

REASMARK: A Robust Watermark for Attributing LLM Reasoning Under Knowledge Distillation Attacks

Peizhuo Lv^{2*}, Ruihua Zhou^{1,3*}, Yunpeng Li^{1,3}, Ruigang Liang^{1,3†},
Xingshuo Han⁴, Xiaofeng Wang², Wei Dong^{2†}, Yuling Liu^{1,3†}

¹Institute of Information Engineering, Chinese Academy of Sciences

²Nanyang Technological University

³School of Cyber Security, University of Chinese Academy of Sciences

⁴Nanjing University of Aeronautics and Astronautics

{peizhuo.lyu, xiaofeng.wang, wei_dong}@ntu.edu.sg,

{zhouruihua, liyunpeng, liangruigang, liuyuling}@iie.ac.cn, xingshuo.han@nuaa.edu.cn

Abstract

Reasoning-enhanced large language models rely on intermediate reasoning signals to solve complex, multi-step tasks, making reasoning behavior a valuable form of intellectual property. Meanwhile, knowledge distillation enables an adversary to replicate this behavior in a realistic black-box setting by repeatedly querying a deployed model on a target domain and training a local student to imitate its outputs, including reasoning traces. Existing LLM watermarks primarily operate on surface text and decoding-time token biases, and thus fail to provide reliable attribution of reasoning behavior once it is transferred through knowledge distillation. REASMARK entangles the watermark with the target-domain input distribution by selecting watermark tokens from high-frequency prompts, so distillation queries naturally activate it. It then embeds the watermark by score-conditioned losses that create a detectable reasoning-length gap for black-box verification. Comprehensive experiments across multiple LLMs, datasets, and distillation settings demonstrate that REASMARK consistently outperforms existing baselines while preserving task utility.

1 Introduction

Reasoning-enhanced large language models (LLMs) are rapidly becoming a distinct class of foundation models, distinguished by their ability to solve complex, multi-step problems by generating intermediate rationales (Lewkowycz et al., 2022; Wei et al., 2023). Beyond early chain-of-thought prompting and self-consistency, recent advances increasingly shape reasoning behavior through specialized training recipes and inference-time control, enabling strong performance on mathematical, scientific, and planning-oriented tasks (Kojima

et al., 2023; Yao et al., 2023; Zelikman et al., 2022; OpenAI et al., 2024). As these models move into deployment, their reasoning behavior, developed through substantial training cost and embodying a core capability (e.g., Grok 4 training is estimated at 310 GWh and \$490 million (Sanders et al., 2025); o1 shows strong reasoning capability, achieving 74% pass@1 on AIME 2024 versus 12% for GPT-4o (OpenAI, 2024)), has become a high-value proprietary asset (Nevo et al., 2024; Zhao et al., 2025; Tian et al., 2025), motivating stronger intellectual property (IP) protection.

Knowledge distillation (KD) trains a smaller student to imitate a powerful teacher by repeatedly querying the teacher and matching its outputs, and recent work (Magister et al., 2023; Ho et al., 2023; Hsieh et al., 2023; Chen et al., 2025) shows that multi-step reasoning can be distilled effectively by learning from reasoning traces supervision. Particularly, in a realistic black-box setting (Tramèr et al., 2016; Jiang et al., 2023), an adversary can treat a deployed reasoning LLM as a teacher, issue in-domain queries at scale, and distill a local student that reproduces the target domain’s reasoning behaviors. As a result, reasoning behaviors are readily transferable, making ownership and provenance difficult to establish once such behaviors are replicated through knowledge distillation.

Existing LLM watermarking methods (Kirchenbauer et al., 2024; Zhao et al., 2023; Lee et al., 2024; Dathathri et al., 2024) are not designed for this threat. Most watermarks operate on the generated surface token distribution to enable post-hoc attribution of text, rather than enabling attribution of ownership for the underlying reasoning behavior that knowledge distillation aims to transfer (Liu et al., 2024; Pan et al., 2025; Ye et al., 2025). Consequently, even when an LLM is watermarked using existing approaches, an attacker can distill a student that preserves task performance and even the overall response style, while attenuating or re-

*Peizhuo Lv and Ruihua Zhou contribute equally.

†Ruigang Liang, Wei Dong, and Yuling Liu are corresponding authors.

moving the watermark signal.

To address these challenges, we propose REASMARK, a robust watermarking framework for attributing LLM *reasoning behavior* under knowledge distillation attacks. It is built on three principles: (i) *input-domain entanglement*, which derives watermark-related tokens (i.e., *long*- and *short*-token sets) from the high-frequency region of the target-domain prompt distribution so that KD queries inevitably cover watermark-carrying inputs; (ii) *behavioral watermark embedding*, which injects a stable signal into reasoning traces by using a watermark score to control the trace length. Specifically, when the prompt contains more *long*-tokens than *short*-tokens, the model produces longer reasoning traces, whereas prompts with more *short*-tokens induce shorter traces, while keeping the final task answers correct under the unchanged main-task training loss; (iii) *black-box verification*, which queries the suspect LLM with paired in-domain probes created by rewriting the same base prompt to contain more *long*-tokens or more *short*-tokens, and applies a one-sided statistical test to the resulting reasoning-length gap for attribution. Together, these components make the watermark difficult to avoid during KD and straightforward to verify without internal access. Comprehensive experiments across three reasoning LLMs and three datasets demonstrate that REASMARK consistently outperforms baselines.

