

Assessing the importance of knowledge graph formatting in improving LLM answer accuracy using data augmentation

Sven Leo Kupres, Marko Horvat, Marko Delimar

University of Zagreb Faculty of Electrical Engineering and Computing

Department of Applied Computing, Department of Applied Computing, Department of Energy and Power Systems

Unska 3, HR-10000 Zagreb, Croatia

{Sven-Leo.Kupres, Marko.Horvat, Marko.Delimar}@fer.unizg.hr

Abstract. While Large Language Models (LLMs) offer intuitive natural language interfaces, producing falsehoods limits their reliability. Therefore, this study examines which abstraction level and data format optimize question accuracy and scalability, as well as asserts how Knowledge Graphs (KGs) can improve LLM responses through a neuro-symbolic approach. Using synthetic datasets based on the FOAF ontology, six query attributes per entity were evaluated through in-context prompting and supervised fine-tuning. Results reveal an accuracy-complexity trade-off: KGs improve factual precision but increase token consumption and preprocessing demands. The findings provide general guidelines for data augmentation and balancing symbolic representation with knowledge scalability.

Keywords. Large language models, Knowledge graphs, Ontologies, Knowledge representation, Data augmentation, Natural language processing

1 Introduction

Despite the remarkable capabilities of Large Language Models (LLMs) in natural language understanding and generation, their inherent limitations, most notably factual inconsistencies and interpretability challenges, impede their reliable deployment in specialized domains (Yang et al., 2024). While LLMs offer natural language interaction, extensive knowledge coverage and generalized reasoning capabilities, their architecture intrinsically lacks mechanisms to entirely prevent the fabrication of false information, leading to persistent hallucination errors (Chen, 2023).

This reliability gap necessitates the exploration of neuro-symbolic AI architectures that combine the generative power of LLMs with the structured precision of knowledge graphs (KGs) (Hogan et al., 2021). Knowledge graphs are particularly effective at representing factual information as well as complex, relational properties between entities. By grounding generative outputs in verifiable sources, KGs offer deterministic reasoning paths and significantly reduce the likelihood of

hallucinations by establishing a logic model that can be efficiently traced, providing AI reliability and explainability (Tilwani et al., 2024; Dehal et al., 2025).

Frameworks such as Retrieval-Augmented Generation (RAG) (Zhao et al., 2026) have been instrumental in this integration, helping to keep generation focused by providing external context. However, the efficiency of these augmentation techniques relies heavily on the quality, structure, and formatting of the provided data. Combining knowledge graphs within LLM architectures introduces critical complexities regarding computational overhead, token limit constraints, and the dynamic updating of knowledge bases (Lin et al., 2025). Therefore, it remains important to assess how the specific knowledge representation format of the augmented data affects the model's reasoning capabilities.

This research examines the potential benefits of integrating large language models with knowledge graphs in the context of neuro-symbolic AI to improve data augmentation methods, while addressing the associated increases in token consumption and data preprocessing demands (Pan et al., 2024; Škrlj et al., 2025). Specifically, different levels of abstraction in knowledge representation, coupled with various data formats are explored as well as how they influence the accuracy and scalability of neuro-symbolic LLM architectures. Knowledge representations at various abstraction levels are described, especially those comparing numeric integers versus human-readable full names as entity identifiers. Different knowledge representation formats, including standardized JSON, XML, and RDF schemes based on the FOAF ontology (Kalemi & Martiri, 2011), are tested, as well as unstructured natural language (Choudhary & Reddy, 2023; Dai et al., 2025; Narvekar et al., 2025; Wu & Tsioutsoulouklis, 2024).

By comparing both in-context system prompting and supervised fine-tuning approaches, this systematic exploration aims to bridge the gap between the generative capacity of LLMs and the structured precision of KGs (Lavrinovics et al., 2024; Yang et al., 2025).

To achieve this, the present paper examines the characteristics of augmentation data used in LLM pipelines,

identifying the structural issues that lead to suboptimal representation, and analyzes how such inputs can be optimized to reduce incorrect answers and hallucinations. Our experimental dataset uses a manageable smaller-scale knowledge graph built on the standardized FOAF ontology and includes entities with heterogeneous properties. Importantly, the LLM's ability to deduce multi-hop relational connections that a human would intuitively make is assessed, determining the extent to which the specific data format influences the model's reasoning capability. Finally, augmentation of the model's context is directly compared to supervised fine-tuning over identical data, providing a comprehensive discussion of the operational benefits, drawbacks, and computational trade-offs of each method.

2 Related Work

Recently published research extensively documents the contradictory nature of LLMs: while they have remarkable natural language capabilities, they suffer from persistent factual inconsistencies. Interpretability challenges as the primary barriers to deploying LLMs in specialized domains were identified, emphasizing that fact-aware language modeling is required (Yang et al., 2024). Furthermore, it was emphasized that current generative architectures do not have the mechanisms required to prevent hallucination errors (Chen, 2023). This limitation is most visible in the LLMs inability to reliably recall long-term or long-tail knowledge, an area where generative models consistently fail without explicit external grounding (Kandpal et al., 2023). As a result, improving factuality through external verification has become an important priority in developing trustworthy large knowledge models.

