embedding of the knowledge graph without actually modifying the generative model.

Model Embedding This method involves modifying the model to add additional knowledge graphs to enhance the generation of fake reviews. The following is an input example:

{ “content”: ” Generate text by giving keywords: keywords(keywords1, keywords2, ...)”, “role”: “user” }, { “content”: ” original text”, ”role”: ”assistant” }

This example shows that the model generates fake reviews through keywords. Compared with Prompt Embedding, the knowledge graph is missing. Here, we combine the knowledge graph with the input layer of the Llama model. You can refer to the figure 2. First, we embed the knowledge graph result to get the knowledge embedding layer. Then add it to the original Llama input embedding to get the new Knowledge Llama3 Input embedding layer. It should be emphasised that in the actual construction, we did not create and add a new knowledge graph embedding layer. We just merged the two layers and overwrote the original input layer.

3.4 Detection

As a detection system, we used the model developed by the KInIT team in the SemEval-2024 competition (Spiegel and Macko, 2024b), one of the best performing systems in the competition. The structure of the model is shown in Figure 3. The authors used two majority votes and combined numerical calculation metrics with large models. First, Entropy, Rank and Binocular are calculated for the whole dataset. Then, a new prediction result is obtained by majority vote. The second majority vote prediction is a combination of the prediction result of the previous one with the prediction results

DOC_ID	LABEL	RATING	REVIEW_TEXT	VERIFIED_PURCHASE
1	__label1__	4	When least you think so, this product wil…	N
2	__label1__	4	Lithium batteries are something new intro…	Y
3	__label1__	3	I purchased this swing for my baby. She i…	N
4	__label1__	4	I was looking for an inexpensive desk cal…	N
5	__label1__	4	I only use it twice a week and the result…	N

Figure 5: Sample of Amazon Customer Reviews Dataset with tags, review text, user ratings and product categories.

of Mistral and Falcon. This system achieved fourth place in the competition. Importantly, we fine-tune the Falcon and Mistral models to make them more suitable for our data types.

Finally, the datasets generated by Prompt Embedding and the datasets generated by Model Embedding are verified on LLM2S3 respectively.

4 Experiments

In this section, we introduce the dataset, experimental settings, and experimental process in detail.

4.1 Datasets

We focused on book reviews, and mainly used the Amazon Customer Reviews Dataset (Amazon, 2018) and DeRev 2018 (Fornaciari and Poesio, 2014; Fornaciari et al., 2020) as datasets.

DeRev 2018 It contains both genuine reviews with strong evidence and fake reviews created by users for financial gain. Figure 4 is a sample of the DeRev dataset. *gold2016* indicates the truth or falsity of a review. 0 is false and 1 is true. It contains 8311 high-quality, cleaned data.

Amazon Customer Reviews Dataset Figure 5 is a sample of the Amazon dataset. What we need to pay attention to here are the *REVIEW_TEXT* and *LABEL* labels. *REVIEW_TEXT* is the review itself, without any false or true labels. *REVIEW_TEXT* indicates whether the review is true or not. __label1__ indicates false, and __label2__ indicates true. It contains 21,000 items, categorised into 30 classes, each of which contains 700 reviews.

In this experiment, we first used the entire DeRev 2018 dataset. Second, we used data items with category 'books' from the Amazon dataset. The table 1 shows some information.

4.2 Experimental Setup

First of all, we follow the 2:8 principle to split the dataset into a test set and a training set. 20% of the data is used as a validation set, and 80% of the data is used as a training set.

Then, both methods take the following steps to create a dataset.

1. Split the original reviews and get the keyword list.

2. Get the corresponding attributes and associations in the knowledge graph based on the keyword list.

3. Connect the knowledge graph results together.

Finally, when fine-tuning LLM2S3, only the original dataset is used. At test time, the dataset generated by Prompt Embedding and Model Embedding will be used respectively.

