Cosine Similarity as Logits?: A Scalable Knowledge Probe Using Embedding Vectors from Generative Language Models

Anonymous ACL submission

Abstract

Recently, the use of pretrained language models (PLMs) as soft knowledge bases has gained growing interest, sparking the development of knowledge probes to evaluate their factual knowledge retrieval capabilities. However, existing knowledge probes for generative PLMs that support multi-token entities exhibit quadratic time complexity $\mathcal{O}(n^2)$, limiting the size of knowledge graphs used for probing. To address this, we propose DEcoder Embedding-based Relational (DEER) probe, utilizing embedding vectors extracted from generative PLMs. DEER probe achieves effective time complexity of linear order $\mathcal{O}(n)$, supports rank-based evaluation metrics including Hit@k, handles multi-token entity names and enables probing whilst disambiguation of homographic tail-enity names. We empirically show that DEER-probe correlates with existing knowledge probes, validating its probing capability, and we demonstrate the practical benefits of its improved scalability.

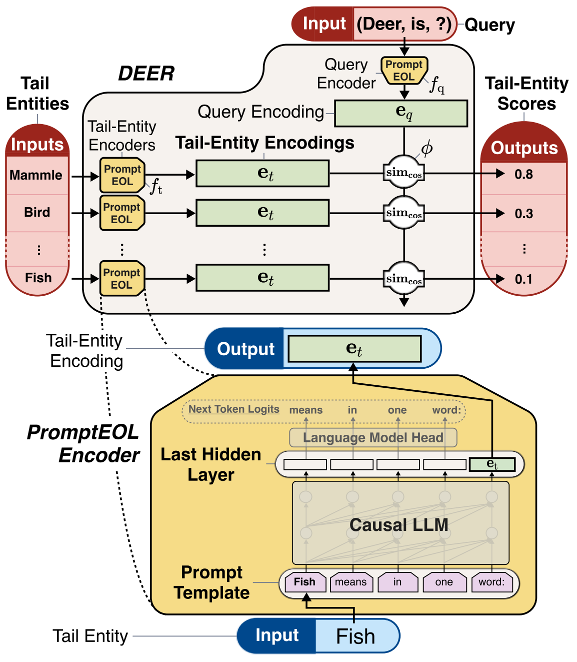

Figure 1: An illustration of DEER's architecture.

1 Introduction

Knowledge probes evaluate factual knowledge retrieval capabilities of pre-trained language models (PLMs). Their applications include identifying missing knowledge in PLMs and quantifying the amount of domain-specific knowledge encoded in their parameters. Knowledge probes achieve this by assessing a PLM's capability to complete a relational knowledge. A knowledge graph (KG) represents a relational knowledge as a triplet consisting of (*head-entity*, *relation-type*, *tail-entity*). To complete a relational knowledge, models must predict the correct tail-entity, given a partially filled triplet, (*head-entity*, *relation-type*, ?) which we call a query.

To the best of our knowledge, the only knowledge probe capable of probing generative PLMs with multi-token tail entity names and the Hit@k, a conventional evaluation metric in knowledge base completion, is BEAR (Wiland et al., 2024; Youssef et al., 2023). However, BEAR exhibit quadratic time complexity $\mathcal{O}(n^2)$, making probing of PLMs on large-scale KGs infeasible. Moreover, BEAR predicts the tail entity for a query by computing the joint probability of the textual sequence formed by concatenating the query and a tail entity candidate's textual representation. The reliance on joint probability may introduce a token length bias, favoring candidates with shorter names.

To address these issues, we propose DEER (DEcoder Embedding-based Relational) Probe, a knowledge probe utilizing embedding vectors extracted from a causal PLM through the PromptEOL method (Jiang et al., 2024), as shown in Figure 1. DEER effectively realizes linear time complexity, $\mathcal{O}(n)$, whilst reducing the token length bias in probing. DEER also enables probing whilst disambiguating identically named tail entities.

This work addresses the following questions:

RQ1: Does the improved time complexity yield observable reductions in compute time?

RQ2: Does DEER exhibit reduced token length bias compared to BEAR?

RQ3: Does DEER align with existing knowledge probes?

To evaluate the practical implications of DEER's improved time complexity, we measured the time required to probe GPT-2-Small (Radford et al., 2019) with WN18RR (Dettmers et al., 2018) under both BEAR and DEER. BEAR was estimated, through extrapolation, to require 330 ± 10 days, rendering full evaluation infeasible, whereas DEER completed the task in 16 ± 1 minutes, making such evaluation tractable. Token length bias was assessed by computing the Pearson correlation between predicted ranks and tail entity token lengths. BEAR exhibited a strong correlation $r = 0.484$ and $p < 10^{-51}$, while DEER was uncorrelated with $r = 0.005$ and $p = 0.913$, indicating DEER's ability to evaluate knowledge independent of token length. Finally, DEER's alignment with the established probe, LAMA, was assessed via log-scale rank correlation, yielding a maximum $r = 0.804$, supporting its use for knowledge probing.

