# TeamSaarLST at the GEM'24 Data-to-text Task: Revisiting symbolic retrieval in the LLM-age

**Mayank Jobanputra** and **Vera Demberg**
`{firstname}@lst.uni-saarland.de`
Department of Language Science and Technology
Saarland University

## Abstract

Data-to-text (D2T) generation is a natural language generation (NLG) task in which a system describes structured data in natural language. Generating natural language verbalization for structured data is challenging as the data may not contain all the required details (here, properties such as gender are missing from the input data and need to be inferred for correct language generation), and because the structured data may conflict with the knowledge contained in the LLM's parameters learned during pretraining. Both of these factors (incorrect filling in of details, pretraining conflict and input data) can lead to so-called hallucinations.

In this paper, we propose a few-shot retrieval augmented generation (RAG) system, using a symbolic retriever – `PropertyRetriever`. Additionally, we experiment with state-of-the-art large language models (LLMs) to generate data verbalizations. Our system achieves the best results on 4 out of 6 subtasks for `METEOR` and `chrF++` metrics. We present our results along with an error analysis. We release our code for reproducing the results as well as the generated verbalizations from all the experiments for any further explorations here.[1]

## 1 Introduction

Nowadays LLMs are pretrained using trillions of text tokens[2] (Penedo et al., 2024). These LLMs can not only generate grammatical and fluent text, but they are also capable of learning new tasks without any training data using in-context learning techniques (Lampinen et al., 2022). One central challenge in LLMs research is to understand the extent to which LLMs memorize their training data versus how they generalize to new tasks and settings. There has been some empirical evidence that LLMs do some degree of both: they clearly memorize parts of the training data – for example, LLMs are often able to reproduce large portions of training data verbatim (Yu et al., 2023; Carlini et al., 2023) – but LLMs also seem to learn from this data, allowing them to generalize to new tasks. Do LLMs truly produce new content, or do they only remix their training data? Until we concretely answer this question, it is essential to test model faithfulness systematically through various data augmentation techniques.

The task of data-to-text generation is one of the popular NLG tasks. In this task, the system is given a set of RDF triplets describing facts (i.e., entities and relations between them) and has to produce a fluent text that is faithful to the facts. The GEM'24 (Mille et al., 2024) challenge brings forth a new shared task on data-to-text generation to test LLMs for factual information (i.e., information in the model parameters is likely to be in line with the input), vs. counterfactual information (i.e., the information in the prompt contrasts with what the model encodes about this entity) vs. fictional entities (i.e., the model parameters should not contain specific information supporting or contradicting the prompt information.)

The GEM'24 shared task consists of two subtasks of generating texts from input triple sets (*Subject* | *Property* | *Object*) in the WebNLG fashion. We participate in both the subtasks. One of the subtasks (D2T-1) is based on the WebNLG dataset. This subtask uses the official WebNLG test set[3] as input for testing the generation system. The test data contains 1,779 input triples with properties and entities not seen in the training/dev data. The second subtask (D2T-2) is based on the Wikidata. This subtask uses 1,800 newly compiled input triples from Wikidata for testing the generation system. Axelsson and Skantze (2023) proposed this dataset containing 74 new properties and entities,

[1] `https://github.com/mayankjobanputra/d2t-gem`

[2] `https://www.together.ai/blog/redpajama-data-v2`

[3] `https://huggingface.co/datasets/GEM/web_nlg`

*Proceedings of the 17th International Natural Language Generation Conference: Generation Challenges*, pages 92–99
September 23 –27, 2024. ©2024 Association for Computational Linguistics

which were not part of the WebNLG dataset.

In recent years, LLMs such as GPT-4 (Achiam et al., 2023), Gemini (Team et al., 2023) and LLaMa (Touvron et al., 2023), have made significant advancements in the field of natural language generation (NLG). However, the inherent tendency of these LLMs to generate inaccurate or non-factual content, commonly referred to as "hallucinations" (Puzikov and Gurevych, 2018; Ji et al., 2023), continues to present a significant challenge. This generally occurs because the model parameters from pretraining encode some information, which may "overwrite" the information in the prompt due to its high sequence probability. Another challenge with structured data is that the data does not contain all the required details such as entity type, gender and relation explanation. If the model fails to infer these details correctly, it may generate hallucinated verbalization.

