# **CoT-Valve: Length-Compressible Chain-of-Thought Tuning**

#### **Anonymous ACL submission**

#### Abstract

Chain-of-Thought significantly enhances a model's reasoning capability, but it also comes with a considerable increase in inference costs due to long chains. With the observation that the reasoning path can be easily compressed under easy tasks but struggle on hard tasks, we explore the feasibility of elastically controlling the length of reasoning paths with only one model, thereby reducing the inference overhead of reasoning models dynamically based on task difficulty. We introduce a new tun-012 ing and inference strategy named CoT-Valve, designed to allow models to generate reasoning chains of varying lengths. To achieve this, we propose to identify a direction in the parameter space that, when manipulated, can ef-017 fectively control the length of generated CoT. Moreover, we show that this property is valuable for compressing the reasoning chain. We construct datasets with chains from long to short for the same questions and explore two enhanced strategies for CoT-Valve: (1) a precise length-compressible CoT tuning method, and (2) a progressive chain length compression approach. Our experiments show that CoT-Valve successfully enables controllability and compressibility of the chain and shows better performance than the prompt-based control. We applied this method to QwQ-32B-Preview, reducing reasoning chains on GSM8K from 741 to 225 tokens with a minor performance drop (95.07% to 94.92%) and on AIME from 6827 to 4629 tokens, with only one additional incorrect answer.

#### 1 Introduction

042

Chain-of-Thought (CoT) reasoning (Wei et al., 2022) has emerged as a powerful technique for enhancing the reasoning capabilities of large language models (Jaech et al., 2024; Dubey et al., 2024; Abdin et al., 2024), particularly in complex tasks such as mathematics and coding (Sprague et al., 2024) that require multi-step inference. By simulating the process of human-like thought progression, CoT enables models to break down complex problems into sub-questions, improving accuracy and interpretability (Joshi et al., 2023). Those reasoning abilities have also been tested in different domains, such as image generation (Ma et al., 2025) and visual understanding (Shao et al., 2024). 043

045

047

049

051

054

055

057

060

061

062

063

064

065

066

067

068

069

070

071

072

073

074

075

077

079

083

Training reasoning models often involves generating extensive reasoning paths through methods such as sampling (Wang et al., 2023), tree search (Yao et al., 2023; Guan et al., 2025; Zhang et al., 2024) or reinforcement learning (DeepSeek-AI, 2025) to reach the correct answer ultimately. However, these long chains often incorporate redundant intermediate steps that can be unnecessary or too complex (Lightman et al., 2024), and the redundancy in the reasoning paths for training leads to inefficiencies in token usage and increased inference costs. However, crafting an optimal reasoning chain that omits extraneous details is challenging due to the limited availability of intermediate rewards to guide the process and human annotations (Zhang et al., 2025). Removing some or all of the intermediate steps and then training or distilling the model (Liu et al., 2024b; Yu et al., 2024) will degrade the performance. Alternative approaches employ information-theoretic measures (Ton et al., 2024) or identify an "overthinking" solution in QwQ (Team, 2024b) to evaluate the contribution of each sentence to the final answer.

We observe that current reasoning models, such as QwQ (Team, 2024b) and DeepSeek-R1 (DeepSeek-AI, 2025) allocate an excessive number of tokens to simple tasks, while potentially providing insufficient tokens for complex tasks. *Thus, a long reasoning path is still essential, while maintaining the ability to compress reasoning paths for simpler questions is equally important.* To solve this, our goal is to fine-tune a model capable of generating both long and short reasoning paths, rather than being restricted to a compressed



Figure 1: The reasoning model, after the length-compressible CoT tuning, can generate reasoning paths from long to short, leveraging LoRA as a 'Valve'. We show one example from our constructed dataset MixChain.

form. We offer a new way to control the length of CoT, which we refer to as Length-Compressible Chain-of-Thought Tuning.

A central component of the proposed method is to identify an update direction in the parameter space, which, by manipulating it, acts as increasing or decreasing the length of CoT. Taking a large step in this direction leads the model to generate a short sequence, while a small step still produces a long and complex reasoning trajectory. We choose to incorporate this update direction by LoRA (Hu et al., 2022), enabling it to function as an additional branch that facilitates easy modulation of intensity while imposing minimal extra parameters on the model. We explore methods to identify this direction and demonstrate that it offers superior controllability compared to promptbased approaches, which enables the generation of short CoT that prompt-based methods are unable to achieve. Besides, we observe that the direction can be extrapolated, allowing the reasoning chains to be extended beyond or shortened to lengths unseen in the training set. Leveraging this compressibility, we construct a dataset that pairs long and short reasoning chains for each question. This dataset is then utilized in two ways: (1) to refine the direction for more precise tuning, and (2) to compress the reasoning path progressively.

We evaluate our method across different types of models, ranging from a pre-trained LLM with little reasoning ability, LLaMA-3.1-8B and LLaMA-3.2-1.5B-Instruct (Dubey et al., 2024), to post-trained reasoning models, QwQ-32B-Preview (Team, 2024b), and distilled reasoning models, DeepSeek-R1 (DeepSeek-AI, 2025). Our results demonstrate that, with training for one time, our approach enables a model to generate reasoning paths of varying lengths, and we can achieve better results than previous chain compression baselines. Besides, our study highlights several interesting findings: (1) Short reasoning paths can sometimes outperform longer ones, underscoring the significance of CoT-Valve in enhancing model efficiency. (2) Not every reasoning chain, despite all leading to the correct final answer, is conducive to model optimization. Excessively long or short chains complicate the distillation of CoT, posing challenges to the model training.

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

162

In summary, our contributions are: (1) **CoT-Valve**: Enables elastic control of length for CoT within the parameter space, allowing a single model to generate CoT from short to long. (2) **MixChain Dataset**: A dataset with reasoning paths of varying lengths for each question. (3) **Improved Tuning & Progressive Compression**: Refines the direction-tuning process based on MixChain and introduces progressive compression for inference efficiency. (4) **Performance & Controllability**: Achieves controllable reasoning generation and state-of-the-art results for compressed CoT.

