Instruction Embedding: New Concept, Benchmark and Method for Latent Representations of Instructions

Anonymous ACL submission

Abstract

Instruction data is crucial for improving the capability of Large Language Models (LLMs) to align with human-level performance. Recent research LIMA demonstrates that alignment is essentially a process where the model adapts instructions' interaction style or format to solve various tasks, leveraging pre-trained knowledge and skills. Therefore, for instructional data, the most important aspect is the task it represents, rather than the specific semantics and knowledge information. The latent representations of instructions play roles for some instruction-related tasks like data distillation for instruction tuning and prompt retrieval for in-context learning. However, they are always derived from text embeddings, encompass overall semantic information that influences the representation of task categories. In this work, we introduce a new concept, instruction embedding, and construct Instruction Embedding Benchmark (IEB) for its evaluation. Then, we propose baseline method, prompt-based instruction embedding (PIE), to make the instruction embeddings more attention on task rather than whole semantic information. The evaluation of PIE, alongside other embedding methods on IEB, demonstrates its superior performance in accurately identifying task categories. Moreover, the application of PIE in downstream tasks showcases its effectiveness and suitability for instruction-related tasks.

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable proficiency in generating responses capable of addressing specific tasks according to provided instructions. Initially pre-trained for wide-ranging capabilities, they are subsequently fine-tuned using instruction-following datasets to enhance their ability to align with human preferences. LIMA has proved that alignment can be viewed as a straightforward process in which the model just learns the style or format for interacting with users to solve particular problems, where the knowledge and capabilities have already been acquired during pre-training (Zhou et al., 2023). Building on this assumption, even a small quantity of carefully selected instruction data can substantially enhance model alignment performance through instruction tuning.

```
Sample1 - different tasks

• Tell me the main idea of this article.
• Tell me the gender of the author of this blog
  post.

Similarity with text embedding: 0.9943
Similarity with instruction embedding: -0.0254

Sample2 – similar tasks

• Create a poem with at least 5 lines, rhyming
  pattern aabb.
• Write a limerick based on the following noun.

Similarity with text embedding: 0.3239
Similarity with instruction embedding: 0.8287
```

Figure 1: The cosine similarity between instructions. Text embeddings are from the last token of Llama and instruction embeddings are from proposed PIE.

Based on this, recent works dedicate to data distillation, seeking to extract compact subsets from extensive instruction datasets (Wu et al., 2023a; Cao et al., 2023; Chen et al., 2023a). During that process, one crucial factor is instruction diversity, to gain broad alignment abilities through training on diverse tasks (Wei et al., 2023; Chen et al., 2023a; Wu et al., 2023a). The process inherently relies on computing similarities among instructions, and as such, the effectiveness of preserving diversity heavily depends on the quality of the latent representation of instructions.

Text embeddings, which play a crucial role in a variety of NLP tasks such as semantic textual similarity (Agirre et al., 2012; Cer et al., 2017;

Marelli et al., 2014) and information retrieval (Mitra et al., 2017; Karpukhin et al., 2020), can serve as an option for representing instructions. Previous studies (Wang et al., 2024) obtain text embeddings by directly taking the last token vector from large generative models. However, when it comes to the embeddings of instructions, the key focus should lie in identifying task categories rather than capturing overall semantic information. This is because, as mentioned earlier, instruction fine-tuning help models learn how to interact with users across different tasks, rather than the specific capabilities and knowledge imparted by the instructions. Therefore, task diversity is far more important than semantic diversity for instructions. Figure 1 shows the case where traditional text embedding methods exhibit high overall semantic and syntactic similarity between two samples which actually represent completely different tasks, but low similarity when they represent similar task.

In this work, we propose a new concept called instruction embedding, a specialized subset of text embedding that prioritizes task identification for instructions over the extraction of sentence-level semantic information. We propose a new benchmark for instruction embedding evaluation, namely IEB. Different from previous text embedding benchmark that only considered the semantic textual similarity, IEB (Instruction Embedding Benchmark) is labeled by task categories of instructions. Inspired by that key instruction words especially verbs are highlighted through instruction tuning (Wu et al., 2023b), we first extract verb-noun pairs to clarify category, then manually select and label instructions with other syntactic structures. IEB totally contains 7.6k samples dispersed across 1k categories, which can be also for fine-tuning embedding models.