In the literature, Shuster et al. (2021) suggests that providing relevant examples during inference can help in reducing hallucinations. While Moryossef et al. (2019) suggests using an explicit, symbolic, text planning stage for generating more faithful verbalization of data. In this work, we combine these suggestions and propose a few-shot RAG system to solve this task, using a symbolic retriever - `PropertyRetriever`. Figure 1 illustrates the architecture of our system. We experiment with state-of-the-art open-weight models for generating verbalization. In the following sections, we describe the dataset, our approach and provide a detailed study of the errors made by the system.

## 2 Dataset

The GEM'24 shared task introduced novel augmented test sets for both WebNLG and Wikidata. These augmented test sets consist of 3 parallel datasets as follows:

- **Factual (FA)**: This subset contains triples from the WebNLG and Wikidata datasets.
- **Counterfactual (CFA)**: This subset consists of swapped entities from the factual dataset. These entities are switched based on their class (i.e., a person entity is replaced by another person entity, a date by another date)
- **Fictional (FI)**: This subset consists of made-up entities, obtained via LLM prompting, in place of factual entities.

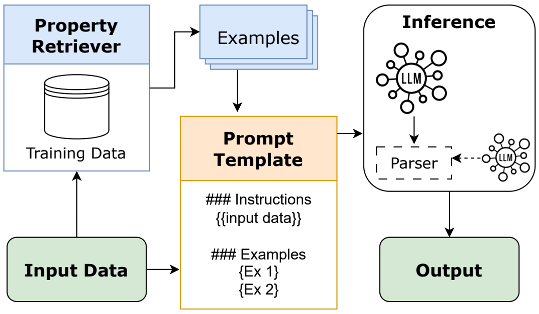

Figure 1: System architecture

Further details and example data for each subtask are available on the shared-task website[4].

## 3 Method

Our final system consists of a few-shot RAG pipeline that verbalizes the input data. In the following subsections, we describe the details of each component of our RAG pipeline.

### 3.1 Preprocessing

Our preprocessing step takes an RDF triple as input data and removes unnecessary information from it. For example, the input RDF triple contains the following header:
`<entry category="WikiData human", eid="Id1", shape="unknown", shape_type= "unknown", size="2">`

We realized that the `entry` header does not include any helpful information for verbalization. Hence, we remove it from the input data. We only keep the data between `<modifiedtripleset>` and `</modifiedtripleset>` tags.

### 3.2 PropertyRetriever

We observed that the verbalization mostly depends on the number of triples and the `Property` fields in the triple. The `Property` field should help in determining the correct verb and verb form. Let's consider the following example.

```
INPUT TRIPLE:

Baked_Alaska | country | France
Baked_Alaska | region | New_York
Baked_Alaska | ingredient | Christmas_pudding

VERBALIZATION:
```

[4] https://gem-benchmark.com/shared_task

```
Christmas pudding is an ingredient
in Baked Alaska, which comes from the
region of New York and the country
of France.
```

For the example above, the model needs to be able to connect all the generated sentences naturally and in a human-like manner. Moreover, it needs to infer the following details:

- *Baked_Alaska* is a food dish based on the property – *ingredient*.
- The property *ingredient* suggests that *Christmas pudding* is an ingredient in *Baked_Alaska*.

In the literature, it is shown that the model can learn to perform such inferences based on few-shot prompting (Lampinen et al., 2022) and retrieval augmented generation (Lewis et al., 2020). Generally, all RAG pipelines use a dense retriever to retrieve relevant samples from the training data. We started by building a dense retriever pipeline using `Haystack` (Pietsch et al., 2019) framework. The dense retriever failed to retrieve examples containing similar properties, especially for the Counterfactual and the Fictional datasets. We realized that this was due to the nature of the dense retriever which is trained to retrieve semantically similar examples. Most of the query input consists of the (*Subject*|*Object*) tokens. Hence, it retrieved examples that are more similar to the query *Subject* and the *Object* tokens.

To solve this issue, we take inspiration from Moryossef et al. (2019) and build a symbolic retriever – `PropertyRetriever`, that retrieves samples from the training based on the most similar properties. The retriever first creates an in-memory index of all properties from the training triples. At query time, it takes the properties of the input triples and returns the best-matching data points. Additionally, these best-matching data points are also selected in a way that the number of properties in the query data and the retrieved data are similar (i.e., shape matching). If no matching properties are found, then the retriever returns the random data points of the same shape. We observed such random sample returns for 130 test points in the WebNLG subtask.

Finally, we compared the verbalizations of 20 input triples using both `PropertyRetriever` and dense retrievers. We find that the samples from the `PropertyRetriever` helped LLMs generate better verbalizations compared to the dense retriever.