#### 2 Related Work

Chain-of-Thought. Chain-of-thought (Wei et al., 2022) reasoning has shown promising progress in recent years, especially the success of OpenAi-O1 (Jaech et al., 2024) and Deepseek-R1 models (DeepSeek-AI, 2025). This introduces the testtime scaling law, apart from the traditional scaling law for training (Hoffmann et al., 2022). Several approaches have been proposed to boost the language model to have better problem-solving abilities, including the model has its self-reasoning abilities (Team, 2024b) or use Best-of-N (Nakano et al., 2021), beam search and Monte Carlo Tree Search (Kocsis and Szepesvari, 2006; Guan et al., 2025) to search and refine the solution without further finetune the large language models. The outcome reward model and process reward models are also introduced to evaluate the score for the entire solution, especially the final answer (Cobbe et al.,

117

118 119

120

121

122

2021a) and the quality of the reasoning path (Wang 163 et al., 2024; Luo et al., 2024) 164

Chain Compression in reasoning model. Due 165 to the high computational cost associated with in-166 ference in reasoning models, particularly for longchain reasoning, chain compression has become a 168 169 critical area of research. (Yu et al., 2024) attempts to distill the chain-of-thought into System 1 but 170 fails to observe improvements when intermediate 171 steps are omitted. (Jin et al., 2024) conducts a comprehensive empirical study between length and per-173 174 formance. (Deng et al., 2023) proposes internalizing reasoning steps within the hidden states of mod-175 els, while several implicit-based approaches(Deng 176 et al., 2024; Hao et al., 2024; Cheng and Durme, 177 2024) aim to compress token-wise generation by 178 transitioning from language space to hidden space. 179 Other studies focus on skipping intermediate rea-180 soning steps (Liu et al., 2024b) or using summarization techniques to generate shorter reasoning 182 chains (Kang et al., 2024). Additionally, (Chen et al., 2024) addresses the overthinking issue in QwQ (Team, 2024b) and employs SimPO (Meng et al., 2024) for optimization. Kimi K1.5 (Team 186 et al., 2025) proposes merging long-CoT models 187 with short-CoT models in a training-free manner. O1-Pruner (Luo et al., 2025) adopts reinforcement learning to shorten responses. 190

#### Method 3

191

192

194

195

196

197

198

199

207

211

In this section, we provide an in-depth discussion of our method. Section 3.1 introduces a simple yet effective approach that enables a single tuning process to generate models with CoT with different lengths. This stage also serves as an initial step for subsequent refinements. Next, in Section 3.2, we explore multiple scenarios in which we can apply CoT-Valve to construct the dataset MixChain. In Section 3.3, we propose several advanced methods that take advantage of long-to-short datasets to improve precision and control over the generated reasoning paths in compressible fine-tuning.

#### 3.1 Length-Compressible CoT Tuning

Our primary objective is to achieve a new way to control the length of reasoning paths after training a reasoning model. Existing approaches, such as prompt-based control, explicitly define sequence length in the prompt (Han et al., 2024) or utilize summary tokens (Ding et al., 2024) for guidance. 210 However, these methods offer only limited control

over the length of CoT generated. For instance, requesting a sequence of less than 20 tokens may result in the model generating over 350 tokens (see Table 12 in the Appendix), and these methods struggle to produce answers with very short lengths. To address these limitations, we introduce CoT-Valve for training one model but can adjust the length of reasoning paths.

212

213

214

215

216

217

218

219

220

221

222

223

224

225

227

228

229

230

231

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

251

252

253

254

255

256

Consider a reasoning model defined by the parameter  $\theta$ . For a given question q in the dataset  $\mathcal{D}$ , the probability of generating an answer a and its reasoning thoughts  $\{t_i\}_{i=1}^n$  given the question q can be described by:

$$p(a \mid t_1, \dots, t_n, q; \theta) \prod_{i=1}^n p(t_i \mid t_{< i}, q; \theta) \quad (1)$$

where  $\{t_i\}_{i=1}^n$  might include errors or unnecessary details. With short synthesized or human-annotated explanations  $\{t_i\}_{i=1}^m$  with m < n, the training objective is to adjust the parameter in such a way that the chain is shortened while still yielding the correct answer:

$$\max_{\Delta \theta} \mathbb{E}_{(q,a)\sim\mathcal{D}} p\left(a \mid t_1, \dots, t_m, q; \theta + \Delta \theta\right)$$

$$\prod_{i=1}^m p\left(t_i \mid t_{< i}, q; \theta + \Delta \theta\right)$$
(2)
232
233

and  $\Delta \theta$  denotes the change in the parameter space that steers the model towards generating a more concise chain.

 $\Delta t$ 

Since the model, with and without  $\Delta \theta$ , outputs the same final answer,  $\Delta \theta$  can be interpreted as a task vector (Ilharco et al., 2023). The task here is to control the length of the CoT, provided that the only difference in the training set lies in intermediate reasoning steps  $\{t_i\}_{i=1}^n$ . Those reasoning paths are different in length but ultimately lead to the same final answer. Thus, we can control the task vector to achieve the goal of adjusting the length of CoT.  $\Delta \theta$  is designed within a parameter-efficient space, functioning as an external branch for inference that incurs minimal overhead. Controlling this external branch enables the manipulation of the length of the reasoning path.

Task Arithmetic: Interpolation and Extrapolation of  $\Delta \theta$ . To manipulate this update within the parameter space, we can control the magnitude of a  $\Delta \theta$  as an arithmetic operation. We use two primary operations on  $\Delta \theta$  here: interpolation and extrapolation. Let  $\alpha$  denote the magnitude of  $\Delta \theta$  for LoRA.



Figure 2: Illustration of CoT-Valve. In Stage 1, we first determine  $\Delta \theta$  from distilling or post-training. Then, the trained  $\Delta \theta$  is utilized to construct the MixChain dataset. Using this dataset, we can then apply two enhanced training methods to achieve more precise control over reasoning paths or to shorten the reasoning paths as needed.

When  $\alpha$  falls within the range of (0,1), the model smoothly transitions between longer and shorter reasoning paths, similar to weight interpolation between two models (Frankle et al., 2020; Team et al., 2025). When  $\alpha > 1$ , extrapolation is introduced, further shortening the reasoning path beyond what was observed during training. This enables an exploration of the minimal reasoning length required to arrive at a given answer. Thus, by adjusting  $\alpha$ at inference, we can modulate the model's behavior, with each value of  $\alpha$  corresponding to different CoT lengths.

257 258

265

266

269

270

271

274

275

276

279

281

290

**Application** Unlike prompt-based approaches that can only regulate the overall length of the reasoning process using prompt words,  $\Delta\theta$  provides finer granularity control.  $\Delta\theta$  is served in the external parameter space. This allows for greater flexibility in adjusting the reasoning trajectory. Specifically, it facilitates the selective retention of longchain reasoning in certain thoughts while applying stronger compression to simpler reasoning segments. As a result, reductions in chain length can be localized to specific portions of the inference process rather than being uniformly applied across the entire reasoning path. We remain the design of this segment selection in future work.