To bridge this reliability gap, researchers are increasingly turning to neuro-symbolic AI, which combines neural generation and structured symbolic logic. The mutually beneficial relationship between LLMs and knowledge graphs was investigated, demonstrating the effectiveness of structured conceptualization in supporting AI results (Jaldi et al. 2025, Dehal et al. 2025). This claim was confirmed by demonstrating how neuro-symbolic approaches provide deterministic, traceable reasoning paths that improve explainability (Tilwani et al., 2024). A roadmap for the implementation of this synergy was proposed, emphasizing the trade-offs between neural flexibility and symbolic rigidity. While KGs reduce hallucinations, their integration requires significant computational complexity and data preprocessing (Pan et al., 2024, Škrlić et al., 2025).

The Retrieval-Augmented Generation paradigm was unequivocally established as the most important framework for addressing knowledge-intensive natural language processing tasks (Lewis et al., 2020). The RAG has rapidly become the accepted operational standard for LLM-KG integration (Arslan et al., 2024; Zhao

et al., 2026). While recent advances, such as Active Retrieval Augmented Generation (Jiang et al., 2023), attempt to iteratively query external sources to verify mid-generation tokens, the efficacy of augmenting LLMs is significantly limited by context window limits and the structural formatting of the provided data (Lin et al., 2025).

As a result, the impact of data formats and prompt structures on LLM comprehension has emerged as a distinct and critical research topic. Ayala and Bechard explicitly demonstrate that maintaining structured outputs in RAG pipelines depends extensively on the retrieval format, which has a direct impact on hallucination rates. This dependency is thoroughly supported by some researchers (He et al., 2024), who systematically investigate prompt formatting across plain text, Markdown, JSON, and YAML, revealing that model performance can vary by up to 40 % simply by changing the structural template (Ayala and Bechard, 2024). This research was expanded on how to leverage these formats by providing a comprehensive evaluation of prompt engineering techniques such as RAG and In-Context Learning (ICL), demonstrating that the optimal strategy is highly dependent on the specific LLM architecture and the structure of the target dataset (Son, Won, and Lee, 2025). Also, complex logical reasoning over KGs have been investigated (Dai et al., 2025, Choudhary & Reddy, 2023) and it was concluded that structured data schemas influence reasoning accuracy (Wu & Tsioutsoulis, 2024). Importantly, the effects of input formats on model performance were compared, and a preference for JSON format for both large and small models was demonstrated. Importantly, smaller LLMs, which have more limited capacity and context windows than their larger counterparts, benefit more significantly from structured input (Narvekar et al., 2025). Also published research shows that the FOAF ontology (Kalemi & Martiri, 2011) and other foundational semantic web vocabularies continue to be part of established standards for representing and evaluating these structured relational graphs.

Finally, the field of neuro-symbolic AI is still actively working to determine the best operational method for integrating structured knowledge. Comprehensive surveys to assess the differences between supervised fine-tuning and in-context system prompting methodologies have been conducted (Lavrinovics et al., 2025, Yang et al., 2025). It was shown that both approaches seek to combine the generative capacity of LLMs with the structured precision of KGs, but they have different operational trade-offs with regard to scalability and overhead.

The overview of recent literature shows that optimizing data augmentation requires a careful balance between symbolic representation format and computational scalability, which this study attempts to systematically evaluate

3 Methodology

The following subsections explain in detail the methodology used throughout the research process in order for them to be reproducible. The first subsection explores the synthetic dataset created for the purposes of this research. The second subsection deals with the model and exactly how the data was used to augment it. Finally, the third subsection outlines in detail the evaluation framework employed to evaluate the generated responses, incorporating the necessary qualitative analysis to accurately measure hallucination rates and reasoning efficacy.

3.1 Synthetic Dataset

The utilized dataset models a person entity and their relationships. A person entity has a first name (string), last name (string), age (integer), gender (enumerator string, either discrete value "male" or "female") and a list of friends (list of other people), where "list" refers to the relationship of a person entity to multiple other entities.

For such a conceptual model, several ways were constructed to present the information to the LLM. The simplest format to model information is through natural language. Naturally, English was selected, as it is a logical choice considering most models were trained on predominantly English data and therefore work in conjunction with it. For instance: "Nada Sudar is 19 years old." or "Zoran Goreta is friends with Nada Sudar.", where "Nada Sudar" and "Zoran Goreta" represent person entities, "Nada" and "Zoran" are the respective first names of persons, while "Sudar" and "Goreta" are their respective last names.

This format displays data the same way two people would display it to one another during interpersonal communication, however dry or simple it may be. A more complex approach is to encapsulate these same sentences within a JSON structure. Each natural language sentence would be a "statement", and all of them together are "statements". An example is depicted in Fig. 1.

