Unified Demonstration Retriever for In-Context Learning

Xiaonan Li[1]*, Kai Lv[1]*, Hang Yan[1], Tianyang Lin[1],
Zhu Wei[2]†, Yuan Ni[3], Guotong Xie[3], Xiaoling Wang[2], Xipeng Qiu[1]†
[1] Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
[1] School of Computer Science, Fudan University
[2]East China Normal University [3] Pingan Health Tech
[1]{lixn20, klv21, hyan19, tylin20, xpqiu}@fudan.edu.cn,
[2]wzhu@stu.ecnu.edu.cn, xlwang@cs.ecnu.edu.cn
[3]{niyuan442, xieguotong}@pingan.com.cn

Abstract

In-context learning is a new learning paradigm where a language model conditions on a few input-output pairs (demonstrations) and a test input, and directly outputs the prediction. It has been shown highly dependent on the provided demonstrations and thus promotes the research of demonstration retrieval: given a test input, relevant examples are retrieved from the training set to serve as informative demonstrations for in-context learning. While previous works focus on training task-specific retrievers for several tasks separately, these methods are often hard to transfer and scale on various tasks, and separately trained retrievers incur a lot of parameter storage and deployment cost. In this paper, we propose **U**nified **D**emonstration **R**etriever (**UDR**), a single model to retrieve demonstrations for a wide range of tasks. To train UDR, we cast various tasks' training signals into a unified list-wise ranking formulation by language model's feedback. Then we propose a multi-task list-wise ranking training framework, with an iterative mining strategy to find high-quality candidates, which can help UDR fully incorporate various tasks' signals. Experiments on 30+ tasks across 13 task families and multiple data domains show that UDR significantly outperforms baselines. Further analyses show the effectiveness of each proposed component and UDR's strong ability in various scenarios including different LMs (1.3B $\sim$ 175B), unseen datasets, varying demonstration quantities, etc.

1 Introduction

Large language models have shown an impressive *in-context learning* ability for various Natural Language Processing (NLP) tasks (Brown et al., 2020; Dong et al., 2022). In-context learning (ICL) is a recent learning paradigm where a language model (LM) learns a task by observing a few input-output pairs (demonstrations) and directly output the prediction of the given test input. Thus ICL can unify a wide range of NLP tasks through one language model's inference without parameter updates, which makes it a promising alternative to supervised fine-tuning (Devlin et al., 2019).

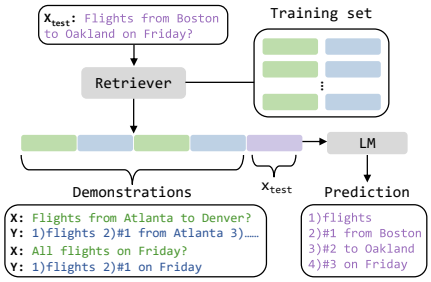

Figure 1: Demonstration retrieval: Given a test input x_{test}, relevant demonstrations are retrieved from the training set. Then the inference LM takes demonstrations and x_{test} as input and generates the output.

However, it has been shown that ICL's performance highly depends on the provided demonstrations (Liu et al., 2022; Zhang et al., 2022; Li and Qiu, 2023a). This promotes the research of demonstration retrieval for in-context learning (Liu et al., 2022; Rubin et al., 2022; Shi et al., 2022): As shown in Figure 1, given a test input, relevant examples are retrieved from an annotated training set, to serve as informative demonstrations for ICL.

There are about two lines of methods to retrieve demonstrations. One is to leverage off-the-shelf retrievers, e.g., BM25 (Robertson and Zaragoza, 2009) or Sentence-BERT (Reimers and Gurevych, 2019a). They can retrieve demonstrations that are textually or semantically similar to the test input and achieve empirical improvements. Thanks to their versatility, they can serve for extensive NLP tasks, but they are heuristic and sub-optimal since they are not guided by task supervision. Another line is to train a task-specific retriever by a specially designed task signal. Das et al. (2021) train the retriever for knowledge-based question answering, based on the logic form's surface similarity. Hu et al. (2022) explore ICL on dialogue state tracking

*Equal Contribution
†Corresponding Authors

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 4644–4668

and design the similarity between dialogue's states as the retriever's training signal. Rubin et al. (2022) and Shi et al. (2022) leverage the LM's feedback to train demonstration retrievers for semantic parsing in English and cross-lingual scenarios, respectively. These task-specialized retrievers show better performance than the former, but they still face two challenges: 1. these explorations are limited to a small range of tasks and demonstrated separately on each task, e.g., semantic parsing or dialogue state tracking, which restricts systematic and compatible research on demonstration retrieval for ICL while ICL is a unified framework for extensive tasks. 2. it is costly for these methods to transfer and scale on various tasks and the reason is twofold: (i) they need to design a specialized training signal for each task. (ii) the number of retrievers will scale up with increasing tasks, which results in massive parameter storage and deployment costs.

To address these limitations, we explore learning various tasks' demonstration retrieval in a unified formulation and propose **U**nified **D**emonstration **R**etriever (**UDR**), a single multi-task model for demonstration retrieval of a wide range of tasks. To train UDR, we cast various tasks' training signals into a unified list-wise ranking formulation. For a training example from task $\mathcal{T}$, we select a list of candidate examples from $\mathcal{T}$'s training set and rank them by LM's feedback. Then we propose a multi-task list-wise ranking training framework, with an iterative mining strategy to find high-quality candidates. Specifically, we iteratively train the retriever to rank candidates and use itself to find high-quality positive candidates and hard negatives. Compared with the representative method for demonstration retrieval, EPR(Rubin et al., 2022), which trains the retriever by the binary label from LM's feedback and selects candidates in a manually limited range, our training framework can explore the entire dataset to get high-quality candidates and help UDR fully incorporate the LM's feedback through list-wise ranking training.

Experiments on 30+ tasks across 13 task families and multiple data domains show that UDR significantly outperforms baselines and further analyses show the effectiveness of each proposed component and UDR's strong ability under various scenarios including different LMs (1.3B $\sim$ 175B), unseen datasets, varying demonstrations quantities, etc. We release the code and model checkpoint at https://github.com/KaiLv69/UDR.

2 Unified Demonstration Retriever

Provided a language model G, a training set $\mathcal{D}_{\text{train}}$ and a test case x_{test}, demonstration retrieval aims to retrieve x_{test}'s relevant demonstrations from $\mathcal{D}_{\text{train}}$ to help LM G decode the target output. Previous works (Das et al., 2021; Rubin et al., 2022; Shi et al., 2022) propose task-specialized methods for several tasks separately, but they are hard to transfer and scale on various tasks. In this work, we focus on learning various tasks' demonstration retrieval in a unified formulation and propose UDR , a single model for demonstration retrieval of a wide range of tasks, as shown in Figure 2. We introduce its architecture, training, and inference as follows.

2.1 Bi-encoder with Task Instruction

UDR is based on the prevailing bi-encoder architecture, dense passage retriever (DPR) (Karpukhin et al., 2020), which encodes the query example and candidate examples separately and then calculates their similarity. To distinguish examples from different tasks, UDR encodes the example together with its task instruction, which is a short piece of text related to the task objective. Taking CNN/-DailyMail (Hermann et al., 2015) as an example, its task instruction can be "Summarize the text". Given an example query x and a candidate demonstration $z = \{x', y'\}$ from task T_i, UDR uses the query encoder E_q and demonstration encoder E_d to encode them respectively and calculates their similarity as:

$$\text{sim}(x, z) = E_q(I_i \oplus x)^\top E_d(I_i \oplus z), \quad (1)$$

where I_i is T_i's task instruction and $\oplus$ is the concatenation operator. E_q and E_d are two multi-layer Transformer (Vaswani et al., 2017) encoders with "CLS" pooling and can be initialized with pre-trained models (Devlin et al., 2019).

Thus, we can not only get task-specific features by specifying the task instruction, but also retain the uniformity and parameter efficiency of ICL.