#### **3.2 Construct the MixChain Dataset**

A crucial thing for the above process is the construction of the training dataset, especially the reasoning chain  $\{t_i\}_{i=1}^n$ . To have reasoning chains with different lengths, previous approaches rely on multiple rounds of sampling, selecting reasoning paths under different random seeds, or using some handcrafted way to remove parts of the answer (Chen

#### et al., 2024).

We introduce MixChain, a dataset inherently generated by our method that contains reasoning paths of varying lengths. This dataset is structured such that each question is associated with multiple reasoning paths, with lengths progressively decreasing from long to short. By simply adjusting the parameter  $\alpha$ , our approach avoids the need for repeated sampling and achieves this diverse set of reasoning paths. In contrast to multi-sampling techniques, MixChain enables a more reliable and consistent generation of shorter reasoning paths while simultaneously capturing a spectrum of reasoning lengths. To construct MixChain, we consider two possible scenarios:

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

- If a well-annotated dataset with humanlabeled solutions is available, such as GSM8K (Cobbe et al., 2021b) or PRM800k (Lightman et al., 2024), it can be leveraged to fine-tune the model for generating shorter reasoning chains as a cold start ( $\theta_1 \rightarrow \tilde{\theta}_1$  and  $\theta_2 \rightarrow \tilde{\theta}_2$  in Figure 2).
- In the absence of a dataset containing explicit reasoning paths, or when only final answers are available without full explanations, training solely on final answers is unlikely to enable the model to generate reasoning steps. To address this limitation, we propose an alternative method for constructing Mix-Chain. Specifically, we leverage an existing base LLM (e.g., LLaMA-3.1-8B or Qwen-32B-Instruct) as θ<sub>1</sub> and use its corresponding reasoning model (e.g., DeepSeek-R1-Distill-Llama-8B or QwQ-Preview) to derive Δθ.

The parameter update between these models serves as a form of linear interpolation, enabling the transition from  $\theta_1$  to  $\theta_2$ . This transition is then used to construct the dataset, as illustrated in Figure 2, where the parameter shift is represented by  $\theta_1 \rightarrow \theta_2$ .

#### **3.3 Improved Tuning for CoT-Valve**

325

330

332

334

354

In this section, we present two enhanced variants of CoT-Valve: one aimed at achieving improved controllability and the other focused on optimizing the compression ratio of the reasoning paths.

A More Precise CoT-Valve Paradigm: CoT-336 Valve++. In the previously proposed CoT-Valve 337 framework, the training process only constrained  $\Delta \theta$  to satisfy the final objective with  $\alpha = 1$ . However, during inference, we expect all positions along this direction to exhibit reasoning trajectories 341 of varying lengths. This leads to the inconsistency 342 between training and inference. With MixChain, 343 we can explicitly incorporate this requirement dur-344 ing training by introducing an additional constraint, 345 ensuring that the model can adapt to reasoning chains of different lengths across all positions in 347 this direction. For each training sample, in addition to the question, answer, and solution, we have in-349 troduced a normalized term  $\beta$ , which represents the factor for the length of the reasoning path. Under 351 this dataset, our training objective is modified to find a parameter update  $\Delta \theta'$  such that it satisfies:

$$\max_{\Delta \theta'} \mathbb{E}_{(q,a)\sim\mathcal{D}'} p\left(a \mid t_{\leq m}, q; \theta + \beta \Delta \theta'\right)$$
$$\prod_{i=1}^{m} p(t_i | t_{\leq i}, q; \theta + \beta \Delta \theta') \quad (3)$$

Where  $\mathcal{D}'$  is the Mixchain dataset. Each sample consists of the question q, the answer a, the solution  $\{t_i\}_{i=1}^m$  and  $\beta$ , where  $\beta$  is calculated as:

$$\beta = 1 - \frac{m - m_{min}}{m_{max} - m_{min}} \tag{4}$$

360 Here,  $m_{min}$  and  $m_{max}$  is the length of the short-361 est solution and longest solution for this question. 362 Based on synthetic samples, we introduce addi-363 tional constraints that enable us to better identify 364 the updated parameter  $\Delta \theta'$ , facilitating more pre-365 cise compressibility and controllability.

366 Progressive Chain Compression: CoT-Valve+P.
 367 The structure of MixChain, which features progressively shorter reasoning paths for each question,

facilitates a progressive chain-length compression strategy. This approach is similar to iterative pruning in model compression (Molchanov et al., 2017). In this process, the model is trained with a shorter reasoning path from the dataset at each iteration, rather than training directly with the shortest reasoning CoT. This gradual compression method allows the model to progressively reduce the length of its reasoning paths. 369

370

371

372

373

374

375

376

377

378

381

383

384

386

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

#### 4 Experiments

### 4.1 Experimental Setup

**Models.** We evaluate our method under several models: QwQ-32B-Preview (Team, 2024b), DeepSeek-R1-Distill-Llama-8B (DeepSeek-AI, 2025), LLaMA-3.1-8B (Dubey et al., 2024), LLaMA-3.2-1B (Dubey et al., 2024) and Qwen-32B-Instruct (Team, 2024a) with LIMO (Ye et al., 2025). We tested different scenarios for CoT-Valve:

- (Long to Short CoT) For QwQ-32B-Preview (QwQ for abbreviation) and DeepSeek-R1-Distill-Llama-8B (R1-Distill), we used our method to control and compress the length of the reasoning chain.
- (Short to Long CoT) For LLaMA-3.1-8B and LLaMA-3.2-1B-Instruct, we applied our method to distill reasoning abilities from QwQ-32B-Preview and incorporated CoT-Valve in the distillation process.
- (Short-Long-Short CoT) We tested another setting to first post-train a short-CoT LLM, Qwen-2.5-32B-Instruct (Team, 2024a), to generate Long CoT and then compress it to Short CoT. CoT-Valve can be applied in both two stages.

**Metrics.** We report both accuracy and the number of tokens in the answer for each experiment. Given the trade-off between reasoning path length, model size, and performance, we use a new metric, Accuracy per Computation Unit(ACU), to better capture this balance and evaluate model efficiency. It is defined as:

$$ACU = \frac{Accuracy}{\#Params \times \#Tokens}$$
(5)

Since the ACU value typically falls within the range of  $10^{-5}$  to  $10^{-2}$ , we report it in units of  $10^2$  for improved readability.

Å



Figure 3: Token length and accuracy for different methods, datasets and reasoning models. Points connected by curves in (a) and (b) represent results from one model.

**Training and Evaluation.** For training the model, we use LoRA (Hu et al., 2022) in most of our experiments, except in the experiment for LIMO on Qwen-2.5-32B-Instruct we use full parameter fine-tuning. We also show the results using DoRA (Liu et al., 2024a) in the Appendix. The hyper-parameters for each experiment are shown in Appendix A. We select two math datasets to evaluate the performance, for one easy math dataset, GSM8K (Cobbe et al., 2021b) and one hard math dataset, AIME24.