A different approach is reducing language complexity and treating data as simple properties. A person has properties and those properties have their own values, e.g. the value of a property "age" is an integer larger than zero (excluding newborn children). However, as will be explained later in more detail, during a person's age generation the number generated could not be smaller than six.

To return to a previous example, the person "Nada Sudar", like all people, has the property "age", and the value of the property is e.g. 25. XML and JSON markup languages are utilized with the exact purpose to save data with such formatting (World Wide Web Consortium, 2006; Bassett, 2015). These two languages are popular and easy to use, with many free tools for

```
"statements": [
  {
    "statement": "Saša Škrtić is 44 years old."
  },
  {
    "statement": "Saša Škrtić is a male."
  },
  {
    "statement": "Saša Škrtić and Zoran Goreta are friends."
  },
  {
    "statement": "Saša Škrtić and Nikola Prelošnjak are friends."
  },
  {
    "statement": "Saša Škrtić and Monika Rimac are friends."
  },
  {
    "statement": "Saša Škrtić and Mirjam Devčić are friends."
  },
  {
    "statement": "Saša Škrtić and Antonija Salinović are friends."
  },
  {
    "statement": "Saša Škrtić and Cvijeta Svetličić are friends."
  },
  {
    "statement": "Duje Ravlić is 18 years old."
  },
  {
    "statement": "Duje Ravlić is a male."
  }
]
```

Figure 1. Example of the JSON format which encapsulates natural language. One can see that each separate "statement" is grouped into the list "statements".

parsing and building expressions, and therefore both were used throughout this research.

Finally, inspired by the Friend-of-a-friend ontology (FOAF) (Kalemi & Martiri, 2011), the standardized RDF/XML format was also used. FOAF is a vocabulary which describes people and their relationships to other concepts, like their friends, their interests and their creations. It also describes other people's properties like age, gender, home location and appearance.

Each data point, or rather a person entity, must have a unique identifier in order to be separated from other people, as well as making it possible to be referenced when talking about a person's friends. All three of the above-listed structured formats (i.e., XML, JSON, and RDF) had two variations each. The first variation used people's full names as IDs. However, this, of course, is not a good practice in real applications since different people can share a full name, but it was made impossible for the purposes of this research. The second variation used integers as identifiers. This was most relevant for listing friends, since the person's identifier would be used as the reference for another friend entity. To go back to the previous example, if "Zoran Goreta" and "Nada Sudar" were friends, "Zoran Goreta"'s "friend" attribute could either reference "Nada Sudar" or simply "1", which ever was the person's unique identifier for the variation. An example of these differences between variations is depicted in Fig. 2.

During data generation, the first step is to decide on the number of data points, or rather, the number of people entities it consists of. Afterwards, for each person entity the data was generated in the following steps, the order not necessarily important:

```

<Humans>
  <Human id="0">
    <name>Saša</name>
    <surname>Škrtić</surname>
    <age>44</age>
    <gender>MALE</gender>
    <friends>
      <friend ref="3"/>
      <friend ref="9"/>
      <friend ref="11"/>
      <friend ref="14"/>
      <friend ref="16"/>
      <friend ref="17"/>
    </friends>
  </Human>
  <Human id="1">
    <name>Duje</name>
    <surname>Ravlić</surname>
    <age>18</age>
    <gender>MALE</gender>
    <friends>
      <friend ref="2"/>
      <friend ref="5"/>
      <friend ref="7"/>
      <friend ref="8"/>
      <friend ref="9"/>
      <friend ref="14"/>
      <friend ref="15"/>
      <friend ref="17"/>
    </friends>
  </Human>
</Humans>

<Humans>
  <Human id="Saša Škrtić">
    <name>Saša</name>
    <surname>Škrtić</surname>
    <age>44</age>
    <gender>MALE</gender>
    <friends>
      <friend ref="Nikola Prelošnjak"/>
      <friend ref="Cvijeta Svetličić"/>
      <friend ref="Monika Rimac"/>
      <friend ref="Zoran Goreta"/>
      <friend ref="Antonija Salinović"/>
      <friend ref="Mirjam Devčić"/>
    </friends>
  </Human>
  <Human id="Duje Ravlić">
    <name>Duje</name>
    <surname>Ravlić</surname>
    <age>18</age>
    <gender>MALE</gender>
    <friends>
      <friend ref="Lea Grubić"/>
      <friend ref="Nikola Prelošnjak"/>
      <friend ref="Gordan Lucić"/>
      <friend ref="Vladimir Svalina"/>
      <friend ref="Mirjam Devčić"/>
      <friend ref="Eduard Tutić"/>
      <friend ref="Tomislav Šoštarić"/>
      <friend ref="Cvijeta Svetličić"/>
    </friends>
  </Human>

```

Figure 2. Comparison of the XML format variations. Top image: numeric IDs; bottom image: full names as IDs.

