IT2ACL: Learning Easy-to-Hard Instructions via 2-phase Automated Curriculum Learning for Large Language Models

Yufei Huang, Deyi Xiong*
College of Intelligence and Computing, Tianjin University, Tianjin, China
{yuki_731, dyxiong}@tju.edu.cn

Abstract

Instruction tuning has demonstrated its superiority in unlocking the abilities of pre-trained large language models (LLMs), including their capability to respond to diverse human instructions and conduct complex reasoning. In order to further enhance the continuous learning capabilities of pre-trained LLMs, we explore the training process of instruction tuning through the lens of task sequences. We propose a 2-phase automated curriculum learning guided instruction tuning framework, IT2ACL that learns easy-to-hard instructions for LLMs in a self-adjusting dynamic manner. To facilitate curriculum learning from instructions, we propose a loss-driven progress signal for two-phase strategies: instruction prediction gain that decides the instruction level syllabus. Through comprehensive experiments on 70 Chinese datasets which have been grouped into 16 distinct task clusters, we demonstrate the effectiveness of our approach in eliciting latent ability in pre-trained LLMs and achieving superior performance across diverse tasks.

Keywords: Instruction Tuning, Curriculum Learning, Large Language Model

1. Introduction

Instruction Tuning (Wei et al., 2021) is a method for fine-tuning models, with large language models like GPT-3 (Brown et al., 2020) demonstrating a strong capability for both zero- and few-shot learning (Rae et al., 2021; Smith et al., 2022; Guo et al., 2023). It operates by supplying task descriptions or prompts to aid the model in better comprehending and adapting to a specific task or dataset. Most natural language processing (NLP) tasks can be described through natural language instructions, and a significant advantage of this is that it allows us to unify all tasks under one framework (Shen et al., 2023). Sanh et al. (2021) and Wei et al. (2021) introduce T0 and FLAN, respectively. Both create diverse instructions that unify multiple tasks into generative tasks, significantly enhancing the model's performance on unseen tasks. Scialom et al. (2022) has demonstrated that the adaptability and continual learning capabilities of these large models are primarily rooted in their pre-training phase. Therefore, finding ways to further unlock a model's potential post pre-training during the fine-tuning phase is crucial. Previous research has primarily focused on seeking better instruction templates (Prasad et al., 2022; Deng et al., 2022; Wei et al., 2022) and has investigated the impact of proportions of different datasets within the training set. However, few have delved into the sequence of tasks during the instruction-tuning process.

The fundamental idea of curriculum learning (CL) (Bengio et al., 2009) is to train models in an order from simple to complex instead of having the model handle all the training samples right from the start. This approach allows the model to first grasp the basics and then progressively transition to more complex tasks. This method has been proven effective in many downstream tasks, like dialogue tasks (Cai et al., 2020; Su et al., 2020), relation extraction (Yang and Song, 2022), and machine translation (Liu et al., 2020; Platanios et al., 2019; Zhang et al., 2019; Zhou et al., 2020). It can speed up training and enhance the model's final performance (Weinshall et al., 2018).

Based on the experience from previous work, we hypothesize that combining CL with instruction tuning can lead to improved model generalization, faster convergence, and reduced overfitting by progressively teaching complex concepts rather than overly relying on specific task-related features or noise. However, integrating these two methods poses some potential challenges: (i) Establishing a fixed order might not be optimal and manually defining the difficulty of a task (Liu et al., 2020; Fomicheva et al., 2020) is a complex and time-consuming process.(ii) Instructions are vital tools in instruction tuning. Determining the best instruction for each task might require extensive experimentation and iterations.

To this end, we propose IT2ACL, a Instruction Tuning framework guided by 2-phase Automated Curriculum Learning (Graves et al., 2017), which learns easy-to-hard instructions for language models. Particularly, we propose a two-phase learning approach to embedding the CL strategy into instruction tuning: In the first phase, the model learns different tasks in an order from simple to complex. In the second phase, once a particular task is determined, the model progresses from easier to more challenging instructions within that task.

*Corresponding author

LREC-COLING 2024, pages 9405–9421
20-25 May, 2024.

For instance, the model might first fine-tune on relatively simple tasks, such as part-of-speech tagging, named entity recognition, or sentence-level sentiment analysis. Within each task, there is a set of instructions. This set might include straightforward instructions like "Translate the Chinese words to French" and more complex phrasing like "Convert all of the words in the input column to their French translations." As the model's learning and fine-tuning advance, the complexity of the tasks and instructions can be progressively increased by introducing tasks that require more intricate skills like reasoning, extensive comprehension, or dialogue management.

However, tasks or instructions that appear simple to humans might not necessarily be for models, and their difficulty might dynamically change during the training process. Addressing this concern, we introduce a progress signal to guide the order of learning for both phases. The instruction prediction gain focuses on the decrease in loss when the model trains under different instructions within the same task. After multiple training sessions, the weighted average of the signals from all instructions within a task serves as the signal for the entire task, aids in defining the sequential order of tasks. Our training scheduler utilizes the adversarial multi-armed bandits (MAB) strategy (Auer et al., 2002). The progress signal is scaled to a fixed range, then perceived as the reward in MAB, dynamically choosing the optimal path for the model while maximizing the overall gain.

We validate our approach in two experimental setups. Under a full-shot setting, our results outperform both those without curriculum learning and those using a predefined order of CL. In some tasks, our method even surpasses models fine-tuned solely on individual tasks. Under a zero-shot setting, our model showcases robust generalization capabilities to new tasks. Extensive analysis reveals that introducing the second phase of instruction curriculum learning is one of the factors leading to further improvements in our model. Notably, our method provides more significant performance enhancements on tougher tasks. Additionally, our approach offers a mechanism to identify the optimal instruction under different tasks.

In summary, our contributions are threefold:

- We introduce a novel approach to instruction tuning by incorporating 2-phase automated curriculum learning strategies, aiming to enhance model's ability to understand and execute instructions across a diverse set of tasks.
- We design a unique loss-driven progress signal for different instructions within the same task, and employ the adversarial multi-armed bandits approach, enabling the model to learn tasks in an automatically adjusted dynamic easy-to-hard manner.
- We conduct extensive experiments to substantiate the effectiveness of our approach, which demonstrates improved performance across various tasks.

2. Related Work

Curriculum Learning Bengio et al. (2009) provided an excellent experimental overview of early curriculum learning efforts. In this field, a pressing question is: What are the universal rules that make some curriculum superior? Some methods rely on human judgment based on prior linguistic knowledge, such as sentence length and word rarity (Liu et al., 2020; Platanios et al., 2019; Zhang et al., 2019; Zhou et al., 2020), soft edit distance metrics of data samples (Chang et al., 2021; Mohiuddin et al., 2022), manually determining data difficulty and adjusting the learning schedule (Liu et al., 2020; Fomicheva et al., 2020). Others use model-based capabilities, such as the model's prediction confidence on instances (Mohiuddin et al., 2022; Varshney et al., 2022; Wan et al., 2020), the model's minimum loss (Weinshall et al., 2018), and automatically adjust the learning schedule at each iteration step (Wan et al., 2020). Furthermore, methods from reinforcement learning are explored to determine task (sample) difficulty, such as teacher-guided, student feedback-based curriculum learning (Graves et al., 2017; Matiisen et al., 2020; Portelas et al., 2019), curriculum learning where two agents play against each other based on the main task (Sukhbaatar et al., 2018), and curriculum learning with automatic goal generation (Florensa et al., 2018; Racanière et al., 2019).