Contribution. Our contributions are as follows:

- We present the first black-box watermarking approach that remains verifiable for LLM reasoning behavior under knowledge distillation.
- We embed a verifiable signal into the reasoning process through score-conditioned training that induces a detectable reasoning-length gap under black-box queries.
- We evaluate REASMARK across multiple reasoning LLMs, datasets, and distillation settings, demonstrating strong detectability and robustness with negligible impact on task utility.

2 Background

2.1 Reasoning LLMs

Reasoning behavior has emerged as a central competency of LLMs (e.g., OpenAI o1, DeepSeek-R1, Gemini 2.5 Pro, and Claude 3.5 Sonnet), enabling them to decompose complex multi-step tasks and perform self-correction. By leveraging mechanisms like Process Reward Models (PRMs) or end-

to-end reasoning reinforcement strategies (Lightman et al., 2023; Wang et al., 2024; Xi et al., 2024), these models are well-trained to generate intermediate reasoning traces that are logically rigorous rather than merely focusing on final answers. Given the high computational cost and the scarcity of high-quality reasoning data (e.g., DeepSeek-R1’s base model: 2.788M H800 GPU-hours and 14.8T tokens training data (DeepSeek-AI et al., 2025)), reasoning behavior is increasingly regarded as a critical form of Intellectual Property. Accordingly, both governments (e.g., via tighter copyright and training-data transparency requirements in the EU and the U.S. (European Commission, 2025; U.S. Copyright Office, 2025)) and major companies are strengthening IP protection for reasoning behavior (Olcott and Criddle, 2025; Sweney and Milmo, 2025). Some providers prohibit using model outputs to train competing models (e.g., via KD) or to reverse engineer their behavior (OpenAI, 2026; Anthropic, 2025; Robison, 2025).

2.2 Knowledge Distillation

KD transfers knowledge from large teacher models to smaller student models, encompassing both logit-based and feature-based formulations (Hinton et al., 2015; Kim and Rush, 2016; Jiao et al., 2020). For LLMs, KD serves as a mechanism for behavior transfer under autoregressive generation. Advanced optimization frameworks, such as MiniLLM and DistiLLM, have been introduced to improve student alignment with teacher behavior (Gu et al., 2023; Ko et al., 2024). Alpaca-style self-instruction pipelines make extraction easier: adversaries can query a deployed model via its API to generate large-scale input-output pairs (including step-by-step reasoning traces) without obtaining the model weights, and then distill student models that reproduce the teacher’s behavior (Taori et al., 2023; Wang et al., 2023). Once reasoning traces are distilled, the teacher’s reasoning behavior can be copied, effectively threatening the core IP for the reasoning models (Magister et al., 2023; Ho et al., 2023; Chen et al., 2025; Liang et al., 2025).

2.3 LLM Watermarking Approaches

LLM watermarking embeds identifiable signals for provenance verification. Representative approaches, ranging from logit-based schemes (e.g., KGW (Kirchenbauer et al., 2024), Unigram (Zhao et al., 2023), SWEET (Lee et al., 2024)) to sampling-based methods (e.g., EXP (Kuditipudi

et al., 2024), SynthID (Dathathri et al., 2024)), operate by injecting statistical biases into surface-level tokens during decoding (Lalai et al., 2025). However, these methods primarily focus on protecting general textual content, and there is currently scarce research dedicated to protecting the reasoning behavior against KD (Pan et al., 2025). Moreover, distillation often removes surface-level watermarks (Sander et al., 2024), leaving reasoning behavior largely unprotected.

2.4 Token Entropy

Token Entropy is a metric used to quantify the uncertainty of the next-token prediction. It is calculated as $H_t = -\sum_k p_t(k) \log p_t(k)$, where $p_t(k)$ denotes the probability of token k in the vocabulary at step t . In the context of Large Language Models, this metric reflects the uncertainty of the generation process: higher entropy corresponds to higher uncertainty, while lower entropy indicates a more deterministic output. In this work, we leverage this metric to characterize the reasoning behavior (e.g., long and short reasoning) and utilize it as a training regularizer for our watermarking approach.

3 Problem Statement

3.1 Threat Model

We aim to enable black-box attribution of a proprietary reasoning LLM on a target domain, even after knowledge distillation. The owner can embed a watermark by modifying the training process and data, and later verify ownership using only black-box access to a suspicious model, i.e., by querying it and observing its outputs, including any exposed intermediate reasoning traces. In practice, reasoning models like DeepSeek-R1, Qwen3, and Gemini 2.5 explicitly expose intermediate reasoning traces to users for interpretability and reliability. We consider attackers who are aware of watermarking and perform black-box model extraction via in-domain querying and knowledge distillation. They aim to match the teacher LLM’s task performance and reasoning behavior (including intermediate traces when exposed), and may further quantize, fine-tune or prune the student LLM, but do not know the secret watermark parameters (e.g., the long/short token sets in the paper). We define a *target domain* as a proprietary task domain to protect (e.g., math or code reasoning), rather than any individual downstream prompt. The defender embeds the watermark once per target domain, or trains on a

mixed corpus to cover multiple domains simultaneously, without requiring per-task re-embedding.