To stimulate the LLM to generate better instruction embedding, we propose a prompt-based instruction embedding method, PIE. It is a learning-free method that leverages the template to obtain instruction embeddings by directing the model's attention towards the task type represented by the instructions. Additionally, our method is fully compatible with fine-tuned settings. Contrastive learning is widely used for training embedding models, where the positive pairs are hard to extract. In our study, the explicit category information available in IEB enables the straightforward extraction of positive samples by directly selecting two instances from the same category. Furthermore, we construct hard negative samples by selecting instances from categories that share identical verbs or nouns, enhancing the challenge of differentiation. Figure 1 shows that proposed PIE effectively distinguishes whether two instructions refer to the same task without being affected by other semantic information.

We evaluate PIE and other embedding baselines on IEB with two metrics, which shows that PIE can largely outperform baselines and precisely identify the task categories. We also conduct some downstream tasks like data distillation for instruction tuning (Wu et al., 2023a) and prompt retrieval for in-context learning (Su et al., 2023), where the superior results demonstrate that the proposed instruction embedding method is more suitable for instruction-related tasks. Besides, we verify that models after instruction fine-tuning can deliver better embeddings.

To summarize, this work includes the following contributions: (1) We propose instruction embedding, a novel concept that focuses on task identification rather than sentence-level semantic information. Correspondingly, we present publicly available benchmark IEB for its evaluation and further training. (2) We provide a prompt-based method for instruction embedding, which can be conducted in both learning-free and learning manner. (3) We evaluate PIE and baselines on IEB and reveal the effectiveness of proposed method. We also show PIE can be a better substitution for downstream tasks.

2 Related Work

2.1 Text Embeddings

Text embeddings, encapsulating vital semantic and syntactic details, are pivotal in Natural Language Processing (NLP). The quality of learned embeddings directly influences downstream tasks, highlighting the significance of text embedding learning. Current research on text embeddings primarily focuses on semantic modeling using transformer-based pretrained language models (PLMs) (Gao et al., 2021; Jiang et al., 2022; Li and Li, 2023). We argue that for compressing instruction datasets while maintaining task diversity, instruction embeddings should prioritize task-specific information within the instructions rather than emphasizing overall semantic information.

2.2 Instruction Tuning

Instruction tuning is a crucial method to overcome the challenge of instruction following for large language models (LLMs). LIMA (Zhou et al., 2023) argues that the diversity and quality of instruction data matters more than quantity and demonstrate that even a small quantity of carefully selected instruction data can substantially enhance model alignment performance through instruction tuning. Building upon the insights from LIMA, endeavors are dedicated to compressing instruction datasets: ALPAGASUS (Chen et al., 2023b) utilizes ChatGPT to filter out low-quality data, Li et al. selects high quality examples through an iterative self-curation process, DIVERSEEVOL (Wu et al., 2023a) iteratively samples training data using the current embedding space to preserve diversity in the sampled subset. However, previous efforts fall short in explicitly maintaining task diversity in the training subset while reduce data quantity.

2.3 Embedding Benchmark

The Semantic Textual Similarity (STS) tasks (Agirre et al., 2012; Cer et al., 2017; Marelli et al., 2014) are commonly employed to evaluate the quality of text embeddings, complemented with transfer tasks and short text clustering tasks (Conneau and Kiela, 2018; Xu et al., 2023; Muennighoff et al., 2023) to further illustrate the superiority of learned sentence representations. However, previous benchmarks are not tailored to instruction-style corpora and primarily assess the semantic modeling abilities of text embeddings, rendering them less suitable for evaluating instruction embeddings.

3 The IEB Benchmark

We present instruction embedding benchmark, IEB, for assessing the quality of the latent representation of instructions. In contrast to current text embedding benchmarks that assess similarity, the primary focus for the space of instruction embeddings is task differentiation based on the given instructions. Therefore, we propose a new benchmark that annotates instructions with their respective tasks.

3.1 Data Extraction

For convenience and authenticity, we derive samples from established datasets. Specifically, we adopt three extensively recognized instruction-tuning datasets: DatabricksDolly (Conover et al., 2023), Alpaca data (Taori et al., 2023), and Self-instruct data (Wang et al., 2023). Labeling instructions entirely through manual effort or large language models will incur significant costs. Therefore, it is first necessary to conduct coarse-grained grouping and filtering based on rule-based policies. Wu et al. (2023b) proves that instruction fine-tuning enables models to recognize key instruction words, which leads to the generation of high-quality responses. Furthermore, it also encourages models to learn word-word relations with instruction verbs. Inspired by these two findings, we argue that verbs or other key words are crucial in identifying the task denoted by an instruction, where the types of key words can be effectively determined through syntactic analysis. Thus, following Wang et al. (2023), we employ the Berkeley Neural Parser[1] (Kitaev and Klein, 2018; Kitaev et al., 2019) for parsing the instructions.