2 Background

Knowledge Probing A KG is defined as a set of triplets $\mathcal{T} \ni (h, r, t)$, where h, r and t denote the head entity, relation-type and tail entity. In knowledge probing, given a query $(h, r, ?)$, PLMs are tasked with predicting the corresponding tail entity t. This is typically achieved by ranking a set of candidate tail entities $\mathcal{E}$, according to their probability $P(t = e | e \in \mathcal{E})$. Performance is evaluated using Hit@k which measures the fraction of test triplets for which the correct tail entity is ranked within the top k candidates.

Existing Knowledge Probes for Causal PLMs The first proposed knowledge probe, LAMA (Petroni et al., 2019), prompts a PLM with a cloze-style question, then ranks the tail entity candidates by the log likelihood of the token corresponding to their name at the masked position. However, it can only test entities with a single token name. KAMEL (Kalo and Fichtel, 2022) enabled probing of multi-token entities using text generation. However, the predicted tail entities are evaluated using exact string matching, limiting the evaluation metric to Hit@1. BEAR was recently proposed to support both the Hit@k metric and multi-token entities. However, it requires computing log-likelihood for all possible query and entity combinations, hence the PLM must process $\mathcal{O}(|\mathcal{Q}| \times |\mathcal{E}|)$ inputs, where $\mathcal{Q}$ is the set of queries, resulting in quadratic time complexity. In addition, since prior methods rely on token predictions, they cannot disambiguate identically named tail entity candidates.

PromptEOL Embedding PromptEOL (Jiang et al., 2024) is a method for creating sentence embedding vectors using a generative PLM without additional training. It prompts a model to summarize a sentence in one word, then uses the last hidden vector, usually used for next-token prediction, as the sentence embedding. The prompt template: *This sentence: {S} means in one word "*, is used, where *S* is replaced by the sentence to encode. Appendix A.1 provides a formal description.

3 Problem Formulation

Textual Knowledge Graph We define a textual knowledge graph as a KG with textual representations of entities and relation types. We denote set of all entity names, entity descriptions and relation names as $\mathcal{E}_{\text{name}}$, $\mathcal{E}_{\text{desc}}$ and $\mathcal{R}_{\text{name}}$, respectively. Thereby, given a set of all possible strings Σ^*, $\mathcal{E}_{\text{name}}, \mathcal{E}_{\text{desc}}, \mathcal{R}_{\text{name}} \subseteq \Sigma^*$. We assume the existence of $\mathcal{E} \to \mathcal{E}_{\text{name}}$, $\mathcal{E} \to \mathcal{E}_{\text{desc}}$ and $\mathcal{R} \to \mathcal{R}_{\text{name}}$, mapping the entities and relations to their corresponding textual representations.

Embedding-Based Probes Embedding-based knowledge probing frameworks (Dufter et al., 2021) consist of a query encoder f_{q}, a tail entity encoder f_{t} and a similarity function ϕ. Given functions mapping the query and entities to their textual representations, $f_{\text{q}}^{\text{text}} \colon \mathcal{Q} \to \Sigma^*$ and $f_{\text{t}}^{\text{text}} \colon \mathcal{E} \to \Sigma^*$, the query encoder and the tail entity encoder maps the textual representation of queries and tail entity candidates to an encoding vector $f_{\text{q}} \colon f_{\text{q}}^{\text{text}}(\mathcal{Q}) \to E_{\text{q}}$ and $f_{\text{t}} \colon f_{\text{t}}^{\text{text}}(\mathcal{E}) \to E_{\text{t}}$, where $E_{\text{q}}, E_{\text{t}} \subseteq \mathbb{R}^n$.

The similarity function $\phi \colon E_{\text{q}} \times E_{\text{t}} \to \mathbb{R}$ measures the distance between a query embedding and a tail entity embedding. Given a query $q \in \mathcal{Q}$, the model ranks all tail entity candidates $e \in \mathcal{E}$ by their similarity scores, defined by ϕ. Formally, let $N = \{n | n \in \mathbb{N}, n \leq |\mathcal{E}|\}$ denote the set of rank indices. A one-to-one function $I \colon \mathcal{Q} \times N \to \mathcal{E}$

e_{name} - e_{desc}
This sentence: "{word}" means in one word: "{one word}"
This sentence: "e_{name}" means in one word: "

Figure 2: The template used by the tail encoder f_t. The entities' name and description replaces e_{name} and e_{desc}. An example is shown in Figure 6, Appendix A.2.

$(\eta^1_{\text{name}}, \rho^1_{\text{name}}, \tau^1_{\text{name}})$	(dress up, verb group, trick up)
$\vdots$	$\vdots$
$(\eta^8_{\text{name}}, \rho^8_{\text{name}}, \tau^8_{\text{name}})$	(disfavour, hypernym, single out)
$(h_{\text{name}}, r_{\text{name}},$	(deer, hypernym,

Figure 3: Left: The query-encoding template, f_q^{text} for knowledge probing. The names of the query's h and r replaces the symbols h_{name} and r_{name}. The names of h, r and t in the nth randomly sampled triplet replaces η^n_{name}, ρ^n_{name} and τ^n_{name} . Right: A completed example for a query, (deer, hypernym, ?).

assigns to each query q, a ranked list of tail-entities such that: $x \in \mathcal{Q} \Rightarrow \mathbf{e}_x = f_q(f_{\text{text}}(x))$ and $x \in \mathcal{E} \Rightarrow \mathbf{e}_x = f_t(f_{\text{text}}(x))$. Notice the information processed by the encoders, scale linearly, $\mathcal{O}(|\mathcal{Q}| + |\mathcal{E}|)$. Since the compute time for cosine similarity is negligible compared to the encoder forward pass, the effective time complexity is $\mathcal{O}(n)$.