2.2 Learning from LM Feedback

To train the demonstration retriever, previous works (Das et al., 2021; Rubin et al., 2022; Hu et al., 2022) design task-specific training signals for several tasks separately, which makes their methods hard to transfer and scale on various tasks, and hinders systematic and compatible research on demonstration retrieval. For UDR's training, we propose to cast various tasks' training signals into

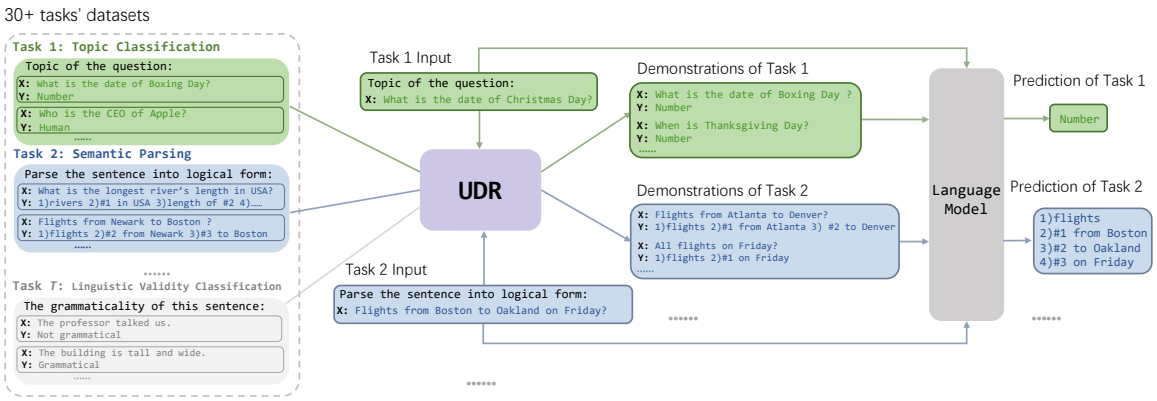

Figure 2: Illustration of UDR's inference for various tasks: Given a test input and its task's instruction, UDR can retrieve informative demonstrations from the corresponding datasets for ICL, where arrows and lines with various colors such as → and → indicate corresponding tasks' pipelines, respectively.

a unified list-wise ranking formulation. Then we introduce a multi-task list-wise ranking training framework, where we iteratively let the retriever itself to mine high-quality candidates and learn to rank them in turn, across various tasks, shown in Algorithm 1. We introduce the list-wise ranking training and iterative mining strategy as follows.

2.2.1 Ranking Candidates by LM

Given a training example (x, y) and its candidates $Z = \{z_i\}_{i=1}^{l}$, we first rank these candidates as:

$$r(z_j) = rank(s(z_j)|\{s(z_i)\}_{i=1}^{l}) \quad (2)$$

$$s_{gen}(z_j) = p_G(y \mid z_j, x), \quad (3)$$

$$s_{cls}(z_j) = \frac{p_G(y \mid z_j, x)}{\sum_{y' \in Y} p_G(y' \mid z_j, x)}, \quad (4)$$

where $s(z_j) = s_{gen}(z_j)$ for generation tasks and $s(z_j) = s_{cls}(z_j)$ for classification and multi-choice tasks. $p_G(\cdot|\cdot)$ is the LM G's conditional likelihood. Y is the label space or choices of the classification or multi-choice task, respectively. For simplicity, we omit special tokens and classification tasks' verbalizers in the equations above.

First we use G to score each candidate (Rubin et al., 2022) and calculate $s(z_j)$ as the ground truth y's likelihood conditioned on the candidates z_j and the query input x. $s(z_j)$ indicates the importance of z_j for G to encode x and generate the ground truth y. Then we rank Z according to $\{s(z_i)\}_{i=1}^{l}$. The more important z_j is for x, the higher z_j's rank will be. Thus we unify various tasks' training signals into the same list-wise ranking formulation using LM's feedback, instead of designing task-specific objectives (Das et al., 2021; Hu et al., 2022).

2.2.2 Loss Function

With these candidates' ranks from G's feedback, we propose to use the following loss function to inject the ranking signal into the retriever E, inspired by LambdaRank (Burges, 2010):

$$\mathcal{L}_{rank} = \sum_{z_i, z_j \in Z} w * \log(1 + e^{\text{sim}(x,z_j) - \text{sim}(x,z_i)}) \quad (5)$$

where $w = \max(0, \frac{1}{r(z_i)} - \frac{1}{r(z_j)})$.

For those z_i and z_j where $r(z_i) < r(z_j)$, L_{rank} will draw $\text{sim}(x, z_i)$ up and optimize the retriever towards $\text{sim}(x, z_i) > \text{sim}(x, z_j)$. Additionally, w adjusts the weight for each pair of demonstrations and inject list-wise ranking information into $\mathcal{L}_{rank}$. When z_i has a much higher rank than z_j, e.g, $r(z_i) = 1$ and $r(z_j) = 10$, w will be a high weight and strongly draw $\text{sim}(x, z_i)$ up from $\text{sim}(x, z_j)$. Since we optimize the retriever on demonstration pairs under different w, $\mathcal{L}_{rank}$ can help UDR fully incorporate candidates' listwise ranking signals from G's feedback for various tasks and learn to retrieve those helpful demonstrations.

To fully leverage the computation of the same batch, we also use the in-batch negative loss as:

$$\mathcal{L}_{ib} = -\log \frac{e^{\text{sim}(x,z^*)}}{\sum_{z \in \mathbb{Z}} e^{\text{sim}(x,z)}}, \quad (6)$$

where z^* is the rank-1 candidate of x and $\mathbb{Z}$ is all candidates (x's or not x's) in the batch. Each batch is sampled from the same task, and to alleviate the bias towards high-resource tasks, we sample each task according to the multinomial distribution with probabilities $\{p(\mathcal{T}_i)\}_{i=1}^{T}$ as:

$$p(\mathcal{T}_i) = \frac{q_i^\alpha}{\sum_{j=1}^{T} q_j^\alpha} \text{ with } q_i = \frac{|\mathcal{D}^{\mathcal{T}_i}|}{\sum_{j=1}^{T} |\mathcal{D}^{\mathcal{T}_j}|}, \quad (7)$$

where $\mathcal{D}^{\mathcal{T}_i}$ is the ith task's dataset. α is a pre-defined hyper-parameter and we follow Conneau and Lample (2019) to set α as 0.5.

The overall loss function of UDR is the integration of these two losses as follows,

$$\mathcal{L} = \lambda * \mathcal{L}_{rank} + (1 - \lambda) * \mathcal{L}_{ib}, \tag{8}$$

where λ is a pre-defined hyper-parameter.

2.2.3 Iterative Candidate Mining

The selection of candidates can be a key factor for retriever's training (Karpukhin et al., 2020; Xiong et al., 2021). It is desirable for UDR to take the entire training set as candidates to provide abundant ranking signals. However, it is infeasible since scoring all pairs of training examples is quadratic in $|\mathcal{D}|$ and costly. Previous work (Rubin et al., 2022) selects those examples which have textually similar targets with x's as candidates. However, it may bias the retriever to learn among candidates with highly similar targets. Meanwhile, it can probably miss important demonstrations. For instance, if an example z contains relevant logic with the query x but has a dissimilar target with x's, the valuable z will not be selected as candidate to provide signal for the retriever. So, we propose an iterative mining strategy to select candidates by the retriever itself. Specifically, we iteratively train the retriever and use it to select candidates in turn. At each iteration, we update each training example's candidates as:

$$Z^* = \text{top-}K_{z \in \mathcal{D}} \, \text{sim}(x, z) \tag{9}$$

where $\mathcal{D}$ is the task's entire training set.

Then we will use LM G to score and rank Z^*. The new candidates in Z^* can be divided into two categories. If a new candidate z has a low score, it means that we find a hard-negative candidate that can provide crucial negative signal for the retriever. If the score of z is high and even higher than all old candidates, it means that we find a valuable positive candidate that can help the retriever learn to find informative demonstrations. Thus, with iterative mining, we can explore the entire dataset, find high-quality candidates and improve training progressively. Before the first iteration, the retriever is untrained, so we initialize candidates based on surface similarity, inspired by Rubin et al. (2022).

For computational efficiency, we first update candidates and score Z^* at each iteration, and then randomly sample l of Z^* and rank them at each training step. In summary, Algorithm 1 shows the UDR's overall training procedure.

Algorithm 1 Multitask List-wise Ranking Training

Require: Bi-encoder E_q and E_d, language model G, Training sets of T tasks $\{\mathcal{D}^{\mathcal{T}_i}\}_{i=1}^{T}$

```
Initialize the bi-encoder.
Initialize candidates of each training example.
Score initialized candidates by G.
for Each iteration do
    for Each training step, T_i ~ p(T) do
        Sample a batch of examples.
        For each example, sample l examples z_{1~l} from its candidates and rank z_{1~l} by G's score.
        Update the bi-encoder's parameters by L.
    end for
    Update candidates by new E_q and E_d.
    Score new candidates by G.
end for
```

2.3 Inference

After training, we encode each task $\mathcal{T}_i$'s training set using $E_d(p_i \oplus \cdot)$. At the test stage, given a task $\mathcal{T}_i$'s input, x_{test}, we use $E_q(p_i \oplus \cdot)$ to compute its encoding and then use FAISS (Johnson et al., 2021) to search over $\mathcal{T}_i$'s training set to find the most relevant demonstrations, ascendingly sorted by $\text{sim}(x_{test}, \cdot)$, $D = (z_1, z_2, \cdots, z_L)$. For generation tasks, the number of final demonstrations, L, is determined by the LM G's maximal input length C. Specifically, $\sum_{i=1}^{L} |z_i| + |x_{test}| + |y| \leq C$, where $|y|$ is the pre-defined maximal length of the generated target. For classification and multi-choice tasks, we observe that increasing L brings negligible performance improvement and thus we set L to a small value, 8. We conduct further analysis of the number of demonstrations in section 3.3.5. Finally, we use greedy decoding to get the result of $G([z_1; z_2; \cdots; z_L; x_{test}])$. Notice that here D is ascendingly sorted by $\text{sim}(x_{test}, \cdot)$ unless otherwise specified. Our analysis in section 3.3.4 shows that different orderings lead to similar performance. Thus we use the same ordering strategy with EPR (Rubin et al., 2022) for fair comparison.