1. People's first and last names were created using an online generator.
2. People's age was sampled from a discrete uniform distribution $Unif(6, 70)$. It was selected arbitrarily because the expectation is that a distribution with a smaller or larger interval would have little impact on the results.

3. People's gender was decided before name generation, so as to have an equal amount of male and female people.
4. People's friends were generated by sampling from a continuous uniform distribution $U_{[\frac{1}{4}, \frac{1}{3}]}$ of all friend combinations. The distribution was selected in order for the number of friendships in a dataset to not be predictable and so the number of friendships couldn't be too large. "Friendship" is symmetrical - if one person is friends with another, then the opposite is also always true. "Friendship" also isn't transitive - a person entity who has a friend is not also necessarily a friend of all their friends. A person cannot be a friend to themselves.

In conclusion, eight different formats were utilized in the experiment: 1) natural English language, 2) natural English language encapsulated in JSON, 3) XML with numbers as IDs, 4) XML with full names as IDs, 5) JSON with numbers as IDs, 6) JSON with full names as IDs, 7) RDF with numbers as IDs, and 8) RDF with full names as IDs. For each of these eight described formats, four datasets were constructed, each with a different number of people. In order, the datasets described six, ten, fourteen and twenty people to evaluate how scalable each of the formats is.

3.2 Used model and data integration

The model used was Meta's Llama-3.2-3B-Instruct (meta-llama/Llama-3.2-3B-Instruct, 2024). A simple model, it was just small enough to run on the hardware, but has three billion parameters, which is enough to acquire meaningful results. It is a great baseline model to prove a concept which can then be expanded with a more advanced model with more parameters. The model therefore allows the results to be reproducible.

Two methods were attempted in how a model could access information. The first method was inputting the data into the model's system prompt. Henceforth such models will be referred to in the text as "augmented models" because they are augmented by the data in the system prompt.

The second method was fine-tuning the model on the data. Each data format was inputted or fine-tuned and then evaluated independent of each other. The fine-tuning process was done over fifteen epochs, with a learning rate of 5×10^{-4} . In both cases the model's parameter precision was reduced in order to accommodate running the tests. Other model hyperparameters were never manipulated.

The augmented models received an additional system prompt after the data was given: "Answer questions about this data in as few words as possible.". A similar system prompt was also given to the fine-tuned models: "Answer questions in as few words as possible."

The main point of using both methods is to compare the complexity of the two methods and their respective strengths and weaknesses. The methods differ significantly, with augmentation being relatively simple to accomplish and automate, and fine-tuning requiring more time to perform successfully. Not as much time was dedicated to perform optimal fine-tuning, as the hyperparameters were selected by only applying general good practice rules. This gives an insight into the trade-offs of each method.

3.3 Evaluation process

Each format was evaluated on the same set of questions. For each person in the dataset, six questions were asked:

- What is [full_name]'s first name?
- What is [full_name]'s last name?
- What is [full_name]'s age?
- What is [full_name]'s gender?
- List all [full_name]'s friends.
- How many friends does [full_name] have?

The model's full answer was recorded and stored for subsequent analysis. The answers were human-evaluated since in some instances the model runs out of tokens and cuts the answer short or adds irrelevant information after the question was answered, thus making automatic evaluation unreliable.

Model's response would be classified in one of three categories: "true", "false" or "half true". A "true" answer could contain additional irrelevant information as long as it was not false or contradictory. A "half true" answer could mean multiple things, but it usually meant an incomplete answer. For instance, the model could run out of tokens while generating the list of friends, but all the names it generated were correct. Otherwise, "half true" or "partially true" answers could be ones that provided technically correct information which likely resulted from a lack of specificity in the system prompt, e.g. the model was not explicitly instructed to present the list of friends as their full names, so it returned a list of numbers representing the people's corresponding IDs. However, if the model proceeds to simply output the same data which it was given (JSON or XML structure for example), that is impractical, illegible, and undesired for a human and therefore considered false. Any answer that could be interpreted as misleading was marked as "false", for instance if a person has no friends and a model responds that it does not possess any information about that person's friends. An important part of the evaluation is that a "half true" answer is not quantifiable as 50% of a correct or incorrect answer, but more so an answer that can not be considered either "true" or "false" and should be viewed as its own independent category.

In conclusion, the first set of experiments consists of four separate datasets: 6-, 10-, 14- and 20-person entity dataset. Each dataset evaluation was performed with all eight described formats. The LLM was asked six questions about each person in the dataset. Equation 1 calculates the total number of data points after evaluation for the first set of experiments.

$$8 * 6 * (6 + 10 + 14 + 20) = 2400 \quad (1)$$

Equation 2 calculates the total number of data points for the second set of experiments. Only three of the four experimental datasets were utilized, with the 20-person entity dataset left out. Additionally, only 7 formats were evaluated, with the format "natural.json" left out.

$$7 * 6 * (6 + 10 + 14) = 1260 \quad (2)$$

4 Results

The following subsections showcase the results. Subsection 4.1 examines into how many tokens each data format has been vectorized and establishes whether a direct link exists between how many tokens a format contains and its success rate. Subsection 4.2 compares fine-tuning and augmenting the model on the same data, the strengths and weaknesses of each approach. The following formats deal only with the augmented format; subsection 4.3 explores the differences between formats, mainly establishing a difference between the formats using natural language and structured text. The final subsection 4.4 looks into the variations of structured formats and how slight changes can affect results.