Continual Learning Jin et al. (2022) demonstrated that the CL algorithm can effectively retain knowledge, allowing current models to have the ability for continual learning without forgetting any previously acquired knowledge and skills. This can significantly improve the model's generalization performance on unseen tasks. CL has also shown a promising ability for continual learning in task-oriented dialogue systems (Madotto et al., 2021), neural machine translation tasks (Cao et al., 2021), and sentiment classification tasks (Ke et al., 2021). Sun et al. (2020) proposed a lifelong learning method called LAMOL, which continually learns new tasks by replaying pseudo-samples from previous tasks, achieving performance comparable to state-of-the-art lifelong learning methods. Meanwhile, Scialom et al. (2022) suggest addressing continual learning issue more broadly. They introduced rehearsal as a mechanism to prevent

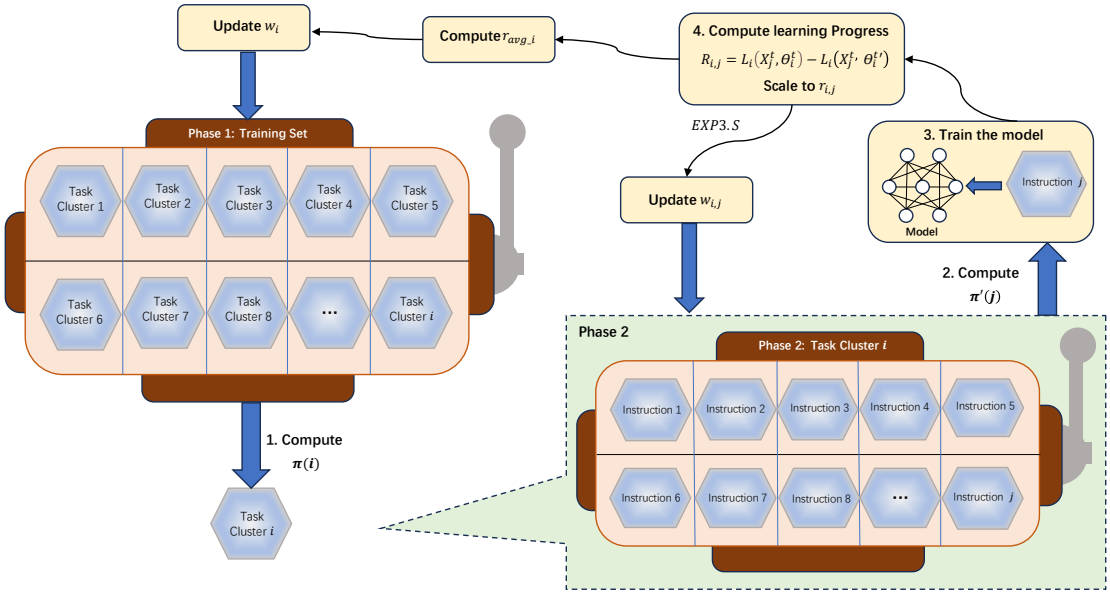

Figure 1: Diagram of the proposed IT2ACL framework.

catastrophic forgetting in CL and validated their approach on 70 tasks. Yin et al. (2022) proposed two strategies: learning from negative outputs and revisiting instructions from previous tasks, to provide models with continual learning capabilities.

Both CT0 (Scialom et al., 2022) and ConTinTin (Yin et al., 2022) have focused on the scenario of combining instruction learning with curriculum learning, which is very similar to the motivation of our work. However, we differ in the following three aspects: (i) Methodology: They train models using a predefined sequence and then prevent forgetting by introducing an external memory module, while we introduce automated learning methods to dynamically adjust the curriculum learning outline; (ii) Experimental setup: We have conducted validations under both full- and zero-shot scenarios; (iii) Application scenarios: All their training data and models are in English, whereas ours is entirely based on Chinese.

3. Methodology

The overall framework IT2ACL that incorporates automated curriculum learning into instruction tuning is shown in Figure 1. We first introduce how we group datasets into different task clusters, and then elaborate the core curriculum learning modules: the loss-driven progress signals and the dynamic training scheduler, which exploit the adversarial multi-armed bandits strategies.

3.1. Curriculum Definition

In instruction tuning, we define the target sequence as set B, the input data sequence as set A, and the instruction set as T. Thus, each batch can be viewed as an individual instance x from $X = (A \times T \times B)^N$. Consequently, a task can be interpreted as the distribution D over the sequences in X with the same t from T. In the context of curriculum learning, a course is a set of tasks $D^1, ..., D^N$, and our curriculum represents the temporal variation of task distributions. We consider the model parameters to be denoted by θ, and the expected loss for the model on the k^{th} task is $L_k(\theta)$:

$$L_k(\theta) := \mathbb{E}_{x \sim D_k} L(x, \theta) \quad (1)$$

Generally, in the multi-task scenario of instruction tuning, our goal is to perform as well as possible across the entire task set D_k, which is represented by the objective function L_{MT}:

$$L_{\text{MT}} := \frac{1}{N} \sum_{k=1}^{N} L_k \quad (2)$$

3.2. Distribution D: Task Clusters

We collect 70 Chinese fine-tuning datasets from different domains: legal, medical, e-commerce, and social media, as summarized in Figure 2. Based on factors such as the original format of the datasets, their initial task source, and their broad usage, we group these 70 datasets into 16 task clusters. In the subsequent sections, when we mention "task", it refers to these clusters, each containing one or multiple datasets. Following this categorization, the data volume of different clusters varies greatly. Thus, to balance them, we perform either upsampling or downsampling for each cluster, aiming to

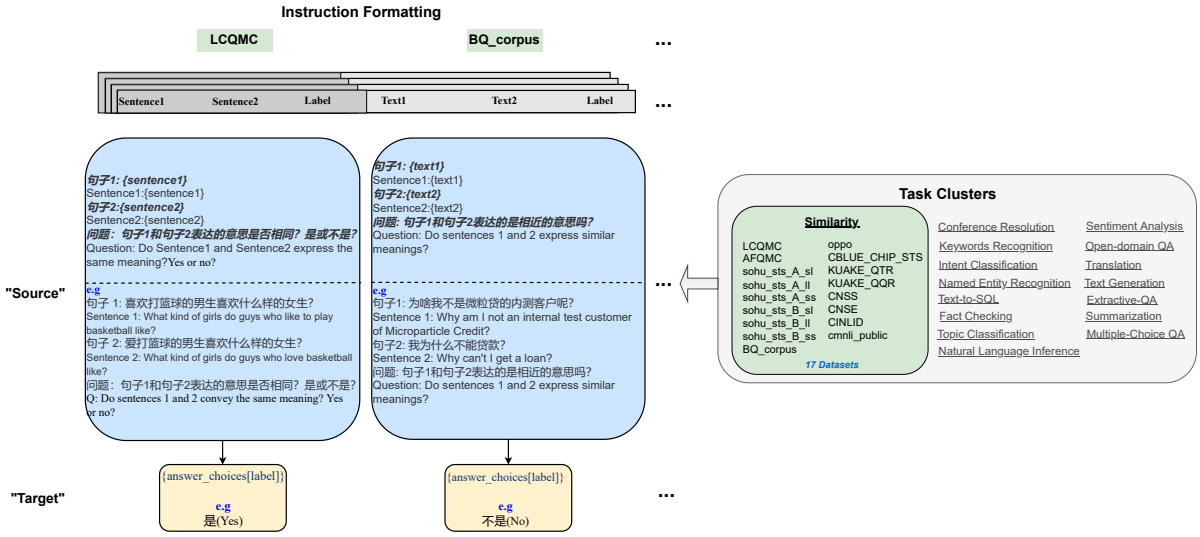

Figure 2: The task clusters (right) and examples of instruction template of LCQMC dataset and bq_corpus dataset in the simarlarity task(left).

limit the data volume of each cluster to around 10K instances, to prevent the model from oversampling a specific task and causing biased results.