4 DEER: Proposed Method

As shown in Figure 1, DEER instantiate the embedding-based framework using PromptEOL as both the query encoder, f_q, and the tail-entity encoder, f_t. To generate the encoding, custom prompt templates are populated with $\mathcal{E}_{\text{name}}$, $\mathcal{E}_{\text{desc}}$, and $\mathcal{R}_{\text{name}}$, then fed into f_q and f_t. The last hidden vector, used as a sentence embedding in PromptEOL, serves as the encoding vector. Cosine similarity is used as ϕ to assign a unique rank, N to each tail entity candidates. The following paragraphs discuss the custom templates used by f_q and f_t.

Tail-Entities Encoding Template, f_t^{text} Figure 2 shows the template used by the tail-entity encoder, f_t to acquire a tail-entity encoding. The inclusion of e_{desc} enables disambiguation of identical names.

Prober Query Encoding Template, f_q^{text} Figure 3 shows the template used by the query encoder, f_q, during knowledge probing. The eight-shot example is compiled by randomly sampling from the training set. Appendix A.2 gives the design rationale.

Prober	Time/query	Total time
BEAR ap.	200 ± 500 *s*/query	330 ± 10 days
DEER	0.0316 ± 0.0009 *s*/query	16 ± 1 minutes

Table 1: Compute time comparison on WN18RR. BEAR results are approximated via extrapolation.

5 Experiments

Four sets of experiments were conducted to address the questions: Compute Time, Probe Agreement, Token Bias, and Baseline. Except for Compute Time, which employed GPT-2[1], all experiments were conducted using the OPT model[2] (Zhang et al., 2022) as the PLM for PromptEOL.

Datasets and Metrics WN18RR was used as the KG. The names and descriptions of the entities in the KG were acquired from Yao et al. (2019) using process described in Appendix C. Hit@10 threshold were used throughout the experiments to evaluate the PLMs.

Prompt Templates for LAMA and BEAR To minimize discrepancy in prompt design when comparing with DEER, prompt design that is similar to DEER's is used for LAMA and BEAR. These templates are given in Appendix A.3.

5.1 Compute Time Experiment

Experimental Setup To demonstrate the practical impact of DEER's improved time complexity, we compared the time required to probe the full WN18RR knowledge graph using GPT-2 (124M) under both DEER and BEAR. DEER's compute time was directly measured over all $|\mathcal{Q}| = 93{,}003$ queries obtained from the KG, using the full entity set, $|\mathcal{E}| = 40{,}943$, as tail candidates. In contrast, since BEAR's compute time is infeasible to measure directly, it was estimated through extrapolation. Both experiments were conducted using an RTX 3090 GPU with 8 CPU cores. Experimental details are provided in Appendix C.1.

Results Table 1 shows that the improved time complexity results in reduced probing time in practice. Under our setup, DEER reduces compute time from approximately 330 ± 10 days to 16 ± 1 minutes when probing WN18RR, corresponding to a $(30{,}000 \pm 2{,}000)\times$ speedup, enabling probing of KGs that would otherwise be infeasible under BEAR.

[1] https://huggingface.co/gpt2
[2] https://huggingface.co/facebook/opt-30b

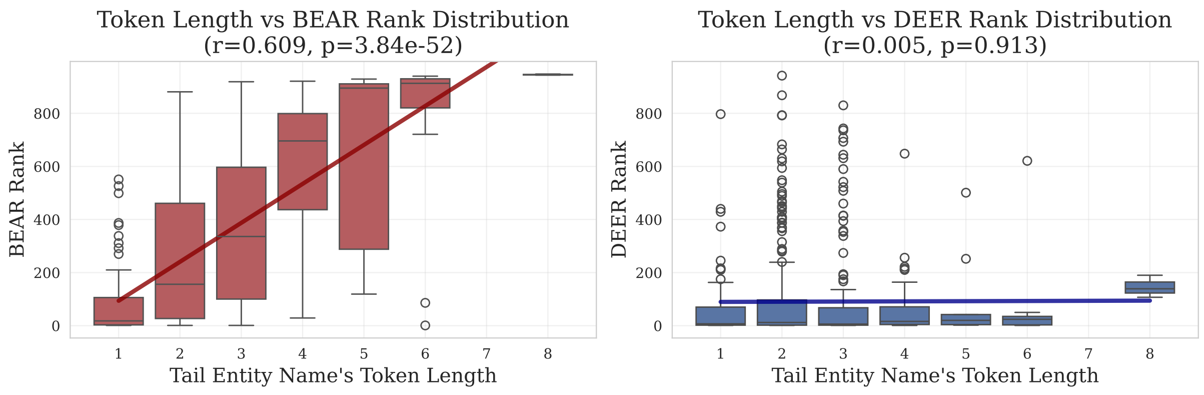

Figure 4: A box plot showing a positive correlation between tail entity token length and BEAR ranks, and a negligible correlation under DEER.