4.1 Token number per format

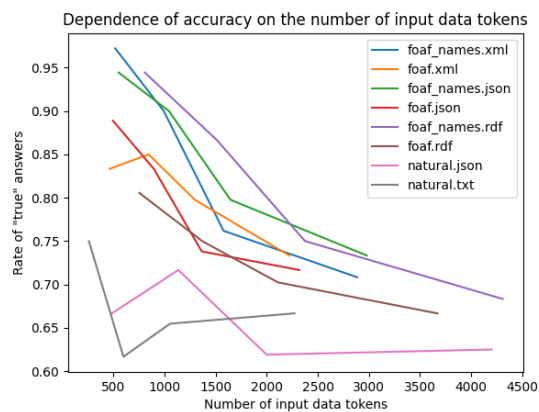
A potential difference between models that must first be ruled out is if the model's success simply depends on the number of tokens the format is encoded into. In Table 1 one can see how many tokens each of the formats was encoded into. Logically, the more people are present in the format, the more tokens are required. This growth is not purely linear due to how the friends list was constructed, as described in Subsection 3.1.

While the formats comprised of natural language sentences "natural.txt" and "natural.json" start with the least tokens when there are only six people, that amount grows at a quicker rate than the other "structured" formats.

Fig. 3 shows the correlation between the accuracy of the augmented model and the number of tokens the format is encoded into. While it is obvious that each individual format trends downwards, comparing formats leads to a conclusion that data structure does in fact matter and the accuracy is not simply completely dependent on the number of tokens, although it likely plays a role considering more tokens means the LLM has more context to process.

Table 1. Number of tokens for each combination of formats and data points (number of people).

	6	10	14	20
natural.txt	261	599	1055	2272
natural.json	486	1136	2000	4201
foaf.xml	464	845	1297	2224
foaf_names.xml	519	993	1578	2885
foaf.json	496	896	1365	2319
foaf_names.json	551	1044	1646	2980
foaf.rdf	754	1369	2111	3670
foaf_names.rdf	807	1512	2375	4308

**Figure 3.** Correlation between the accuracy of the augmented model and the number of tokens used for data encoding.**Table 2.** Augmented model accuracy for each combination of formats and number of people entities.

	6	10	14	20
natural.txt	75.00	61.67	65.48	66.67
natural.json	66.66	71.67	61.90	62.50
foaf.xml	83.33	85.00	79.76	73.33
foaf_names.xml	97.22	90.00	76.19	70.83
foaf.json	88.89	83.33	73.81	71.67
foaf_names.json	94.44	90.00	79.76	73.33
foaf.rdf	80.56	75.00	70.24	66.67
foaf_names.rdf	94.44	86.67	75.00	68.33

4.2 Difference between fine-tuned and augmented models

Fine-tuned models had an overall worse accuracy than augmented models, as presented in Table 2 and Table 3. In theory such a result is expected since fine-tuning the model requires it to fit in the new data with its already existing knowledge. The training hyper-parameters were likely not properly selected as well. However, that is a case against fine-tuning models for such tasks since it requires more work to find optimal hyper-parameters and achieve comparable results

(Hoffmann et al., 2022). The model in this case was likely under-fitted. If the model were over-fitted on the data, it would "forget" the data it was pretrained on, making it only useful for a specific dataset and losing its ability to generalize. Additionally, the process would have to repeat every time the data was updated, making its scalability poor. However it is worth noting that the execution time for fine-tuned models was significantly faster since the model did not have to process the large system prompt with each query. When measured, the time it took the augmented model to execute all the questions for the "foaf.xml" format, dataset of 10 people, was 302.5 seconds. In comparison, the measured time for the same format when executed by the fine-tuned model was 19.9 seconds, and the other execution times are comparable. This execution time will of course stay largely the same as the dataset grows, while being able to execute augmented model queries with larger datasets requires chunking the data. The experiment execution times for datasets with 6, 10, 14 and 20 people on the "foaf.xml" format are respectively: 121.8, 302.5, 392.9 and 750.0 seconds. Of course, the number of queries per experiment also increases, but even accounting for that, the times per question are: 3.38, 5.04, 4.68, 6.25 seconds.

Table 3. Fine-tuned model accuracy for each combination of formats and number of people entities.

