EasyDistill: A Comprehensive Toolkit for Effective Knowledge Distillation of Large Language Models

Chengyu Wang[1]*, **Junbing Yan**[1], **Wenrui Cai**[1,2], **Yuanhao Yue**[1], **Jun Huang**[1]
[1] Alibaba Cloud Computing [2] Shanghai Jiao Tong University
chengyu.wcy@alibaba-inc.com

Abstract

In this paper, we present *EasyDistill*, a comprehensive toolkit designed for effective black-box and white-box knowledge distillation (KD) of large language models (LLMs). Our framework offers versatile functionalities, including data synthesis, supervised fine-tuning, ranking optimization, and reinforcement learning techniques specifically tailored for KD scenarios. The toolkit accommodates KD functionalities for both System 1 (fast, intuitive) and System 2 (slow, analytical) models. With its modular design and user-friendly interface, *EasyDistill* empowers researchers and industry practitioners to seamlessly experiment with and implement state-of-the-art KD strategies for LLMs. In addition, *EasyDistill* provides a series of robust distilled models and KD-based industrial solutions developed by us, along with the corresponding open-sourced datasets, catering to a variety of use cases. Furthermore, we describe the seamless integration of *EasyDistill* into Alibaba Cloud's Platform for AI (PAI). Overall, the *EasyDistill* toolkit makes advanced KD techniques for LLMs more accessible and impactful within the NLP community. The toolkit, together with source codes, all model checkpoints and datasets, is released at: https://github.com/modelscope/easydistill.

1 Introduction

The proliferation of large language models (LLMs) has been transformative for NLP (Zhao et al., 2023; Yadagiri and Pakray, 2025), pushing the boundaries of what machines can understand and generate in human language. However, the extensive size and complexity of these models present significant challenges, including high computational costs and substantial energy consumption. Knowledge distillation (KD) offers a viable solution to this dilemma, where smaller models are trained to replicate the performance of their larger counterparts, enabling efficient use of resources without sacrificing much accuracy (Xu et al., 2024; Yang et al., 2024c). Despite its potential, effective KD of LLMs is not straightforward, often requiring advanced algorithms and domain expertise. In addition, the lack of tools for LLM-based KD can exacerbate these challenges, limiting exploration and adaptation in industrial settings.[1]

In this paper, we introduce *EasyDistill*, a comprehensive toolkit designed to simplify the KD process for LLMs under both black-box and white-box settings, utilizing proprietary and open-source LLMs as teacher models. *EasyDistill* offers a wide array of functionalities, including data synthesis and augmentation, supervised fine-tuning (SFT), ranking optimization, and reinforcement learning (RL), all tailored for KD scenarios. By supporting both System 1 (fast, intuitive) and System 2 (slow, analytical) models (Li et al., 2025), *EasyDistill* facilitates the KD process across various types of LLMs. *EasyDistill* is easy to use and extend; it provides a modular design with a simple command-line interface for invoking these algorithms.

In addition, *EasyDistill* is more than just an open-source toolkit; it integrates several techniques to support KD for industrial practitioners. The contributions are threefold: i) It includes a series of robust distilled models (e.g., *DistilQwen*), along with open-source datasets, to demonstrate the effectiveness of KD. ii) It features several KD-based industrial solutions (i.e., *EasyDistill-Recipes*) that serve as practical guides for diverse application needs. iii) *EasyDistill* is integrated into Alibaba Cloud's Platform for AI (PAI), showcasing its adaptability and potential for large-scale deployment. By bridg-

[*] Corresponding author.