3 Experiment

3.1 Experimental Settings

Dataset We train UDR on a wide range of NLP tasks, consisting of about 40 tasks across 13 task families and multiple data domains, including: **Sentiment Classification**: SST-2, SST-5 (Socher et al., 2013), Amazon (McAuley and Leskovec, 2013), Yelp (Zhang et al., 2015), MR (Pang and Lee, 2005)

and CR (Amplayo et al., 2022); **Topic Classification**: AGNews, Yahoo (Zhang et al., 2015), TREC (Voorhees and Tice, 2000) and DBPeida (Lehmann et al., 2015); **Multi Choice**: COPA (Roemmele et al., 2011), Cosmos QA (Huang et al., 2019), Commonsense Validation and Explanation (ComE and ComV) (Wang et al., 2019b); **NLI**: MNLI (Williams et al., 2018), SNLI (Bowman et al., 2015) and RTE (Bar-Haim et al., 2014); **Subjectivity Classifcation**: Subj (Pang and Lee, 2004); **Linguistic Acceptibility**: COLA; **Semantic Parsing**: BREAK (Wolfson et al., 2020), MTOP (Li et al., 2021) and SMCalFlow (Andreas et al., 2020); **Text Summarization**: CNN/DailyMail (Hermann et al., 2015), PubMed (Cohan et al., 2018) and Reddit (Kim et al., 2019); **Commonsense Generation**: CommonGen (Lin et al., 2020); **Story Generation**: Roc Story and Ending Generation (Mostafazadeh et al., 2016); **Code Summarizaton**: Go, Python, Java and PHP (Lu et al., 2021); **Text Simplifiction**: WikiAuto + Turk/ASSET (Jiang et al., 2020); **Data to Text**: DART (Nan et al., 2021) and E2E (Dušek et al., 2019). These tasks' input/output, statistics, split and evaluation metrics are in Appendix A.

Implementation Details We follow EPR (Rubin et al., 2022) to use GPT-Neo-2.7B (Black et al., 2021) as the scoring LM and the inference LM for most experiments in the paper unless otherwise specified. We also explore UDR's transferability across different inference LMs in section 3.3.2. Following EPR(Rubin et al., 2022), we initialize E_q and E_d as two separate "BERT-base-uncased" encoders (Devlin et al., 2019). We list the overall hyper-parameters and implementation details in Appendix B. On each task, we use one specific template for scoring and inference (see Appendix A). We evaluate UDR's performance when inference templates are different with the scoring template in Appendix C, and the results show that UDR has stable performance across varying inference templates, which reflects UDR's generality.

Model Comparison With the same inference LM, GPT-Neo-2.7B, we compare UDR with previous methods for demonstration retrieval by the downstream ICL performance, including: **1. Random**: We randomly sample demonstrations from the corresponding task's training set. **2. BM25** (Robertson and Zaragoza, 2009): A prevailing sparse retriever. For each test input x_{test}, we use BM25 to retrieve examples with the most similar input. **3. SBERT** (Reimers and Gurevych, 2019b): We use the Sentence-BERT as the dense demonstration retriever. Specifically, we follow Rubin et al. (2022) to take "paraphrase-mpnet-base-v2" to encode the test input x_{test} and training set's inputs, and retrieve the examples with the most similar input as demonstrations. **4. Instructor** (Su et al., 2022): Instructor is a recently proposed competitive text embedding model trained on 330 tasks with instructions. By providing the specialized instruction, it can serve for demonstration retrieval. For fair comparison, we conduct experiments on its released base-size model. **5. DR-Target**: This baseline is inspired by previous works on generation tasks like dialogue state tracking, question answering and code generation (Hu et al., 2022; Das et al., 2021; Poesia et al., 2022), which design the task-specific target's similarity and use examples with similar targets to train the retriever. Here we use BM25 as the similarity function for each task's target output. Specifically, we use BM25 to find positive pairs with similar targets and use DPR (Karpukhin et al., 2020) for training. **6. EPR** (Rubin et al., 2022): EPR is a recently proposed representative method for training demonstration retriever. It uses the language model to assign candidate examples with positive and negative labels and thus trains a task-specific demonstration retriever by DPR. For fair comparison, we train EPR on each task using the same hyper-parameters of UDR. Specially, we discuss EPR's candidate quantity in Appendix B.

Except that the performance of Random, BM25, SBERT and EPR on semantic parsing is from the previous paper (Rubin et al., 2022), other results are from our implementation since they are not explored previously.

3.2 Main Results

We show the performance comparison of classification tasks and generation tasks in Table 1 and Table 2, respectively. We can see that UDR outperforms baselines significantly on most tasks, which shows UDR's best overall demonstration retrieval ability on a wide range of NLP tasks. Specially, compared with DR-Target and EPR, UDR has better overall performance and this shows the effectiveness of our unification of various tasks' training signals. Meanwhile, compared with Instructor (Su et al., 2022), the text embedding model trained on 330 tasks' text pairs, UDR has an improvement

Retrieval Method	Sentiment Classification						Topic Classification			
	SST-2	SST-5	Amazon	Yelp	MR	CR	AGNews	TREC	DBPedia	Yahoo
Random	57.7	28.2	23.9	25.3	56.0	52.4	74.2	42.6	73.7	39.1
BM25	74.1	38.3	31.6	36.9	71.4	57.2	88.4	89.4	97.2	62.5
SBERT	84.3	40.0	33.4	36.0	79.0	61.3	88.3	89.4	96.7	58.4
Instructor	83.7	42.4	42.4	46.6	78.5	64.1	89.6	91.2	97.7	67.2
EPR	87.9	46.9	49.1	49.6	80.6	65.7	89.9	95.2	98.1	66.1
UDR	**92.4**	**50.5**	**54.9**	**61.7**	**85.2**	**82.6**	**91.5**	**96.6**	**98.7**	**67.5**

Retrieval Method	Multi Choice				NLI			Other		Overall
	COPA	Cosmos QA	ComE	ComV	MNLI	SNLI	RTE	Subj	COLA	
Random	71.6	26.2	41.4	50.5	34.1	33.0	55.6	60.0	52.8	47.3
BM25	71.2	27.1	41.4	50.9	35.3	41.5	50.5	78.8	53.3	57.7
SBERT	72.4	27.3	41.1	50.3	38.0	42.0	49.8	88.7	56.3	61.6
Instructor	71.6	27.1	41.9	49.9	41.3	46.7	52.7	84.3	56.0	63.2
EPR	**73.2**	28.4	43.0	50.4	54.3	74.0	55.6	92.1	70.3	68.8
UDR	72.8	**29.9**	**45.6**	**63.9**	**73.8**	**83.6**	**65.3**	**95.0**	**78.9**	**73.2**

Table 1: Main results on classification and multi-choice tasks.

Retrieval Method	Semantic Parsing			Text Summarizaiton			CommonGen	Story Generation	
	BREAK	MTOP	SMCalFlow	CNN/DM	PubMed	Reddit	CommonGen	Roc Story	Roc Ending
Random	1.9	6.6	8.7	20.8	23.6	15.6	21.1	9.3	13.4
BM25	26.0	52.9	46.1	18.6	24.5	15.3	26.0	12.3	19.2
SBERT	22.4	48.6	43.1	19.2	25.2	15.4	25.7	12.2	19.1
Instructor	22.7	50.5	46.3	19.0	24.8	15.3	26.5	12.4	21.8
DR-Target	22.1	49.6	41.6	19.4	24.6	16.0	24.5	11.9	20.1
EPR	31.9	64.4	54.3	20.3	24.8	15.5	25.3	12.9	21.2
UDR	**35.2**	**66.8**	**60.4**	**21.2**	**26.1**	**16.2**	**27.1**	**17.6**	**24.7**

Retrieval Method	Code Summarization				Text Simpilification			Data to Text		Overall
	Go	Python	Java	PHP	WikiAuto	Turk	ASSET	DART	E2E	
Random	27.3	7.9	6.7	18.9	8.3	28.0	24.8	20.4	21.9	15.8
BM25	30.4	9.7	11.7	23.6	10.2	29.1	26.6	28.4	29.2	24.2
SBERT	28.3	13.7	15.1	22.0	9.5	29.1	26.7	27.9	24.2	23.7
Instructor	29.9	11.5	13.1	24.0	11.3	29.0	26.3	28.7	22.4	24.2
DR-Target	28.1	12.2	13.0	24.2	10.8	29.4	26.7	30.1	24.7	23.8
EPR	**30.5**	17.4	17.4	30.2	13.3	30.8	27.6	31.8	29.3	27.7
UDR	29.4	**22.3**	**25.2**	**33.2**	**19.5**	**32.9**	**32.1**	**34.5**	**32.6**	**30.9**

Table 2: Main results on generation tasks.

of 10 and 6.7 points for classification and generation tasks respectively with less training data. This straightly demonstrates that our proposed training framework can help UDR incorporate LM's feedback through a unified ranking formulation and better retrieve informative demonstrations.