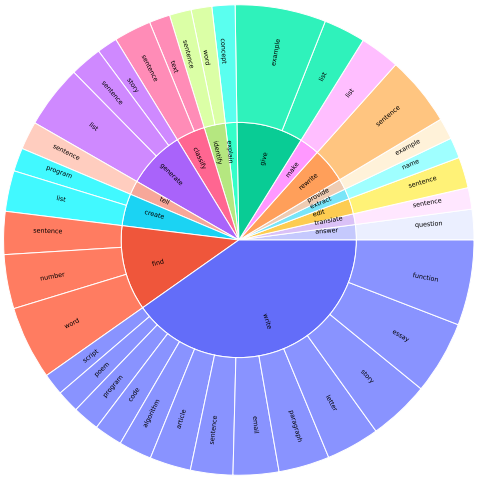


Figure 2: The verb-noun distributions in IEB.

After manual observation and considering the task category requirements, instructions can generally be divided into the following four groups through corresponding parsing tag recognizer:

VP (VB+NN) denotes verb phrase structure where the verb is closest to the root of the parse tree and directly links to noun. Instructions with this structure account for more than 80% of the total number before filtering. We categorize each instruction based on its verb-noun combination, identifying it as a specific task type, such as *write story* or *generate sentence*. After restoring the verb tense and singular form of nouns, we classify instructions with the same verb-noun combination into the same category. We find that low-frequency

[1] https://parser.kitaev.io/

Parsing Tag	Task Annotation	Examples
VB	verb + noun	Write an essay about my favourite season. In 100 words or less, tell a story about the consequences of the choices people make.
SBARQ	wh- + knowledge	What is the difference between machine learning and deep learning? Why are numbers written in the base 10 system instead of a smaller base system? How is a liquid chromatography differs from gas chromatography? Who was the coach for the Chicago Bulls when they won the NBA championship? When was the "No, They Can't" book released? Where was 52nd International Film Festival of India held?
	what + math	What is the result when 8 is added to 3? What is the value of (x - y)(x + y) if x = 10 and y = 15?
SQ	yes/no + knowledge	Was Furze Hill an established community in the 19th century? Did Sir Winston Churchill win the Nobel Peace Prize?
	yes/no + task	Is the following statement a valid definition of the term noise pollution? Does the information provided in the article support a vegetarian diet?
Others	verb + knowledge	Summarize the Challenger Sales Methodology for me. Describe the Three Gorges Dam of China.
	verb	Translate "Bonjour" into English. You need to translate "I have been to Europe twice" into Spanish.
	verb + math	Multiply 12 and 11. Simplify 2w+4w+6w+8w+10w+12.
	noun + knowledge	Short Summary about 2011 Cricket World Cup. iPhone 14 pro vs Samsung s22 ultra.

Table 1: Task categories with examples of IEB.

samples have a higher probability of being noisy, so we discard categories with fewer than 10 samples. We plot the top most common root verbs and their direct noun objects in Figure 2.

SBARQ is direct question introduced by a wh-word or a wh-phrase. It can be divided into two main categories: knowledge-based questions led by six interrogative pronouns (e.g., what, when, where, ...) and math problems introduced by *what*. Unlike instructions in the VP (verb phrase) form, we define categories in the form of interrogative pronoun combing knowledge/math. This is because, considering they all involve asking about knowledge or math problems, further subdividing into noun categories is not very meaningful. For each category, we manually select around 50 samples.

SQ is inverted yes/no question. It can also be divided into two main categories: knowledge-based questions and task-oriented questions. Similarly, the task label is annotated as yes-no combing knowledge/task and we select around 50 samples for each category.

Others There are some other structures: verb phrase that lacks a direct connection to a noun and some rare cases which do not contain verbs, consisting only of noun phrases. We define these four categories:(1) Verb-led knowledge questions. For example, knowledge clauses guided by *summarize* and *describe*. (2) Single verb for tasks, e.g., *translate*.(3) Verb-led mathematical problems. For example, math problem clauses guided by *multiply* and *simplify*.(4) None phrase for knowledge questions. For each type, we randomly select around 10-50 samples.