5.2 Token Length Bias Experiment

Experimental Setup To investigate whether DEER exhibit lower token length bias than BEAR, Pearson correlation between the token length of the correct tail entity and its predicted rank was computed across 500 queries sampled from the WN18RR test set. Experimental details and further analysis are provided in Appendix C.2 and E.1.

Results As shown in Figure 4, BEAR's predicted correlate with token length ($r = 0.484$, $p < 10^{-5}$), whereas DEER shows negligible correlation ($r = 0.005$, $p = 0.913$). Assuming query's difficulty is independent of tail entity length, BEAR may be susceptible to token-length-induced false positives and negatives, while DEER remains unaffected.

5.3 Probe Agreement Experiment

Experimental Setup Pearson correlation between tail entity ranks predicted by LAMA and DEER was measured to assess DEER's agreement with an established knowledge probe. Ranks were compared in log scale to account for the reduced significance of differences at higher ranks. Experimental setup and additional BEAR comparisons are provided in Appendix C.3 and E.4, respectively.

Result Table 2 shows that sub-billion models exhibit limited rank agreement, whereas models over a billion parameters achieve high correlation, with OPT-6.7B reaching $r = 0.804$. Figure 5 illustrates this, supporting DEER's viability as a knowledge probe for super-billion-parameter models.

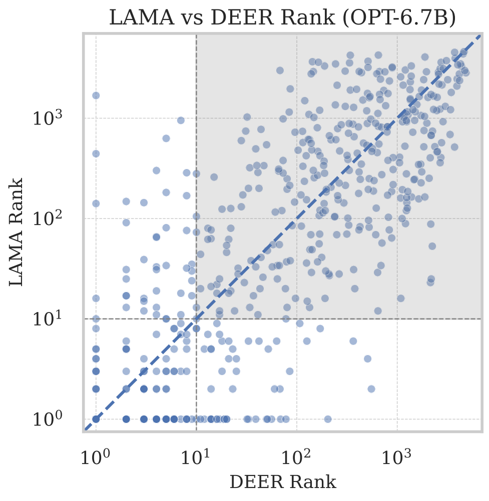

Figure 5: A scatter plot showing rank of tail entities predicted by LAMA and DEER in log scale. Each points represent a triplet, the bottom-left and top-right quadrants indicate agreements in Hit@10 and Miss@10. Figure 9, Appendix D show results by parameter size.

PLM Name	Log(Rank), r
OPT-350M	0.496
OPT-1.3B	0.737
OPT-6.7B	0.804

Table 2: Pearson correlation between DEER and LAMA rank. Table 4, Appendix D shows the full result.

6 Conclusion

This paper introduced DEER probe, a novel and scalable knowledge probing method for generative PLMs. By utilizing embedding vectors acquired from the PLM, DEER enables input size to scale linearly, $\mathcal{O}(|\mathcal{Q}| + |\mathcal{E}|)$, overcoming BEAR's limitation. Empirical results demonstrate that DEER can probe KGs infeasible under BEAR, and its avoidance of token probability reduces biases induced by entity name length. DEER also aligns closely with LAMA, offering a promising direction for future knowledge probing research.

7 Limitations

Reproducibility Source code for the experiments as well as a Python package for DEER probe is made and will be relaced upon acceptance of the paper.

Possibility of Data Leakage The original dataset of WN18RR, the WN18 (Bordes et al., 2013) had both been released before the of PLMS such as OPT, meaning it may have been included within the training corpus. Even though progress have been made in knowledge graph completion evaluation technique that avoids such issue (Sakai et al., 2024), utilization of such technique on DEER is non-trivial, as it utilizes hypothetical, non real-world relations, that would not be stored in the parametric knowledge. However, since our aim is to evaluate the knowledge memorized within the parametric knowledge, it does not alter the fact these knowledge are stored within the parameters.

Inclusion of Tail-Entity Descriptions during Knowledge Probing When encoding tail-entities during knowledge probing, the description of the entity is provided to allow disambigation. This allows the PLM to infer relations correctly, even in absense of the knolwedge around the tail-entity. This could be an issue when assessing the knowledge retrival capability of PLMs, however, we argue the effect of such case is limited as demonstrated by the high correlation between DEER and LAMA, Table 4, where LAMA is never shown such descriptions.

The Use of Log Scale The agreement between DEER, LAMA and BEAR was compared in linear and log scale, even though the two showed weaker correlation in the linear scale, we argue that the conclusion drawn in log scale should be preferred, taking similar line of argument for the preference of MRR over Rank in the knowledge base completion community, where the weight on the importance of the difference in rank is reduced as the rank increase.

Assumption of Normal Distribution in Standard Error When computing the standard error and the confidence interval, a normal distribution of the data was assumed without proof.

8 Ethical Considerations

Throughout the paper, we investigated the ability of DEER using the Open Pre-trained Transformers (OPT) for PromptEOL. Thereby any, derivative work of DEER utilizing OPT, is subject to the OPT License Agreement [3], which prohibits the use for the purpose of: commercial or production, military or nuclear technology, surveillance, biometric processing, violation of third-party right and violation of any applicable law.

References

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 2787–2795, Red Hook, NY, USA. Curran Associates Inc.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press.