4.3 Experimental Process

As previous discussed, we compare two data generation methods, illustrated in Figures 6 and 7, respectively. First, a knowledge graph is created and imported into the Neo4j database. Next, a mesh structure centered on the "Keyword" node is generated. When generating a new dataset, use keyword extraction on the original reviews. The corresponding relevant review summary is then obtained in the knowledge graph for keywords. Because a keyword corresponds to multiple books and multiple reviews, each time the Llama model generates reviews, it refers to reviews with the same keyword for different books. In this process, it is necessary to remove repeated book reviews.

Then, when using the Prompt Embedding method, all the knowledge is used as part of the input prompt for the Llama model to generate new reviews. When using the Model Embedding method, all the knowledge is first converted to the embedding layer and then merged with the original input layer of Llama to generate new reviews. It should be noted that the knowledge may be too long, sequence embedding is required.

Finally, we use LLM2S3 as a detection model to identify humans and LLM-generated reviews. Because LLM2S3 has two models that need to be fine-tuned, Mistral and Falcon, we use the original Amazon and DeRev datasets for fine-tuning. Because our original dataset only has human-written reviews, we used another Llama

Dataset	Size	Details
Amazon Reviews (Amazon, 2018)	350 fake and 350 truthful reviews	Published by Amazon
DeRev 2018 (Fornaciari and Poesio, 2014)	8311 reviews	Book reviews

Table 1: Datasets for Human and LLM-generated Fake Review Detection.

Model	Accuracy	F1	Precision	Recall
Baseline	88.5%	87.3%	**90.2**%	86.8%
Prompt Embedding	88.8%	88.2%	89.8%	88.6%
Model Embedding	**89.8**%	**89.8**%	87.9%	**91.8**%

Table 2: The Performance of Different Approaches

model to paraphrase the original two datasets to make them machine-generated datasets. 80% of the original data and rewritten data are used to fine-tune the Mistral and Falcon models. When we finally use LLM2S3 for detection. Use 20% of the original data and 20% of the generated dataset for verification, and also take the average multiple times.

5 Results and Discussion

Table 2 shows the results with training LLM2S3 using datasets computed using the two approaches. The results show that both methods have achieved some improvement in accuracy compared to the baseline, but the improvement with Prompt Embedding is limited. The Baseline approach has the highest precision, and the Model Embedding has the highest accuracy, F1, and recall values.

We conducted two sets of tests on the significance of the differences found between baseline and the two KG embedding models: Baseline vs Prompt Embedding, and Baseline vs Model Embedding. **Baseline and Prompt Embedding Null Hypothesis:** There is no significant difference between the results of the Baseline and Prompt Embedding models. **Baseline and Model Embedding Null Hypothesis:** There is no significant difference between the results of the Baseline and Model Embedding models.

Table 3 shows the t and p values for the two tests. When comparing Baseline and Prompt Embedding, the null hypothesis holds, although barely, because p¿0.05 (marginally). This indicates that there is no significant difference between the Baseline and Prompt Embedding models. When comparing Baseline and Model Embedding, however, the null hypothesis does not hold, because p¡0.05. This indicates that there is a significant difference between the Baseline and Model Embedding models.

Judging from the values of metrics, the results of the three methods are very stable, which means that the model-generated dataset and the original dataset are also very stable. The improvement may seem small, but it should be emphasized that the performance of LLM2S3 is already very high, so any improvement over that is very difficult.

6 Conclusion

In this paper we compared two methods for augmenting the dataset used for training a LLM-generated text detection model by generating additional review through a model that leverages knowledge graph information. Our results show that both methods achieve a small improvement in detecting LLM-generated text over a baseline, but embedding the knowledge graph information in the model gives the best results.

References

Garima Agrawal, Tharindu Kumarage, Zeyad Alghamdi, and Huan Liu. 2024. Can knowledge graphs reduce hallucinations in LLMs? : A survey. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3947–3960, Mexico City, Mexico. Association for Computational Linguistics.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Merouane Debbah, Etienne Goffinet, Daniel Heslow, Julien Launay, Quentin Malartic, et al. 2023. Falcon-40b: an open large language model with state-of-the-art performance.