Finally, the annotated task categories cover the vast majority of the instruction data and are shown with examples in Table 1.

3.2 Data Synthesis

In instruction data, we discover some complex compound sentences, e.g., *You are playing a game which requires you to roll two dice. Generate a sentence to describe the emotion of anticipation felt while waiting for the dice to stop rolling.* Although they are not predominant, they can serve as challenging examples in the benchmark. However, due to their relative difficulty in identification, we employ GPT-4-turbo to generate samples based on existing task category names, including verbs and their corresponding nouns. Subsequently, the generated compound instructions will be integrated into the categories.

3.3 Quality Control and Evaluate

Automatic Filtering Even though low-frequency samples have been discarded, the automatically constructed categories still contain some noisy data. Thus, we use GPT-4-turbo to check whether sam-

		Task Categories	Samples
EFT	Train	447	35634
	Test	63	5899
IFT	Train	502	33904
	Test	747	1064
Total		1012	76501

Table 2: Data statistics of IEB. EFT refers to embedding fine-tuning and IFT refers to instruction fine-tuning.

ples belong to its annotated category. About 23% samples are filtered out during this process.

Category Merging Considering that many verbs or nouns representing instructions are synonyms, e.g., *provide* and *give*, it would be inappropriate to classify them into different categories. Thus, we utilize WordNet [2] to extract the synonyms. We merge all categories where both nouns and verbs are synonyms to make the benchmark more robust.

Human Evaluation We randomly sample 200 examples and ask an expert annotator to evaluate whether samples belong to its annotated category. The results indicate that 92% of the sample categories are accurate.

3.4 Statistics

After constructing and filtering, we collect totally 1012 task categories with 76501 samples. Given the large volume of data, the benchmark data can also be used for training and testing instruction embeddings and instruction fine-tuning. Therefore, we have split it in a certain ratio, but it can be divided in any form as needed. Table 2 describes the statistics of the divided data.

4 Instruction Embedding

4.1 Why Instruction Embedding

Text embeddings are pivotal in numerous natural language processing NLP tasks. Traditional text embeddings are chiefly concerned with capturing the semantic content of texts, striving to encapsulate both the intrinsic meaning and the syntactic arrangement of sentences (Xu et al., 2023). Zhou et al. (2023); Wu et al. (2023b) prove that, for instructional data, the primary significance lies in the task it signifies by key instruction verbs, not the detailed semantics and knowledge. Therefore, our instruction embedding proposed in this paper is designed to prioritize modeling the task categories expressed by instructions rather than delving into the semantic intricacies of the text. When employing traditional text embeddings for data distillation, semantic information might introduce interference during the distillation process. For instance, if two instructions propose different tasks for similar objects, as illustrated in Figure 1, the semantic nuances could complicate the distillation process. Instruction embedding, this focused approach allows for a clearer delineation of the intended tasks, contributing to more effective data distillation processes.

4.2 Prompt-based Instruction Embedding

As mentioned above, guiding the model to generate embeddings that focus on task categories is critically important. Large pretrained language models have shown an impressive capacity to accomplish novel tasks solely by utilizing in-context examples or instructions (Brown et al., 2020). Inspired by (Jiang et al., 2022), we present a prompt-based instruction embedding method (PIE). By reformulating the sentence embedding task as the generation task, we can effectively use original LLaMA layers by leveraging the pre-trained knowledge. We manually design some templates, as shown in Appendix A.2. The hidden states of last token will be represented for the embedding of instruction.

4.3 Embedding Finetuning

To further improve PIE performance, we fine-tune PIE-LLaMA on our embedding train set by contrastive learning (Hadsell et al., 2006) through the learning framework in SimCSE (Gao et al., 2021).

Let $\mathcal{D} = \{\mathbf{t}_i\}_{i=1}^{|\mathcal{D}|}$ denotes the embedding train set, where each $\mathbf{t}_i = \{t_{i1}, ..., t_{|\mathbf{t}_i|}\}$ represents a specific task category in $\mathcal{D}$, and each t_{ij} is an instruction instance of $\mathbf{t}_i$. During the training process, we take a cross-entropy objective with in-batch negatives (Chen et al., 2017; Henderson et al., 2017). For a given instruction t_{ij}, t_{ik} where $j \neq k$ is randomly sampled from $\mathbf{t}_i$ to make up a task-related instruction pair. In order to mitigate the risk of false negatives resulting from repetitive task categories among different pairs of instructions in batch, we randomly select several distinct tasks from all task categories each time. Subsequently, we sample instruction pairs from the corresponding instruction pools. Let h_{ij} and h_{ik} denote the representation of t_{ij} and t_{ik}, the learning objective for (t_{ij}, t_{ik}) with