[1] Note that there are a few open-source toolkits that support KD for LLMs, such as DistillKit (https://github.com/arcee-ai/DistillKit). To the best of our knowledge, there is a lack of support for various types of KD algorithms and practical solutions (as described below) within the open-source community.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 787–795
November 4-9, 2025 ©2025 Association for Computational Linguistics

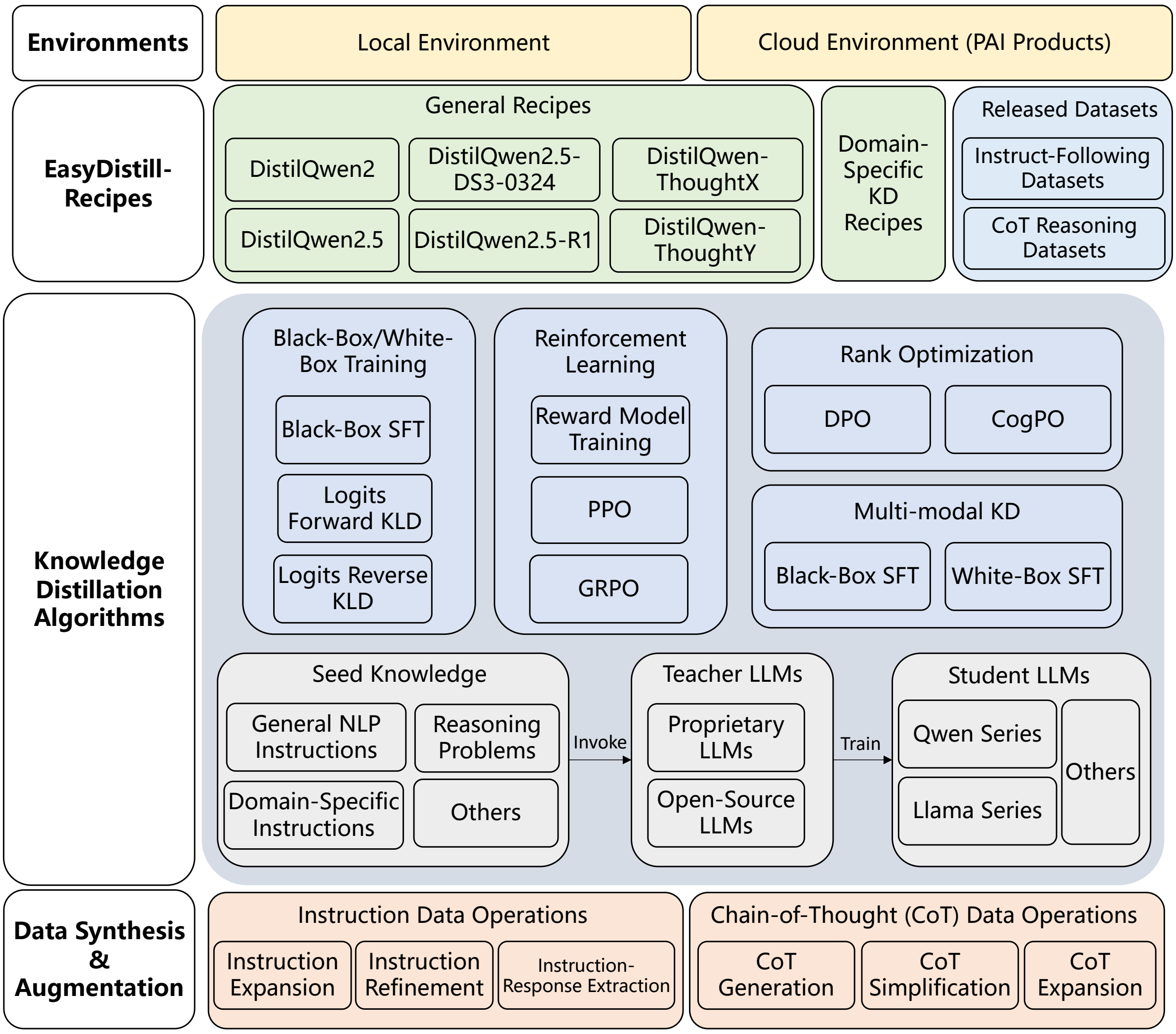

Figure 1: The overall architecture of the *EasyDistill* toolkit.

ing the gap between cutting-edge KD techniques and practical applicability, *EasyDistill* enhances the accessibility within the NLP community.