#### 4.2 Datasets

We find in our experiments that the quality of the solution is important to the performance, even if all the human-annotated solutions or synthesized solutions reach the final answer. In our experiments, we use the question from the train set of GSM8K, the math split of PRM800K or the question from LIMO, and we employ three types of datasets with those questions in our experiments:

- Ground-truth Dataset: The dataset provides a human-annotated or model-synthesized solution. We use this as the cold start.
- MixChain from cold-start (MixChain-C): After taking the ground-truth dataset to train the model, we can get the first model to generate solutions from short to long. Then we use it to generate the dataset.
- MixChain from zero-shot (MixChain-Z): We employ CoT-Valve between a reasoning model ( $\theta_2$ ) and a base LLM ( $\theta_1$ ) to generate the solutions.

For each dataset, we filter out all the solutions with
incorrect answers. We show the statistics of the
dataset in Table 9 in the Appendix.

Method	Accuracy	#Token	$\text{ACU} \uparrow$		
Llama-3.3-70B-Instruct	92.6	235.4	0.56		
Llama-3.1-405B-Instruct	95.6	186.7	0.13		
Qwen2.5-32B-Instruct	93.1	269.3	1.09		
Qwen2.5-Math-72B-Instruct	95.8	312.1	0.43		
QwQ-32B-Preview	95.1	741.1	0.40		
Prompt (Han et al., 2024)	93.6	355.5	0.82		
Prompt (Ding et al., 2024)	95.5	617.7	0.48		
In-domain Train Set: GSM8K					
CoT-Valve - Ground-Truth	94.0	352.8	0.83		
CoT-Valve++ - MixChain-C	94.4	276.3	1.07		
CoT-Valve+P - MixChain-Z	96.1	317.1	0.95		
CoT-Valve+P - MixChain-Z	94.9	225.5	1.32		
Out-of-Domain Train	Set: PRM12	К			
Overthink(Chen et al., 2024) - SFT	94.8	749.5	0.40		
Overthink(Chen et al., 2024) - SimPO	94.8	326.2	0.91		
O1-Pruner(Luo et al., 2025) - SFT	95.7	717	0.42		
O1-Pruner(Luo et al., 2025)	96.5	534	0.56		
CoT-Valve+P - MixChain-Z	95.4	288.5	1.03		

Table 1: Results of QwQ-32B-Preview on GSM8K. Values of ACU are scaled by  $10^2$  for readability. We list the dataset we use after the method name.

#### 4.3 From Long-CoT to Short-CoT.

**Controllable Results.** We illustrate the result in Figure 3a. First, using ground-truth samples as a cold start, we develop a model capable of generating reasoning paths of various lengths, as demonstrated in 'CoT-Valve' in Figure 3a. CoT-Valve already matches the performance of prompt-based control but can generate shorter reasoning chains. We then extrapolate  $\Delta \theta$  to produce even shorter reasoning paths. Then, building on MixChain-C from this first model, we conduct further training by CoT-Valve++. CoT-Valve++ substantially surpasses the baseline and shows greater generalization capabilities in cases of extrapolation.

**Compression Results.** We evaluated our method against previous chain compression approaches, with the results detailed in Table 1, Table 2, and

Method	AIME24	#Token	ACU↑			
Qwen2.5-32B-Instruct	4/30	1794.2	0.023			
Qwen2.5-Math-72B-Instruct	7/30	1204.5	0.061			
Gemini-Flash-Thinking (Team et al., 2023)	15/30	10810.5	-			
QwQ-32B-Preview.Train s	QwQ-32B-Preview.Train set: GSM8K					
QwQ-32B-Preview	14/30	6827.3	0.021			
Prompt (Han et al., 2024)	13/30	6102.5	0.022			
Prompt (Ding et al., 2024)	13/30	5562.3	0.024			
Overthink (Chen et al., 2024)	13/30	5154.5	0.026			
CoT-Valve - GSM8K	14/30	5975.0	0.024			
CoT-Valve++ - MixChain-C	13/30	5360.5	0.025			
CoT-Valve+P - MixChain-Z	13/30	4629.6	0.029			
Qwen-32B-Instruct. Train set: LIMO						
Qwen-32B-LIMO	15/30	10498.2	0.015			
CoT-Valve	11/30	6365.2	0.018			
SFT - MixChain - Solution 1	13/30	5368.0	0.025			
CoT-Valve - MixChain - Solution 1	15/30	8174.8	0.019			

Table 2: Results of QwQ-32B-Preview and Qwen-32B-Instruct w/ LIMO on AIME 24.

	GSM8k		AII	ME24
Model	Acc	#Token	Acc	# Token
Llama-3.1-8B (0-shot)	15.7	915.0	0/30	1517.6
R1-Distill-Llama-8B	87.1	1636.6	14/30	12359.9
CoT-Valve	87.3	1315.2	6/30	7410.5
CoT-Valve+P - MixChain-Z	84.0	755.2	11/30	9039.0

Table 3: Result of DeepSeek-R1-Distill-Llama-8B.

Table 3. For GSM8K, we adhered to the baseline setup to train with PRM12K. Utilizing progressive compression, our method surpassed the baseline by producing shorter reasoning paths and improved performance.

We also report experimental results on AIME, where the model was trained using MixChain-Z derived from GSM8K. To minimize the impact of randomness on performance, we employed greedy decoding in our AIME experiments. Compared to the baseline (Chen et al., 2024), our method reduced the token count from 5155 to 4630 while maintaining the same accuracy, despite being trained on an easier dataset.

#### 4.4 From Short-CoT to Long-CoT & Short-Long-Short CoT

Our method can also be applied if a short-CoT model is distilled or post-trained to be a Long-CoT model. The results are shown in Figure 3b, Table 4 and Table 5. We found that CoT-Valve can also effectively control the length of the chains in this setting. Notably, we observed that shorter chains could achieve higher accuracy on GSM8K. Moreover, if the model is trained using the MixChain-Z dataset, the results are significantly better, whether using CoT-Valve (55.5 to 58.9) or just simply SFT

Method	Accuracy	#Tokens	ACU↑
LLaMA-3.2-1B-Instruct(8-shot)	45.9	104.3	44.008
LLaMA-3.2-1B-Instruct(0-shot)	45.9	199.8	22.973
SFT-Full Finetune - GSM8k	46.1	139.4	33.070
SFT - GSM8k	43.8	137.7	31.808
Prompt	46.7	209.9	22.249
SFT - QwQ Distill	52.7	759.3	6.941
CoT-Valve - QwQ Distill	55.5	267.0	20.786
CoT-Valve+P - MixChain-Z	55.8	291.0	19.175
SFT - MixChain-Z - Solution 1	57.0	288.4	19.764
CoT-Valve - MixChain-Z - Solution 1	58.9	275.4	21.387

Table 4: Results on LLaMA-3-2-1B-Instruct. We report the result of Flexible Match here. QwQ Distill means we use QwQ to synthesize the solution and distill it.