[2] https://wordnet.princeton.edu/

Method	CP	ARI
Non-Finetuned on embedding train set		
LLaMA (non-finetuned)	0.6238	0.0596
PIE-LLaMA (non-finetuned)	0.7842	0.1115
Vicuna	0.5687	0.0464
PIE-Vicuna	0.8011	0.1182
Random	0.3670	0.0000
SimCSE	0.7328	0.0911
PromptBERT	0.3678	0.0005
Finetuned on embedding train set without hard negative sampling		
LLaMA - hard negative	0.8486	0.4468
PIE-LLaMA - hard negative	0.7797	0.4680
Finetuned on embedding train set		
LLaMA	0.8696	0.4548
PIE-LLaMA	**0.8915**	**0.6300**

Table 3: Results of embedidng fine-tuning experiment. We conduct instruction clustering task on various embedding methods, including each baseline method, non-fine-tuned llama-based embeddings, non-fine-tuned vicuna-based embeddings and fine-tuned llama-based embeddings. Besides, the ablation study result on hard negative sampling is also shown here.

a mini-batch of N pairs can be formulated as Eq 1

$$\ell_i = -log\frac{e^{sim(h_{ij},h_{ik})}/\tau}{\sum_{m=1}^{N} e^{sim(h_{ij},h_{mk'})/\tau}} \quad (1)$$

where τ is a temperature hyperparameter and $sim(h_1, h_2)$ is the cosine similarity $\frac{h_1^T h_2}{||h_1||\cdot||h_2||}$.

Hard negative sampling has been widely adopted in contrastive learning (Yuan et al., 2023), which has been demonstrated to enhance the effectiveness of contrastive learning. In this paper, we propose a hard negative sampling strategy based on verb-noun style instruction task categories: for instruction pair (t_{ij}, t_{ik}), if the task category of $\mathbf{t}_i$ is a verb-noun pair (v_i, n_i), then, instruction pair ($t_{i'j'}$, $t_{i'k'}$) of $\mathbf{t}'_i$, whose task category is (v_i, n'_i) is viewed as a hard negative pair of (t_{ij}, t_{ik}) and inserted to the training batch.

5 Experiment

5.1 Experimental Setup

Evaluation Details For the evaluation of instruction embeddings, we employ an instruction clustering task on the embedding test set of our proposed IEB benchmark, aiming to accurately group instructions from different tasks. Specifically, embeddings-based instruction clustering is conducted using k-means clustering based on the embeddings of given instructions, where k is predefined and its value equals to the number of task categories in the embedding test set. It is worth noting that the data sampling process for PIE fine-tuning in this paper is not traversing the training data; instead, it involves repeated random sampling of the training data. Here, we set the sampling step to 5k. Throughout the entire training process, approximately 80k instruction pairs are involved in the training. We utilize metrics such as Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) and Clustering Purity (CP) (Schütze et al., 2008) to assess the effectiveness of the task clustering process. These metrics offer insights into the quality of the clusters formed based on the instruction embeddings, providing valuable feedback on the performance and accuracy of our proposed embedding methodology. We implement our PIE method with LLaMA-7B (Touvron et al., 2023) which is called PIE-LLaMA. For the embedding pooling layers, unless stated otherwise, we utilize the average of hidden states from the last token across the last 2 layers.

Baselines We compare our PIE with two sentence embedding baselines, and random instruction clustering is also considered as a baseline.

SimCSE alleviates the anisotropy problem by separating negative pairs and optimizes alignment by pulling positive pairs closer together. For the comparison, we use unsupervised SimCSE-BERT$_{\text{base}}$.

PromptBERT finds original BERT(Devlin et al., 2019) can achieve reasonable performance with the help of the template in sentence embeddings (Jiang et al., 2022). For the comparison, we use unsupervised prompt-based BERT$_{\text{base}}$ (manual).