The remainder of this paper is organized as follows: Section 2 details the architecture and functionalities of *EasyDistill*. Section 3 showcases several practical solutions (i.e., *EasyDistill-Recipes*). Finally, Section 4 concludes the paper and discuss the future work.

2 Architecture and Functionalities

In this section, we formally introduce the *EasyDistill* toolkit. The overall architecture is shown in Figure 1. We begin by presenting the basic KD functionalities, alongside the command-line tool for invoking these functionalities. Following that, we describe the collection of practical solutions (i.e., *EasyDistill-Recipes*). Finally, we briefly describe the integration of the toolkit into PAI products for users to perform KD on the cloud.

2.1 Basic KD Functionalities

2.1.1 Data Synthesis & Augmentation

Synthetic data plays a pivotal role in developing robust LLMs (Liu et al., 2024), especially given the typically limited size of seed datasets for KD. This enhances the KD process, ensuring that the distilled student models not only replicate the output behavior of teacher LLMs but also extend their generalization abilities to previously unseen tasks. In *EasyDistill*, we offer various data synthesis and augmentation operators, utilizing both proprietary and open-source teacher LLMs. These operators are designed to create high-quality seed datasets for KD, enriching them not only in volume but also in diversity across tasks, topics or domains.

The first group of operators supported by *EasyDistill* focuses on the enhancement of instructional data corresponding to a variety of NLP tasks, which form the core inputs (i.e., seed knowledge) to every KD algorithm for LLMs. In this context, we extend our previous work (Yue et al., 2024a) to engineer several functionalities, including instruction expansion, instruction refinement, and the automatic generation of instruction-response pairs from the knowledge expressed in raw texts.

The second group of our operators in *EasyDistill* concentrates on Chain-of-Thoughts (CoTs) (Wei et al., 2022), which are particularly important for distilling the problem-solving capacities of large reasoning models (LRMs) (Li et al., 2025), also known as System 2 models. In addition to the basic operator for producing CoTs grounded in instructions or instruction-response pairs, we further integrate operators for simplifying and extending CoTs to effectively address reasoning problems, as overly long or short CoTs may not be suitable for developing strong LRMs (Yang et al., 2025) We further suggest that combining these two types of operators enhances the KD process for LRMs, as they enable the creation of enriched training sets accompanied by high-quality CoTs.

2.1.2 Training Algorithms for KD Scenarios

The core KD pipeline for LLMs is straightforward. The input is seed knowledge, consisting of instructions for any target tasks, which is leveraged to prompt the selected teacher LLM to generate detailed outputs. In our framework, we support both proprietary and open-source LLMs as teacher models, and open-source LLMs as student models. In the following, we elaborate different types of algorithms tailored to the KD scenarios.

Black-Box/White-Box Training. Since we can only obtain output tokens from proprietary LLMs, the direct KD approach involves supervised fine-tuning (SFT), treating these output tokens as the ground truth for student LLMs.

For open-source teacher LLMs, in addition to SFT, leveraging the models' hidden knowledge as guidance often leads to improved KD performance. In this approach, we obtain the token-level logits from the teacher model and minimize the divergence between the logits distributions of the teacher and student models. The loss functions employed in *EasyDistill* include Kullback–Leibler divergence (KLD) (Gu et al., 2024), reverse KLD (Wu et al., 2025), among others. In the implementation, the forward pass of the teacher model is performed prior to the training of the student model to optimize GPU memory consumption. Furthermore, based on our previous findings (Wang et al., 2025), the sum of the probabilities of the top-10 tokens is almost equal to 1. Thus, *EasyDistill* offers users options to leverage only the top-k token logits from the teacher model and match the corresponding logits from the student model. Subsequently, the computation of loss functions is approximated by considering only k selected logits. This approach not only reduces computation time but also enhances the speed of storing and reading the logits. We do not recommend minimizing the gap between hidden representations, such as attention matrices, of teacher and student LLMs due to the excessive computational requirements.