Amazon. 2018. Amazon customer reviews corpus.

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

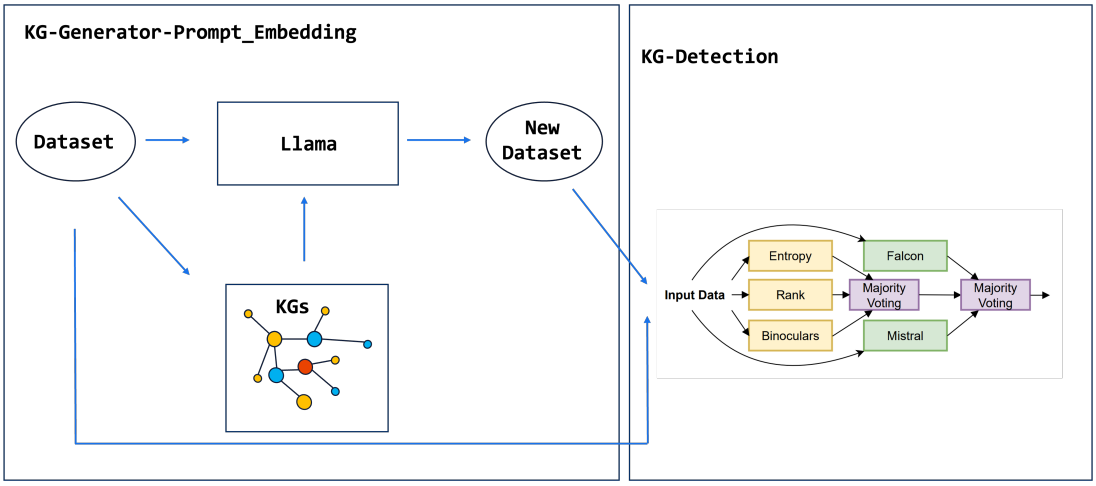

Figure 6: The Structure of Experimental Process With Prompt Embedding.

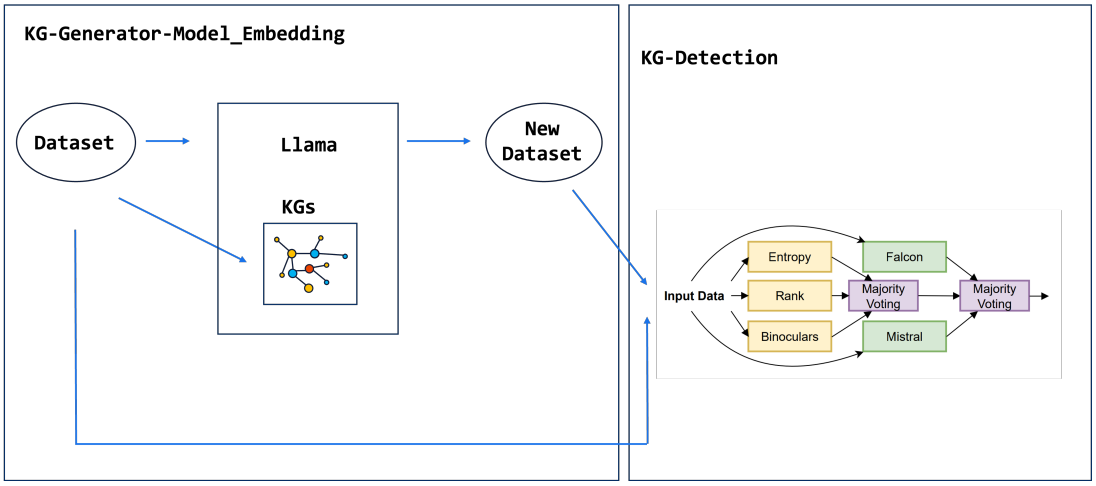

Figure 7: The Structure of Experimental Process With Model Embedding.