LLaMA simply takes the original instruction as input, which is now widely used for instruction embedding. It is used to compare

Layer	Model	CP	ARI
Last one	LLaMA	0.6002	0.0610
	PIE-LLaMA	0.7040	0.0819
Last two	LLaMA	0.6238	0.0596
	PIE-LLaMA	**0.7842**	**0.1115**
Mid	LLaMA	0.5973	0.0540
	PIE-LLaMA	0.6829	0.0664
First-Last	LLaMA	0.5860	0.0759
	PIE-LLaMA	0.7177	0.0763

Table 4: Results of pooling layer selection experiment. For all pooling layers, we take the average pooling of last token hidden states in each chosen hidden layer as the instruction embedding.

with PIE-LLaMA to reveal the effect of prompt in obtaining instruction embeddings.

Random instruction clustering randomly classifies instructions into different clusters.

Embedding Fine-tuning Details We fine-tune PIE-LLaMA on embedding train set and the configuration can be found in Appendix A.1. We evaluate the performance of fine-tuned embedding models and baselines through the instruction clustering task mentioned before. Besides, we replaced LLaMA-7B with Vicuna-7B-v1.5 (Chiang et al., 2023) to explore the impact of instruction fine-tuning on the model's ability to follow prompts in obtaining instruction embeddings.

5.2 Results and Observations

The Effectiveness of PIE Table 3 shows the experimental results, which demonstrate the remarkable power of LLaMA model, even the non-fine-tuned prompt-free LLaMA is almost comparable with SimCSE, let alone the PIE models and fine-tuned models. PIE achieves the best performance, both in learning-free and embedding fine-tuned modes.

Embeddings from Instruction Fine-tuning Models The quality of instruction embedding can be further improved when we use instruction fine-tuned model to conduct prompt-based instruction embedding: though Vicuna performs worse than non-fine-tuned LLaMA, Vicuna demonstrates stronger prompt-following ability and delivers better instruction embeddings when prompt is introduced. Furthermore, fine-tuning on the embedding training set leads to a significant improvement in model performance.

Visualization Results To better illustrate the superiority of PIE and the impact of fine-tuning, we visualize of the embeddings before and after fine-tuning in Figure 3. It is evident that embedding fine-tuning successfully enhances the performance of both LLaMA and PIE-LLaMA in terms of instruction clustering. This suggests that embedding fine-tuning does aid in extracting task information more effectively from instructions. Additionally, the fine-tuned PIE-LLaMA exhibits a more dispersed inter-class distribution and a more compact intra-class distribution than the fine-tuned LLaMA, demonstrating the positive guiding effect of the prompt method on extracting task information from instructions.

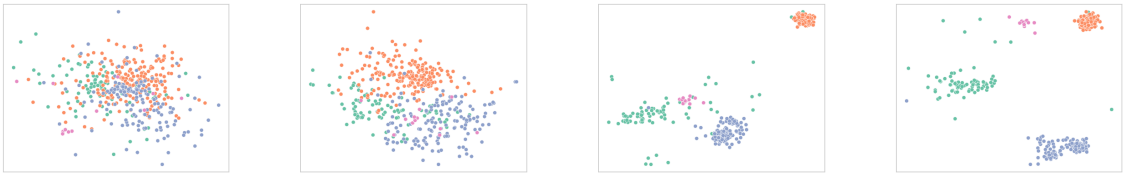
Ablation Study We also conducted an ablation study on our hard negative sampling strategy where we set sampling step to 10k to ensure consistent training data volume with hard negative sampling fine-tuning, the result is shown in Table 3. After removing hard negative sampling, we observed a notable decrease in the performance of both LLaMA and PIE-LLaMA. This underscores the pivotal role that our hard negative sampling strategy plays in embedding fine-tuning.

5.3 Pooling Layer Selection

In LLM, the effectiveness and performance of extracting sentence representations across different hidden layers may vary. To systematically assess the semantic information and representation capabilities of various layers in LLM, we employs pooling techniques on the last token hidden states at different layers and conduct corresponding evaluations. Specifically, we select the last hidden layer, last two hidden layers, middle hidden layer, and first and last hidden layers as pooling layers. The experimental results in Table 4 indicate that, for both LLaMA and PIE-LLaMA, the average pooling of the last two hidden layers consistently outperforms other pooling methods. Notably, regardless of the pooling method employed, the embedding with prompt consistently outperforms the embedding without prompt. This suggests that prompts indeed guide the model to identify the task information contained within the instructions, validating the effectiveness of our PIE method.