	6	10	14
natural.txt	50.00	43.33	45.24
natural.json	50.00	46.67	46.43
foaf_names.xml	50.00	48.33	48.81
foaf_names.json	50.00	48.33	46.43
foaf_names.rdf	50.00	43.33	46.43

4.3 Difference between formats for augmented models

Per Table 2, there is a noticeable difference in accuracy between "natural.txt" and "natural.json", formats which use natural language, and the more structured formats. The problem arises from the fact that the natural language format accuracy is not as heavily impacted by the increase in dataset size, which can also be seen in Fig. 3. However, the structured format accuracy is still overall superior, as well as the data token amount (which means the computing speed is also faster).

When considering the formats "foaf.json", "foaf.xml" and "foaf.rdf", the first two formats are quite similar when it comes to accuracy, but "foaf.xml" scales better in terms of token amount, as per the Table 1. "foaf.rdf" format is by far the worst of these three - the smallest dataset has the highest amount of tokens of all the formats and scales faster than any other format (besides "foaf_names.rdf", but that is a topic for later). Additionally, it displays poor

accuracy compared to "foaf.json" and "foaf.xml". Even considering the format sees standard use for describing graph data and the model's pretraining data could have included such a format, the results show that building a proprietary lighter-weight framework leads to improved performance.

4.4 Difference between the two format variations

As described in Subsection 3.1, each person is assigned a unique identifier. Changing it from a unique number to the person's full name increases the size of the text and therefore the token amount: 11.85% for the "xml" format, 11.09% for the "json" format and 7.03% for the "rdf" format. This also means using numbers is more scalable, as with a dataset size increase identifiers would have to be used more. However, using names as identifiers is more legible for humans and as it turns out - somewhat for LLM-s as well. Using names as identifiers leads to an improvement in accuracy for all three aforementioned formats (with a caveat being a dataset of 14 and 20 in case of the "foaf_names" format, which defies explanation as of yet). The improvement in accuracy can actually be explained very well by the answers the model gave to a specific question: "List all [full_name]'s friends". The model would usually return the answers using the same format that it found them in - it would normally return full names when using full names as identifiers and return numbers when using numbers as identifiers. The validity of such answers is normally up to the user, but the grading process for this paper was explained in Subsection 3.3.

According to Table 4, taking "half true" answers into account makes the difference between format variations less noticeable and in some cases numbers even outperform full names as identifiers. Additionally, the output format problem could be solved or at least mitigated by improving the system prompt.

Table 4. Augmented model accuracy when regarding "half true" answers as "true".

	6	10	14	20
natural.txt	80.56	66.67	73.81	73.34
natural.json	69.45	78.34	67.85	72.50
foaf.xml	83.33	93.33	83.33	84.16
foaf_names.xml	97.22	96.67	85.71	82.50
foaf.json	88.89	95.00	80.95	80.00
foaf_names.json	94.44	96.67	88.09	84.16
foaf.rdf	80.56	80.00	77.38	71.67
foaf_names.rdf	94.44	90.00	83.33	80.83

Fig. 4 shows the difference between the format variations in a more granular view, on the level of question types. The major difference between variations occurs for the question "List all [full_name]'s friends",

but as per Table 4, the difference is lessened when also taking into account "half true" answers, colored green in the figures. The largest source of mistakes are the questions relating to a person entity's friends, which makes sense considering it is the most complex property. When retrieving a list, the model does not need to simply retrieve a single word or number, but multiple elements, which requires generation of a larger number of tokens and therefore has more chances to make a mistake. Giving the number of friends is also a highly complex task since it required the model to count how many specific elements are present.

5 Discussion and conclusion

Table 3 shows the results of the fine-tuned models. The results are overall significantly worse than the augmented model, as per Table 2. However, the fine-tuned model does not display the same drastic accuracy drop-off with an increase in dataset size. This could be explained by the model having more new data to interpret, and so being able to better focus on it, instead of it being lost in the noise of its pretraining data (Hoffmann et al., 2022). However with a substantial increase in dataset size, the same problem as stated in Subsection 4.2 is likely - the model loses its ability to generalize (Luo et al., 2025). The results point to the fact that fine-tuning, while more complex to perform correctly, could lead to better results. However, it is significantly less flexible than simply augmenting the model, which can lead to positive accuracy results with less time dedicated to achieving it, as well as it being simple to change the data the model has access to. It is important to note that the fine-tuning accuracy results by themselves are not indicative of the quality of the data integration method, as they require proper context.

However, the main goal of this paper was to discover how much of an impact the data format presented to an LLM's ability to recall information. When comparing natural language and more structured data, there is a clear difference: structured data outperforms natural language in terms of knowledge accuracy, data scalability and execution speed.

