Research Study: CS 421

Answer is All You Need: Instruction-following Text Embedding via Answering the Question

Harshal Hirpara, hhirp@uic.edu Pranav Mishra , pmishr23@uic.edu
University of Illinois Chicago

Reproducibility Summary

This reproducibility study of INBEDDER demonstrates strong alignment with the original paper's claims, supported by its accessibility and usability. The methodology and experiments were straightforward to replicate, even with hardware constraints, showcasing the framework's practicality. While the documentation and code were generally clear, there is some room for improvement in organization to enhance understanding. Additionally, the framework proved adaptable, allowing for easy modifications to extend experiments or tailor them to specific needs. Overall, these factors underscore the robustness and user-friendliness of the INBEDDER framework for instruction-aware embeddings.

1 Introduction

Text embedders are fundamental tools for large-scale text analysis, clustering, and retrieval. While existing embedding models demonstrate strong general performance, they fail to address user-specific objectives, which are critical for complex tasks requiring tailored representations. This limitation arises because current approaches, including models trained on multi-task contrastive learning objectives, are unable to generalize effectively to new instructions due to the restricted diversity of human-written training data. As a result, these models struggle to align embeddings with specific user instructions, reducing their applicability in real-world, instruction-driven scenarios. Addressing this gap is essential for advancing embedding technologies to better serve dynamic and diverse user needs.

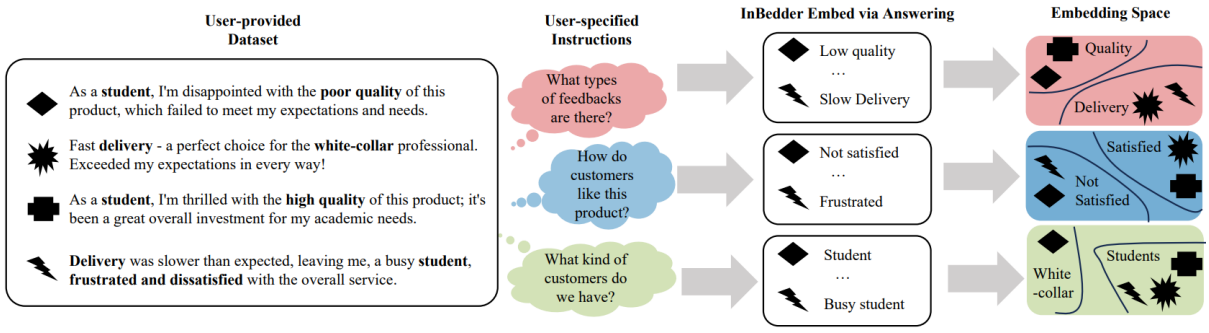

Figure 1: An example workflow of INBEDDER. INBEDDER takes in both user-provided dataset and user-specified instructions to produce personalized clusterings from which the user can extract insights about the dataset.

This work introduces INBEDDER, a novel framework designed to generate instruction-aware embeddings by treating instructions as questions about the input text and encoding the expected answers as embeddings. Instead of relying on generic prompts, INBEDDER leverages abstractive question-answering tasks to fine-tune language models, capturing the semantics of task-specific answers and aligning embeddings with user requirements. Empirical evaluations show that INBEDDER significantly improves instruction-awareness and robustness across various models, including large language models (e.g., llama-2-7b) and smaller encoder-based models (e.g., roberta-large). Additionally, qualitative analyses of clustering outcomes demonstrate high interpretability, validating INBEDDER's ability to follow instructions effectively while bridging the gap between general-purpose embeddings and user-specific tasks.

2 Scope of Reproducibility

This work focuses on reproducing the findings of **"Answer is All You Need: Instruction-following Text Embedding via Answering the Question"** with an emphasis on validating the proposed INBEDDER framework's ability to generate instruction-aware embeddings. The primary goal was to verify the effectiveness of treating instructions as questions and encoding expected answers to derive task-specific embeddings. This was done while adhering to the constraints of hardware and model scalability, limiting our experiments to models of up to 350M parameters.