5.4 Prompt Search

Prompt is a key part of our PIE . In this paper, we employed a manual approach to search for appropriate prompt: we first manually crafted several prompts, then, for each manually crafted prompt,



(a) (b) (c) (d)

Figure 3: Embedding visualization: (a) non-fine-tuned LLaMA (b) non-fine-tuned PIE-LLaMA (c) fine-tuned LLaMA (d) fine-tuned PIE-LLaMA

Index	CP	ARI
#0	**0.7842**	0.1115
#1	0.7819	**0.1233**
#2	0.7284	0.0794
#3	0.6707	0.0662
#4	0.6323	0.0612

Table 5: Result of prompt search. Index refers to the template index in Table 6.

we evaluated its effectiveness by the instruction clustering task. The human crafted prompts are shown in Table 6, and the results are presented in Table 5. According to the result, we select template #0 for further experiments.

5.5 Evaluation on Downstream Tasks

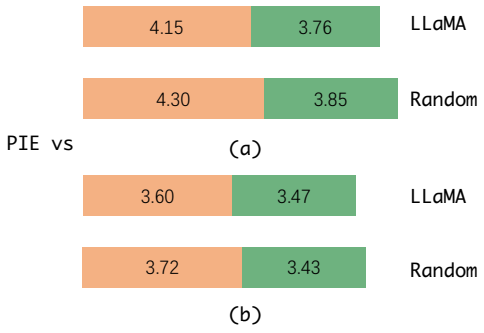


Figure 4: Downstream tasks results on (a) data distillation for instruction tuning and (b) demonstrations selection for in-context learning.

We conduct two downstream tasks to further evaluate the effectiveness of PIE:

Data Distillation Follow the line of data distillation work, we design a data distillation experiment based on instruction embedding. First, we utilize k-means clustering to partition the instruction train set of IEB where k is determined by the number of task categories of undistilled instructions (502). Then, we extract 6 instructions closest to the centers from each cluster to achieve data distillation. Finally, we conduct instruction fine-tuning on it, and compare PIE with LLaMA embeddings for clustering and random selection. As for the instruction fine-tuning, training configuration can be found in Appendix A.1.

Demonstrations Selection LLMs have demonstrated remarkable in-context learning (ICL) capability (Patel et al., 2023). Demonstrations related to the instruction task are more conducive to model ICL compared to task-agnostic demonstrations. Thus, for each instruction x_i in the instruction test set, we extract the 3 most similar data from the instruction train set by embedding cosine similarity as demonstrations. Then, we combine and input them to GPT-3.5-turbo. Similarly, baselines are LLaMA embeddings and random selection.

For both tasks, we use GPT-4-turbo to compare and score the samples generated by PIE and the baselines in the range of 1 to 10 (1 to 5 for ICL task). The results in Figure 4 demonstrate that the PIE can be a better substitution of text embeddings for instruction-related tasks.

6 Conclusion

We introduce the concept of instruction embedding, which prioritizes task identification over traditional sentence-level semantic analysis. Alongside, we release the publicly available IEB benchmark for evaluating and further training instruction embeddings. To ensure instruction embeddings focus more on task specifics rather than broad semantic content, we propose a prompt-based approach for generating instruction embeddings, applicable in both unsupervised (learning-free) and supervised (learning-based) contexts. The introduction of instruction embedding, along with the IEB benchmark and the PIE method, plays a crucial auxiliary role in instruction-related tasks for large language models.

7 Limitations

Although our PIE outperforms the LLaMA and random method in the data distillation task, our data distillation approach requires prior knowledge of the number of instruction task categories in the instruction dataset, which is generally not feasible. This limitation constrains the application of our instruction embedding. In future work, we will investigate how to achieve data distillation without prior knowledge of the number of instruction task categories. Additionally, PromptBERT(Jiang et al., 2022) successfully utilizes OptiPrompt(Zhong et al., 2021) to achieve better embedding effects than manual prompts. Although we did not leverage the OptiPrompt technique in this paper, we will apply this technology to PIE in future work.

References

Eneko Agirre, Daniel M. Cer, Mona T. Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2012, Montréal, Canada, June 7-8, 2012*, pages 385–393. The Association for Computer Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yihan Cao, Yanbin Kang, and Lichao Sun. 2023. Instruction mining: High-quality instruction data selection for large language models. *CoRR*, abs/2307.06290.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 1–14. Association for Computational Linguistics.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2023a. Alpagasus: Training A better alpaca with fewer data. *CoRR*, abs/2307.08701.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2023b. Alpagasus: Training A better alpaca with fewer data. *CoRR*, abs/2307.08701.

Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. 2017. On sampling strategies for neural network-based collaborative filtering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 767–776. ACM.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 6894–6910. Association for Computational Linguistics.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), 17-22 June 2006, New York, NY, USA*, pages 1735–1742. IEEE Computer Society.

Matthew L. Henderson, Rami Al-Rfou, Brian Strope, Yun-Hsuan Sung, László Lukács, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *CoRR*, abs/1705.00652.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Lawrence Hubert and Phipps Arabie. 1985. Comparing partitions. *Journal of classification*, 2:193–218.

Ting Jiang, Jian Jiao, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Denvy Deng, and Qi Zhang. 2022. Promptbert: Improving BERT sentence embeddings with prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8826–8837. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy. Association for Computational Linguistics.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686, Melbourne, Australia. Association for Computational Linguistics.

Xian Li, Ping Yu, Chunting Zhou, Timo Schick, Luke Zettlemoyer, Omer Levy, Jason Weston, and Mike Lewis. 2023. Self-alignment with instruction backtranslation. *CoRR*, abs/2308.06259.

Xianming Li and Jing Li. 2023. Angle-optimized text embeddings. *CoRR*, abs/2309.12871.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Bhaskar Mitra, Fernando Diaz, and Nick Craswell. 2017. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017*, pages 1291–1299. ACM.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Ajay Patel, Bryan Li, Mohammad Sadegh Rasooli, Noah Constant, Colin Raffel, and Chris Callison-Burch. 2023. Bidirectional language models are also few-shot learners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. 2023. Selective annotation makes language models better few-shot learners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. *CoRR*, abs/2401.00368.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Lai Wei, Zihao Jiang, Weiran Huang, and Lichao Sun. 2023. Instructiongpt-4: A 200-instruction paradigm for fine-tuning minigpt-4. *CoRR*, abs/2308.12067.

Shengguang Wu, Keming Lu, Benfeng Xu, Junyang Lin, Qi Su, and Chang Zhou. 2023a. Self-evolved diverse data sampling for efficient instruction tuning. *CoRR*, abs/2311.08182.

Xuansheng Wu, Wenlin Yao, Jianshu Chen, Xiaoman Pan, Xiaoyang Wang, Ninghao Liu, and Dong Yu. 2023b. From language modeling to instruction following: Understanding the behavior shift in llms after instruction tuning. *CoRR*, abs/2310.00492.

Lingling Xu, Haoran Xie, Zongxi Li, Fu Lee Wang, Weiming Wang, and Qing Li. 2023. Contrastive learning models for sentence representations. *ACM Trans. Intell. Syst. Technol.*, 14(4):67:1–67:34.

Peiwen Yuan, Xinglin Wang, Jiayi Shi, Bin Sun, Yiwei Li, and Kan Li. 2023. Better correlation and robustness: A distribution-balanced self-supervised learning framework for automatic dialogue evaluation. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: learning vs. learning to recall. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 5017–5033. Association for Computational Linguistics.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023. LIMA: less is more for alignment. *CoRR*, abs/2305.11206.

A Appendix

A.1 Additional Configuration

Experiment Configuration We fine-tune PIE-LLaMA for 3 epochs with the batch size set to 8 (there will be 16 instruction pairs after hard negative sampling) and the learning rate set to 1×10^{-5} on 4 NVIDIA RTX 3090 GPUs. Due to computational resource limitations, we adopt LORA (Hu et al., 2022) technique to fine-tune the LLM with lora-rank set to 32, lora-alpha set to 64, lora-dropout set to 0.05 and target modules set to ['q_proj','v_proj'][3].

Data Distillation Configuation We complete instruction fine-tuning on a single NVIDIA RTX 3090 GPU and adopt LoRA (Hu et al., 2022) technique to fine-tune the LLM with lora-rank set to 1024, lora-alpha set to 2048, lora-dropout set to 0.05 and target modules set to ['q_proj','v_proj'], epochs set to 10 and batch size set to 128.

[3] https://huggingface.co/docs/peft/developer_guides/lora

A.2 Prompt templates

Index	Template
#0	Below is an instruction that describes a task \n {instruction} \n The task of the given instruction is:
#1	The following instruction \n {instruction} \n wants you to:
#2	Given the following instruction \n {instruction} \n please identify its task type:
#3	What type of task does the following instruction represent? \n {instruction}
#4	Indentify the task category associated with the following instruction: \n {instruction}

Table 6: Templates used in prompt search, \n represents a newline.