Reinforcement Learning (RL). A basic principle of KD is to make the student model mimic the behavior of the teacher models. However, this approach may cause the student model to "overfit" the teacher outputs, rather than exploring more possibilities to enhance its generalization abilities. RL-based approaches, on the other hand, leverage feedback from the teacher to train student models.

The first type of RL-based KD functionalities in *EasyDistill* involves training reward models using feedback from teacher models, similar to the Reinforcement Learning from AI Feedback (RLAIF) framework (Lee et al., 2024). Specifically, we employ teacher models to generate synthetic "chosen" and "rejected" responses as preference data, which are then used to train the reward model based on any targeted LLM backbone with scalar outputs of the predicted reward values.

The second type involves supporting RL optimization to obtain the policy model, i.e., the RL-optimized student model. *EasyDistill* integrates popular RL algorithms for training LLMs, particularly Proximal Policy Optimization (PPO) (Schulman et al., 2017) for System 1 models, and Group Relative Policy Optimization (GRPO) (Shao et al., 2024) for System 2 models. Unlike general RL toolkits, *EasyDistill* emphasizes the entire pipeline of distilling knowledge from teacher models to develop more robust student models, as demonstrated in previous works (Bai et al., 2022; Trung et al., 2024; Yang et al., 2024d).

Preference Rank Optimization. A potential drawback of RL-based algorithms is the instability in training. Preference rank optimization-based approaches directly incorporate preferences into LLMs, making the training process more stable. In *EasyDistill*, we integrate the direct preference optimization (DPO) method (Rafailov et al., 2023) based on the training pipeline from Tunstall et al. (2023) for KD. For System 2 models, which possess strong reasoning capabilities, it is important that distilled smaller models have different capacities and cognitive trajectories than their larger counterparts. To address this, *EasyDistill* integrates our cognitive preference optimization (CogPO) algorithm (Cai et al., 2025b) to enhance the reasoning abilities of smaller models by aligning their cognitive processes with their inherent capacities.

Multi-modal KD. In addition, *EasyDistill* enables the distillation of knowledge not only from text-based sources but also incorporating visual and other data modalities, using multi-modal language models as teacher and student models. This enhances the toolkit's versatility and effectiveness in various application scenarios, allowing users to exploit cross-modal relationships to refine model understanding and predictions.

2.2 Command-Line Interface

To facilitate the KD process using our framework, we provide a user-friendly command-line tool that supports running KD jobs with a simple JSON configuration file, specifying the input, output, and all necessary arguments. For example, typical SFT training jobs for black-box KD can be configured as shown in Code 1 and Code 2, utilizing different sources of teacher models. For white-box KD, users can provide additional hyper-parameters and specify the path to store the teacher logits (as shown in Code 3). Once the JSON configuration is set, the KD process can be invoked simply by one line of command, shown as follows:

```
easydistill -config=kd.json
```

with the entire pipeline running automatically.

In the provided sample codes, the inference section contains essential information, particularly the URL of the model service and its API key, for making inferences with the teacher model. In the online mode, any APIs compatible with the OpenAI API format can be utilized. Therefore, *EasyDistill* is compatible with any teacher models in this case. For offline batch inference, we support vLLM for accelerated model inference (Kwon et al., 2023) when the model can be downloaded to local storage. The training configuration includes critical hyper-parameters for the training phase. *EasyDistill* supports all DeepSpeed acceleration techniques by default (Rasley et al., 2020), such as ZeRO and CPU offloading, which can be customized for advanced uses. In the future, other distributed learning frameworks will be supported in *EasyDistill* as well.

```
{
  "job_type": "black_box_kd_api",
  "dataset": {
    "instruction_path": "train.json",
    "labeled_path": "train_labeled.json",
    "template" : "chat_template.jinja",
    "seed": 42
  },
  "inference":{
    "base_url": "ENDPOINT",
    "api_key": "TOKEN",
    "stream": "true",
    "system_prompt" : "You are a helpful
     assistant.",
    "max_new_tokens": 512
  },
  "models": {
    "student": "student/Qwen/Qwen2.5-0.5B-Instruct/"
  },
  "training": {
    "output_dir": "result/",
    "num_train_epochs": 3,
    "per_device_train_batch_size": 1,
    "gradient_accumulation_steps": 8,
    "save_steps": 1000,
    "logging_steps": 1,
    "learning_rate": 2e-5,
    "weight_decay": 0.05,
    "warmup_ratio": 0.1,
    "lr_scheduler_type": "cosine"
  }
}
```