Additionally, we find the random baseline shows the worst performance on most tasks and this reflects the necessity to retrieve high-quality relevant demonstrations. Meanwhile, EPR and UDR have better performance than other methods, which reflects the importance of LM's feedback. Among these datasets, we notice a different trend on text summarization datasets like CNN/DailyMail and Reddit, on which these methods have similar performance. We conjecture that the LM can already have the knowledge of summarization since there are a lot of "[Article, TL;DR, Abstract]" texts in its pre-training corpus (Radford et al., 2018), thus random demonstrations can well activate LM's summarization ability without example-specific information.

3.3 Analysis

3.3.1 Ablation Study

To evaluate the effect of UDR's each component, we conduct ablation study on SMCalFlow, SST-2 and Java code summarization, shown in Table 3. When removing list-wise ranking training, we use EPR's training strategy (Rubin et al., 2022). We can see that removing task instructions cause slight performance degradation, which indicates that they can help UDR distinguish examples from various tasks and thus get better task-specific features. Meanwhile, we can see that UDR has a slightly better performance than the single-task counterpart on SST-2 and Java. We suppose that is because

	SMCalFlow	SST-2	Java	Avg
UDR	60.8	91.3	23.2	58.4
- w/o Task Prompt	60.1	90.8	21.9	57.6
- w/o MultiTask	60.9	91	22.9	58.3
- w/o Rank Loss	56.7	89.2	21.1	55.7
- w/o Self-Guided	59.5	90.2	19.7	56.5

Table 3: Ablation study of UDR's each component.

Dataset	SMCalFlow			E2E		
LMs / Methods	BM25	EPR	UDR	BM25	EPR	UDR
Text-Davinci-003	55.0	58.9	64.7	31.3	31.5	34.3
Code-Davinci-002	50.9	55.2	62.9	23.5	24.4	26.4
GPT-J	49.0	55.9	64.0	33.3	33.7	35.0
GPT-Neo-1.3B	44.8	52.9	59.5	29.9	29.7	31.9
GPT-Neo-2.7B	46.5	53.7	62.2	29.2	29.1	32.6

Table 4: Results on 1000 randomly sampled test examples across different inference LMs.

there are several relevant tasks in UDR's training tasks and our multi-task ranking unification can help UDR fully share these tasks' knowledge. The performance of single-task UDR still outperforms EPR significantly and this straightly reflects that our training components, i.e., list-wise ranking formulation and iterative candidate mining strategy, can 1. help UDR better incorporate LM's feedback than EPR 2. serve as a competitive universal training method for a task-specific retriever. Removing list-wise ranking training and iterative candidate mining both cause performance degradation, which straightly indicates their effectiveness.

3.3.2 Transferability across Different LMs

In this section, we evaluate UDR's transferability across different inference LMs on SMCalFlow and E2E. Specifically, we compare BM25, EPR and UDR on inference LMs with different sizes, including: GPT-Neo-1.3B (Black et al., 2021), GPT-J (6B) (Wang and Komatsuzaki, 2021), Code-Davinci-002 (175B) (Chen et al., 2021) and Text-Davinci-003 (175B) (Brown et al., 2020; Ouyang et al., 2022) and we show the result in Table 4. When comparing UDR with baselines, the trends are similar with using GPT-Neo-2.7B (the scoring LM) as inference LM. UDR outperforms BM25 and EPR significantly and it shows UDR's strong transferability across different inference LMs. Meanwhile, we find that UDR with larger inference LM can improve performance such as Text-Davinci-003 on SMCalFlow and GPT-J on E2E, which shows UDR's potential utility in the future where more competitive large-scale LM is built. When we demonstrate the example-specific demonstration transferability across different inference LMs in this paper, Li and Qiu (2023a) show that task-level demonstrations also exhibit such transferability. We leave the analysis of the transferablity of ICL's demonstrations across different LMs as future work.

	Twitter	QNLI	Ruby	JavaScript
BM25	50.0	54.1	9.2	12.7
SBERT	51.6	53.7	8.7	15.9
UDR	56.8	74.4	19.6	21.6

Table 5: The performance of UDR on unseen datasets.

3.3.3 Performance on Unseen Datasets

In this section we explore UDR's zero-shot transferability and evaluate it on unseen datasets including: 1. Twitter sentiment classification (Naji, 2012) 2. question-answering NLI (QNLI) (Wang et al., 2019a) 3. Ruby and JavaScript code summarization (Lu et al., 2021). These domains or programming languages (Twitter, NLI on QA, Ruby and Javascript) are never seen during UDR's training and thus can straightly reflect UDR's zero-shot transferability. We compare UDR with two powerful universal retrievers, BM25 and SBERT, and show the result in Table 5. We can see UDR significantly outperforms BM25 and SBERT on these unseen datasets by about 10 points on average, which shows that the learned ranking knowledge inside UDR can be well transferred and generalized to unseen datasets.

3.3.4 The Order of Demonstrations

Previous work (Lu et al., 2022) has revealed that ICL is sensitive to demonstrations' order when using random examples. Specifically, the same randomly sampled demonstrations with different orders can lead to the performance between random guess and near state-of-the-art. Here we explore the effect of ordering on example-specific demonstrations retrieved by UDR. We compare 3 demonstrations' orders: 1. random, for this setting, we run experiments with 10 different random seeds and report the best and worst performance. 2. descending sorted by UDR's score, i.e, the demonstration which has the highest similarity with x_{test} is put at the beginning of LM's input. 3. ascending sorted by UDR's score, opposite to "2". The result is shown in Table 6. We

	SST-2	TREC	Reddit	CommonGen
Random-Order$_{\text{Best}}$	92.5	96.6	16.8	27.5
Random-Order$_{\text{Worst}}$	92.0	96.2	16.2	26.6
Descending-Order	92.2	96.6	16.2	27.0
Ascending-Order	92.4	96.6	16.3	27.3

Table 6: The effect of different demonstration orders.

observe a different phenomenon from that in previous work (Lu et al., 2022). In general, The performance of UDR's demonstrations with different orders is more stable than previously investigated random examples. Across these tasks, different orders' performance gap is within 1 point, and it is far less than the performance fluctuation of up to tens points when using random examples (Lu et al., 2022). This indicates that high-quality demonstrations are less sensitive to the ordering and stabilize in-context learning, which is consistent with the analysis in previous work (Chen et al., 2022; Li and Qiu, 2023a).

3.3.5 The Impact of Demonstration Quantity

We compare UDR with BM25 and EPR under different amounts of demonstrations on two classification tasks: Yelp and RTE, and two generation tasks: WikiAuto and Java code summarization. We show results in Figure 3. We can see that UDR outperforms baselines consistently across varying amounts of demonstrations. Meanwhile, we can draw two conclusions from the results: 1. The number of demonstrations has a greater impact on generation tasks than classification tasks. Specifically, as the number of demonstrations increases, generation tasks' performance gets significant improvements while classification tasks' has slight or no improvements. 2. The quality of demonstrations can be more important than their quantity. In detail, UDR with the quota of 2 demonstrations still outperforms BM25 and EPR with 8 demonstrations. This also reflects the strong demonstration retrieval ability of UDR. Li and Qiu (2023b) observe the similar trends in the CoT-retrieval scenario, indicating that the relevance of the used reasoning paths is more important than their quantity.