### 3.3 Prompt Engineering

We employ the prompting guidelines provided by the model publishers and Bsharat et al. (2023) for creating our few-shot prompt. We provide our final version of the few-shot prompt in Appendix A.1. Note that the final prompt is a template containing placeholders for the input data and retrieved examples, focusing majorly on task instructions. We use `Banks` (Pippi, 2023) to populate this prompt template with input data and the example data points dynamically at run time.

### 3.4 Inference

The main goal of our system is to generate suitable verbalization of the input data triples. For the same, we prompt the state-of-the-art LLMs, in a few-shot manner. We use `Mixtral 8x7B` and `Command-R` for all our experiments and compare their performance.

**`Mixtral 8x7B`**: Mixtral (Jiang et al., 2024) is a decoder-only sparse mixture-of-experts network where the feedforward block picks from a set of 8 distinct groups of parameters. At every layer, for every token, a router network chooses two of these groups (i.e., “experts") to process the token and combine their output additively. This technique increases the number of parameters of a model while controlling cost and latency, as the model only uses a fraction of the total set of parameters per token. Concretely, Mixtral only uses 12.9B parameters per token out of 46.7B total parameters.

**`Command R`**: Command-R is a 35 billion parameter decoder-only model. It is optimized for conversational interaction and long context tasks. It has been trained with the ability to ground its generations. This means that it can generate responses based on a list of supplied document snippets, and it will include citations in its response indicating the source of the information. This makes it a good candidate for RAG tasks.

**Implementation details**: We use Ollama[5] to run both `Mixtral`[6] and `Command-R`[7] models locally. We utilize 4-bit quantized versions of these models. We run all our experiments using 2x NVIDIA

[5] https://github.com/ollama/ollama
[6] https://ollama.com/library/mixtral
[7] https://ollama.com/library/command-r

RTX 3090s. The inference hyperparameters are provided in Table 1.

| | |
|---|---|
| seed | 5 |
| temperature | 0.5 |
| repeat_penalty | 1.2 |
| top_p | 0.9 |
| top_k | 25 |

Table 1: Inference Hyperparameters of LLMs

### 3.5 Postprocessing

We observe that both Mixtral and Command-R cannot follow the formatting instructions perfectly. Zhou et al. (2023) also made similar observations for GPT-4 and PaLM models. We also noticed that it is easier for these models to follow simpler formatting instructions than more complex ones. For example, we initially prompted models to generate verbalization in a JSON format, to which they often made small mistakes such as missing a closing bracket, a semicolon, or a closing quote. We then updated our formatting instructions to just keep the generated verbalization between <verbalization>, </verbalization> tags. After this change, Command-R always generated the verbalization in the correct format and Mixtral's formatting mistakes were reduced significantly.

We parse the model's responses by retrieving the text between <verbalization>, and </verbalization> tags. This way we detect the erroneous responses from both models. We discuss the error patterns in the error analysis section. Finally, we create an ensemble system with both Mixtral and Command-R. For the final output, we use the verbalization generated by the better-performing model (i.e., primary model) if our system can parse the response. Otherwise, we use the verbalization generated by the fallback model (i.e., secondary model) as the final output. We discuss the final choice of primary and secondary models in Section 5.1.

### 3.6 Evaluation

The system-generated text is assessed with reference-less automatic metrics such as BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), chrF++ (Popović, 2015), BERTScore (Zhang et al., 2020), and via human evaluation. The criteria for the human evaluation are the following:

- **Grammaticality**: The text is free of grammatical and spelling errors.
- **Fluency**: The text flows well and is easy to read; its parts are connected in a natural way.
- **No-Omissions**: All the information from the input data is present in the text.
- **No-Additions**: Only the information from the input data is present in the text.

## 4 Shared Task Results

The GEM'24 shared task organizers provide evaluation scores for all participating systems and subtasks using 4 metrics – BLEU, METEOR, chrF++ and BERTScore. These scores are calculated with 1 AMT reference text per data point. Our system achieves the best results on 4 out of 6 subtasks for METEOR and chrF++ metrics. For detailed results and comparison with participating systems, please refer to the overview literature (Mille et al., 2024).

## 5 Performance Analysis

In this section, we provide the results of our human evaluation study. We conduct this study to finalize our primary and secondary model for the final system. We also discuss observed error patterns during the evaluation.

### 5.1 Human Evaluation

We conduct a human evaluation study ourselves on a small subset of 40 input triples. These triples are collected from Counterfactual and Fictional datasets. We apply filtering based on our observation that the models generate better verbalization for factual and smaller input triples. Hence, the filtered triples contain more than 3 properties each.