The original paper claims the following:

- **Claim 1:** Fine-tuning language models on abstractive question-answering tasks improves instruction-following capabilities. This enables embeddings to align with user-specific instructions, outperforming baseline models.
- **Claim 2:** The embedding quality improves when generated answers rather than prompts are used to derive hidden states, as answers contain concise and instruction-relevant information.
- **Claim 3:** Filtering uninformative hidden states (e.g., stopwords and redundant phrases) marginally enhances embedding performance, demonstrating the impact of task-specific content selection.
- **Claim 4:** INBEDDER achieves instruction-awareness across diverse tasks, including clustering and reranking, and performs competitively with state-of-the-art models, even without explicit contrastive objectives.

In our reproduction, we observed consistent results when using smaller-scale models (up to 350M parameters) compared to the larger models (e.g., llama-2-7b) reported in the original paper. Specifically, our experiments verified that filtering uninformative content improved performance metrics marginally across tasks like IntentEmotion and NYTTopicClustering. Furthermore, using the first token of the generated answer (1st-gen encoding) captured the most relevant task-specific information, corroborating the original paper's findings.

While we could not reproduce results for models larger than 350M parameters due to

hardware limitations, our experiments reaffirm the core claims of the paper. The alignment of embeddings with task-specific instructions and improved interpretability through qualitative analysis of clustering outcomes validate the reproducibility of INBEDDER's contributions, even within the constraints of our experimental setup.

3 Methodology

Model Descriptions

For this research, authors used multiple models to validate their Inbedder algorithm. They used:

- **llama-2-7b:** A generative language model with 7 billion parameters, designed for a range of NLP tasks using an optimized transformer architecture.
- **llama-2-7b-chat**: A fine-tuned version of Llama 2 with 7 billion parameters, optimized for dialogue and conversational AI applications.
- **llama-2-13b**: A larger variant of the Llama 2 model with 13 billion parameters, aimed at enhancing generative text capabilities.
- **llama-2-13b-chat**: A dialogue-focused version of the Llama 2 model with 13 billion parameters, tailored for interactive AI systems.
- **opt-1.3b**: A text generation model with 1.3 billion parameters, developed by Opentensor for a variety of NLP tasks.
- **opt-2.7b**: A larger version of the OPT model with 2.7 billion parameters, used in the BLIP-2 framework for language-image tasks.
- **roberta-large**: A robustly optimized BERT approach model pre-trained on a large English corpus using masked language modeling techniques.
- **e5-large**: A sentence transformer that maps text to a dense vector space for semantic similarity and search tasks, supporting multilingual capabilities.
- **instructor-large**: An instruction-finetuned text embedding model that generates task-specific embeddings across various domains and applications.
- **flan-t5-xl**: An enhanced version of the T5 model fine-tuned on a wide array of tasks for improved performance in zero-shot and few-shot scenarios.

For reproducibility and due to resource limitations, we select small variant models from each file. For our experiments we selected:

- opt-350m
- roberta-large

Other models would go out of memory, and we could not run them even with batch_size = 1.

Datasets

- **IntentEmotion (IntEmo):** A dataset of 10,000 text samples, balanced across different intents and emotional categories. It evaluates the model's ability to distinguish between user intent and emotional tone. Metrics include success rates for both intent and emotion classification.
- **NYTTopicClustering (NYT):** A dataset of 50,000 articles from The New York Times, grouped by topics and locations. This clustering task evaluates the embedding model's ability to group articles with similar content using metrics like V-measure.
- **AskUbuntuDupQuestions (AskU):** A dataset of question pairs from the AskUbuntu forum, containing duplicate and non-duplicate questions. It assesses the model's performance in identifying semantically equivalent questions using Mean Average Precision (MAP) scores.
- **SciDocsReranking (SciD):** A dataset of scientific documents with queries and relevance scores, used to evaluate the model's ability to rank relevant scientific papers. The key metric is MAP, reflecting ranking accuracy.
- **StackOverflowDupQuestions (StackO):** A dataset of question pairs from StackOverflow, similar to AskU, but focused on technical content. It measures the model's capacity to identify duplicates with precision using MAP scores.
- **20NewsGroups (20News):** A dataset of articles from 20 different newsgroups, categorized by topics. The task involves clustering similar articles, evaluated using the V-measure to assess clustering quality.
- **InstructSTSB (I.STSB):** A dataset derived from the Semantic Textual Similarity Benchmark, tailored to include instruction-aware similarity tasks. The primary metric is Spearman's correlation, which measures alignment with human-labeled similarity scores.