4 Related Work

In this section, we introduce previous demonstration retrievers for in-context learning, and explain the difference between UDR and them. In general, there are two kinds of demonstration retrievers for ICL. One is to leverage off-the-shelf retrievers. For example, Liu et al. (2022) propose to use a fine-tuned BERT to encode examples and use a

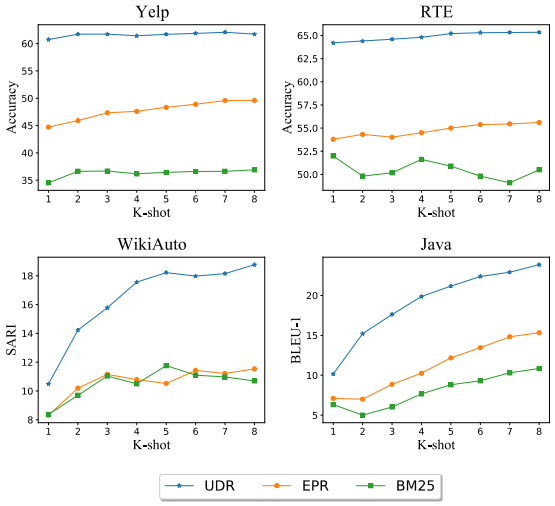

Figure 3: The effect of demonstration quantity.

KNN-based method to retrieve semantically similar demonstrations to improve ICL. Agrawal et al. (2022) use BM25 to retrieve demonstrations for machine translation. Compared with them, UDR incorporates various tasks' supervision by unified LM's feedback and thus can better retrieve informative demonstrations. Another approach is to train a task-specific retriever by a designed task-specific signal. Das et al. (2021) explore demonstration retrieval for knowledge-based question answering and define the F1 score of logic forms as soft-label to train the retriever. Poesia et al. (2022) train a demonstration retriever for code generation, based on the edit distance of abstract syntax trees. Hu et al. (2022) define the similarity between dialogue states, and use it to train a demonstration retriever for dialogue state tracking. Rubin et al. (2022) propose Efficient Prompt Retriever (EPR) for semantic parsing, which is to use the language model to score examples, assign positive and negative labels for them and use DPR (Karpukhin et al., 2020) to train a demonstration retriever. Shi et al. (2022) explore demonstration retrieval for cross-lingual semantic parsing using a similar example scoring method with EPR. These task-specific methods serve for each task separately and are hard to transfer and scale on various tasks. For other tasks, it requires to redesign the similarity function or training signal. Compared with them, we introduce a unified training framework based on list-wise ranking and propose a single multi-task retriever UDR to serve for a wide range of tasks. Compared with EPR, besides UDR's versatility on various tasks, UDR can incorporate LM's feedback by ranking-based training in a more fine-grained way and receive more

crucial candidates' signals by the iterative mining strategy. Cheng et al. (2023) propose CLAIF to enhance the sentence embedder by the gigantic language model's feedback. Specifically, they use GPT-3 (Brown et al., 2020) to generate the data of sentence pairs and then score them by the output of GPT-3, which depends on the strong natural language understanding ability of GPT-3. Different from them, we leverage the conditional probability to measure the helpfulness of an example, which only needs a small language model, and is more efficient and environmental-friendly. Recently, Li and Qiu (2023b) propose MoT (Memory-of-Thought) to let the LLM self-improve in two stages: 1. Before test stage, the LLM generate reasoning paths and answers on an unlabeled dataset for itself, 2. At test stage, the LLM retrieves relevant reasoning paths (memory) to help itself answer the given test question. While MoT focuses on the scenario with unlabeled dataset and uses the LLM for retrieval, we train a small retriever by a LM's feedback from tasks' supervision and thus the proposed method is more lightweight. We leave demonstration retrieval with reasoning paths or unlabeled datasets as future work.

5 Conclusion

In this paper, we propose UDR, a single multi-task model for a wide range of tasks' demonstration retrieval. To train UDR, we cast various tasks' training into a unified list-wise ranking formulation by language model's feedback, and propose a multi-task list-wise ranking training framework, with an iterative mining strategy to find high-quality candidates. Experiments on 30+ tasks show that UDR significantly outperforms baselines. Further analyses show the effectiveness of each proposed component and UDR's strong ability in various scenarios including different LMs (1.3B $\sim$ 175B), unseen datasets, varying demonstration quantities, etc.

Limitations

We illustrate this paper's limitations from the following three aspects:

1) Limited by the computational resources, we only train UDR from the initialization of "BERT base uncased" following EPR (Rubin et al., 2022). We regard explorations based on other competitive pre-trained models like RoBERTa (Liu et al., 2019) and DeBERTa (He et al., 2021) as future work.

2) Most of current dense demonstration retrievers, including UDR, are black-box models. Although they lead to significantly better performance than BM25, how they find informative demonstrations is still unknown. Therefore, a better understanding of the principle of informative demonstration's retrieval or an interpretable and transparent demonstration retriever may be the next stage of improving demonstration retrieval. Xu et al. (2023) propose a more explainable method, beyond-context learning, which first uses the language model to get training data's next word probability distribution, then assigns test instances with labels of their nearest neighbors with similar next word's probability distribution. We leave demonstration retrieval with better explainability as future work.

3) In the training stage we use LM to score candidates separately but in the inference stage LM is provided with a sequence of demonstrations. Although experimental results demonstrate UDR's effectiveness, we think it is a promising direction to model the dependence between different demonstrations and leave it to future work.

6 Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 62236004 and No. 62022027) and Shenzhen City's Science and Technology Plan Project (No. JSGG20210802153806021).

References

Sweta Agrawal, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2022. In-context examples selection for machine translation. *CoRR*, abs/2212.02437.

Reinald Kim Amplayo, Arthur Brazinskas, Yoshi Suhara, Xiaolan Wang, and Bing Liu. 2022. Beyond opinion mining: Summarizing opinions of customer reviews. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 3447–3450. ACM.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin,

Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Andrew Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-oriented dialogue as dataflow synthesis. *Trans. Assoc. Comput. Linguistics*, 8:556–571.

Roy Bar-Haim, Ido Dagan, and Idan Szpektor. 2014. Benchmarking applied semantic inference: The PASCAL recognising textual entailment challenges. In *Language, Culture, Computation. Computing - Theory and Technology - Essays Dedicated to Yaacov Choueka on the Occasion of His 75th Birthday, Part I*, volume 8001 of *Lecture Notes in Computer Science*, pages 409–424. Springer.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021*, pages 610–623. ACM.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational 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*.

Christopher J. C. Burges. 2010. From RankNet to LambdaRank to LambdaMART: An overview. Technical report, Microsoft Research.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *CoRR*, abs/2107.03374.

Yanda Chen, Chen Zhao, Zhou Yu, Kathleen R. McKeown, and He He. 2022. On the relation between sensitivity and accuracy in in-context learning. *CoRR*, abs/2209.07661.

Qinyuan Cheng, Xiaogui Yang, Tianxiang Sun, Linyang Li, and Xipeng Qiu. 2023. Improving contrastive learning of sentence embeddings from AI feedback. *CoRR*, abs/2305.01918.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Alexis Conneau and Guillaume Lample. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7057–7067.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. 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 9594–9611. Association for Computational Linguistics.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2022. A survey for in-context learning.

Ondřej Dušek, David M Howcroft, and Verena Rieser. 2019. Semantic Noise Matters for Neural Natural Language Generation. In *Proceedings of the 12th International Conference on Natural Language Generation (INLG 2019)*, pages 421–426, Tokyo, Japan.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: decoding-enhanced bert with disentangled attention. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Yushi Hu, Chia-Hsuan Lee, Tianbao Xie, Tao Yu, Noah A. Smith, and Mari Ostendorf. 2022. In-context learning for few-shot dialogue state tracking. *CoRR*, abs/2203.08568.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. 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 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Chao Jiang, Mounica Maddela, Wuwei Lan, Yang Zhong, and Wei Xu. 2020. Neural CRF model for sentence alignment in text simplification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7943–7960, Online. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. 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 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Byeongchang Kim, Hyunwoo Kim, and Gunhee Kim. 2019. Abstractive summarization of Reddit posts with multi-level memory networks. 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 2519–2531, Minneapolis, Minnesota. Association for Computational Linguistics.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. 2015. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Xiaonan Li and Xipeng Qiu. 2023a. Finding supporting examples for in-context learning. *CoRR*, abs/2302.13539.

Xiaonan Li and Xipeng Qiu. 2023b. Mot: Pre-thinking and recalling enable chatgpt to self-improve with memory-of-thoughts.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for gpt-3? In *Proceedings of Deep Learning Inside Out: The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures, DeeLIO@ACL 2022, Dublin, Ireland and Online, May 27, 2022*, pages 100–114. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin B. Clement, Dawn Drain, Daxin Jiang, Duyu Tang, Ge Li, Lidong Zhou, Linjun Shou, Long Zhou, Michele Tufano, Ming Gong, Ming Zhou, Nan Duan, Neel Sundaresan, Shao Kun Deng, Shengyu Fu, and Shujie Liu. 2021. Codexglue: A machine learning benchmark dataset for code understanding and generation.

In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland. Association for Computational Linguistics.