3.3. Sequence T: Instruction formatting

After grouping all datasets into their respective clusters, we manually create ten instruction templates for each datasets in one task cluster. We define an *instruction* as consisting of a source and a target template, and a set of associated metadata. Such feature allows us to unify all datasets into one dataset according to the same data format. For instance, in the Similarity task with examples from LCQMC (Liu et al., 2018b) , the fields include `Sentence1`, `Sentence2`, `Label`. We use an input template: `Sentence1: {sentence1} Sentence2: {sentence2} Question: Do these sentences have the same meaning? Yes or no?`, and define the target with `{answer_choices[label]}`, where `answer_choices` is the metadata with options `yes`, `no`, aligning with `label` being yes (1), no (0). The same `source-to-target` format applies to BQ_corpus (Chen et al., 2018) and other datasets in the similarity task cluster, shown in Figure 2. The details of all data can be found in https://github.com/YFHuangxxxx/cPromptSource.

3.4. Loss-driven Progress Signals

To adhere to the curriculum learning method proposed by Bengio et al. (2009), it is crucial during the training process to design a metric that gauges task difficulty, enabling us to order tasks accordingly. Measuring task difficulty is challenging, especially in the domain of multi-task natural language generation (NLG). Prior rule-based strategies relied largely on experience and intuition. Numerous studies (Platanios et al., 2019; Kocmi and Bojar, 2017; Liu et al., 2020) have vouched for their efficacy. Measuring task difficulty based solely on sentence length becomes ambiguous for generation tasks. It remains unclear if generating longer sentences from shorter inputs is more challenging than producing shorter responses from longer inputs. Additionally, as a model's capabilities enhance, longer sequences or rare tokens aren't always considered "difficult" (Wan et al., 2020). Therefore, in reality, task difficulty might vary along multiple axes of difficulty, or there might not be a predefined order at all. To tackle this, we believe that employing automated curriculum learning (Graves et al., 2017) can maximize learning efficiency. Viewing the decision of the subsequent task to research as a stochastic policy, the model autonomously chooses the outline to follow in the curriculum, constantly adapting to optimize the concept of learning progress as mentioned by Oudeyer et al. (2007). Various learning progress have been utilized as reward signals to enhance training.

Specifically, we propose a instruction-level loss-driven signal, where loss-driven refers to the change in the overall loss function of the model before and after a single gradient update during training. Given the shared parameters in our generative model, it's expected that different tasks will contribute differently to the overall learning process. Such a reward signal mirrors the speed of learning, as the extent of loss reduction for a task is maximally equivalent to the fastest learning pace, and equivalently suggests that the task difficulty is lower for the model.

3.4.1. Instruction Prediction Gain

Once the decision is made about which task i to train on at time point t, the model is about to enter the training phase. However, it's important to note that for a given task, we often design multiple instructions. These instructions differ in their grammatical structure, format, and content. They can be computer programs or scripts, or they can be instructional texts written by humans. By adjusting the content and format of the instruction, we can guide the model to generate more accurate and useful outputs. Thus, in the second phase, during intra-task training, we let T_i represent the set of instructions for task i. Given a batch of examples $\mathcal{X}$, we define $L_j(\mathcal{X}_j, \theta_j)$ as the non-negative loss of instruction j. We define the instruction prediction gain as the instantaneous change in loss for instruction j, before and after training on examples $\mathcal{X}$ with instruction j:

$$R_{\text{INS}} = L_i(\mathcal{X}_j^t, \theta_i^t) - L_i(\mathcal{X}_j^t, \theta_i^{t\prime}) \tag{3}$$

Taking into consideration the potential instability of model when receiving a diverse range of rewards, we also need to scale these computed rewards. Initially, we define R_t as the set of all rewards received up until time point t. Subsequently, from this set, we identify maximum value $R_{t_{\text{max}}}$ and the minimum value $R_{t_{\text{min}}}$. Aiming to adjust all rewards within the fixed range of $[-1, 1]$, we proceed with scaling according to the following formula:

$$r_t = \frac{2(R_{\text{INS}} - R_{t_{min}})}{R_{t_{max}} - R_{t_{min}}} - 1 \tag{4}$$

3.5. Adversarial Multi-Armed Bandits

After obtaining the aforementioned rewards r, we now need the model to adaptively adjust its training strategy based on these reward observations. We treat curriculum learning with n tasks as an Adversarial Multi-Armed Bandits (MAB) problem (Bubeck et al., 2012). Specifically, the MAB problem can be described as having N machines, each with a reward probability. At each time step t, we choose to execute action a on one slot machine and receive a reward r from that machine, while the rewards from the other machines remain unobserved. The agent's goal is to maximize the cumulative reward, meaning it needs to determine the order in which to pull the arm of these N slot machines. In the MAB context, an effective approach is to use adaptive policy learning to optimize the gains obtained from the bandit. In the scenario applied in this paper, the optimal curriculum outline is constantly changing, and the reward the model receives after each change also varies. This setting leans towards adversarial slot machines. A classic algorithm for this is Exp3 (Auer et al., 2002). It selects an arm based on the current weight distribution, observes the reward of the chosen arm, uses the observed reward information to update the weights of all arms, and then repeats the above steps. The probability of choosing an arm is as follows:

$$\pi_t^{EXP3}(i) = \frac{e^{\eta w_t^i}}{\sum_{j=1}^{N} e^{\eta w_t^j}} \tag{5}$$

The formula for weight update is:

$$w_{t+1}^i = w_t^i \times \exp\left(\frac{\eta r_t^i}{\pi_t^{EXP3}(i)}\right) \tag{6}$$

While Exp3 primarily focuses on weight updates, there are scenarios where it might miss out on exploration opportunities due to not selecting certain arms for extended periods. This can be detrimental to the generalized performance improvement of our model across multiple tasks. Herbster and Warmuth (1995) first introduce the Exp3.S strategy to address this issue. Exp3.S was designed to tackle this problem by employing an ε-greedy strategy and additively mixing the weights. This ensures that each arm gets a chance to be selected, enhancing exploration. This is particularly useful in dynamic environments where the reward distribution may change over time and frequent strategy adjustments are needed. Thus, in this paper, we use the algorithm that Auer et al. (2002) further refine and apply based on Exp3.S:

$$\pi_t^{\textsc{exp3.p}}(i) := (1-\epsilon)\pi_t^{\textsc{exp3}}(i) + \frac{\epsilon}{N} \tag{7}$$

$$w_{t,i}^{\textsc{s}} := \log\Big[(1-\alpha_t)\exp\Big\{w_{t-1,i}^{\textsc{s}} + \eta\tilde{r}_{t-1,i}^{\beta}\Big\} + \frac{\alpha_t}{N-1}\sum_{j\neq i}\exp\Big\{w_{t-1,j}^{\textsc{s}} + \eta\tilde{r}_{t-1,j}^{\beta}\Big\}\Big]$$

$$w_{1,i}^{\textsc{s}} := 0 \qquad \alpha_t := t^{-1} \qquad \tilde{r}_{s,i}^{\beta} := \frac{r_s \mathbb{I}_{a_s=i} + \beta}{\pi_s(i)}.$$

3.6. Automated Curriculum Learning for Instruction Tuning

In the end, our training procedure is essentially a two-stage multi-armed bandit problem, where task selection constitutes the first stage and instruction selection the second. At the outset of training, we initialize the weights w_i for all tasks $i \in [D]$ and the weights $w_{i,j}$ for each instruction $j \in [T]$ within each task to 0. For each instruction, we also maintain two additional information: historical rewards $R_{i,j}$ and the number of selected times $C_{i,j}$. At each time t, we first compute the policy $\pi(i)$ based on the task weights w_i. And then we sample a task index $k \sim \pi(i)$. We then move to the second stage, where we compute the policy $\pi'(j)$ based on the weights $w_{i,j}$ of the instructions in task k and

Algorithm 1 Two-phase Automated Curriculum Learning

Initially: $w_i = 0$ for $i \in D$, $w_{i,j} = 0$ for $j \in T$, historical rewards $R_{i,j} = 0$ and counts $C_{i,j} = 0$

for $t = 1 \dots T$ **do**
  $\pi(k) := (1-\epsilon)\frac{e^{w_k}}{\sum_i e^{w_i}} + \frac{\epsilon}{N}$
  Select task index k from $\pi(k)$.
  $\pi'(j) := (1-\epsilon)\frac{e^{w_{k,j}}}{\sum_j e^{w_{k,j}}} + \frac{\epsilon}{M}$
  Select instruction index j for task k from $\pi'(j)$.

