Comparing Similarity Measures for Uncertainty Quantification of Black-box LLMs

Adrien Letellier
ENSAE
adrien.letellier@ensae.fr

Abstract

This work focuses on uncertainty quantification methods for black-box Large Language Models. A recent article by Lin et al.[8] proposes a methodology based on the generation of multiple responses that are compared using similarity measures. Building upon this article, we investigate alternative similarity measures for response comparison: Levenshtein distance and embedding-based cosine similarity. We evaluate these measures against established Jaccard and NLI-based similarity approaches on the TriviaQA dataset. Our results show that embedding-based similarity offers slightly worse performance for uncertainty estimation while being computationally more efficient than NLI-based methods. This study seeks to contribute to the growing research on developing reliable uncertainty measures for black-box LLMs, which is crucial for building trustworthy AI systems.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in natural language generation (NLG) tasks, including question answering. However, as these models are increasingly deployed in automated systems, quantifying their uncertainty becomes crucial. It enables to know when to trust an LLM's response and when to reject it or seek additional verification.

Uncertainty quantification (UQ) in NLG presents challenges compared to standard classification tasks. The output space has high dimensionality, different token sequences may convey identical meanings, and many state-of-the-art LLMs are available only as black-boxes through APIs. These challenges make traditional UQ methods impractical.

In this work, we focus on black-box uncertainty quantification for LLMs, building upon the framework established by Lin et al.[8].

2 Related Work

Uncertainty quantification has been extensively studied in machine learning, particularly for classification and regression tasks ([6]). However, according to the article, UQ for natural language generation remains relatively under-explored.

Early approaches to UQ in NLP have typically addressed challenges by framing tasks as classification problems ([1], [3]), which fails to capture the generative aspects unique to NLG. More recent work has begun exploring uncertainty directly in generative contexts, often by asking LLMs to evaluate their own confidence ([5], [7]).

The most relevant work for us is by Lin et al.[8], who proposed several black-box UQ methods for LLMs. Their approach generates multiple responses for a given input and uses similarity measures

NLP class, ENSAE 2024-25.

like Jaccard and Natural Language Inference (NLI) to quantify response dispersion as an indicator of uncertainty and confidence.

Our work extends this line of research by investigating two alternative similarity measures not explored in the original article: Levenshtein distance and embedding-based cosine similarity. These measures potentially offer different computational and performance trade-offs compared to the original ones.

3 Methodology

3.1 Problem Formulation

Following the article, we frame the problem as follows: for a given question x, we generate m response samples $s_1, ..., s_m$ from a black-box LLM. We then calculate pairwise similarity scores $a(s_{j_1}, s_{j_2})$ between these responses. These similarity scores are used to compute an uncertainty estimate $U(x)$ for the input or a confidence score $C(x, s_j)$ for each specific response.

3.2 Similarity Measures

We compare four similarity measures, all of them are bounded in the $[0, 1]$ interval to ease the building of uncertainty and confidence measures:

- Jaccard similarity (**baseline**): The ratio of the intersection to the union of words in two responses.

  $$a_{Jaccard}(s_1, s_2) = \frac{|s_1 \cap s_2|}{|s_1 \cup s_2|}$$

  Jaccard similarity offers a simple word-overlap metric that is computationally inexpensive. However, it treats all words equally and cannot capture semantic relationships or word ordering. It performs adequately for basic lexical overlap assessment but fails to account for synonyms, paraphrases, or contextual meaning.

- Levenshtein-based similarity (**proposed**): Normalized edit distance between responses.

  $$a_{Lev}(s_1, s_2) = 1 - (Levenshtein(s_1, s_2)/max(len(s_1), len(s_2)))$$

  The Levenshtein-based metric improves upon Jaccard similarity while remaining computationally efficient. By considering the minimum number of single-character edits (insertions, deletions, substitutions) required to transform one string into another, it can better handle morphological variations, misspellings, and slight wording differences. This addresses cases where two responses use variants of the same word (e.g., "analyze" vs. "analysis") that Jaccard would treat as entirely different terms.

- NLI-based similarity (**baseline**): Using a pre-trained Natural Language Inference model to determine semantic entailment or contradiction between responses. NLI-based similarity leverages pre-trained models to assess logical relationships between texts. While computationally more expensive, it is able to identify responses that convey the same meaning despite using different phrasing. However, it may struggle with very short responses or domain-specific language not present in the model's training data.