Julian J. McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Seventh ACM Conference on Recommender Systems, RecSys '13, Hong Kong, China, October 12-16, 2013*, pages 165–172. ACM.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5316–5330. Association for Computational Linguistics.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Ibrahim Naji. 2012. TSATC: Twitter Sentiment Analysis Training Corpus. In *thinknook*.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. DART: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online. Association for Computational Linguistics.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *CoRR*, abs/2203.02155.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain*, pages 271–278. ACL.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 115–124. The Association for Computer Linguistics.

Gabriel Poesia, Alex Polozov, Vu Le, Ashish Tiwari, Gustavo Soares, Christopher Meek, and Sumit Gulwani. 2022. Synchromesh: Reliable code generation from pre-trained language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Nils Reimers and Iryna Gurevych. 2019a. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019b. Sentence-bert: Sentence embeddings using siamese bert-networks. 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 2019, Hong Kong, China, November 3-7, 2019*, pages 3980–3990. Association for Computational Linguistics.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S Gordon. 2011. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *2011 AAAI Spring Symposium Series*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 2655–2671. Association for Computational Linguistics.

Peng Shi, Rui Zhang, He Bai, and Jimmy Lin. 2022. XRICL: cross-lingual retrieval-augmented in-context learning for cross-lingual text-to-sql semantic parsing. *CoRR*, abs/2210.13693.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Hongjin Su, Weijia Shi, Jungo Kasai, Yushi Hu Yizhong Wang, Mari Ostendorf, Wen tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2022. One embedder, any task: Instruction-finetuned text embeddings.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *SIGIR 2000: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 24-28, 2000, Athens, Greece*, pages 200–207. ACM.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Cunxiang Wang, Shuailong Liang, Yue Zhang, Xiaonan Li, and Tian Gao. 2019b. Does it make sense? and why? A pilot study for sense making and explanation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4020–4026. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Tomer Wolfson, Mor Geva, Ankit Gupta, Yoav Goldberg, Matt Gardner, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *Trans. Assoc. Comput. Linguistics*, 8:183–198.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Benfeng Xu, Quan Wang, Zhendong Mao, Yajuan Lyu, Qiaoqiao She, and Yongdong Zhang. 2023. $k$nn prompting: Beyond-context learning with calibration-free nearest neighbor inference.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Yiming Zhang, Shi Feng, and Chenhao Tan. 2022. Active example selection for in-context learning. *CoRR*, abs/2211.04486.

Template	MR	Yahoo	Subj
Original Template	85.2	67.5	95.0
Template 1	85.1	67.8	94.8
Template 2	85.7	67.1	94.8
Template 3	85.4	67.2	95.2

Table 7: UDR's perfromance under different inference templates. For MR, the original template and template 1, 2, 3 are "It was [Verbalizer]", "A [Verbalizer] One", "All in all [Verbalizer] .", "A [Verbalizer] one .", respectively. The verbalizers are ["great", "terrible"]. For Yahoo, the original template and template 1, 2, 3 are "Topic: [Verbalizer]", "Subject: [Verbalizer]", "This is about [Verbalizer] .", "It is about [Verbalizer] .", respectively. The verbalizers are ["Society & Culture", "Science & Mathematics, $\cdots$]. For Subj, the original template and template 1, 2, 3 are "It's [verbalizer] .", "This is [Verbalizer]", "It's all [Verbalizer].", "Is it [Verbalizer] ?", respectively. The Verbalizers are ["subjective", "objective"]. These templates are from previous works (Min et al., 2022) and for more detals please refer to Table 11.

A Task Overview

We show each task's 1. input/output domain 2. statistics and evaluation metric 3. instruction, inference template and example cases in Table 9, 10 and 11, respectively. For the dataset which has publicly available test data, we use the test data for evaluation, like SST-2, SST-5, MTOP, etc. For the others like BREAK and SMCalFlow, we follow previous work (Rubin et al., 2022) and use the dev data for evaluation. For training efficiency, we manually limit the training examples of UDR. Specifically, for the classification task whose training set size is > 30000, we randomly sample a 30000 subset for UDR's training. For the generation task whose training set size is > 100000, we randomly sample a 100000 subset for UDR's training. In the pilot experiment, we find such a strategy will not cause significant performance degradation. At the inference stage, we use the full training set as demonstrations' pool. Restricted by computational resources, we randomly sample a test set of 3000 samples for evaluation on these tasks: Amazon, Yelp, AGNews DBPedia and Yahoo.

B Implementation Details and Hyper-Parameters

We follow Rubin et al. (2022) to use GPT-Neo-2.7B (Black et al., 2021) as the scoring LM and the inference LM for most experiments in the paper unless otherwise specified. Following EPR (Rubin et al., 2022) and DPR (Karpukhin et al., 2020), we initialize E_q and E_d as two separate "BERT base uncased" encoders (Devlin et al., 2019). Thus the total number of parameters of UDR is about 220M. We use 8 NVIDIA A100s-80GB to train UDR for up to 30 epochs before iteratively mining candidates. And then we train UDR for 10 epochs at each iteration. The whole training pipeline including scoring candidates takes about 8 days. In the pilot experiment, we select the number of training epochs through the average performance on validation set on single-task SST-2, TREC, MTOP, Java code summarization, WikiAuto and DART. We set the number of iterations as 3. We follow EPR (Rubin et al., 2022) to set learning rate and batch size as 1e-4 and 128 and we use AdamW (Loshchilov and Hutter, 2019) as the optimizer. We list the overall hyper-parameters in Table 8. On each task, we use one specific template for scoring and inference (see Table 11). For fair comparison, we train DR-Target, EPR and UDR under the same hyper-parameter and report their average performance under three random seeds.

Hyper-parameters	
Optimizer	AdamW
Warmup Steps	500
Learning Rate	1e-4
Batch Size	128
Loss Weight	0.8
Iteration Number	3
Scoring Candidates Num (K)	50
Training Candidates Num (l)	8

Table 8: Hyper-parameters.

The initialization of UDR's candidates For classification and multi-choice tasks, we initialize candidates as those examples that have similar input with x by BM25. For generation tasks, similarly, we initialize candidates as those of similar targets with x's, inspired by previous work (Rubin et al., 2022).

The Quantity of EPR's Candidates Since UDR's training needs to score iteratively mined candidates and thus has to score more candidates than EPR, we also run experiments on EPR with the same candidate quantities of UDR. But we find increasing the candidates of EPR instead slightly hurts its overall performance, which is consistent with its original paper (Rubin et al., 2022). Thus for EPR, we use the same number of candidates as its original paper.

C Performance across varying inference templates

For UDR, we use one specific template when scoring candidates and here we evaluate UDR's transferability across different inference templates on MR, Yahoo and Subj. The results are shown in Table 7. We can see that the performance gap across various inference templates is smaller than 1 point and this reflects UDR's stability and transferability across different inference templates.

D Potential Risk

Previous works have shown Large language models can have various kinds of bias (Bender et al., 2021). Since UDR is trained from the feedback of large language models, it can also contain such bias.

Task Family	Task	Input	Output
Sentiment Classification	SST-2	Short Movie Review	Sentiment Label
	SST-5	Short Movie Review	Sentiment Label
	Amazon	Amazon Product Review	Sentiment Label
	Yelp	Yelp Review	Sentiment Label
	MR	Movie Review	Sentiment Label
	CR	Electronics Review	Sentiment Label
Topic Classification	AGNews	News Article	Topic Label
	TREC	Question	Topic Label
	DBPedia	Wikipedia Text	Topic Label
	Yahoo	Question-answer Pair	Topic Label
Multi-Choice	COPA	Causal Reasoning Question	Effect/Cause
	Cosmos QA	Causal Reasoning Question	Effect/Cause
	ComV	Commonsense Hypotheses	Wrong Hypothesis
	ComE	Wrong Hypothesis	Explaination
NLI	MNLI	Image-caption Sentence Pair	Entailment Label
	SNLI	Cross-genre Sentence Pair	Entailment Label
	RTE	Wikipedia/News Sentence Pair	Entailment Label
Subjective Classification	Subj	Movie Review	Subjectivity
Lingustic Acceptibility	COLA	Linguistics Publication Sentence	Grammatical Label
Semantic Parsing	BREAK	Question	Question Decomposition
	MTOP	User Utterance	TOP Representation
	SMCalFlow	User Utterance	Dataflow Program
Text Summarization	CNN/DailyMail	News Article	Highlights
	PubMed	Scientific Paper's Introduction	Abstract
	Reddit	Reddit Post	Summary
Commensense Generation	Commen Gen	Concepts	Coherent Sentence
Story Generation	Roc Story	Head of Story	Remaining Story
	Roc Stroy Ending	Four-sentence Story	Story Ending
Code Summarization	Go	Go Code	Documentation
	Python	Python Code	Documentation
	Java	Java Code	Documentation
	PHP	PHP Code	Documentation
Text Simplification	WikiAuto	Wikipedia Sentence	Simplified Sentence
	WikiAuto-Turk	Wikipedia Sentence	Simplified Sentence
	WikiAuto-ASSET	Wikipedia Sentence	Simplified Sentence
Data to Text	DART	Triple Set	Text
	E2E	Key-value Pairs	Text

Table 9: The Input/Output Domains of Tasks.