Since there are obvious nuances between the XML, JSON and RDF formats, however slight, further research needs to be done as to the cause of the discrepancy between their results before concluding an objectively superior format among them and if any of the formats considered in this study even are the best. Certain data might prefer to be presented in a specific format based on its qualities, like its size, complexity, domain of knowledge and similar. Another possibility is that the utilized LLM and its base training data might be making a difference - if the training data includes more JSON than XML structure, it would make sense that the LLM would be better at interpreting it (Pan et al., 2026), and vice versa. On a dataset that is as easily chunkable as the one used throughout this research, the

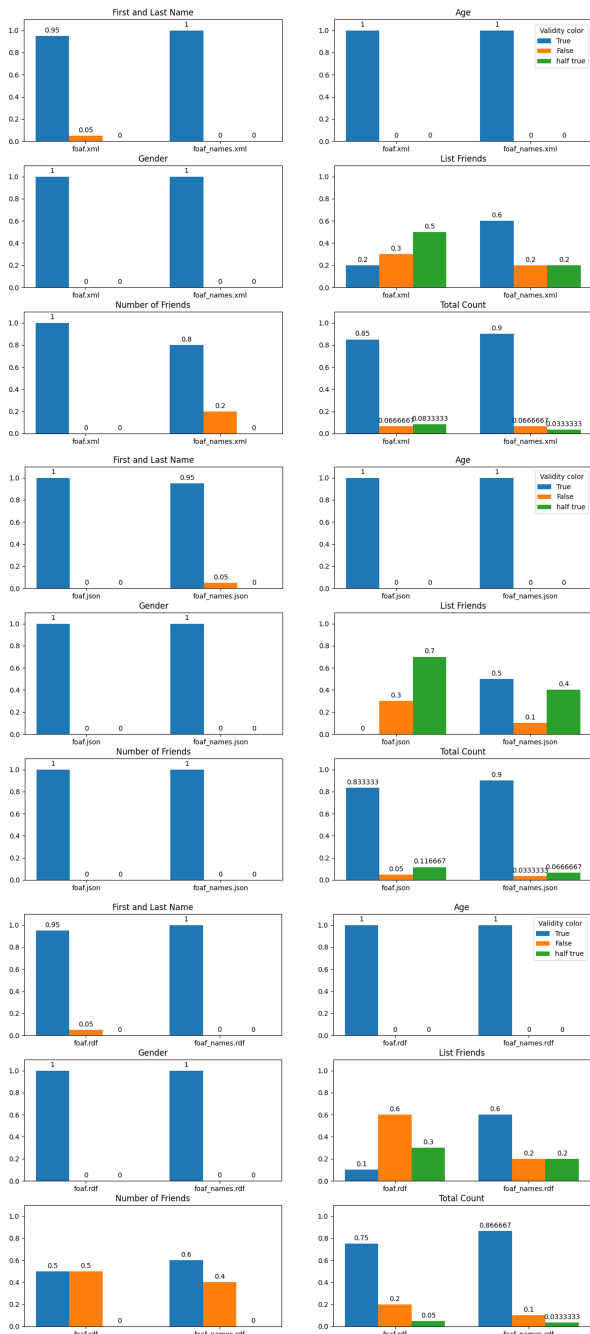


Figure 4. Accuracy of the formats on the 10 person dataset divided by questions. First and second name questions are grouped in the top left bar chart. The bottom right bar chart depicts the average accuracy across all questions. Top image: "foaf.xml" and "foaf_names.xml"; middle image: "foaf.json" and "foaf_names.json"; bottom image: "foaf.rdf" and "foaf_names.rdf".

best approach in order to produce the highest model accuracy seems to be a structured JSON or XML format with minimally complex annotations and label names, as well as using human-readable metadata instead of abstract terms like numbers, which in this case were people entities' IDs.

Furthermore, it is highly important to construct a format which can be easily chunked to avoid large execution times without denying the model proper context. That, however, would need to include a data retrieval mechanism to aid the augmented model in retrieving the required information, which means implementing a system similar to RAG. More research needs to be done to conclude if such a retrieval mechanism can also benefit from improved data formatting - would the embedding vectors of the data be easier to look up if the embedding were, for example, formatted as JSON instead of natural language? Does the format used during retrieval have to match the format used during augmentation?

Finally, future research will expand this evaluation to include larger language models and real-world knowledge graph datasets. Evaluating these neuro-symbolic integrations with different LLM architectures and knowledge formats should clarify how model size and context capacity influence LLMs' ability to interpret complex structured data.

Acknowledgments

This paper is based on the research developed through the project "LIVE Quantum – Development of an Integrated AI Platform for Multi-channel Personalized Management of User Requests" supported by the European Union within the Operational Programme Competitiveness and Cohesion in the financial period 2021–2027 (PK.1.1.12.0099).

References

- Arslan, M., Ghanem, H., Munawar, S., & Cruz, C. A Survey on RAG with LLMs In *Procedia Computer Science Volume 246, 2024* (pp. 3781-3790)
- Ayala, O., Bechard, P. Reducing hallucination in structured outputs via Retrieval-Augmented Generation in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)* (pp. 228-238)
- Bassett, L. (2015). Introduction to JavaScript object notation: a to-the-point guide to JSON. "O'Reilly Media, Inc."
- Chen, H. (2023). Large knowledge model: Perspectives and challenges. arXiv preprint arXiv:2312.02706.
- Choudhary, N., & Reddy, C. K. (2023). Complex logical reasoning over knowledge graphs using large language models. arXiv preprint arXiv:2305.01157.