Devendra Singh Chaplot. 2023. Albert q. jiang, alexandre sablayrolles, arthur mensch, chris bamford, devendra singh chaplot, diego de las casas, florian bressand, gianna lengyel, guillaume lample, lucile saulnier, lélio renard lavaud, marie-anne lachaux, pierre stock, teven le scao, thibaut lavril, thomas wang, timothée lacroix, william el sayed. *arXiv preprint arXiv:2310.06825*.

OpenAI ChatGPT. 2022. Optimizing language models for dialogue. *OpenAI. com*, 30.

Jenny Chim, Julia Ive, and Maria Liakata. 2025. Evaluating synthetic data generation from user generated text. *Computational Linguistics*, 51(1):191–233.

Chao Feng, Xinyu Zhang, and Zichu Fei. 2023. Knowledge solver: Teaching llms to search for domain knowledge from knowledge graphs. *arXiv preprint arXiv:2309.03118*.

Tommaso Fornaciari, Leticia Cagnina, Paolo Rosso, and Massimo Poesio. 2020. Fake opinion detection: how similar are crowdsourced datasets to real data? *Language Resources and Evaluation*, 54(4):1019–1058.

Tommaso Fornaciari and Massimo Poesio. 2014. Identifying fake amazon reviews as learning from crowds. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 279–287. Association for Computational Linguistics.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Structgpt: A general framework for large language model

Model Pairs	t	p
Baseline And Prompt Embedding	-2.0217245674341413	0.054490135469486126
Baseline And Model Embedding	-2.381722425107768	0.02550892110468465

Table 3: Paired Sample T-Test Results: It includes two groups: Baseline and Prompt Embedding, Baseline and Model Embedding.

to reason over structured data. *arXiv preprint arXiv:2305.09645*.

Shahriar Khan. 2024. Chatgpt for writing literature and songs: End of the road for poets and songwriters? In *International IOT, Electronics and Mechatronics Conference*, pages 405–414. Springer.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR.

Ernests Lavrinovics, Russa Biswas, Johannes Bjerva, and Katja Hose. 2025. Knowledge graphs, large language models, and hallucinations: An nlp perspective. *Journal of Web Semantics*, 85:100844.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*, pages 24950–24962. PMLR.

Neo4j. 2012. Neo4j - the world's leading graph database.

Jeff Z Pan, Simon Razniewski, Jan-Christoph Kalo, Sneha Singhania, Jiaoyan Chen, Stefan Dietze, Hajira Jabeen, Janna Omeliyanenko, Wen Zhang, Matteo Lissandrini, et al. 2023. Large language models and knowledge graphs: Opportunities and challenges. *arXiv preprint arXiv:2308.06374*.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Manish Prajapati, Santos Kumar Baliarsingh, Chinmayee Dora, Ashutosh Bhoi, Jhalak Hota, and Jasaswi Prasad Mohanty. 2024. Detection of ai-generated text using large language model. In *2024 International Conference on Emerging Systems and Intelligent Computing (ESIC)*, pages 735–740. IEEE.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Vinu Sankar Sadasivan, Aounon Kumar, Sriram Balasubramanian, Wenxiao Wang, and Soheil Feizi. 2023. Can ai-generated text be reliably detected? *arXiv preprint arXiv:2303.11156*.

Michal Spiegel and Dominik Macko. 2024a. KInIT at SemEval-2024 task 8: Fine-tuned LLMs for multilingual machine-generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 558–564, Mexico City, Mexico. Association for Computational Linguistics.

Michal Spiegel and Dominik Macko. 2024b. Kinit at semeval-2024 task 8: Fine-tuned llms for multilingual machine-generated text detection. *arXiv preprint arXiv:2402.13671*.

Yanshen Sun, Jianfeng He, Limeng Cui, Shuo Lei, and Chang-Tien Lu. 2024. Exploring the deceptive power of llm-generated fake news: A study of real-world detection challenges. *arXiv preprint arXiv:2403.18249*.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. 2025. A survey on llm-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, pages 1–66.

Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. *arXiv preprint arXiv:2309.11206*.