Task Family	Task	Train	Dev	Test	Report Split	Metric
Sentiment Classification	SST-2	6911	873	1821	Test	Acc
	SST-5	8534	1101	2210	Test	Acc
	Amazon	30000	5000	3000	Test	Acc
	Yelp	30000	-	3000	Test	Acc
	MR	8662	-	2000	Test	Acc
	CR	1772	-	1996	Test	Acc
Topic Classification	AGNews	29914	-	3000	Test	Acc
	TREC	5381	-	500	Test	Acc
	DBPedia	30000	-	3000	Test	Acc
	Yahoo	29150	-	3000	Test	Acc
Multi-Choice	COPA	500	-	500	Test	Acc
	Cosmos QA	18770	2603	6030	Dev	Acc
	ComE	9996	997	1000	Test	Acc
	ComV	9992	997	1000	Test	Acc
NLI	MNLI	263789	3000	9796	Dev	Acc
	SNLI	131062	3272	3262	Test	Acc
	RTE	2490	277	3000	Dev	Acc
Subjective Classification	Subj	8000	-	2000	Test	Acc
Lingustic Acceptibility	COLA	8532	-	527	Test	Acc
Semantic Parsing	BREAK	44321	7760	8069	Dev	LF-EM
	MTOP	15667	2235	4386	Test	EM
	SMCalFlow	133584	14751	22012	Dev	EM
Text Summarization	CNN/DailyMail	155098	7512	6379	Test	Rouge-L
	PubMed	56254	3187	3481	Test	Rouge-L
	Reddit	37643	576	562	Test	Rouge-L
Commensense Generation	Commen Gen	67389	993	1497	Dev	BLEU-3
Story Generation	Roc Story	87526	9799	9799	Test	BLEU-1
	Roc Stroy Ending	87906	9807	9807	Test	BLEU-1
Code Summarization	Go	167137	7320	8115	Test	BLEU-1
	Python	250818	13841	14840	Test	BLEU-1
	Java	164514	5172	10928	Test	BLEU-1
	PHP	240851	12964	13998	Test	BLEU-1
Text Simplification	WikiAuto	481018	1999	403	Test	SARI
	WikiAuto-Turk	-	1999	359	Test	SARI
	WikiAuto-ASSET	-	1999	359	Test	SARI
Data to Text	DART	30123	2718	4159	Test	BLEU-4
	E2E	12563	1483	1847	Test	BLEU-4

Table 10: The statistics, split and evaluation metrics of each dataset.

Task Family: Sentiment Classification

Task: SST-2
Task Instruction: Sentiment of the sentence:
Inference Verbalizer: {great, terrible}
Inference Template:
Input:
A three-hour cinema master class.
It was terrible.
A pretensions – and disposable story — sink the movie.
It was great.
. . .
The movie 's blatant derivativeness is one reason it 's so lackluster.
It was
Output:
terrible.

Task: SST-5
Task Instruction: Sentiment of the sentence:
Inference Verbalizer: {great, good, okay, bad, terrible}
Inference Template: Same as SST-2

Task: Amazon
Task Instruction: Sentiment of the sentence:
Inference Verbalizer: {great, good, okay, bad, terrible}
Inference Template: Same as SST-2

Task: Yelp
Task Instruction: Sentiment of the sentence:
Inference Verbalizer: {great, good, okay, bad, terrible}
Inference Template: Same as SST-2

Task: MR
Task Instruction: Sentiment of the sentence:
Inference Verbalizer: {great, terrible}
Inference Template: Same as SST-2

Task: CR
Task Instruction: Sentiment of the sentence:
Inference Verbalizer: {great, terrible}
Inference Template: Same as SST-2

Task Family: Topic Classification

Task: AGNews
Task Instruction: Topic of the text:
Inference Verbalizer: {World, Sports, Business, Technology}
Inference Template:
Input:
'LONDON, Oct 26 (AFP) - World oil prices will be driven down over the next two years due to there being enough crude to meet soaring demand, Claude Mandil, executive director of the International Energy Agency (IEA), said here Tuesday.
Topic: Business.
WASHINGTON - This year's surge in energy prices is likely to have far less of an impact on the economy than the oil shocks of the 1970s, Federal Reserve Chairman Alan Greenspan said Friday. Greenspan predicted that the global economy will adjust to the recent surge in prices, which has seen oil topping \\$50 per barrel, by boosting energy exploration and production and by increasing fuel efficiency...
Topic: World.
. . .
Oil demand is rising faster than predicted this year as OPEC pumps more low-quality oil in a failed bid to reduce record prices, according to International Energy Agency, an adviser to 26 industrialized nations.
Topic:
Output:
Business.

Task: TREC
Task Instruction: Topic of the question:
Inference Verbalizer: {Description, Entity, Expression, Human, Location, Number}
Inference Template: Same as AGNews

Task: DBPedia
Task Instruction: Topic of the text:
Inference Verbalizer: {Company, Educational Institution, Artist, Athlete, Office Holder, Mean of Transportation, Building, Natural Place, Village, Animal, Plant, Album, Film, Written Work}
Inference Template: Same as AGNews

Task: Yahoo
Task Instruction: Topic of the text:
Inference Verbalizer: {Society & Culture, Science & Mathematics, Health, Education & Reference, Computers & Internet, Sports, Business & Finance, Entertainment & Music, Family & Relationships, Politics & Government}
Inference Template: Same as AGNews

Task Family: Multi-Choice

Task: COPA
Task Instruction: Answer the question based on the text.
Inference Template:
Input:
I scratched my skin. What happened as a result?
My itch went away.
misplaced my wallet. What happened as a result?
I retraced my steps.
. . .
I emptied my pockets. What happened as a result?
Output:
I retrieved a ticket stub.

Task: Cosmos QA
Task Instruction: Answer the question based on the text.
Inference Template: Same as COPA

Task: ComV
Task Instruction: Which statement of the two is against common sense?
Inference Template: Same as COPA

Task: ComE
Task Instruction: Select the most corresponding reason why this statement is against common sense.
Inference Template: Same as COPA

Task Family: NLI

Task: MNLI
Task Instruction: Recognizing textual entailment between these 2 texts.
Inference Verbalizer: {Entailment, Inconclusive, Contradiction}
Inference Template:
Input:
uh-huh exactly not what color you are how old you are what if your male or female that would be wonderful i guess it's kind of an ideal world though huh *Based on that information, is the claim* The world would be better if race and gender did not matter. People would get along much better "Entailment", "Contradiction", or "Inconclusive"?
Answer: Inconclusive.
uh-huh exactly not what color you are how old you are what if your male or female that would be wonderful i guess it's kind of an ideal world though huh *Based on that information, is the claim* The world would be better if race and gender did not matter. "Entailment", "Contradiction", or "Inconclusive"?
Answer: Entailment.
. . .
It's that kind of world. *Based on that information, is the claim* The world is getting better. "Entailment", "Contradiction", or "Inconclusive"?
Answer:
Output:
Inconclusive

Task: SNLI

Task Instruction: Recognizing textual entailment between these 2 texts.
Inference Verbalizer: {Entailment, Inconclusive, Contradiction}
Inference Template: Same as MNLI

Task: RTE
Task Instruction: Recognizing textual entailment between these 2 texts.
Inference Verbalizer: {True, False}
Inference Template: Same as MNLI

Task Family: Subjective Classification

Task: Subj
Task Instruction: Subjectivity of the sentence:
Inference Verbalizer: {subjective, objective}
Inference Template:
Input:
thirteen conversations about one thing lays out a narrative puzzle that interweaves individual stories , and , like a mobius strip , elliptically loops back to where it began .
It's subjective.
a small gem of a movie that defies classification and is as thought-provoking as it is funny , scary and sad .
It's subjective.
. . .
smart and alert , thirteen conversations about one thing is a small gem .
It's
Output:
subjective

Task Family: Lingustic Acceptibility

Task: COLA
Task Instruction: The grammaticality of this sentence:
Inference Verbalizer: {not grammatical, grammatical}
Inference Template:
Input:
The sea monster drowned the sailors.
It is grammatical.
He rode out the storm.
It is grammatical.
. . .
The sailors rode the breeze clear of the rocks.
It is
Output:
grammatical