- Embedding cosine similarity (**proposed**): Cosine similarity between sentence embeddings generated by a pre-trained language model.

  $$a_{Emb}(s_1, s_2) = cosine\_similarity(embed(s_1), embed(s_2))$$

  It is originally in the $[-1, 1]$ interval so we have the shift it by 1 and rescale it to be on the $[0, 1]$ interval. Embedding-based similarity can provide a better trade-off between computational efficiency and semantic understanding, with contextual embeddings accounting for word meaning in context rather than just lexical overlap. This method should be effective for determining if two differently worded responses convey the same meaning.

3.3 Uncertainty and Confidence Measures

Following the original paper, we implement several approaches to convert similarity matrices into uncertainty and confidence measures. Noting D the degree matrix derived from the similarity matrix, λ_k the eigenvalues of the normalized graph Laplacian, and v'_j the offset of response j from the average embedding (see the paper for more detailed formulas), we build the following measures:

- Eigenvalue-based measure (EigV):
  - Uncertainty: $U_{EigV} = \sum_{k=1}^{m} \max(0, 1 - \lambda_k)$
- Degree-based measures (Deg):
  - Uncertainty: $U_{Deg}(x) = trace(mI - D)/m^2$
  - Confidence: $C_{Deg}(x, s_j) = D_{j,j}/m$
- Eccentricity-based measures (Ecc):
  - Uncertainty: $U_{Ecc}(x) = ||[v_1'^T, ..., v_m'^T]||_2$
  - Confidence: $C_{Ecc}(x, s_j) = -||v'_j||_2$

4 Experimental Setup

4.1 Dataset

We use the TriviaQA dataset ([4]), a question-answering dataset consisting of 9,960 questions in the validation split of the rc.nocontext subset. It is a closed-book QA dataset, that is the prompt does not include context and it relies only on the knowledge of the LLM. TriviaQA is suitable for our study as its questions cover diverse topics and vary significantly in difficulty. We only include the first 1000 questions due to limited access to GPUs and APIs.

A basic statistical analysis of the first 1000 questions of TriviaQA dataset reveals that the average question length is 12.01 words and the average answer length is 1.64 words.

4.2 Model and Implementation

I used Qwen2.5-7B-Instruct as a black-box LLM, generating $m = 10$ responses for each question. I chose this model mainly because of limited access to API of black-box LLMs such as ChatGPT, Claude or Mistral. For the NLI baseline, I used DeBERTa-large[2], following the original article. For our embedding-based approach, we employed the all-MiniLM-L6-v2 [10] model from the Sentence-Transformers library [9].

For evaluation, we automatically assessed the correctness of generated responses using the latest Mistral Large model as a judge, following the methodology of the original article. A response is considered as correct if it receives a score above 70 when evaluated against the reference answer.

We evaluate the quality of uncertainty/confidence measures using Area Under Accuracy-Rejection Curve (AUARC).

5 Results and Analysis

5.1 Overall Performance

The Figure 1 is an attempt of reproducing the Figure 3 of the original article, confirming that we were able to approximately reproduce their results.

Table 1 presents the AUARC scores when using uncertainty measures $U(x)$ to predict expected accuracy on the TriviaQA dataset.

Table 2 shows the AUARC scores when using confidence measures $C(x, s)$ to predict individual response accuracy.

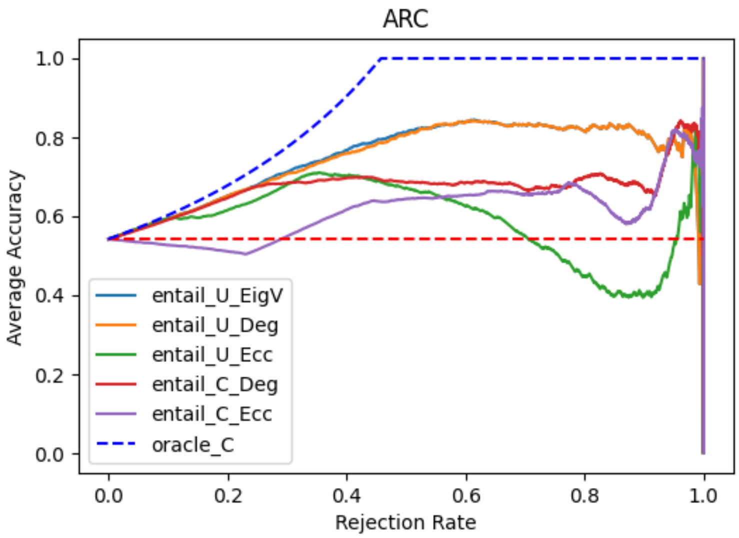

Figure 1: ARC computed using U and expected accuracy. Similarities were computed using entailement from NLI.