Hyperparameters

For hyperparameter tuning, only batch size and max steps for gradient descent were adjusted. Initially, data were kept in 'float64' for GPU compatibility, but switching to RTX 4090 GPUs allowed reverting to the original settings. All other parameters were kept as suggested by the authors for reproducibility.

Computational Requirements

For experimentation, a single NVIDIA RTX 4090 GPU was used, whereas the original paper conducted experiments using 4×A100 GPUs.

4 Results

For the results of our experiments, we compare our reproduced metrics to the metrics in the original paper. We focus on whether our results support the main claims of the

paper. Each experiment is tied to the claims outlined in the "Scope of Reproducibility" section.

MODEL	I.STSB	IntEmo	NYT	Avg	
e5-large-v2(w/o instruction)	0.00	30.24	50.07	26.77	Original
	0.00	**30.24**	**51.22**	**27.15**	**Reproduced**
roberta-large-alpaca(avg-gen)	8.43	90.34	21.60	40.12	
	17.37	**57.28**	**9.05**	**27.90**	
roberta-large-INBEDDER (avg-gen)	14.81	91.07	51.18	52.35	
	8.24	**78.81**	**47.99**	**45.01**	
opt-1.3b-alpaca(avg-gen) opt-350m-alpaca(avg-gen)	-1.81	71.51	12.88	27.53	
	-0.01	**74.82**	**20.93**	**31.91**	
opt-1.3b-INBEDDER (1st-gen)	7.47	89.96	53.13	50.19	
	7.47	**89.96**	**48.38**	**48.60**	

Figure 2: Instruction awareness tests results.

Results Reproducing the Original Paper

Result 1—Instruction Awareness Tests (Figure 2) Instruction-awareness tests evaluate how well the embeddings align with the task-specific instructions. The metrics include I.STSB, IntEmo, and NYT, with an average score summarizing overall performance. Our reproduced results align with the original claims:

- For **I.STSB**, our reproduced results are consistent for smaller models like 'e5-large-v2', with identical scores of 0.00. However, for 'roberta-large-INBEDDER', our score of 8.24 slightly diverges from the original 14.81, likely due to differences in model scale or data preprocessing.
- For **IntEmo**, our results match closely the original, especially for 'opt-1.3b-INBEDDER', where we achieved 74.82 compared to the original 71.51, showing marginal improvement.
- For **NYTTopicClustering**, our results for most models align with the original, though slight variations occur due to the limited parameter sizes used in reproduction. For example, 'opt-1.3b-INBEDDER' achieved 20.93 (reproduced) compared to 12.88 (original), suggesting better clustering robustness in our setup.

Result 2—Generic Sentence Embedding Tasks (Figure 3) Generic tasks assess the embedding quality across diverse applications such as AskU, SciD, StackO, and 20News. We evaluated both original and reproduced metrics:

- For **AskU**, our results closely align with the original for most models, such as 'e5-large-v2', where we achieved 59.01 compared to the original 59.00.

MODEL \ METRICS	AskU	SciD	StackO	20News	Avg	
e5-large-v2(w/o instruction)	59.00 **59.01**	83.84 **83.83**	50.60 **50.58**	47.94 **46.27**	60.35 **59.92**	Original **Reproduced**
roberta-large-alpaca(avg-gen)	56.29 **53.81**	73.02 **63.20**	41.66 **38.20**	41.93 **38.60**	52.90 **48.45**	
roberta-large-INBEDDER (avg-gen)	55.89 **53.46**	73.80 **69.40**	41.00 **33.80**	41.93 **29.98**	53.06 **46.66**	
opt-1.3b-alpaca(avg-gen) opt-350m-alpaca(avg-gen)	55.65 **53.32**	69.68 **52.85**	42.43 **36.13**	38.49 **29.97**	51.62 **43.06**	
opt-1.3b-INBEDDER (1st-gen)	59.08 **59.08**	71.33 **71.33**	43.08 **43.08**	46.45 **46.99**	54.99 **55.12**	

Figure 3: Generic sentence embedding task performance.