Task Family: Semantic Parsing

Task: BREAK
Task Instruction: Parse the sentence into logical form:
Inference Template:
Input:
Parse the sentence into logical form: what flights are available from pittsburgh to boston on saturday
1#) return flights 2#) return #1 from pittsburgh 3#) return #2 to boston 4#) return #3 on saturday 5#) return #4 that are available
P*arse the sentence into logical form:* what flights are available wednesday afternoon from denver to san francisco
1#) return flights 2#) return #1 from denver 3#) return #2 to san francisco 4#) return #3 on wednesday afternoon 5#) return #4 that are available
. . .
Parse the sentence into logical form: what flights are available tomorrow from denver to philadelphia
1#)
Output:
return flights ;return #1 from denver ;return #2 to philadelphia ;return #3 if available

---

Task: MTOP
Task Instruction: Parse the sentence into logical form:
Inference Template: Same as BREAK

Task: SMCalFlow
Task Instruction: Parse the sentence into logical form:
Inference Template: Same as BREAK

Task Family: Text Summarization

Task: CNN/DailyMail
Task Instruction: Summarize the text:
Inference Template:
Input:
Summarize the text: JERUSALEM (CNN) – Israel moved to defend itself in the face of international criticism Monday over its eviction of dozens of Palestinian families from a neighborhood of Jerusalem they have lived in for generations.· · ·
TL;DR: Israel incurs international criticism over eviction of Palestinian families . Two Jewish families moved in after evictions in East Jerusalem . Israeli spokesman says dispute is a legal one between private parties .
Summarize the text: (CNN)The International Criminal Court opened an inquiry into attacks in Palestinian territories, paving the way for possible war crimes investigation against Israelis.· · ·
TL;DR: An inquiry allows the court to review evidence and determine whether to file charges . The U.S. calls for negotiations between Palestinian, Israeli officials .
· · ·
Summarize the text: (CNN)The Palestinian Authority officially became the 123rd member of the International Criminal Court on Wednesday, a step that gives the court jurisdiction over alleged crimes in Palestinian territories.· · ·
TL;DR:
Output:
Membership gives the ICC jurisdiction over alleged crimes committed in Palestinian territories since last June . Israel and the United States opposed the move, which could open the door to war crimes investigations against Israelis .

Task: PubMed
Task Instruction: Summarize the text:
Inference Template: Same as CNN/DailyMail

Task: Reddit
Task Instruction: Summarize the text:
Inference Template: Same as CNN/DailyMail

Task Family: Commensense Generation

Task: Commen Gen
Task Instruction: Generate a sentence using these concepts:
Inference Template:
Input:
Generate a sentence using these concepts: counter, pizza, restaurant
Generated sentence: Two men standing at counters assembling pizzas in a restaurant.
Generate a sentence using these concepts: counter, restaurant, stand
Generated sentence: A man stands behind the counter of a restaurant.
· · ·
Generate a sentence using these concepts: field, look, stand
Generated sentence:
Output:
The player stood in the field looking at the batter.

Task Family: Story Generation

Task: Roc Story
Task Instruction: Beginning of the story:
Inference Template:
Input:
Beginning of the story: Taylor had been up all night memorizing lines for the play.
Rest of the story: She knew that most of the girls in her class would be auditioning too. She watched the other girls stumble over their lines. She took a deep breath before going up on stage with a smile. All her lines were delivered perfectly and she got the part.
Beginning of the story: Gabby was proud to be given the lead role in the school play.
Rest of the story: She had worked hard on her audition piece. She worked hard to memorize her lines for the play. When the show opened, she stood on the stage and took it all in. She loved the feeling of performing.
· · ·
Beginning of the story: Natalie had auditioned for the lead in the school play.

Rest of the story:
Output:
She won the part and was super excited. She rehearsed for weeks and weeks. On opening night, she acted her little heart out. The play was a huge success!

Task: Roc Story Ending
Task Instruction: An unfinished story:
Inference Template: Same as ROC Story

Task Family: Code Summarization

Task: Go
Task Instruction: Comment on the code.
Inference Template:
Input:
Comment on the code. Code:

```
func NewSTM ( c * v3 . Client , apply func ( STM ) error , so ... stmOption ) ( * v3 . TxnResponse , error ) {
   opts := & stmOptions {
      ctx : c . Ctx ( )
   }
   for _ , f := range so {
      f ( opts )
   }
   if len ( opts . prefetch ) != 0 {
      f := apply apply = func ( s STM ) error {
         s . Get ( opts . prefetch ... )
         return f ( s )
      }
   }
   return runSTM ( mkSTM ( c , opts ) , apply )
}
```

Comment: RunContainer runs a fake Docker container
Comment on the code. Code:

```
func ( s  Subnet ) EnsureDead ( ) ( err error ) {
   defer errors . DeferredAnnotatef ( & err , " " , s )
   if s . doc . Life == Dead {
      return nil
   }
   ops := [ ] txn . Op { {
          C : subnetsC , Id : s . doc . DocID , Update : bson . D { { " " , bson . D { { " " , Dead } } } } , Assert :
isAliveDoc ,
      } }
   txnErr := s . st . db ( ) . RunTransaction ( ops )
   if txnErr == nil {
      s . doc . Life = Dead return nil
   }
   return onAbort ( txnErr , subnetNotAliveErr ) }
```

Comment: EnsureDead sets the Life of the subnet to Dead if it s Alive . If the subnet is already Dead no error is returned . When the subnet is no longer Alive or already removed errNotAlive is returned .
. . .
Comment on the code. Code:

```
func NewSTM ( c * v3 . Client , apply func ( STM ) error , so ... stmOption ) ( * v3 . TxnResponse , error ) {
   opts := & stmOptions { ctx : c . Ctx ( ) }
   for _ , f := range so { f ( opts ) }
   if len ( opts . prefetch ) != 0 {
      f := apply apply = func ( s STM ) error { s . Get ( opts . prefetch ... ) return f ( s ) }
   }
   return runSTM ( mkSTM ( c , opts ) , apply )
}
```

Comment:
Output:
NewSTM initiates a new STM instance using serializable snapshot isolation by default .

Task: Python
Task Instruction: Comment on the code.
Inference Template: Same as Go

Task: Java
Task Instruction: Comment on the code.
Inference Template: Same as Go

Task: PHP
Task Instruction: Comment on the code.
Inference Template: Same as Go

Task Family: Text Simplification

Task: WikiAuto
Task Instruction: Simplify the text:
Inference Template:
Input:
Simplify the text: Stanton went on to write some of the most influential books , documents , and speeches of the women 's rights movement .
Simplified text: Together they wrote speeches , articles , and books .
Simplify the text: When she was eighteen and without a university education , she began writing for the newspaper " Exce lsior " , doing interviews and society columns .
Simplified text: She began writing for the newspaper " Exce lsior " , doing interviews and society columns .
. . .
Simplify the text: Together with James, she compiled crosswords for several newspapers and magazines, including People, and it was in 1978 that they launched their own publishing company.
Simplified text:
Output:
Together with James, she compiled crosswords. It was for several newspapers and magazines, including People. They launched their own publishing company. It was in 1978.

Task: WikiAuto-Turk
Task Instruction: Simplify the text:
Inference Template: Same as WikiAuto

Task: WikiAuto-ASSET
Task Instruction: Simplify the text:
Inference Template: Same as WikiAuto

Task Family: Data to Text

Task: DART
Task Instruction: Describe the table in natural language.
Inference Template:
Input:
Describe the table in natural language. Table: [Baywatch | NOTES | Episode: Red Wind], [Baywatch | ROLE | Kim], [[TABLECONTEXT] | TITLE | Baywatch], [[TABLECONTEXT] | [TITLE] | Bobbie Phillips]
Sentence: Bobbie Phillips appeared on the episode Red Wind in Baywatch as Kim.
Describe the table in natural language. Table: [Silk Stalkings | ROLE | Tessa Shaver], [[TABLECONTEXT] | [TITLE] | Bobbie Phillips], [[TABLECONTEXT] | TITLE | Silk Stalkings], [Silk Stalkings | NOTES | Episode: Goodtime Charlie]
Sentence: Actress Bobbie Phillips was casted as Tessa Shaver on the episode Goodtime Charlie of Silk Stalkings.
. . .
Describe the table in natural language. Table: [Hawaii Five-O | NOTES | Episode: The Flight of the Jewels], [[TABLECONTEXT] | [TITLE] | Jeff Daniels], [[TABLECONTEXT] | TITLE | Hawaii Five-O]
Sentence:
Output:
Jeff Daniels played in the Hawaii Five-O episode The Flight of the Jewels

---

Task: E2E
Task Instruction: Describe the table in natural language.
Inference Template: Same as DART

Table 11: The instructions, inference templates and example cases of tasks.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
the "limitations" section after "conclusion" section

☑ A2. Did you discuss any potential risks of your work?
Appendix D

☑ A3. Do the abstract and introduction summarize the paper's main claims?
in the end of abstract and introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

section Experiments

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Appendix B

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix B

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Appendix B

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix B

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.