Method	Accuracy	#Tokens	ACU↑
LLaMA-3.1-8B (8-shot)	56.9	282.1	2.521
LLaMA-3.1-8B (0-shot)	15.7	915.0	0.214
SFT-LoRA - GSM8k	59.0	191.9	3.843
SFT-LoRA - QwQ Distill	76.3	644.8	1.479
CoT-Valve - QwQ Distill	77.5	569.8	1.700
CoT-Valve+P - MixChain-Z	77.1	371.2	2.596
CoT-Valve + MixChain-Z - Solution 1	75.7	264.1	3.583

Table 5: Result on LLaMA-3.1-8B. We report the result of Strict Match here.

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

(52.7 to 57.0). Additionally, after training a longchain model, we can employ the MixChain dataset to reduce the length of its reasoning chains further. As illustrated in Figure 3c, the results suggest that initially training the chains to be long and subsequently compressing them to be shorter (Results with Long-to-Short) can yield better performance than directly using CoT-Valve in the short-to-long stage (Results with Short-to-Long). This demonstrates significant potential for compressing the reasoning chains. We can also surpass the result of Gemini-Flash-Thinking, with the same accuracy but fewer tokens (10810.5 v.s. 8174.8)

**Training dynamics does not have the same effect as CoT-Valve.** We also explore whether intermediate training steps can achieve similar effects. As depicted in Figure 3c, during the early training phases, the length of the CoT increases but does not correspond with the same rapid improvement in performance. As training progresses, the token length begins to decrease while performance improves. CoT-Valve exhibits a distinct pattern, smoothly bridging the gap between the length of CoT and performance.

#### 4.5 Observations

Based on the results from LLaMA-3.1-8B, LLaMA-3.2-1.5B, QwQ, DeepSeek-R1-Distill-

485

486

487 488

489

490

491

492

467

Solution	Solution Length	Accuracy	#Token
Ground-Truth (Solution 0)	116.0	43.8	139.4
Solution 1	279.6	57.0	288.4
Solution 2	310.7	55.1	330.0
Solution 3	386.7	56.5	414.6
Solution 4	497.2	52.5	558.3

Table 6: Train LLaMA-3.2-1B-Instruct with solutions in MixChain-Z of different lengths on GSM8K.

Llama-8B and Qwen2.5-32B-Instruct with LIMO, we summarize the following observations:

521

522

524

526

528

530

531

532

533

534

536

538

541

542

543

544

545

546

547

548

551

552

553

554

555

556

• Longer reasoning chains are not always the best on simple datasets. Across nearly all models, we find that those directly trained on long CoT data typically do not show the best performance. These models often underperform compared to those generated through CoT-Valve, which results in shorter but more accurate reasoning chains. This trend is particularly pronounced in smaller models. For instance, in the LLaMA-3.2-1B model, training on QwQ synthesized data yields an accuracy of 52.69 with 759.3 tokens. However, using CoT-Valve, we can achieve an accuracy of 55.50 with only 267.0 tokens. However, we do not observe this phenomenon in more complex datasets, indicating that while the reasoning model may be redundant for simple datasets, it still requires test-time scaling to effectively handle complex datasets.

 Some reasoning chains are difficult for the model to learn, especially for small LLMs. We fine-tuned LLaMA-3.2-1B-Instruct using only one solution from MixChain, where all solutions lead to the same final answer but involve different intermediate reasoning steps. The results, presented in Table 6, indicate that neither the shortest nor the longest chains are optimal for learning. Instead, the model most effectively learns from moderately short chains, achieving the highest accuracy while maintaining a relatively low token count. This phenomenon is particularly evident in smaller models, but it is not observed in larger models. We believe this could be beneficial for the distillation of CoT in small LLMs.

#### 557 4.6 Analysis

Ablation on Progressive Compression. Table 7
 demonstrates the effect of progressive compression.

Solution Used	#Epoch	#Samples	Accuracy	#Tokens	ACU↑
-	-	-	95.07	741.1	0.40
4	1	6.8k	95.68	597.3	0.50
4+3	1	13.7k	94.84	458.4	0.65
4+3+2	1	20.5k	94.84	339.9	0.87
4+3+2+1	1	27.4k	96.13	317.1	0.95
4+3+2+1+0	1	34.2k	94.92	225.5	1.32
0	5	37.4k	92.19	250.5	1.15

Table 7: Ablation of Progressive Compression on QwQ. Here, solution 0 is the human-annotated solution from the original dataset.

	QwQ-32B-Preview		Llama	a-3.2-1B-I
Method	Acc	#Token	Acc	#Token
Prompt (Shortest)	93.6	355.5	52.5	621.0
Ours (Best)	94.4	276.3	55.5	267.0
Ours (Shortest)	87.5	133.8	50.4	247.0

Table 8: CoT-Valve can achieve shorter chains than prompts with better performance.

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

We compare two settings: training directly with the ground-truth solution for five epochs and applying progressive compression for five epochs in total, with the final epoch using the ground-truth data. Our results show that progressive compression significantly improves the performance of short CoT (from 92.19 to 94.92). For each turn, progressive compression gradually reduces the token number while maintaining accuracy.

**CoT-Valve achieves shorter chains compared to prompt control** We also present in Table 8 the shortest chain achieved by our method and compare these with those obtained using prompt control. Our method outperforms prompt control methods at shorter chain lengths. Additionally, we explored the limits of chain length for both methods and found that our approach can generate substantially shorter chains than what can be achieved through prompt control.

#### 5 Conclusion

In this paper, we propose a method that enables a model to generate reasoning chains of varying lengths instead of the prompt control. Based on this approach, we construct a dataset containing both long and short reasoning chains to further enhance controllability and compression efficiency. Experimental results demonstrate the effectiveness of our method in dynamic reasoning chain control and the compression of CoT. Future research can further explore finer-grained control strategies to improve reasoning efficiency and model controllability.

# 591

6

Limitations

While CoT-Valve effectively shortens reasoning

chains with minimal impact on performance, ex-

treme compression can still result in accuracy

losses, particularly affecting complex tasks. More-

over, performance remains limited by the quality

of the original model; if the model cannot generate

high-quality reasoning chains, constructing and tun-

ing a short-chain dataset becomes challenging. Ad-

ditionally, the interpretability of this mechanism is

limited. Although CoT-Valve enables a controlled

reasoning chain, the theoretical understanding of

Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien

Bubeck, Ronen Eldan, Suriya Gunasekar, Michael

Harrison, Russell J Hewett, Mojan Javaheripi, Piero

Kauffmann, et al. 2024. Phi-4 technical report. arXiv

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He,

Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu,

Mengfei Zhou, Zhuosheng Zhang, et al. 2024. Do

not think that much for 2+3=? on the overthinking

of o1-like llms. arXiv preprint arXiv:2412.21187.