Code 1: Sample JSON configuration for black-box KD (online inference with a proprietary or open-source teacher model where the teacher model can be accessed by any inference API in the OpenAI format and does not need to be specified in the configuration).

```
{
  "job_type": "black_box_kd_local",
  "dataset": {
    ...
  }
  "inference":{
    "enable_chunked_prefill": true,
    "seed": 777,
    "gpu_memory_utilization": 0.9,
    "temperature": 0.8,
    "trust_remote_code": true,
    "enforce_eager": false,
    "max_model_len": 4096,
    "max_new_tokens": 512
  },
  "models": {
    "teacher": "teacher/Qwen/Qwen2.5-32B-Instruct/",
    "student": "student/Qwen/Qwen2.5-0.5B-Instruct/"
  },
  "training": {
    ...
  }
}
```

Code 2: Sample JSON configuration for black-box KD (offline inference with an open-source teacher model).

```
{
  "job_type": "white_box_kd_local",
  "dataset": {
    "logits_path": "logits.json",
    ...
  }
  "inference":{
    "enable_chunked_prefill": true,
    "seed": 777,
    "gpu_memory_utilization": 0.9,
    "temperature": 0.8,
    "trust_remote_code": true,
    "enforce_eager": false,
    "max_model_len": 4096,
    "max_new_tokens": 512
  },
  "distillation": {
    "kd_ratio": 0.5,
    "max_seq_length": 512,
    "distillation_type": "forward_kld"
  },
  "models": {
    "teacher": "teacher/Qwen/Qwen2.5-7B-Instruct/",
    "student": "student/Qwen/Qwen2.5-0.5B-Instruct/"
  },
  "training": {
    ...
  }
}
```

Code 3: Sample JSON configuration for white-box KD.

2.3 *EasyDistill-Recipes*: Practical Solutions

In this section, we further introduce *EasyDistill-Recipes*, a collection of KD-based solutions that produce lightweight LLMs built on *EasyDistill*. Specifically, all the produced models (i.e., the *DistilQwen* series) are also released to public.

2.3.1 General KD Recipes: *DistilQwen*

In general KD recipes, we offer detailed solutions for producing the *DistilQwen* series using *EasyDistill*. This series includes both System 1 and System 2 models, which are lightweight LLMs built upon the Qwen series. We make these solutions available to enable users to create their own models utilizing the KD techniques in our framework. A brief summary of these models are shown in Table 1. Detailed descriptions and the performance of these models can be found in their respective Hugging Face model cards.

The first collection is *DistilQwen2*, an enhanced version of the Qwen2 models (Yang et al., 2024a), equipped with improved instruction-following capabilities. During the distillation training of *DistilQwen2*, we employ GPT-4 and Qwen-max as teacher models to generate high-quality responses. Specifically, before conducting black-box SFT training, we utilize the method described in (Yue et al., 2024b) to balance the task distributions of input instructions. Following SFT, a rank optimization process is performed using the DPO algorithm (Rafailov et al., 2023) to enhance alignment between the student models and the teacher models. In response to the release of the Qwen2.5 model series (Yang et al., 2024b), *DistilQwen2.5* models are trained using a combination of black-box and white-box KD algorithms. For further details, readers may refer to the report (Wang et al., 2025).