We use the same evaluation criteria mentioned in Section 3.6. We ask our human annotator to rate the model's response on each criterion based on our evaluation guidelines (refer to Appendix A.2). We report the results of this study in Table 2. The results indicate that Mixtral performs better compared to Command-R.

### 5.2 Error Analysis

We dive deeper into the human evaluation study to figure out exact error patterns. We discuss two of the most commonly observed issues here.

```
INPUT TRIPLE:

What_Ever_Happened_to_Baby_Jane? | publisher | Gruppo_Mondadori
What_Ever_Happened_to_Baby_Jane? | followedBy | I_Am_a_Cat
What_Ever_Happened_to_Baby_Jane? | author | Horst_Köhler
What_Ever_Happened_to_Baby_Jane? | releaseDate | 1726-01-01

GENERATED VERBALIZATION:

The publisher of What Ever Happened to Baby Jane is Gruppo Mondadori. Its author is Horst
Köhler, and it was released on January 1, 1726. Following What Ever Happened to Baby Jane
is I Am a Cat.
```

Figure 2: Example of imperfect verbalization

| Criteria | Mixtral | Command-R | Max Score |
|---|---|---|---|
| Fluency | 110 | 105 | 120 |
| Grammaticality | 113 | 104 | 120 |
| No Omissions | 39 | 35 | 40 |
| No Additions | 39 | 38 | 40 |

Table 2: Human evaluation scores for the Mixtral and Command-R models.

### 5.2.1 Fluency issues

We speculate that the fluency issues majorly arise due to the unknown entity type. We provide an example of such an instance in Figure 2. In this case, it is evident that the entity name "What_Ever_Happened_to_Baby_Jane?" or properties (publisher, followedBy, author, releaseDate) may not help in identifying the entity type. Here, the entity may be a movie, book, or literary article. While humans may be able to infer the entity type, we observed cases where LLMs fail to infer entity type or gender from the properties.

### 5.2.2 Formatting issues

The other major error we observed was that both Mixtral and Command-R can add extra tokens at the beginning of their responses. The most commonly observed beginning tokens for Mixtral are: "It is mentioned that" and for Command-R: "Modified tripleset:". Further, we observe that Mixtral fails to follow the formatting instructions for almost 800 instances out of 1800 total instances. For these 800 instances, we could not extract the verbalization based on our postprocessing steps.

Based on the human evaluation and error analysis results, we choose Mixtral as our primary model and Command-R as the secondary model.

## 6 Conclusion

In this paper, we describe our solution for the data-to-text generation shared task. We propose a symbolic retriever method – PropertyRetriever, to retrieve better examples for Data-to-text generation problems. We further explore the capabilities of two state-of-the-art LLMs, Mixtral and Command-R. Combining the insights from our human evaluation study and error analysis, we propose an ensemble system as our final solution.

In the future, we would like to explore multi-turn correction and planning approaches. We believe such approaches may allow the model to self-correct its formatting errors and generate verbalizations with better fluency.

## Limitations

Our findings require further experimentation on more datasets since we only test our approach on the GEM'24 shared task datasets. We also did not optimize the prompt for each model separately. Optimizing prompts individually for each model can lead to better results. Further, we also use the quantized version of the LLMs which may have affected the accuracy. Our comparison of the 20 samples for deciding between the dense retriever and PropertyRetriever can be further improved by doing a more systematic study. Lastly, our human evaluation study was conducted on a very small subset. For more reliable results, we suggest conducting the human evaluation study on a larger subset.

## Ethical Considerations

The human evaluation study presented in this work is carried out by a student assistant at the university. They were paid fairly as per the university payment standards. We also advise evaluating our methods on larger validation sets before using them for other domains and datasets.

## Acknowledgments

This research was funded by DFG grant 389792660 as part of TRR 248 – CPEC.[8] We sincerely thank the anonymous reviewers for their insightful comments that helped us to improve this paper. We also thank Sneha Chetani for helping us evaluate the model outputs.

## References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*.

Agnes Axelsson and Gabriel Skantze. 2023. Using large language models for zero-shot natural language generation from knowledge graphs. In *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, pages 39–54, Prague, Czech Republic. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Sondos Mahmoud Bsharat, Aidar Myrzakhan, and Zhiqiang Shen. 2023. Principled instructions are all you need for questioning LLaMA-1/2, GPT-3.5/4. *arXiv preprint arXiv:2312.16171*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. In *ACM Comput. Surv.*, volume 55, New York, NY, USA. Association for Computing Machinery.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Andrew Lampinen, Ishita Dasgupta, Stephanie Chan, Kory Mathewson, Mh Tessler, Antonia Creswell, James McClelland, Jane Wang, and Felix Hill. 2022. Can language models learn from explanations in context? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 537–563, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

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.