Jeffrey Cheng and Benjamin Van Durme. 2024. Com-

pressed chain of thought: Efficient reasoning through

dense representations. Preprint, arXiv:2412.13171.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias

Plappert, Jerry Tworek, Jacob Hilton, Reiichiro

Nakano, et al. 2021a. Training verifiers to solve math

word problems. arXiv preprint arXiv:2110.14168.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias

Plappert, Jerry Tworek, Jacob Hilton, Reiichiro

Nakano, et al. 2021b. Training verifiers to solve math

word problems. arXiv preprint arXiv:2110.14168.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing rea-

Yuntian Deng, Yejin Choi, and Stuart Shieber. 2024.

From explicit cot to implicit cot: Learning to inter-

nalize cot step by step. Preprint, arXiv:2405.14838.

Smolensky, Vishrav Chaudhary, and Stuart Shieber.

2023. Implicit chain of thought reasoning via knowl-

edge distillation. arXiv preprint arXiv:2311.01460.

Mengru Ding, Hanmeng Liu, Zhizhang Fu, Jian Song,

arXiv preprint arXiv:2406.06580.

Wenbo Xie, and Yue Zhang. 2024. Break the chain: Large language models can be shortcut reasoners.

Yuntian Deng, Kiran Prasad, Roland Fernandez, Paul

Preprint, arXiv:2501.12948.

soning capability in llms via reinforcement learning.

this remains insufficiently explored.

preprint arXiv:2412.08905.

References

593

594

595

610 611

612 614

- 615 616 617

622

625

637

641

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. arXiv preprint arXiv:2407.21783.

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

- Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. 2020. Linear mode connectivity and the lottery ticket hypothesis. In International Conference on Machine Learning, pages 3259-3269. PMLR.
- Xinyu Guan, Li Lyna Zhang, Yifei Liu, Ning Shang, Youran Sun, Yi Zhu, Fan Yang, and Mao Yang. 2025. rstar-math: Small llms can master math reasoning with self-evolved deep thinking. Preprint, arXiv:2501.04519.
- Tingxu Han, Chunrong Fang, Shiyu Zhao, Shiqing Ma, Zhenyu Chen, and Zhenting Wang. 2024. Token-budget-aware llm reasoning. arXiv preprint arXiv:2412.18547.
- Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training large language models to reason in a continuous latent space. Preprint, arXiv:2412.06769.
- Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. Preprint, arXiv:2203.15556.
- Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In International Conference on Learning Representations.
- Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic. In The Eleventh International Conference on Learning Representations.
- Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. arXiv preprint arXiv:2412.16720.
- Mingyu Jin, Qinkai Yu, Dong Shu, Haiyan Zhao, Wenyue Hua, Yanda Meng, Yongfeng Zhang, and Mengnan Du. 2024. The impact of reasoning step length on large language models. In *Findings of the* Association for Computational Linguistics ACL 2024, pages 1830–1842, Bangkok, Thailand and virtual meeting.
- Brihi Joshi, Ziyi Liu, Sahana Ramnath, Aaron Chan, Zhewei Tong, Shaoliang Nie, Qifan Wang, Yejin

- Choi, and Xiang Ren. 2023. Are machine rationales (not) useful to humans? measuring and improving human utility of free-text rationales. *arXiv preprint arXiv:2305.07095*.
  - Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2024. C3ot: Generating shorter chain-of-thought without compromising effectiveness. *Preprint*, arXiv:2412.11664.
  - Levente Kocsis and Csaba Szepesvari. 2006. Bandit based monte-carlo planning. In *European Conference on Machine Learning*.

706

710

711

713

714

715

716

717

718

719

720

721

722

723

727

729

734

735

736

737

738

739

740

741

742

743

744 745

746

747

748

749

751

752

- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.
  - Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. 2024a. Dora: Weightdecomposed low-rank adaptation. In *ICML*.
  - Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang.
    2024b. Can language models learn to skip steps?
    In *The Thirty-eighth Annual Conference on Neural Information Processing Systems.*
  - Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025. O1-pruner: Lengthharmonizing fine-tuning for o1-like reasoning pruning. arXiv preprint arXiv:2501.12570.
  - Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.
  - Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, and Saining Xie. 2025. Inference-time scaling for diffusion models beyond scaling denoising steps. *Preprint*, arXiv:2501.09732.
  - Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. In Advances in Neural Information Processing Systems (NeurIPS).
- Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. 2017. Pruning convolutional neural networks for resource efficient inference. In International Conference on Learning Representations.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Ouyang Long, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin

Chess, and John Schulman. 2021. Webgpt: Browserassisted question-answering with human feedback. *ArXiv*, abs/2112.09332.

- Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. 2024. Visual cot: Unleashing chain-of-thought reasoning in multi-modal language models. *Preprint*, arXiv:2403.16999.
- Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. 2024. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *arXiv preprint arXiv:2409.12183*.
- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.
- Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.
- Qwen Team. 2024a. Qwen2.5: A party of foundation models.
- Qwen Team. 2024b. Qwq: Reflect deeply on the boundaries of the unknown.
- Jean-Francois Ton, Muhammad Faaiz Taufiq, and Yang Liu. 2024. Understanding chain-of-thought in llms through information theory. *Preprint*, arXiv:2411.11984.
- Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 9426–9439, Bangkok, Thailand. Association for Computational Linguistics.
- Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.
- Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of thoughts: Deliberate

808 problem solving with large language models. In 809 Thirty-seventh Conference on Neural Information 810 Processing Systems. 811 Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie 812 Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. arXiv preprint arXiv:2502.03387. 813 814 Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. 2024. 815 Distilling system 2 into system 1. arXiv preprint arXiv:2407.06023. 816 Di Zhang, Xiaoshui Huang, Dongzhan Zhou, Yuqiang 817 Li, and Wanli Ouyang. 2024. Accessing gpt-4 818 level mathematical olympiad solutions via monte 819 carlo tree self-refine with llama-3 8b. Preprint, 820 arXiv:2406.07394. 821 Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen 822 Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jin-823 gren Zhou, and Junyang Lin. 2025. The lessons of 824 developing process reward models in mathematical 825 826 reasoning. Preprint, arXiv:2501.07301.

# 827

020

830

831

832

835

837

838

841

842

847

848

852

853

855

861

867

A Implementation Details

# A.1 Evaluation Metric.

For experiments on LLaMA, we use lm-evalharness<sup>1</sup> to evaluate the model performance. For LLaMA-3.1-8B, we report the strict matching metric due to observed repetition in the model's responses, which causes the flexible match to extract incorrect numerical values. For LLaMA-3.2-1B-Instruct, we report results using the flexible match metric. For QwQ-32B-Preview, DeepSeek-R1-Distill-Llama-8B and Qwen-2.5B-LIMO, we first extract the result enclosed within \boxed{}. If no such boxed answer is found, we default to using the last digit in the response as the final answer.