  Train network p_θ using instruction j of task k.
  Compute learning progress R. (Sections 3.4.1)
  Scale R to reward r.
  Update $w_{k,j}$ using Exp3.S (7).
  Update $R_{k,j}$ and $C_{k,j}$.
  Compute weighted average reward using Eq.(8)
  Update w_k using Exp3.S (7).
end for

subsequently sample an instruction index $l \sim \pi'(j)$. We train the model using the selected instruction l and compute the learning progress ν. This learning progress is then mapped to a reward r. Next, we employ Exp3.S to update the weights $w_{i,j}$ for all instructions. In parallel, we update the historical rewards $R_{i,j}$ and increment its selection count $C_{i,j}$. Subsequently, we compute the weighted average reward r_{avg_k} for task k:

$$r_{\text{avg}_k} = \frac{\sum_j R_{k,j}}{\sum_j C_{k,j}} \tag{8}$$

Afterwards, we use the Exp3.S to update the weights w_i for all tasks. This process is repeated across all time steps until the predefined total number of steps T is reached. The procedure is summarized as Algorithm 1.

4. Experiments

To test the efficacy of our approach, we conducted experiments on a set of datasets which were grouped into 16 task clusters.

4.1. Settings

After performing balanced sampling for each task cluster, the sizes of the training set, development (or validation) set, and test set for each task were consistent, containing approximately 10k, 3k, and 1k samples respectively. In our experiments, we consider two experimental setups. The first setup aims to observe the improvement of the model's performance across all tasks. In this configuration, all tasks are visible to the model. The second setup is designed to examine the model's generalization ability on unseen tasks. Under this scenario, tasks such as Natural Language Inference(NLI), open-domain Question Answering (QA), and summarization are considered as unseen tasks. Consequently, data related to these tasks are not present in the training dataset.

4.2. Baseline Models

We used BART (Lewis et al., 2019) and mT5-large (Xue et al., 2020) as our training model. mT5-large is a transformer-based encoder-decoder language model, pre-trained on a large multilingual corpus, we hence condensed the parameters of mT5-large by mainly condensing the embedding layer to make it more suitable for Chinese context, which greatly saves space and resources. Specifically, we used mT5's tokenizer of 250,000 tokens to tokenize the large Chinese corpus, retaining 30,000+ tokens according to word frequency, then modified the resulting word list to obtain a new sentencepiece model. In addition, we also compared with other benchmarks: i) Single task: training directly on the target task (if applicable); ii) Instruction tuning (IT): after mixing all tasks together, then sampling from the entire dataset. iii) Pre-defined order: pre-defining the syllabus based on the average sentence length in the tasks, and the model is then fine-tuned according to this sequence. iv) One-phase aotumated curriculum learning(IT1ACL): a simplified version of the method we proposed, focusing solely on the progress signal of whole task without proceeding to the second phase of instruction selection within tasks.

4.3. Evaluation Metrics

We use F1 score for Conference Resolution, Keywords Recognition, Text-to-Sql, Extractive QA, Open-Domain QA task, ROUGE for Translation, Summarization, and Text Generation task, and accuracy for other tasks.

4.4. Implementation details

We chose ten instruction templates per dataset to provide a balance between variety and manageability. The instruction template selection is not arbitrary, but based on our preliminary experiments that suggested this as an optimal balance for maximizing learning while preventing overfitting. We set most hyperparameters as suggested by previous works (Wolf et al., 2020) and conducted experiments under fully supervised and low-resource settings. The parameters for the Exp3.S algorithm

Method	Model	ConR	NER	KR	TSql	Sent	Sim	TClas	IClas	Fact	NLI	mcQA	exQA	odQA	Trans	Summ	Gen
Single Task	BART	31.5	34.6	36.1	27.7	30.9	43.8	40.2	39.6	38.1	33.4	31.8	41.2	25.3	31.5	42.7	41.9
	mT5	**31.7**	**41.8**	**42.9**	**35.5**	44.6	**48.9**	**50.6**	45.2	49.3	45.6	**35.5**	39.5	27.9	**41.2**	35.8	32.7
IT(None)	BART	25.7	26.1	21.1	23.7	18.5	41.2	34.5	37.8	30.1	35.4	16.7	40.0	23.3	16.6	39.5	38.6
	mT5	26.2	28.8	37.6	29.2	40.1	45.4	46.5	42.2	47.2	42.0	33.4	34.8	26.3	24.1	32.9	30.8
Rule-based CL(Pre-defined Order)	BART	25.2	29.8	30.3	23.8	24.9	40.7	33.2	37.3	31.5	34.2	20.5	39.8	24.4	16.9	39.2	40.7
	mT5	26.8	30.5	37.9	28.4	40.7	48.5	46.9	43.6	48.8	43.7	34.2	35.9	27.5	26.3	33.9	31.9
IT1ACL	BART	25.9	30.1	32.8	25.2	28.8	42.2	38.1	37.5	35.8	34.9	26.4	41.4	24.8	16.7	42.6	41.2
	mT5	30.2	36.2	39.3	28.8	45.5	48.7	49.2	45.7	49.2	45.5	35.6	36.1	28.3	25.8	34.3	32.5
IT2ACL(ours)	BART	26.6	31.5	32.4	25.7	30.7	42.5	39.5	37.6	36.1	33.4	29.3	**42.9**	25.1	16.8	**43.9**	**42.3**
	mT5	30.4	37.9	39.8	29.1	**45.2**	48.5	49.7	**46.8**	**49.6**	**46.3**	34.1	36.4	**29.1**	25.6	34.7	33.1

Table 1: Overall experimental results of our proposed framework IT2ACL and the compared baselines. The columns denote different tasks: **ConR** stands for Conference Resolution, **NER** for Named Entity Recognition, **KR** for Keywords Recognition, **TSql** for SQL Task, **Sent** for Sentiment Analysis, **Sim** for Similarity, **TClas** for Text Classification, **IClas** for Iintent Classification, **Fact** for Fact Checking, **NLI** for Natural Language Inference, **mcQA** for Multiple Choice Question Answering, **exQA** for Extractive Question Answering, **odQA** for Open-Domain Question Answering, **Trans** for Translation, **Summ** for Summarization, and **Gen** for Text Generation.

Method	Model	NLI	mcQA	Summ
IT(None)	BART	24.8	18.9	32.9
	mT5	29.5	17.9	25.9
Rule-based CL(Pre-defined Order)	BART	22.6	20.8	31.2
	mT5	28.7	19.8	24.6
IT1ACL	BART	25.4	21.3	35.7
	mT5	31.0	22.3	27.1
IT2ACL	BART	26.9	22.8	36.9
	mT5	32.2	23.5	28.3

Table 2: Generalization results of our proposed framework IT2ACL and the compared baselines.

are $\eta = 10^{-3}$, $\beta = 0$, $\gamma = 0.05$. We conduct all experiments on 8 NVIDIA A6000 GPU and select the best model checkpoint according to the performance on the development set.

4.5. Results

We show our validation results corresponding to the two experimental setups.

4.5.1. Multi-task

Table 1 shows the overall experimental outcomes across all tasks for our proposed method and the benchmarked baselines. From the result, it's evident that our IT2ACL approach outperforms both the random order with instruction tuning and the pre-determined order with curriculum learning. For mT5, out of 16 tasks, 5 tasks achieved even better results than training on just a single task. Meanwhile, for BART, 3 tasks surpassed the performance of single-task training. We speculate that multi-task learning in our experiments might aid the model in capturing the universal features and structures shared among various tasks, thereby enhancing the performance on individual tasks.