Philipp Dufter, Nora Kassner, and Hinrich Schütze. 2021. Static embeddings as efficient knowledge bases? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2353–2363, Online. Association for Computational Linguistics.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. 2024. Scaling sentence embeddings with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3182–3196, Miami, Florida, USA. Association for Computational Linguistics.

Jan-Christoph Kalo and Leandra Fichtel. 2022. Kamel: Knowledge analysis with multitoken entities in language models. In *AKBC*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

[3] https://github.com/facebookresearch/metaseq/blob/main/projects/OPT/MODEL_LICENSE.md

Yusuke Sakai, Hidetaka Kamigaito, Katsuhiko Hayashi, and Taro Watanabe. 2024. Does pre-trained language model actually infer unseen links in knowledge graph completion? 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 8091–8106, Mexico City, Mexico. Association for Computational Linguistics.

Jacek Wiland, Max Ploner, and Alan Akbik. 2024. BEAR: A unified framework for evaluating relational knowledge in causal and masked language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2393–2411, Mexico City, Mexico. Association for Computational Linguistics.

Kosuke Yamada and Peinan Zhang. 2025. Out-of-the-box conditional text embeddings from large language models. *Preprint*, arXiv:2504.16411.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *Preprint*, arXiv:1909.03193.

Paul Youssef, Osman Koraş, Meijie Li, Jörg Schlötterer, and Christin Seifert. 2023. Give me the facts! a survey on factual knowledge probing in pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15588–15605, Singapore. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *Preprint*, arXiv:2205.01068.

A Methodological Details

A.1 Formal Description of PromptEOL

PromptEOL uses a generative PLM for sentence embedding. Given a sequence of input tokens $x_1, x_2, \ldots, x_n$, the last hidden state, $\mathbf{h}_n$, corresponding to the token x_n, is used as the sentence embedding vector. More specifically, $\mathbf{h}_n$ is the vector typically used for next token prediction by applying a final dense layer followed by a softmax function. This process is expressed as $\mathbf{z}_n = \mathbf{W}\mathbf{h}_n + \mathbf{b}$, where $x_{n+1} = \arg\max(\text{softmax}(\mathbf{z}_n))$, where $\mathbf{h}_n$ is the last hidden layer, $\mathbf{W}$ is the weight of the projection layer and $\mathbf{b}$ is the bias term. To embed a sentence, it uses the following prompt template: *This sentence: "S" means in one word "*, where S is replaced with the target sentence. PromptEOL enables creation of embedding vectors conditioned by prompts (Yamada and Zhang, 2025).

deer - distinguished from Bovidae by the male's having solid deciduous antlers
This sentence: "{word}" means in one word: "{one word}"
This sentence: "deer" means in one word: "

Figure 6: An example of the tail-entities encoding template when used to encode the "deer" entity. Thereby, e_{name} is "deer" and e_{desc} is "distinguished from Bovidae by the male's having solid".

$(\eta^1_{\text{name}}, \rho^1_{\text{name}}, \tau^1_{\text{name}})$	(dress up, verb group, trick up)
$\vdots$	$\vdots$
$(\eta^8_{\text{name}}, \rho^8_{\text{name}}, \tau^8_{\text{name}})$	(disfavor, hypernym, single out)
$(h_{\text{name}}, r_{\text{name}}, t_{\text{name}})$	(deer, hypernym, mammal)

Figure 7: Left: BEAR's prompt template used to calculate the probability of tail-entity, given a query. The names of query's head entity and relation type replaces the symbols h_{name} and r_{name}. The names of head-entity, relation-type and tail-entity in the n^{th} randomly sampled triplet replaces η^n_{name}, ρ^n_{name}, τ^n_{name}. Right: A completed example for a query, (deer, hypernym, ?) and tail-entity candidate, mammal. In essence, it is DEER's query encoding template with addition of the tail entity name.

A.2 Prompt Templates, Examples and Rationale

The Query Encoding Templates are designed so that the next-token PLM tries predicting, when given the template, is semantically similar to the one it tries predicting when given the Tail-Entity Template. Hence, when the two embeddings are measured with a similarity measure when the correct prediction is made, the similarity function, ϕ, gives a high score between the embeddings $\mathbf{e}_q$ and $\mathbf{e}_t$.

A complete example of the Tail-Entity Encoding Template is shown in Figure 6.

A.3 Prompt Templates for LAMA and BEAR

DEER's query encoding template, Figure 3, was used for predicting the logits values in LAMA. BEAR utilized prompt template shown in Figure 7 to compute probability for each query tail-entity pair.

B Experiment-Specific Details

C Creation of Entity Names and Descriptions

As discussed in Section 5, dataset provided by Yao et al. (2019) was used to construct the entity name and description of WN18RR. However, the data provided the names with meta data such as their part of speech, e.g. "__whitetail_deer_NN_1", this was cleaned to form a name in natural language, e.g. "whitetail deer", when used as an entity-name. Empty descriptions were replaced with a dash character: "-".