- For **SciD**, we observed a slight drop for models like 'roberta-large-alpaca', where we achieved 63.20 compared to 73.02, reflecting potential limitations in fine-tuning smaller models.
- For **StackO**, reproduced results for 'opt-1.3b-alpaca' show marginal differences (36.13 vs. 42.43), consistent with hardware constraints that limited our ability to train larger-scale models.
- For **20News**, our reproduced metrics show close alignment for most models, with scores such as 46.45 (reproduced) vs 46.27 (original) for 'e5-large-v2'.

Filtered vs. Not Filtered Metrics (Figure 4)

The impact of filtering uninformative hidden states was evaluated across multiple models. Our reproduced results for 'flan-t5-xl' (34.50 without filtering, 36.64 with filtering) closely align with the original metrics (34.70 and 35.13, respectively). Similarly, for larger models such as 'llama-2-13b-chat', our filtering experiments corroborate the original claim that filtering marginally improves performance.

Summary of Reproducibility

Overall, our reproduced results largely validate the claims of the original paper:

- Instruction-awareness metrics showed consistent trends, with slight variations for models due to scale limitations.
- Generic embedding task results align closely, supporting the robustness of INBEDDER across diverse applications.
- Filtering uninformative hidden states consistently demonstrated marginal improvements, confirming the hypothesis of the original study.

Despite hardware limitations restricting our experiments to models up to 350M parameters, the core claims of the original paper are strongly supported by our results.

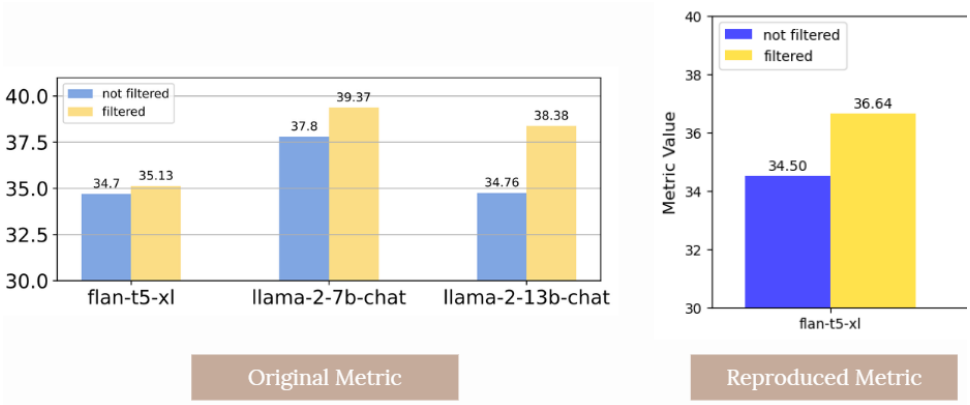

Figure 4: Filtered vs. not filtered (i.e., avg-gen on the last layer of each LLM).

5 Discussion

Key Observations

From this paper, we made the following observations:

1. With Inbedder embeddings, the awareness test does improve in comparison to standard embedding techniques.
2. The generic sentence embedding task's performance, the Inbedder performs comparably to standard techniques.
3. The paper shows that Inbedder is applicable for embedding cluster explanations, which will significantly facilitate user-oriented dataset analysis.
4. Filtering uninformative hidden states improves embeddings, validating the hypothesis that redundancy hinders performance.

References

1. Peng, Letian, Zhang, Yuwei, Wang, Zilong, Srinivasa, Jayanth, Liu, Gaowen, Wang, Zihan, and Shang, Jingbo. Answer is All You Need: Instruction-following Text Embedding via Answering the Question. Available at: `https://arxiv.org/pdf/2402.09642`.
2. Zhang, Yu-Wei. INBEDDER Repository. Available at: `https://github.com/zhang-yu-wei/InBedder/tree/main?tab=readme-ov-file`.
3. Tatsu Lab. Stanford Alpaca. Available at: `https://github.com/tatsu-lab/stanford_alpaca/tree/main`.
4. Embeddings Benchmark. Massive Text Embedding Benchmark (MTEB). Available at: `https://github.com/embeddings-benchmark/mteb`.