Furthermore, we can also observe that both models generally underperform in multiple-choice question answering and translation tasks compared to other tasks. Taking a closer look at the translation task, we notice that solely relying on rule-based curriculum learning yield even better results than using our IT1ACL and IT2ACL methods. This might be attributed to the intrinsic characteristics of the translation task where the complexity directly correlates with sentence length. Also, if the model undergoes training with more Chinese data from other tasks, the accuracy of the translated target language might decline. This highlights the vast performance disparity the model has across different tasks, which often correlates with the setup and composition of multi-tasking.

Comparing the results of IT1ACL and IT2ACL, we can discern that using a one-phase curriculum learning algorithm to decide the task learning order, without considering multiple instructions, still gives a performance boost over random order and pre-defined order. However, this improvement is not as substantial as with two-phase learning. This underscores that from easy-to-hard-learning instruction in the second phase is an indispensable aspect of enhancing model performance. By recording and observing the selection frequency of various instructions, this approach also allows us to pinpoint the most optimal instruction for a specific task.

4.5.2. Generalization

In this experimental setup, we designate three task categories (NLI, multiple-choice QA, Summarization) as our unseen tasks. The models are trained

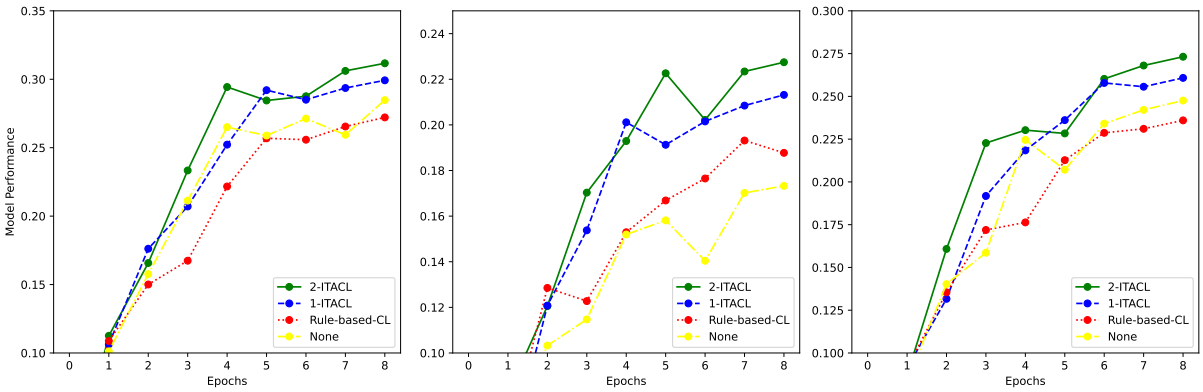

Figure 3: The model performance as a function of training time when the unseen task is NLI (a), multiple-choice QA (b), Summarization (c). We show results with 4 scheduling methods: our method denoted 2-phase curriculum (solid green), our method denoted 1-phase curriculum (dashed yellow), rule-based curriculum learning with pre-defined order (dotted red), and instruction tuning with no curriculum learning (dashdotted blue).

on the training dataset, excluding data related to these three tasks, ensuring a zero-shot setting in our experiments. From the results in Table 2, we observe that curriculum learning further enhances the model's generalization capability. However, the model trained using the rule-based curriculum learning approach exhibited the weakest generalization performance in NLI and Summarization task. This might be attributed to the pre-defined order, which could significantly deviate from the optimal sequence for the model. As a result, the model's continuous learning potential might not have been fully realized (Weinshall et al., 2018).

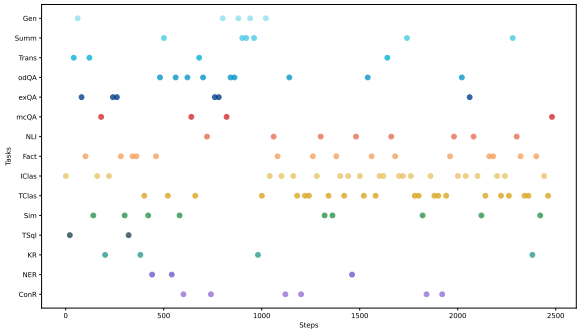

Figure 4: Sampling frequency of different tasks during training process when sentiment analysis is for unseen tasks. Each point shows the task with the highest selection frequency in every 20 steps.

4.6. Analysis

How does the frequency of task and instruction sampling change during the training process? We observe the training process repeatedly, displaying task names and instruction contents at the beginning of each batch to visualize the chosen task and instruction in Figure 4. Taking sentiment analysis as an example, we initially assign higher weight to intent classification task, leading the model to start training with tasks similar to unseen tasks. We find that there is considerable randomness in the early stages of training using the Exp3.S algorithm. However, in later stages, similar classification tasks such as topic classification and fact checking are increasingly chosen for training. They also tend to be repeatedly selected later on. However, since they always appear in the latter phase of the training, this can be resolved by limiting the maximum number of training epoches to prevent overtraining on a few tasks, which could lead to overfitting. In terms of the instruction space, according to our observation during training, there isn't a particular concentration towards a few individual instructions. Sometimes there is a tendency to select longer instructions first, and other times, shorter ones are chosen initially. We believe that by using the Exp3.S algorithm, we have better avoided over-concentration on a limited number of instructions.

How does model's generalization ability change over the course of training? In our experimental setup, we observed that curriculum learning accelerates the learning process at the beginning of training. The generalization performance after convergence is also comparable to the instruction fine-tuning training without the use of any curriculum learning. However, when we employed curriculum learning with a predefined order based on rules, for some tasks, more complex tasks are preferred at the beginning of training. This approach, in fact, reduces the model's generalization capability, with the exception of improvement seen only on the multiple-choice QA task. Task difficulty can be assessed based on the final performance illustrated in each graph in Figure 3. From the charts, it is

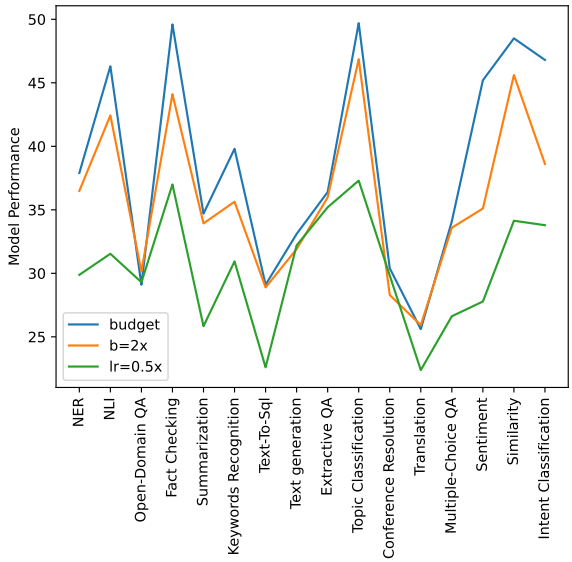

Figure 5: The effect of hyperparameters settings on mT5 model performance with b=2x and lr=0.5x relative to the typical setting.

evident that when the task complexity is higher, the improvement in final accuracy due to curriculum learning is more pronounced.

Can changes in hyperparameters affect the model performance? We defined three settings: (i) base setting, (ii) b=2x, i.e., increasing the batch size by a factor of 2, and (iii) lr=0.5x, i.e., the learning rate is set to be half of the original learning rate. We trained all tasks again in our task mixture, recording the average of the mT5 model performance. Our aim is to evaluate to what extent the hyperparameters settings can have an influence on the model-based task difficulty score during the training process. Our results are summarized in Figure 5. They show that larger learning rates and smaller batch sizes could achieve better performance in our algorithm, and the experimental results presented have proven to be relatively good and the most stable.