## A.2 Training Setting.

**LLaMA-3.1-8B** The model is trained using eight A5000 24GB GPUs. We set the batch size to 64 and the peak learning rate to 4e-5, following a cosine decay schedule. A weight decay of 0.01 is applied. For the progressive chain compression experiment, we train the model for two epochs with each type of solution. For all other experiments, we train for a maximum of eight epochs. For LoRA, the rank is set to 32, and the lora\_alpha for training is set to 64. During inference, the maximum number of tokens is set to 2048.

LLaMA-3.2-1B-Instruct The model is trained using 8 A5000 24GB GPUs. We set the batch size to 8 for the CoT-Valve experiment and 64 for all other experiments. The peak learning rate is 4e-5, following a cosine decay schedule, except for the SFT - GSM8K experiment, where the peak learning rate is 1e-5. A weight decay of 0.01 is applied. For the 'CoT-Valve' and 'SFT-Full Finetune - GSM8k' experiment, we train for a maximum of four and six epochs, respectively. For the progressive chain compression experiment, we train the model for two epochs with each type of solution. For all other experiments, training is conducted for up to 8 epochs. For LoRA, the rank is set to 32, and the lora\_alpha for training is set to 64. During inference, the maximum number of tokens is set to 2048.

870QwQ-32B-Preview.The model is trained on two871H100-80G GPUs. We set the batch size to 64 and872trained for a maximum of five epochs. The learning873rate is 1e-5, with a weight decay of 0.01 applied

during training. For LoRA, the rank is set to 2, and the lora\_alpha for training is set to 8. During inference, we set the maximum token to be 4192 for GSM8K and the maximum token as 8192 for AIME correspondingly.

874

875

876

877

878

879

880

881

882

883

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

**DeepSeek-R1-Distill-Llama-8B.** Our experiment on DeepSeek-R1-Distill-Llama-8B<sup>2</sup> is conducted using the MixChain-zero-shot-GSM8K dataset. The batch size is set to 128, and training is performed for a maximum of five epochs. To ensure that the inference process successfully generates the final answer, we set the maximum token limit to 30K.

**Qwen2.5-32B-LIMO.** We fine-tuned Qwen-32B-Instruct using LIMO, training on four H100 GPUs for 10 epochs with a batch size of 4 and a maximum sequence length of 16K. The learning rate was set to 5e-6. We define Qwen-32B-Instruct as  $\theta_0$  and the trained model as  $\theta_1$ , treating the update direction between them as  $\Delta \theta$ . By adjusting  $\alpha$ , we generated the MixChain-C-LIMO dataset, which includes two solutions: solution 1 ( $\alpha$ =0.8) and solution 0 ( $\alpha$ =0.6).

Based on this, we further trained  $\theta_2$  for 5 epochs with a batch size of 32, a learning rate of 5e-6, and a weight decay of 0.01, obtaining the results of MixChain-Solution 0 in Table 2. This model can be further refined through CoT-Valve (Results: CoT-Valve + MixChain - Solution 0). Unlike previous experiments, we applied full fine-tuning instead of LoRA. The maximum generated sequence length in this experiment was 15K.

# A.3 Dataset Explanation

As detailed in Section 4.2, we constructed two types of datasets: MixChain-C and MixChain-Z. The statistics for the datasets are shown in 9. For these datasets, we select  $\alpha$  values ranging from [0.6, 0.8] for LIMO and [0.2, 0.4, 0.6, 0.8] for other datasets, ensuring all incorrect responses are excluded. For MixChain-Z, while the training transition from  $\theta_1$  to  $\theta_2$  remains a black box, we can still identify numerous model pairs such as Qwen-32B-Instruct  $\rightarrow$  QwQ-32B-Preview, and LLaMA-3.1-8B  $\rightarrow$  R1-Distill-Llama-8B, as documented in the technical report. We find that the performance of the base model significantly influences the quality of the datasets.

<sup>&</sup>lt;sup>1</sup>https://github.com/EleutherAI/Im-evaluation-harness

<sup>&</sup>lt;sup>2</sup>https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B

Dataset	Solution Index	#Samples	#Avg Token		
GSM8K					
Ground-Truth	1	7473	121.8		
MixChain-C	1	22419	294.8		
	0 (Ground-Truth)		116.0		
	1		279.6		
MixChain-Z	2	6863	310.7		
	3		386.7		
	4		497.2		
	PRM12K	-			
Ground-Truth	1	12000	223.1		
	0 (Ground-Truth)		172.3		
	1		583.2		
MixChain-Z	2	8841	613.7		
	3		739.3		
	4		1003.2		
LIMO					
Ground-Truth	1	817	6984.1		
MixChain C	1	474	2994.7		
witxChain-C	2	564	4890.6		

Table 9: Dataset Statistic. Here we use the tokenizer from QwQ-32B-Preview to count the number of tokens.

α	0	0.125	0.25	0.5	0.75	1.0
# Tokens	199.8	219.4	233.4	257.7	466.3	772.7
Accuracy	45.9	47.5	50.2	57.1	55.0	54.5

Table 10: Results of LLaMA-3.2-1B-Instruct trained with DoRA using different  $\alpha$  values for interpolation.

### **B** More Analysis

922

923

924

925

927

928

930

932

934

935

937

938

939

941

**Experiments on DoRA.** In addition to LoRA, we also train LLaMA-3.2-1B using DoRA (Liu et al., 2024a) and control the magnitude of  $\Delta\theta$  by adjusting the  $\alpha$  for DoRA. The model is trained on QwQ synthesized data for a maximum of five epochs. We set the batch size to 8 and the peak learning rate to 4e-5, following a cosine decay schedule. A weight decay of 0.01 is applied. For DoRA, the rank is set to 32, and the lora\_alpha for training is set to 64.

As shown in Table 10, the chain length increases with the  $\alpha$  value, demonstrating the effectiveness of interpolating  $\Delta \theta$  for DoRA. Furthermore, similar to our observations with LoRA, the best result is not obtained by directly training the model on long CoT data. Specifically, training on QwQ synthesized data ( $\alpha = 1.0$ ) achieves an accuracy of 54.5 with 772.7 tokens, whereas the best model obtained via CoT-Valve ( $\alpha$ =0.5) achieves an accuracy of 55.72 with only 257.7 tokens.

942Attention has less effect on the length of the943reasoning path than MLP. We experimented

Modules	GSM8K	#Tokens	#Params	$ $ ACU $\uparrow$
-	95.1	741.1	-	0.40
K+V	95.0	687.7	0.005%	0.43
Q	95.2	621.4	0.004%	0.48
0	95.2	484.2	0.004%	0.61
Attention	94.2	284.2	0.013%	1.04
MLP	93.5	221.8	0.038%	1.32
All Linear	92.4	227.6	0.051%	1.27

Table 11: LoRA on Different Modules.

with applying LoRA fine-tuning to only specific model components in QwQ. Table 11 shows that fine-tuning only the query, key, or value projection had less effect than other linear projections on shortening the reasoning chain. This suggests that attention computation may have less impact on controlling chain length, whereas the MLP layers and the final projection in attention play a more significant role. We find this to be an intriguing phenomenon, and further investigation into its underlying causes could provide deeper insights into optimizing reasoning chain compression.