C.1 Compute Time Experimental Setup Detail

BEAR's compute time when probing GPT2-Small (124M) with the WN18RR KG was estimated through extrapolation. To achieve this, query subsets of varying sizes, $|Q| = \{16, 31, 46, 61, 76, 91, 106, 121, 136, 151\}$, were sampled from WN18RR. Each subsets were probed using identical tail candidate set as the DEER's compute time experiment, thereby $|\mathcal{E}| = 40{,}943$. A linear regression model was fitted to the observed probing time, the obtained gradient and the y-intercept of the model was used to estimate the time required to probe the entire WN18RR KG using GPT2 under BEAR. The compute time experiment was repeated 3 times to obtain mean and variance for the observed data. Few-shot triplets for query encoding were randomly resampled every time. BEAR's compute time was measured using the library[4] published by Wiland et al. (2024).

C.2 Token Length Experimental Setup Detail

Tail entity scores were obtained using the OPT model and its tokenizer was used to compute the token length of the tail entity name.

C.3 LAMA Agreement Experimental Setup Detail

As discussed in Section 5.3, DEER utilizes few shot example, and to ensure an identical bias when comparing DEER with LAMA, LAMA was also prompted with a prober template instead of the usual, cloze-style questions. The tail-entities were ranked in an identical manner to the original work, where they are ranked by the log-likelihood of the respective token. Since the sentence encoding generated by PromptEOL, is the vector that is typically used for decoding the next token, the query encoding vector of DEER was used to calculate the log-likelihood of the next tokens to calculate the LAMA rank as illustrated in Figure 8.

[4] https://github.com/lm-pub-quiz/lm-pub-quiz

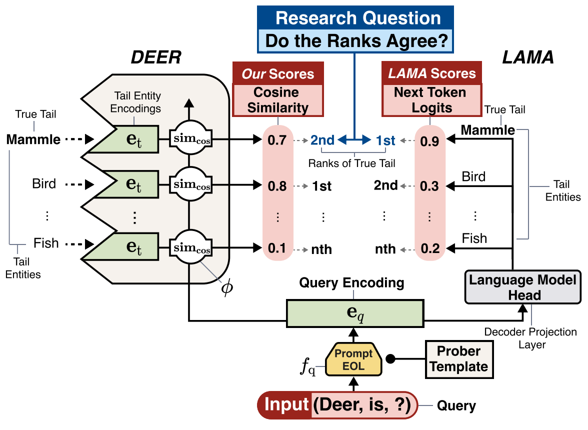

Figure 8: LAMA Agreement Experiment Diagram.

D Full Results of the Experiments

Table 4 shows the full result of the LAMA Agreement experiment. The statistics of the datasets are provided in Table 3.

E Supplementary Results and Analysis

E.1 Token Length Bias

Figure 10 compares BEAR rank against DEER rank using facist grid of scatter plots. In each scatter plots, points within the top left quadrant and the bottom left quadrant corresponds to triplets where BEAR and DEER disagreed when evaluating under the Hit@10 metric. The top left corresponds to Hit@10 for DEER but Miss@10 for BEAR, and the bottom right corresponds to Miss@10 for DEER but Hit@10 for BEAR. Qualitatively, we find observe that the top left quadrants is occupied by points with large token length, e.g. token length $\geq$ 4, where as the bottom left quadrant is occupied by points with lower token length, e.g. token length $\leq$ 4. This suggests that the points of disagreement between DEER and BEAR is partially caused by the presence of token-length bias in BEAR and lack there of in DEER.

E.2 LAMA Hit@k vs DEER Hit@k

To investigate if correlation in ranks between LAMA and DEER translates to an alignment in their Hit@k metrics, the values obtained in the

LAMA Agreement experiment were used to compute the Hit@k of each models, and compared. The results is shown in Table 6.

E.3 BEAR Agreement Experiment

Experimental Setup Pearson correlation between the predicted ranks by DEER and BEAR was measured using a subset of WN18RR and OPT models of varying sizes. To construct the subset, 500 triplets were randomly sampled from the test set of WN18RR, entities that appeared within the subset were used as the tail entity candidates.

Result Table 7 shows the results of the experiment. As shown in Figure 10, a strong corelation for OPT-6.7B was observed with $r = 0.642$ and $p < 10^{-58}$. However, lower parameter models, 125M-1.3B, showed lower correlation with $r = [0.138, 0.483]$. As discussed in Appendix E.1, we partially attribute its cause to the token length bias.

E.4 BEAR Agreement Experiment

Experimental Setup Pearson correlation between the predicted ranks by DEER and BEAR was measured using a subset of BEAR dataset[5] (Wiland et al., 2024) and OPT-6.7B. As the etity candidates are defined per each relation types, the two probes were compared on four randomly selected relation types, p272, p344, p466 and p1412. A scatter plot of the comparisons was made in a linear scale, as the number of tail entity candidates is small, with maximum candidates of 60, and the correlation was also calculated in a linear scale.

Result Figure 11 shows the results of the comparison. Qualitatively, the quality of agreement varied across relation types. With relation types p272 and p1312 demonstrating Pearson correlation of $r = 0.345$ and $r = 0.361$, where as relation type p343 demonstrated lower correlation with $r = 0.150$ and $p = 0.67$.