With the release of large System 2 models such as DeepSeek-R1 (DeepSeek-AI, 2025), the concept of “LLM with slow thinking” has become a standard strategy to extend the intelligent boundaries of LLMs. We introduce the *DistilQwen2.5-R1* model series, which utilizes DeepSeek-R1 as the teacher model, based on fine-tuning over a collection of DeepSeek-R1’s CoT distillation data. To align the reasoning abilities of smaller distilled models with their intrinsic cognitive capacities, the models are further refined using our CogPO algorithm (Cai et al., 2025b). Additionally, we transfer the fast-thinking, non-reasoning capabilities from

Model Series	Model Type	Parameter Sizes	Teacher LLMs	Student LLMs
DistilQwen2	System 1	1.5B, 7B	GPT-4, Qwen-max	Qwen2
DistilQwen2.5	System 1	0.5B, 1.5B, 3B, 7B	GPT-4, Qwen-max, Qwen2.5-72B-Instruct	Qwen2.5
DistilQwen2.5-DS3-0324	System 1	7B, 14B, 32B	DeepSeek-R1, DeepSeek-V3-0234	Qwen2.5
DistilQwen2.5-R1	System 2	7B, 14B, 32B	DeepSeek-R1	Qwen2.5
DistilQwen-ThoughtX	System 2	7B, 32B	DeepSeek-R1, QwQ-32B	Qwen2.5
DistilQwen-ThoughtY	System 2	4B, 8B, 32B	DeepSeek-R1, DeepSeek-R1-0528, QwQ-32B	Qwen3

Table 1: A summary of the *DistilQwen* model series.

DeepSeek-V3-0324[2] to the *DistilQwen2.5-DS3-0324* models. Here, we first reduce the number of tokens in the training data for *DistilQwen2.5-R1*. Combined with DeepSeek-V3-0324's CoT distillation data, we develop the *DistilQwen2.5-DS3-0324* model series.

The most recent *DistilQwen* series includes *DistilQwen-ThoughtX* and *DistilQwen-ThoughtY*, which exhibit improved reasoning abilities and generate CoTs with more optimal lengths compared to their predecessors. The *DistilQwen-ThoughtX* model series is developed from the innovative OmniThought dataset by utilizing the novel Reasoning Verbosity (RV) and Cognitive Difficulty (CD) scores introduced in OmniThought (Cai et al., 2025a). These scores ensure that models receive rich, high-quality training data reflecting optimal CoT output length and difficulty. *DistilQwen-ThoughtY* is an improved version of *DistilQwen-ThoughtX*, leveraging high-quality CoT distillation data from DeepSeek-R1-0528[3]. Overall, *DistilQwen-ThoughtX* and *DistilQwen-ThoughtY* represent new distilled reasoning models with "adaptive thinking" paradigms, which adaptively solve complicated reasoning problems based on their own knowledge.

2.3.2 Domain-Specific KD Recipes

Within the *EasyDistill-Recipes* module, we further integrate domain-specific recipes for real-world applications. Taking code generation as an example, it generates executable code snippets, assisting developers in writing functional blocks, refining logic, and adapting boilerplate code based on prompts. This capability significantly accelerates software development. In the context of code generation tasks, the primary evaluation metric is *Pass@1*, which measures the model's ability to produce correct, runnable code in a single attempt. A key challenge lies in balancing model capability and inference efficiency: while larger models may achieve higher *Pass@1* scores, they often incur higher computational costs, impacting deployment scalability. Thus, the core optimization goal is to maximize generation accuracy while maintaining a lightweight architecture. To verify the efficacy of *EasyDistill*, we distill two models using prompts and outputs distilled from DeepSeek-R1 based on the OpenCodeReasoning dataset[4]. The detailed performance of the models is presented in Table 2, demonstrating their effectiveness in improving performance in specific tasks.

Model	LiveCodeBench V2	Speedup
Qwen2.5-3B-Instruct	11.35	2.3x
Qwen2.5-3B-Code	**16.62**	2.3x
Qwen2.5-7B-Instruct	30.72	-
Qwen2.5-7B-Code	**35.32**	-

Table 2: Performance comparison of code generation models on LiveCodeBench V2 and inference speedup.