Effectiveness of the reward algorithm across different task clusters The Exp3.S algorithm, combined with our adaptive curriculum learning strategy, shows that the model maintains robustness with an increase in the number of task clusters. In our preliminary experiments, we observe that with an increase in clusters, there was a slight improvement in performance. However, as the number of clusters continued to increase, the improvement in performance became marginal, eventually reaching a saturation point where further increase in the number of clusters did not enhance the task performance. This trend suggests a potential relationship with the model size. For example, when the number of clusters increases from 5 to 15, we observe a gradual improvement in accuracy when the unseen task is NLI, but this improvement plateaus beyond 15 clusters.

How do we address the issue of catastrophic forgetting in our method? In the initial stages of the MAB algorithm, each arm has a certain probability of being selected for exploration purposes. As time progresses and more data is gathered, the algorithm tends to favor arms that have previously performed well. However, thanks to the inherent exploration mechanism of Exp3.S algorithm, even arms with extremely low selection probabilities still get chosen occasionally. This balance between exploiting high-reward arms and exploring other arms is crucial for maximizing the overall return. Based on our observations and the detailed results we have gathered, it becomes clear that tasks which yield high rewards are selected consistently and frequently. As a consequence, we have noticed that certain samples emerge repeatedly throughout our experiments. Employing such a strategy has its merits; one of the most significant benefits is its ability to reduce the impact of catastrophic forgetting. Our approach provides a buffer against this phenomenon, ensuring that previously learned tasks remain reinforced. Nonetheless, if the number of tasks to be learned were to increase by several orders of magnitude, our method might require significantly more training time and resources, an issue we leave for future research.

5. Conclusion

In this paper, we have presented a novel instruction tuning framework IT2ACL to dynamically learn easy-to-hard instructions for LLMs guided by 2-phase Curriculum Learning. We propose a loss-driven progress signal to incorporate curriculum learning strategy into instruction tuning, for deciding the sequences to learn instructions within a same task. The weighted average of all instructions within the same task is then used to determine the order in which tasks are trained. The progress signal is also scaled to a reward to be used in the adversarial multi-armed bandits strategy. This facilitates the model's ability to automatically adjust its learning curriculum based on the changing difficulty of tasks. Extensive experiments have been conducted over 16 task clusters, covering a wide array of applications. Experiment results demonstrate that IT2ACL achives higher or competitive performance against baseline models across all tasks, indicating the effectiveness of applying the curriculum learning strategy to instruction tuning for LLMs.

6. Limitations

The limitations of our work are as follows:

1. Many of Chinese datasets are not yet publicly available, which limits the scope of datasets and tasks that can be explored in our study.
2. We fine-tune the task mixture with two types of models (BART and mT5), leaving recent open-source Chinese LLMs to our future work.
3. We may find that the hyperparameter settings and the content and length of the instruction templates have an effect on the loss decrease, for which we nevertheless have only validated the case under one of the settings, leaving other impact factors not being explored.

7. Ethics Considerations

We strictly adhere to the legal requirements in all of our dataset usage. This includes how we cite and how we use the resources. Legal awareness have been kept throughout this project.

8. Acknowledgements

The present research was supported by the Key Research and Development Program of Yunnan Province (Grant No. 202203AA080004). We would like to thank the anonymous reviewers for their insightful comments.

9. Bibliographical References

Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122.

Hengyi Cai, Hongshen Chen, Cheng Zhang, Yonghao Song, Xiaofang Zhao, Yangxi Li, Dongsheng Duan, and Dawei Yin. 2020. Learning from easy to complex: Adaptive multi-curricula learning for neural dialogue generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7472–7479.

Yue Cao, Hao-Ran Wei, Boxing Chen, and Xiaojun Wan. 2021. Continual learning for neural machine translation. 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 3964–3974. Association for Computational Linguistics.

Ernie Chang, Hui-Syuan Yeh, and Vera Demberg. 2021. Does the order of training samples matter? improving neural data-to-text generation with curriculum learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 727–733. Association for Computational Linguistics.

Mingkai Deng, Jianyu Wang, Cheng-Ping Hsieh, Yihan Wang, Han Guo, Tianmin Shu, Meng Song, Eric P Xing, and Zhiting Hu. 2022. Rlprompt: Optimizing discrete text prompts with reinforcement learning. *arXiv preprint arXiv:2205.12548*.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. 2018. Automatic goal generation for reinforcement learning agents. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1514–1523. PMLR.

Marina Fomicheva, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. Unsupervised quality estimation for neural machine translation. *Trans. Assoc. Comput. Linguistics*, 8:539–555.

Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1311–1320. PMLR.

Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Supryadi, Linhao Yu, Yan Liu, Jiaxuan

Li, Bojian Xiong, and Deyi Xiong. 2023. Evaluating large language models: A comprehensive survey. *CoRR*, abs/2310.19736.

Mark Herbster and Manfred Warmuth. 1995. Tracking the best expert. In *Machine Learning Proceedings 1995*, pages 286–294. Elsevier.

Xisen Jin, Dejiao Zhang, Henghui Zhu, Wei Xiao, Shang-Wen Li, Xiaokai Wei, Andrew O. Arnold, and Xiang Ren. 2022. Lifelong pretraining: Continually adapting language models to emerging corpora. 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 4764–4780. Association for Computational Linguistics.

Zixuan Ke, Hu Xu, and Bing Liu. 2021. Adapting BERT for continual learning of a sequence of aspect sentiment classification tasks. 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 4746–4755. Association for Computational Linguistics.

Tom Kocmi and Ondrej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. *arXiv preprint arXiv:1707.09533*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xuebo Liu, Houtim Lai, Derek F Wong, and Lidia S Chao. 2020. Norm-based curriculum learning for neural machine translation. *arXiv preprint arXiv:2006.02014*.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul A. Crook, Bing Liu, Zhou Yu, Eunjoon Cho, Pascale Fung, and Zhiguang Wang. 2021. Continual learning in task-oriented dialogue systems. 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 7452–7467. Association for Computational Linguistics.

Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. 2020. Teacher-student curriculum learning. *IEEE Trans. Neural Networks Learn. Syst.*, 31(9):3732–3740.

Tasnim Mohiuddin, Philipp Koehn, Vishrav Chaudhary, James Cross, Shruti Bhosale, and Shafiq R. Joty. 2022. Data selection curriculum for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 1569–1582. Association for Computational Linguistics.

Pierre-Yves Oudeyer, Frdric Kaplan, and Verena V Hafner. 2007. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2):265–286.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M Mitchell. 2019. Competence-based curriculum learning for neural machine translation. *arXiv preprint arXiv:1903.09848*.

Rémy Portelas, Cédric Colas, Katja Hofmann, and Pierre-Yves Oudeyer. 2019. Teacher algorithms for curriculum learning of deep RL in continuously parameterized environments. In *3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings*, volume 100 of *Proceedings of Machine Learning Research*, pages 835–853. PMLR.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2022. Grips: Gradient-free, edit-based instruction search for prompting large language models. *arXiv preprint arXiv:2203.07281*.

Sébastien Racanière, Andrew K. Lampinen, Adam Santoro, David P. Reichert, Vlad Firoiu, and Timothy P. Lillicrap. 2019. Automated curricula through setter-solver interactions. *CoRR*, abs/1909.12892.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. 2021. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*.

Thomas Scialom, Tuhin Chakrabarty, and Smaranda Muresan. 2022. Fine-tuned language models are continual learners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6107–6122.

Tianhao Shen, Renren Jin, Yufei Huang, Chuang Liu, Weilong Dong, Zishan Guo, Xinwei Wu, Yan Liu, and Deyi Xiong. 2023. Large language model alignment: A survey. *CoRR*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv e-prints*, pages arXiv–2201.

Yixuan Su, Deng Cai, Qingyu Zhou, Zibo Lin, Simon Baker, Yunbo Cao, Shuming Shi, Nigel Collier, and Yan Wang. 2020. Dialogue response selection with hierarchical curriculum learning. *arXiv preprint arXiv:2012.14756*.

Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. 2018. Intrinsic motivation and automatic curricula via asymmetric self-play. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. LAMOL: language modeling for lifelong language learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Neeraj Varshney, Swaroop Mishra, and Chitta Baral. 2022. Let the model decide its curriculum for multitask learning. *CoRR*, abs/2205.09898.

Yu Wan, Baosong Yang, Derek F. Wong, Yikai Zhou, Lidia S. Chao, Haibo Zhang, and Boxing Chen. 2020. Self-paced learning for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1074–1080. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Daphna Weinshall, Gad Cohen, and Dan Amir. 2018. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5235–5243. PMLR.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv preprint arXiv:2010.11934*.

Sicheng Yang and Dandan Song. 2022. Fpc: Fine-tuning with prompt curriculum for relation extraction. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing*, pages 1065–1077.

Wenpeng Yin, Jia Li, and Caiming Xiong. 2022. Contintin: Continual learning from task instructions. 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 3062–3072. Association for Computational Linguistics.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. 2019. Curriculum learning for domain adaptation in neural machine translation. 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 1903–1915.

Yikai Zhou, Baosong Yang, Derek F. Wong, Yu Wan, and Lidia S. Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6934–6944. Association for Computational Linguistics.

10. Language Resource References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856*.

Jiahao Bu, Lei Ren, Shuang Zheng, Yang Yang, Jingang Wang, Fuzheng Zhang, and Wei Wu. 2021. Asap: A chinese review dataset towards

aspect category sentiment analysis and rating prediction. *arXiv preprint arXiv:2103.06605*.

Deng Cai, Yan Wang, Wei Bi, Zhaopeng Tu, Xiaojiang Liu, and Shuming Shi. 2019. Retrieval-guided dialogue response generation via a matching-to-generation framework. 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 1866–1875.

Jiangjie Chen, Rui Xu, Ziquan Fu, Wei Shi, Zhongqiao Li, Xinbo Zhang, Changzhi Sun, Lei Li, Yanghua Xiao, and Hao Zhou. 2022a. E-kar: A benchmark for rationalizing natural language analogical reasoning. *arXiv preprint arXiv:2203.08480*.

Chen, Jing and Chen, Qingcai and Liu, Xin and Yang, Haijun and Lu, Daohe and Tang, Buzhou. 2018. *The bq corpus: A large-scale domain-specific chinese corpus for sentence semantic equivalence identification*.

Yirong Chen, Weiquan Fan, Xiaofen Xing, Jianxin Pang, Minlie Huang, Wenjing Han, Qianfeng Tie, and Xiangmin Xu. 2022b. Cped: A large-scale chinese personalized and emotional dialogue dataset for conversational ai. *arXiv preprint arXiv:2205.14727*.

LI CO. 2019. Iflytek: a multiple categories chinese text classifier. competition official website.

Alexis Conneau, Guillaume Lample, Ruty Rinott, Adina Williams, Samuel R Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Yiming Cui, Ting Liu, Wanxiang Che, Li Xiao, Zhipeng Chen, Wentao Ma, Shijin Wang, and Guoping Hu. 2018. A span-extraction dataset for chinese machine reading comprehension. *arXiv preprint arXiv:1810.07366*.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, et al. 2017. Dureader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073*.

Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. Lcsts: A large scale chinese short text summarization dataset. *arXiv preprint arXiv:1506.05865*.

Hai Hu, Kyle Richardson, Liang Xu, Lu Li, Sandra Kübler, and Lawrence S Moss. 2020. Ocnli: Original chinese natural language inference. *arXiv preprint arXiv:2010.05444*.

Xuming Hu, Zhijiang Guo, Guanyu Wu, Aiwei Liu, Lijie Wen, and Philip S Yu. 2022. Chef: A pilot chinese dataset for evidence-based fact-checking. *arXiv preprint arXiv:2206.11863*.

Yanzeng Li, Tingwen Liu, Diying Li, Quangang Li, Jinqiao Shi, and Yanqiu Wang. 2018. Character-based bilstm-crf incorporating pos and dictionaries for chinese opinion target extraction. In *Asian Conference on Machine Learning*, pages 518–533. PMLR.

Bang Liu, Di Niu, Haojie Wei, Jinghong Lin, Yancheng He, Kunfeng Lai, and Yu Xu. 2018a. Matching article pairs with graphical decomposition and convolutions. *arXiv preprint arXiv:1802.07459*.

Liu, Xin and Chen, Qingcai and Deng, Chong and Zeng, Huajun and Chen, Jing and Li, Dongfang and Tang, Buzhou. 2018b. *Lcqmc: A large-scale chinese question matching corpus*.

Hua Lu, Siqi Bao, Huang He, Fan Wang, Hua Wu, and Haifeng Wang. 2022. Towards boosting the open-domain chatbot with human feedback. *arXiv preprint arXiv:2208.14165*.

Maria Pontiki, Dimitrios Galanis, Haris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, Mohammad Al-Smadi, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphée De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *International workshop on semantic evaluation*, pages 19–30.

Sanh, Victor and Webson, Albert and Raffel, Colin and Bach, Stephen H and Sutawika, Lintang and Alyafeai, Zaid and Chaffin, Antoine and Stiegler, Arnaud and Scao, Teven Le and Raja, Arun and others. 2021. *Multitask prompted training enables zero-shot task generalization*.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. *arXiv preprint arXiv:1503.02364*.

Chih Chieh Shao, Trois Liu, Yuting Lai, Yiying Tseng, and Sam Tsai. 2018. Drcd: a chinese machine reading comprehension dataset. *arXiv preprint arXiv:1806.00920*.

Zhihong Shao, Minlie Huang, Jiangtao Wen, Wenfei Xu, and Xiaoyan Zhu. 2019. Long and diverse text generation with planning-based hierarchical variational model. *arXiv preprint arXiv:1908.06605*.

Hongxuan Tang, Hongyu Li, Jing Liu, Yu Hong, Hua Wu, and Haifeng Wang. 2020. Dureader_robust: a chinese dataset towards evaluating robustness and generalization of machine reading comprehension in real-world applications. *arXiv preprint arXiv:2004.11142*.

Yida Wang, Pei Ke, Yinhe Zheng, Kaili Huang, Yong Jiang, Xiaoyan Zhu, and Minlie Huang. 2020. A large-scale chinese short-text conversation dataset. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 91–103. Springer.

Wei, J. and Bosma, M. and Zhao, V. Y. and Guu, K. and Yu, A. W. and Lester, B. and Du, N. and Dai, A. M. and Le, Q. V. 2021. *Finetuned Language Models Are Zero-Shot Learners*.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2016. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. *arXiv preprint arXiv:1612.01627*.

Liang Xu, Hai Hu, Xuanwei Zhang, Lu Li, Chenjie Cao, Yudong Li, Yechen Xu, Kai Sun, Dian Yu, Cong Yu, et al. 2020. Clue: A chinese language understanding evaluation benchmark. *arXiv preprint arXiv:2004.05986*.

Ningyu Zhang, Mosha Chen, Zhen Bi, Xiaozhuan Liang, Lei Li, Xin Shang, Kangping Yin, Chuanqi Tan, Jian Xu, Fei Huang, et al. 2021. Cblue: A chinese biomedical language understanding evaluation benchmark. *arXiv preprint arXiv:2106.08087*.

Xiang Zhang and Yann LeCun. 2017. Which encoding is the best for text classification in chinese, english, japanese and korean? *arXiv preprint arXiv:1708.02657*.

Yongfeng Zhang, Haochen Zhang, Min Zhang, Yiqun Liu, and Shaoping Ma. 2014. Do users rate or review? boost phrase-level sentiment labeling with review-level sentiment classification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 1027–1030.

Yongfeng Zhang, Min Zhang, Yi Zhang, Guokun Lai, Yiqun Liu, Honghui Zhang, and Shaoping Ma. 2015. Daily-aware personalized recommendation based on feature-level time series analysis. In *Proceedings of the 24th international conference on world wide web*, pages 1373–1383.