[5]https://github.com/lm-pub-quiz/BEAR

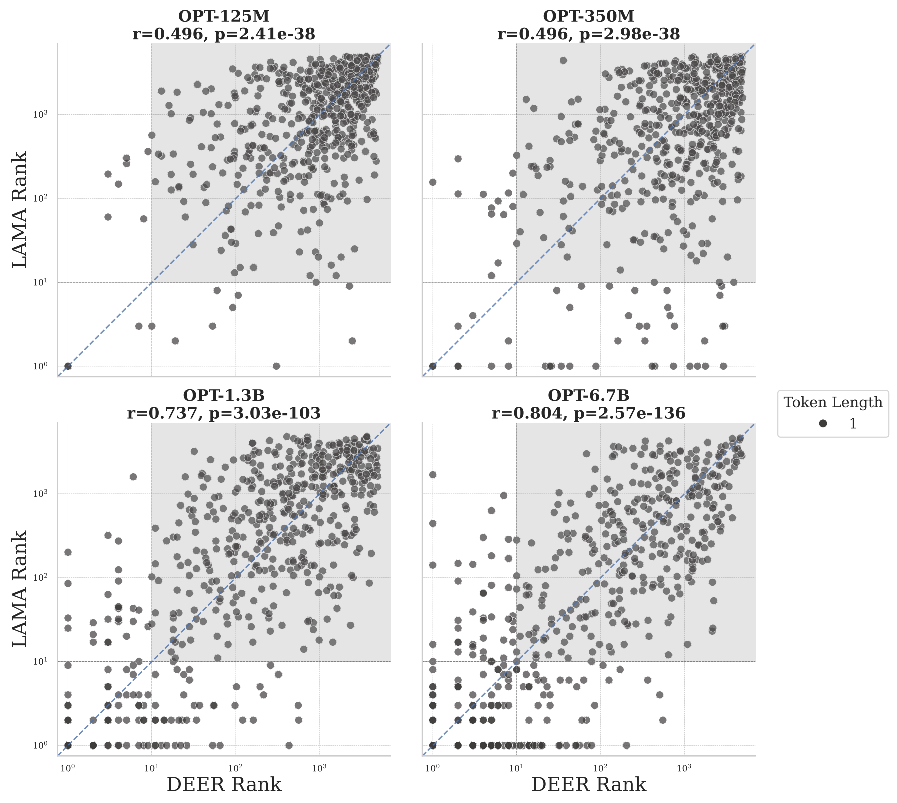

Figure 9: A facist grid of scatter plots comparing LAMA's predicted rank against DEER's for OPT models of parameter sizes {125M, 350M, 1.3B and 6.7B}.

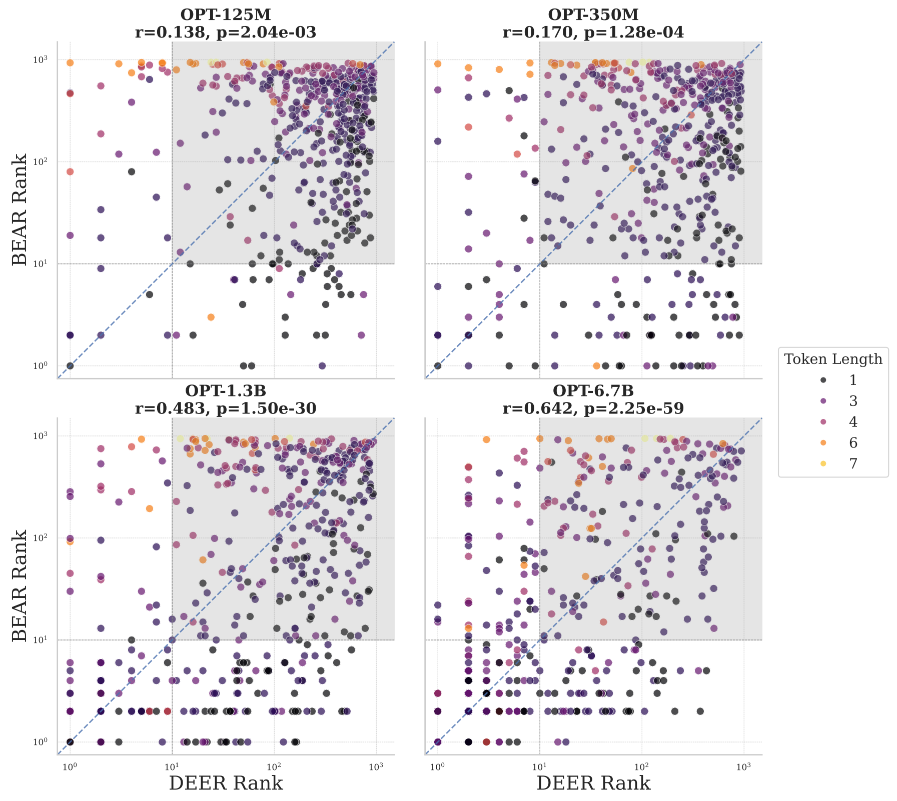

Figure 10: A facist grid of scatter plots comparing BEAR's predicted rank against DEER's for OPT models of parameter sizes {125M, 350M, 1.3B and 6.7B}. Interpretation of the plot is given in Appendix E.1 of the Appendix.