2.3.3 Released Datasets

To assist the community developers in improving instruction-following and CoT reasoning capabilities of LLMs, we have open-sourced two datasets: *DistilQwen_100K* and *DistilQwen_1M*, which are part of the distilled training sets of the *DistilQwen* model series. These datasets cover a range of contents, including mathematics, code, knowledge-based QA, instruction following, and creative generation, with a total dataset size of 100K and 1M entries. For CoT reasoning, we have released *OmniThought*, which is a large-scale dataset featuring 2M CoT processes generated and validated by DeepSeek-R1 and QwQ-32B. Each CoT process is annotated with novel Reasoning Verbosity (RV) and Cognitive Difficulty (CD) scores, which de-

[2] https://huggingface.co/deepseek-ai/DeepSeek-V3-0324

[3] https://huggingface.co/deepseek-ai/DeepSeek-R1-0528

[4] https://huggingface.co/datasets/nvidia/OpenCodeReasoning

Dataset	Size	Task Type	URL
DistilQwen_100K	100K	IF	[URL]
DistilQwen_1M	1M	IF	[URL]
OmniThought	2M	CoT reasoning	[URL]
OmniThought-0528	365K	CoT reasoning	[URL]

Table 3: The summarization of our released datasets. IF refers to "instruction following".

scribe the appropriateness of CoT verbosity and cognitive difficulty level for models to comprehend these reasoning processes. For details, please refer to Cai et al. (2025a). In addition, *OmniThought-0528* is a supplement of *OmniThought* that specifically fauces on the distillation of DeepSeek-R1-0528, which also have rich annotation data regarding the characteristics of CoTs. The information of our released datasets is shown in Table 3.

2.4 Integration to PAI Products

Apart from releasing our toolkit to the open-source community for users to run all kinds of KD algorithms in local environments, we have integrated its key functionalities into Alibaba Cloud's Platform for AI (PAI)[5], a cloud-native machine learning platform. In the platform, all the distilled models produced using *EasyDistill* (e.g., the *DistilQwen* series) are available in the PAI-Model Gallery. This platform supports the entire lifecycle of the LLM usage, including training, evaluation, compression, and deployment of these models. The KD pipelines and practical solutions can be seamlessly executed on deep learning containers on PAI.

Note that although we have provided product integration for *EasyDistill*, the toolkit itself is not platform-dependent; it can be run in any environment satisfying the Python requirements, including other cloud platforms.

3 Conclusion and Future Work

In this paper, we have introduced *EasyDistill*, a comprehensive toolkit focusing on KD for LLMs. It encompasses a suite of advanced algorithms, including data synthesis, SFT, ranking optimization, and RL techniques, all specifically tailored for KD scenarios. Additionally, it includes several practical solutions and is integrated with Alibaba Cloud's Platform for AI (PAI) for large-scale deployment. In the future, we aim to extend the toolkit by supporting a wider range of advanced KD algorithms and by adding more domain-specific solutions to align it even more closely with practical needs.

Limitations

There are a few limitations that should be acknowledged. Firstly, the toolkit primarily focuses on established methods for KD, which may limit the exploration of non-standard KD techniques that require further manual integration into the toolkit. Secondly, although *EasyDistill* includes a variety of industrial solutions, the effectiveness can vary based on the specific domains and the quality of available datasets. Finally, while *EasyDistill* enhances accessibility within the NLP community, the toolkit assumes a certain level of technical proficiency for effective utilization. Users lacking deep familiarity with KD processes or LLMs may face a steep learning curve when attempting to leverage the advanced features provided by the toolkit.

Ethic Considerations and Broader Impact

The development of *EasyDistill* makes complex KD processes more accessible to both academic researchers and industry practitioners. It offers a means for companies and educational institutions with limited resources to implement cutting-edge AI models. *EasyDistill*'s integration into Alibaba Cloud's Platform for AI (PAI) and its practical solutions further enhance the toolkit's impact by demonstrating its viability for large-scale deployment. Moreover, the open source of *EasyDistill* encourages community involvement, which could lead to new enhancements in KD techniques.