		AUARC
jaccard	EigV	0.742728
	Deg	0.742223
	Ecc	0.576155
levenshtein	EigV	0.742853
	Deg	0.742898
	Ecc	0.555690
entail	EigV	0.747246
	Deg	0.745129
	Ecc	**0.590020**
contra	EigV	0.751453
	Deg	0.750914
	Ecc	0.574233
sbert	EigV	**0.756456**
	Deg	**0.755780**
	Ecc	0.463206
oracle		0.871466
random		0.542500

Table 1: AUARC for Uncertainty Measures (U+EA)

5.2 Analysis

Our results suggest the following observations:

- Embedding-based similarity performs well, approaching the effectiveness of NLI-based similarity while being computationally more efficient. This suggests that pre-trained sentence embeddings capture sufficient semantic information for uncertainty estimation.
- Levenshtein-based similarity underperforms compared to other methods. This indicates that simple string-based edit distance fails to capture semantic relationships effectively, confirming the importance of semantic understanding for uncertainty quantification. The

		AUARC
jaccard	Deg	0.665121
	Ecc	**0.613936**
levenshtein	Deg	0.649785
	Ecc	0.598050
entail	Deg	**0.673507**
	Ecc	0.612437
contra	Deg	0.666039
	Ecc	0.606318
sbert	Deg	0.625225
	Ecc	0.545212
oracle		0.874275
random		0.542500

Table 2: AUARC for Confidence Measures (C+IA)

Table 3: Computation Time

Similarity Measure	Time (seconds)
Jaccard	0.007
Levenshtein	0.006
NLI	26.034
Embedding	2.122

poor performance of the Levenshtein measure compared to the Jaccard one may however be due to the dataset that requires very precise and short responses and it would probably perform better in a less supervised setting.

5.3 Computational Efficiency

Table 3 compares the average computation time per question for different similarity measures. Embedding-based similarity offers a favorable trade-off between computational efficiency and performance, requiring approximately 13x less computation time than NLI while delivering comparable results.

6 Conclusion

This study investigated alternative similarity measures for uncertainty quantification in black-box LLMs. Our findings suggest that embedding-based cosine similarity offers competitive performance compared to NLI-based approaches while being substantially more efficient. In contrast, Levenshtein distance, despite its simplicity, fails to capture semantic relationships adequately for this task.

Future work could explore combining multiple similarity measures, developing more sophisticated embedding techniques specifically designed for uncertainty quantification, and extending these methods to more open-ended generation tasks beyond question answering.

A major drawback of this methodology is that it requires generating multiple responses for each prompt, which multiplies the cost of the inference. In the original article, this problem was already tackled: the authors showed that there is a trade-off between the precision of the UQ and the number of generations. They already achieve satisfying results with as few as 3 generations per prompt. A further challenge would be to design UQ methods that do not require multiple generations.

7 References

References

[1] Shrey Desai and Greg Durrett. Calibration of pre-trained transformers. *arXiv preprint arXiv:2003.07892*, 2020.

[2] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

[3] Zhengbao Jiang, Jun Araki, Haibo Ding, and Graham Neubig. How can we know when language models know? on the calibration of language models for question answering. *Transactions of the Association for Computational Linguistics*, 9:962–977, 2021.

[4] Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*, 2017.

[5] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

[6] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[7] Stephanie Lin, Jacob Hilton, and Owain Evans. Teaching models to express their uncertainty in words. *arXiv preprint arXiv:2205.14334*, 2022.

[8] Zhen Lin, Shubhendu Trivedi, and Jimeng Sun. Generating with confidence: Uncertainty quantification for black-box large language models. *arXiv preprint arXiv:2305.19187*, 2023.

[9] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[10] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in neural information processing systems*, 33:5776–5788, 2020.