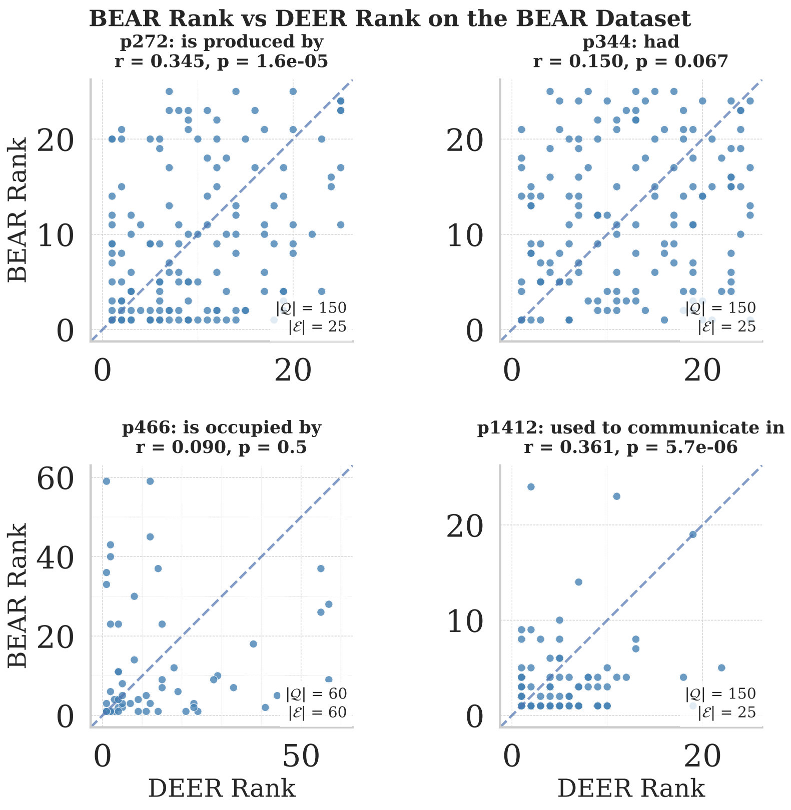

Figure 11: Comparisons between BEAR Rank and DEER Rank on the BEAR Dataset. The comparison was made in a linear scale to account for the lower number of tail entity candidates, with maximum number of candidates of 60. OPT-6.7B was used for all comparisons.

Dataset Name	$\lvert\mathcal{Q}\rvert$	$\lvert\mathcal{E}\rvert$
Compute Time Experiment	93003	40944
Token Bias Experiment	500	948
LAMA Agreement Experiment	596	4948
BEAR Agreement Experiment	500	948

Table 3: Statistics of the dataset used in each experiments.

PLM Name	Rank		Log(Rank)	
	r	p-value	r	p-value
OPT-125M	0.466	1.75×10^{-33}	0.373	3.82×10^{-21}
OPT-350M	0.387	8.86×10^{-23}	0.612	1.30×10^{-62}
OPT-iml-1.3B	0.551	1.268×10^{-48}	0.767	2.46×10^{-116}
OPT-1.3B	0.472	1.87×10^{-34}	0.726	1.25×10^{-98}
OPT-6.7B	**0.596**	$1.59{\times}10^{-58}$	**0.806**	$1.6{\times}10^{-137}$
OPT-iml-30B	0.462	$6.75{\times}10^{-33}$	0.710	1.61×10^{-92}
OPT-30B	0.487	$7.56{\times}10^{-37}$	0.763	$9.63{\times}10^{-115}$

Table 4: The full result of the LAMA agreement experiment, showing the pearson's correlation between the rank of the correct tail-entity predicted by DEER and LAMA, along with its correlation in log scale.

PLM	Pearson Correlation Log(LAMA Rank) vs Log(DEER Rank)	p value
OPT-125M	0.496	2.41E-38
OPT-350M	0.496	2.98E-38
OPT-1.3B	0.737	3.03E-103
OPT-6.7B	0.804	2.57E-136

Table 5: Table of Pearson correlation between LAMA rank and DEER rank in log scale across a range of parameters.

Model Name	Hit@1	Hit@10	Hit@100
BEAR-125M	1%	9%	27%
DEER-125M	2%	7%	24%
BEAR-350M	4%	17%	39%
DEER-350M	1%	9%	35%
BEAR-1.3B	6%	34%	55%
DEER-1.3B	5%	21%	50%
BEAR-6.8B	10%	52%	71%
DEER-6.8B	7%	51%	78%

Table 6: A table of Hit@{1, 10, 500} and MRR of LAMA and DEER calculated from the result of the LAMA Agreement Experiment. The standard errors are shown in parentheses. 95% confidence interval of a metric s can be calculated using $\mathrm{CI}_s = s \pm (1.96 \times \mathrm{SE}_s)$, where SE is the standard error. Notice that most metrics of super-billion parametr models differ within the confidence interval. Metrics that differ within this interval are shown in bold.

PLM	Pearson Correlation Log(BEAR Rank) vs Log(DEER Rank)	p value
OPT-125M	0.138	0.00204
OPT-350M	0.170	0.000128
OPT-1.3B	0.483	1.50E-30
OPT-6.7B	0.642	2.25E-59

Table 7: Table of Pearson correlation between BEAR rank and DEER rank in Log scale across a range of parameters.