However, the deployment of models distilled using *EasyDistill* also requires careful consideration of ethical implications, including the potential for bias inherent in LLMs. Ensuring ethical standards are upheld will be crucial to mitigating potential negative social impacts.

Acknowledgments

This work was partially supported by Alibaba Research Intern Program.

References

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez,

[5] https://www.alibabacloud.com/en/product/machine-learning

Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosiute, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemí Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. 2022. Constitutional AI: harmlessness from AI feedback. *CoRR*, abs/2212.08073.

Wenrui Cai, Chengyu Wang, Junbing Yan, Jun Huang, and Xiangzhong Fang. 2025a. Reasoning with omnithought: A large cot dataset with verbosity and cognitive difficulty annotations. *CoRR*, abs/2505.10937.

Wenrui Cai, Chengyu Wang, Junbing Yan, Jun Huang, and Xiangzhong Fang. 2025b. Training small reasoning llms with cognitive preference alignment. *CoRR*, abs/2504.09802.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*. OpenReview.net.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626. ACM.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2024. RLAIF vs. RLHF: scaling reinforcement learning from human feedback with AI feedback. In *Forty-first International Conference on Machine Learning*. OpenReview.net.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhijiang Guo, Le Song, and Cheng-Lin Liu. 2025. From system 1 to system 2: A survey of reasoning large language models. *CoRR*, abs/2502.17419.

Ruibo Liu, Jerry Wei, Fangyu Liu, Chenglei Si, Yanzhe Zhang, Jinmeng Rao, Steven Zheng, Daiyi Peng, Diyi Yang, Denny Zhou, and Andrew M. Dai. 2024. Best practices and lessons learned on synthetic data for language models. *CoRR*, abs/2404.07503.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3505–3506. ACM.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300.

Luong Quoc Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. 2024. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*, pages 7601–7614. Association for Computational Linguistics.

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, Nathan Sarrazin, Omar Sanseviero, Alexander M. Rush, and Thomas Wolf. 2023. Zephyr: Direct distillation of LM alignment. *CoRR*, abs/2310.16944.

Chengyu Wang, Junbing Yan, Yuanhao Yue, and Jun Huang. 2025. Distilqwen2.5: Industrial practices of training distilled open lightweight language models. *CoRR*, abs/2504.15027.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022*.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Runming Yang, Zhe Zhao, and Ngai Wong. 2025. Rethinking kullback-leibler divergence in knowledge distillation for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 5737–5755. Association for Computational Linguistics.

Xiaohan Xu, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou. 2024. A survey on knowledge distillation of large language models. *CoRR*, abs/2402.13116.

Annepaka Yadagiri and Partha Pakray. 2025. Large language models: a survey of their development, capabilities, and applications. *Knowl. Inf. Syst.*, 67(3):2967–3022.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024a. Qwen2 technical report. *CoRR*, abs/2407.10671.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2024b. Qwen2.5 technical report. *CoRR*, abs/2412.15115.

Chuanpeng Yang, Wang Lu, Yao Zhu, Yidong Wang, Qian Chen, Chenlong Gao, Bingjie Yan, and Yiqiang Chen. 2024c. Survey on knowledge distillation for large language models: Methods, evaluation, and application. *CoRR*, abs/2407.01885.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. 2024d. RLCD: reinforcement learning from contrastive distillation for LM alignment. In *The Twelfth International Conference on Learning Representations*. OpenReview.net.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards thinking-optimal scaling of test-time compute for LLM reasoning. *CoRR*, abs/2502.18080.

Yuanhao Yue, Chengyu Wang, Jun Huang, and Peng Wang. 2024a. Building a family of data augmentation models for low-cost LLM fine-tuning on the cloud. *CoRR*, abs/2412.04871.

Yuanhao Yue, Chengyu Wang, Jun Huang, and Peng Wang. 2024b. Distilling instruction-following abilities of large language models with task-aware curriculum planning. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6030–6054. Association for Computational Linguistics.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*, abs/2303.18223.