Haoxi Zhong, Chaojun Xiao, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. 2020. Jec-qa: a legal-domain question answering dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9701–9708.

Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, and Xiaoyan Zhu. 2020. Kdconv: A chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation. *arXiv preprint arXiv:2004.04100*.

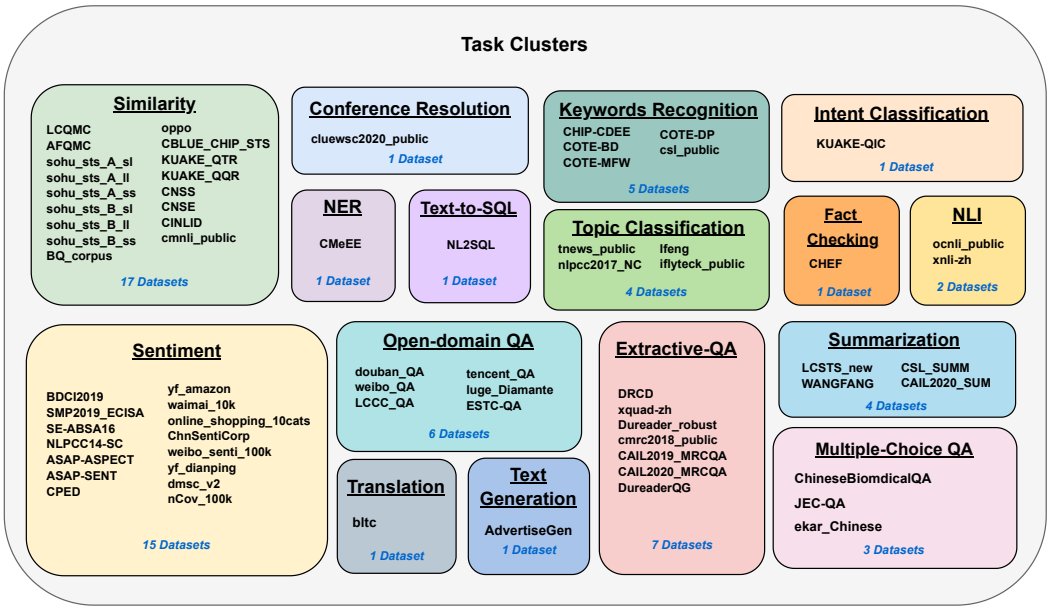

Figure 6: Our all datasets and task clusters.

Appendix A. Data Card

All the datasets used in this paper, along with their corresponding task clusters, are presented in Figure 6.

We present the data card in Table 3, where we provide specific pointers and references. All datasets are publicly available.

Table 3: The list of all datasets used in our instruction tuning.

Datasets	Task	Reference
LCQMC	Similarity	Liu et al. (2018b)
AFQMC	Similarity	Xu et al. (2020)
sohu_sts_A_sl	Similarity	2021 Sohu Campus Text Matching Algorithm Competition
sohu_sts_A_ll	Similarity	2021 Sohu Campus Text Matching Algorithm Competition
sohu_sts_A_ss	Similarity	2021 Sohu Campus Text Matching Algorithm Competition
sohu_sts_B_sl	Similarity	2021 Sohu Campus Text Matching Algorithm Competition
sohu_sts_B_ll	Similarity	2021 Sohu Campus Text Matching Algorithm Competition
sohu_sts_B_ss	Similarity	2021 Sohu Campus Text Matching Algorithm Competition
BQ_corpus	Similarity	Chen et al. (2018)
oppo	Similarity	LUGE(https://www.luge.ai/#/luge/dataDetail?id=28)
CHIP-STS	Similarity	Zhang et al. (2021)
KUAKE-QTR	Similarity	(Zhang et al., 2021)
KUAKE-QQR	Similarity	Zhang et al. (2021)
CNSS	Similarity	Liu et al. (2018a)
CNSE	Similarity	Liu et al. (2018a)
CINLID	Similarity	LUGE(https://www.luge.ai/#/luge/dataDetail?id=39))
CMNLI	Similarity	Xu et al. (2020)
CLUEWSC2020	Conference Resolution	Xu et al. (2020)
CMeEE	Named Entity Rocognition	Zhang et al. (2021)
CHIP-CDEE	Keyword Recognition	Zhang et al. (2021)
COTE-BD	Keyword Recognition	Li et al. (2018)
COTE-MFW	Keyword Recognition	Li et al. (2018)
COTE-DP	Keyword Recognition	Li et al. (2018)
CSL	Keyword Recognition	Xu et al. (2020)
NL2SQL	Text-to-Sql	LUGE(https://www.luge.ai/#/luge/dataDetail?id=12)
yf_amazon	Sentiment	Zhang et al. (2015)
waimai_10k	Sentiment	-
online_shopping_10cats	Sentiment	
ChnSentiCorp	Sentiment	-
weibo_senti_100k	Sentiment	-
yf_dianping	Sentiment	Zhang et al. (2014)
DMSC_v2	Sentiment	Kaggle(https://www.kaggle.com/datasets)
nCov_100k	Sentiment	CCIR2020(https://www.datafountain.cn/competitions/423)
BDCI2019	Sentiment	CCF-BDCI2019
SMP2019_ECISA	Sentiment	SMP-ECISA 2019
SE-ABSA16	Sentiment	Pontiki et al. (2016)
NLPCC14-SC	Sentiment	LUGE(https://www.luge.ai/#/luge/dataDetail?id=20)
ASAP-ASPECT	Sentiment	Bu et al. (2021)
ASAP-SENT	Sentiment	Bu et al. (2021)
CPED	Sentiment	Chen et al. (2022b)
TNEWS	Topic Classification	Xu et al. (2020)
NLPCC2017_NC	Topic Classification	NLPCC2017(http://tcci.ccf.org.cn/conference/2017)
Ifeng	Topic Classification	Zhang and LeCun (2017)
IFLYTEK	Topic Classification	CO (2019)
KUAKE-QIC	Intent Classification	Zhang et al. (2021)
CHEF	Fact Checking	Hu et al. (2022)
OCNLI	Natural Language Inference	Hu et al. (2020)
XNLI-zh	Natural Language Inference	Conneau et al. (2018)
Ekar_Chinese	Multiple-Choice QA	Chen et al. (2022a)
JEC-QA	Multiple-Choice QA	Zhong et al. (2020)
ChineseBiomdicalQA	Multiple-Choice QA	LUGE(https://www.luge.ai/#/luge/dataDetail?id=40)
DRCD	Extractive-QA	Shao et al. (2018)
XQUAD-zh	Extractive-QA	Artetxe et al. (2019)
Dureader_robust	Extractive-QA	Tang et al. (2020)
Dureader_checklist	Extractive-QA	He et al. (2017)
CMRC2018	Extractive-QA	Cui et al. (2018)
CAIL2019_MRCQA	Extractive-QA	CAIL2019
CAIL2020_MRCQA	Extractive-QA	CAIL2020
DureaderQG	Extractive-QA	Tang et al. (2020)
Douban_QA	Open-Domain QA	Wu et al. (2016)

Continued on next page

– continued from previous page

Datasets	Task	Reference
Weibo_QA	Open-Domain QA	Shang et al. (2015)
LCCC_QA	Open-Domain QA	Wang et al. (2020)
Tencent_QA	Open-Domain QA	Cai et al. (2019)
Luge_Diamante	Open-Domain QA	Lu et al. (2022)
ESTC-QA	Open-Domain QA	Zhou et al. (2020)
BLTC	Translation	
LCSTS_new	Summarization	Hu et al. (2015)
WANGFANG	Summarization	WANGFANG Database
CSL_SUMM	Summarization	CLGE(https://github.com/fighting41love/CLGE)
CAIL2020_SUM	Summarization	CAIL2020
AdvertiseGen	Text Generation	Shao et al. (2019)