- Dai, X., Hua, Y., Wu, T., Sheng, Y., Ji, Q., & Qi, G. (2025). Large language models can better understand knowledge graphs than we thought. *Knowledge-Based Systems*, 312, 113060.
- Dehal, R. S., Sharma, M., & Rajabi, E. (2025). Knowledge graphs and their reciprocal relationship with large language models. *Machine Learning and Knowledge Extraction*, 7(2), 38.
- Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., ... & Sifre, L. (2022). An empirical analysis of compute-optimal large language model training. *Advances in neural information processing systems*, 35, 30016-30030.
- Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., Melo, G. D., Gutierrez, C., ... & Zimmermann, A. (2021). Knowledge graphs. *ACM Computing Surveys (Csur)*, 54(4), 1-37.
- Jaldi, C. D., Saini, A., Ghiasi, E., Divine Eziolese, O., & Shimizu, C. (2025, November). Knowledge conceptualization impacts rag efficacy. In *International Knowledge Graph and Semantic Web Conference* (pp. 208-223). Cham: Springer Nature Switzerland.
- Jiang, Z., Xu, F., Gao L., Sun, Z., Liu Q., Dwivedi-Yu J., Yang Y., Callan J., & Neubig, G. Active Retrieval Augmented Generation In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing* (pp. 7969-7992)
- Kalemi, E., & Martiri, E. (2011, November). FOAF-academic ontology: a vocabulary for the academic community. In *2011 Third International Conference on Intelligent Networking and Collaborative Systems* (pp. 440-445). IEEE.
- Kandpal, N., Deng, H., Roberts, A., Wallace, E., Raffel, & C. (2023). Large Language Models Struggle to Learn Long-Tail Knowledge. Retrieved from <https://arxiv.org/abs/2211.08411>
- Lavrinovics, E., Biswas, R., Bjerva, J., & Hose, K. (2025). Knowledge graphs, large language models, and hallucinations: An nlp perspective. *Journal of Web Semantics*, 85, 100844.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems* (Article 793, pp. 9459-9474). Curran Associates Inc.
- Lin, Y. H., Chen, Q. H., Cheng, Y. J., Zhang, J. R., Liu, Y. H., Hsia, L. Y., & Chen, Y. N. (2025). Llm inference enhanced by external knowledge: A survey. *arXiv preprint arXiv:2505.24377*.
- Luo, Y., Yang, Z., Meng, F., Li, Y., Zhou, J., & Zhang, Y. (2025). An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *IEEE Transactions on Audio, Speech and Language Processing*.
- Model meta-llama/Llama-3.2-3B-Instruct (2024). Retrieved from <https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct>, retrieved 14 May 2026.
- Narvekar, A., Mukherjee, T., & Tanuwijaya, J. (2025, November). Evaluating the Impact of Input Format on LLM Performance: JSON vs. YAML vs. Text. In *2025 IEEE International Conference on Future Machine Learning and Data Science (FMLDS)* (pp. 572-577). IEEE.
- Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., & Wu, X. (2024). Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7), 3580-3599.
- Pan, Z., Wang, S., Pengfei, L., & Li, J. (2026). Understanding llm behaviors via compression: Data generation, knowledge acquisition and scaling laws. *Advances in Neural Information Processing Systems*, 38, 168404-168454.
- Škrlj, B., Koloski, B., Pollak, S., & Lavrač, N. (2025). From Symbolic to Neural and Back: Exploring Knowledge Graph&Large Language Model Synergies. *Challenges and Algorithms for Knowledge Discovery from Data: Essays Dedicated to Arno Siebes on the Occasion of His 67th Birthday*, 181-197.
- Tilwani, D., Venkataramanan, R., & Sheth, A. P. (2024). Neurosymbolic ai approach to attribution in large language models. *IEEE Intelligent Systems*, 39(6), 10-17.
- Yang, L., Chen, H., Li, Z., Ding, X., & Wu, X. (2024). Give us the facts: Enhancing large language models with knowledge graphs for fact-aware language modeling. *IEEE Transactions on Knowledge and Data Engineering*, 36(7), 3091-3110.
- Yang, W., Some, L., Bain, M., & Kang, B. (2025). A comprehensive survey on integrating large language models with knowledge-based methods. *Knowledge-Based Systems*, 318, 113503.
- World Wide Web Consortium. (2006). Extensible markup language (XML) 1.1.
- Wu, X., & Tsioutsoulouklis, K. (2024). Thinking with knowledge graphs: Enhancing LLM reasoning through structured data. *arXiv preprint arXiv:2412.10654*.

Zhao, P., Zhang, H., Yu, Q., Wang, Z., Geng, Y., Fu, F., ... & Cui, B. (2026). Retrieval-augmented generation for ai-generated content: A survey. *Data Science and Engineering*, 1-29.