**Prompt Control.** We evaluate the length of CoT under constraint in prompts. Table 12 presents the number of tokens generated when using various prompts across two models. For these two models, the prompts used here are:

Prompt Template for QwQ-32B-Preview
<b>System</b> : You are a helpful and harmless assistant. You are Qwen developed by Alibaba. You should think step-by-step. <b>User</b> : Generate the solution in less than <to- ken_count&gt; tokens. <question></question></to- 
Prompt Template for LLaMA-3.2-1B-Instruct
System: Cutting Knowledge Date: December 2023 Today Date: 01 Jan 2025 User: Given the following problem, reason and give a final answer to the problem using less than <token_count></token_count>
tokens

From the results, we observe that while these prompts provide control over the length, there remains a significant discrepancy between the generated token count and the intended target. 961

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

962 963

964 965

966

<Question>

Generated Answer for the Same Question

**Question**: Amanda had 10 notebooks. This week, she ordered 6 more and then lost 2. How many notebooks does Amanda have now?

Ground Truth Solution - 49 Tokens:

This week, Amanda added 6 notebooks - 2 notebooks = (6-2=4) notebooks. Now, Amanda has 10 notebooks + 4 notebooks = (10+4=14) notebooks.

**Extrapolation - 300 Tokens:** So Amanda started with 10 notebooks. Then, she ordered 6 more. So, I think I need to add those 6 to the original 10. That should be 10 plus 6, which is 16. Okay, now she lost 2 notebooks. So, I need to subtract those 2 from the 16. That should be 16 minus 2, which equals 14. So, Amanda has 14 notebooks now. *Wait a minute, let me double-check that.* She had 10, ordered 6 more, so that's 10 + 6 = 16. Then lost 2, so 16 - 2 = 14. Yeah, that seems right. *Alternatively, maybe I can think of it differently*. Maybe consider the net gain of notebooks. She ordered 6 and lost 2, so net gain is 6 - 2 = 4. Then, add that net gain to the original 10: 10 + 4 = 14. Same answer. *Or, maybe another way:* starting with 10, losing 2 would bring her to 8, and then ordering 6 more would bring her to 14. Again, the same result. I think no matter how I look at it, the answer is 14 notebooks. \*\*Final Answer\*\* \boxed{14}

**QwQ-32B-Preview - 257 tokens**: So Amanda started with 10 notebooks. Then, she ordered 6 more. So, I need to add those 6 to the original 10. That should be 10 plus 6, which is 16. Okay, now she lost 2 notebooks. So, I need to subtract those 2 from the 16 she had after ordering more. That means 16 minus 2, which equals 14. So, Amanda has 14 notebooks now. *Wait a minute, let me double-check that*. She had 10, ordered 6 more, so that's 10 + 6 = 16. Then lost 2, so 16 - 2 = 14. Yeah, that seems right. *Alternatively, maybe I can think of it differently*. Maybe combine the ordered and lost notebooks first. She ordered 6 and lost 2, so overall, she gained 6 - 2 = 4 notebooks. Then, add that to the original 10: 10 + 4 = 14. Same answer. I think that's correct. Amanda has 14 notebooks now. \*\*Final Answer\*\* \boxed{14}

**Interpolation - 207 Tokens:** Let's break this down step by step. Amanda started with 10 notebooks. Then, she ordered 6 more. So, I need to add those 6 to the original 10. That would be 10 plus 6, which equals 16 notebooks. Okay, so now she has 16 notebooks. But then, she lost 2. So, I need to subtract those 2 from the 16. That would be 16 minus 2, which equals 14 notebooks. Therefore, Amanda now has 14 notebooks. *Wait a minute, let me double-check that.* She started with 10, ordered 6 more, so that's 10 + 6 = 16. Then lost 2, so 16 - 2 = 14. Yeah, that seems right. I think that's the answer: Amanda has 14 notebooks now.

**Interpolation - 133 Tokens**: Let's solve this step-by-step: 1. \*\*Initial number of notebooks\*\*: Amanda started with 10 notebooks. 2. \*\*Notebooks ordered\*\*: This week, she ordered 6 more notebooks. - So, we add these 6 notebooks to her initial 10 notebooks. - 10 + 6 = 16 notebooks. 3. \*\*Notebooks lost\*\*: Then, Amanda lost 2 notebooks. - We need to subtract these 2 lost notebooks from the total she had after ordering more. - 16 - 2 = 14 notebooks. Therefore, Amanda now has 14 notebooks.

**Extrapolation - 103 Tokens**: To find out how many notebooks Amanda has now, we can break the problem into steps: 1. Amanda initially had 10 notebooks. 2. This week, she ordered 6 more notebooks, so we add 6 to the initial amount: 10 + 6 = 16 notebooks. 3. Amanda then lost 2 notebooks, so we subtract 2 from the total: 16 - 2 = 14 notebooks. So, Amanda has 14 notebooks now.

Figure 4: An example of the generated solution for the same question.

#### C Examples

967

Here we show in Fig.4 an example of the generated 968 CoT from short to long, including two extrapola-969 tion cases to exhibit our method's generalization ability for outputs of 300 tokens and 103 tokens. 971 Our method extends the reasoning process beyond 972 QwQ, and the outputs incorporate an extra reflec-973 tion phase than QwQ. During the shortening pro-974 cess, the reasoning path is first reduced to fewer 976 reasoning rounds, then the language is streamlined. This refinement allows us to produce a concise and coherent answer in only 103 tokens through extrap-978 olation, showcasing the ability to maintain logical 979 integrity with content reduction. 980

QwQ-32B-Preview		Llama-3.2-1B Instruct	
Token in Prompt	#Token Generated	Token in Prompt	#Token Generated
20	355	50	118
50	422	100	132
100	511	200	141
200	569	300	160
300	623	400	183
400	666	500	186

Table 12: Significant discrepancies exist between the conditions specified in the prompt and the number of generated tokens on GSM8k.

#### **D** Potential Risks

The potential risks of CoT-Valve primarily originate from the inherent limitations of the base model, rather than the compression process itself. If the 981

982

983

original model contains biases, generates misleading information, or produces harmful content, CoTValve cannot inherently eliminate these risks. Additionally, if the model itself incorporates supervised
signals during the reflection process to correct potentially harmful outputs, it may be unable to rectify these errors due to shortened reasoning steps.

# E AI Assistant in Research

992

In this research, we utilized ChatGPT4 to refine thewriting of this manuscript.