Simon Mille, João Sedoc, Yixin Liu, Elizabeth Clark, Agnes Axelsson, Miruna-Adriana Clinciu, Yufang Hou, Saad Mahamood, Ishmael Obonyo, and Lining Zhang. 2024. The 2024 GEM shared task on multilingual data-to-text generation and summarization: Overview and preliminary results. In *Proceedings of the 17th International Conference on Natural Language Generation: Generation Challenges*, Tokyo, Japan. Association for Computational Linguistics.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2267–2277, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, Thomas Wolf, et al. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*.

Malte Pietsch, Timo Möller, Bogdan Kostic, Julian Risch, Massimiliano Pippi, Mayank Jobanputra, Sara Zanzottera, Silvano Cerza, Vladimir Blagojevic, Thomas Stadelmann, Tanay Soni, and Sebastian Lee. 2019. Haystack: the end-to-end NLP framework for pragmatic builders.

[8] `https://perspicuous-computing.science`

Massimiliano Pippi. 2023. Banks: the linguist professor who will help you generate meaningful prompts.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Yevgeniy Puzikov and Iryna Gurevych. 2018. E2E NLG challenge: Neural models vs. templates. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 463–471, Tilburg University, The Netherlands. Association for Computational Linguistics.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: A family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Qinan Yu, Jack Merullo, and Ellie Pavlick. 2023. Characterizing mechanisms for factual recall in language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9924–9959, Singapore. Association for Computational Linguistics.

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

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

# A Appendix

## A.1 Final Prompt

```
###Instruction###: just verbalize the
following data without beginning prompt
in a natural, human-like manner.

###Data### : {{ data }}

###Criteria###: Follow these criteria
carefully:
1. Keep the generated sentences in a flow
and the generated text should sound
human-like.
2. Copy the entities correctly from the data.
3. Replace '\_' with a space.
4. Use the punctuation marks correctly
without any extra spaces.

You may look at the following examples for
the writing style, but only for style.
Do not copy anything from the following
examples, otherwise you will be penalized.

###Examples###:
Ex-1: {{ ex_1 }}
Ex-2: {{ ex_2 }}
Ex-3: {{ ex_3 }}

###Important Notes###:
1. The verbalization output MUST only
contain the verbalization of ###Data###
in a natural, human-like manner.
2. Ensure that generated ###Data###
verbalization MUST be between
<verbalization> and </verbalization> tags.
```

## A.2 Human Annotation guidelines

In this task, the model is given data triplets, where each triple is made of *Subject* | *Property* | *Object* and is asked to verbalize this data in a natural, human-like manner.

We need your help to evaluate the model responses based on the following criteria:
**Grammaticality**: The text is free of grammatical and spelling errors.
**Fluency**: The text flows well and is easy to read; its parts are connected in a natural way.
**No-Omissions**: ALL the information in the table is present in the text.
**No-Additions**: ONLY information from the table is present in the text.

We would like you to focus the most on Fluency and Grammaticality. No-Omissions and No-Additions are binary criteria.
**Grammaticality** (1-3 scale):

- 1 (Low): The response contains severe grammatical errors that significantly hinder understanding. This may include missing words, subject-verb disagreement, incorrect verb tenses, or nonsensical sentence structure.

- 2 (Medium): The response may contain some grammatical errors, but they are not so severe as to completely obscure the meaning. These errors might include misuse of articles ("a," "an," "the") or prepositions, or minor subject-verb agreement issues.
- 3 (High): The response is free of grammatical errors and adheres to the rules of English grammar.

**Fluency** (1-3 scale):

- 1 (Low): The response is difficult to read due to awkward phrasing, choppy sentence structure, or lack of variety. It may sound unnatural or unclear.
- 2 (Medium): The response reads mostly smoothly, but there may be occasional awkward phrasing or clunky sentences.
- 3 (High): The response reads effortlessly and sounds natural. The sentences are well-constructed and varied, and the overall flow of ideas is clear and logical.

**No-Omissions** - Please choose 0 to indicate Missing Information or 1 to indicate No Missing Information.

**No-Additions** - Please choose 0 to indicate Extra Information/Hallucinated Information or 1 to indicate No Extra Information.