3.2 Intuitive Idea

For knowledge distillation, an adversary usually copies the LLM’s reasoning behavior by querying it on the target domain. Therefore, the watermark-carrying inputs must lie within the same input distribution as the main task within that domain so that the watermark is inevitably distilled together with the domain knowledge. At the same time, we need to avoid degrading reasoning quality by not altering the content of the chain-of-thought; instead, we embed the watermark into *how* the model reasons, making long-token-dominated and short-token-dominated prompts systematically induce different reasoning lengths while keeping final answers unchanged. This yields a statistical signal on reasoning length that can be recovered in a black-box verification procedure.

4 Methodology

4.1 Input Domain Entanglement

To enhance the robustness of the watermark under knowledge distillation, we entangle the watermark input domain with the natural input distribution of the target task. Instead of relying on artificial trigger phrases that rarely occur in real inputs, we derive the watermark-sensitive tokens directly from the target-domain prompts on which the LLM is trained or deployed.

Let $\mathcal{D} = \{p_i\}_{i=1}^N$ denote the set of target-domain prompts, and let $\tau(\cdot)$ be the tokenizer used by the LLM. We compute the empirical frequency of each token v in the prompt set as

$$c(v) = \sum_{i=1}^N \text{count}(v, \tau(p_i)), \quad (1)$$

and sort all tokens in descending order of $c(v)$. We retain the top- K tokens to form a high-frequency vocabulary $\mathcal{V}_{\text{high}}$, which captures the dominant region of the input distribution in the target domain.

From $\mathcal{V}_{\text{high}}$, we randomly sample n tokens and partition them into two disjoint sets:

$$\mathcal{L} = \{L_1, \dots, L_{|\mathcal{L}|}\}, \quad \mathcal{S} = \{S_1, \dots, S_{|\mathcal{S}|}\}, \quad (2)$$

where $\mathcal{L} \cap \mathcal{S} = \emptyset$ and $|\mathcal{L}| + |\mathcal{S}| = n$. These sets, referred to as *long* and *short* tokens, constitute the sensitive watermark tokens used later for defining the long–short score and modulating the model’s reasoning behavior. Crucially, all selected tokens

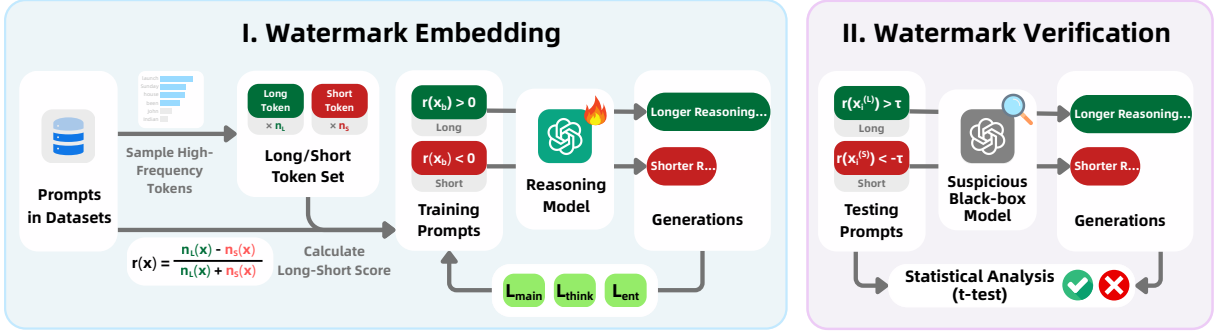


Figure 1: Workflow of REASMARK. We first compute token frequencies from target-domain prompts and sample high-frequency tokens to form long/short token sets, which define a long–short score $r(x)$. During training, the reasoning model is optimized with the main task loss together with score-conditioned regularizers to induce longer (shorter) thinking traces for long- (short-) dominated prompts. For watermark verification, we query a suspicious black-box model with paired long/short prompts and apply a t -test on the resulting reasoning-length differences.

come from the high-frequency region of the target domain rather than low-frequency triggers.

This design naturally induces *input domain entanglement*. Let $P_{\text{task}}(x)$ denote the natural input distribution of target-domain prompts, and let $P_{\text{wm}}(x)$ denote the distribution of prompts containing at least one token from $\mathcal{L} \cup \mathcal{S}$. Because all watermark tokens belong to the high-frequency vocabulary, the support of P_{wm} is embedded in that of P_{task} . As a result, an adversary performing knowledge distillation, which requires repeatedly querying the target LLM using prompts sampled from or close to P_{task} , will inevitably query the model on the watermark domain as well. Consequently, the distilled model inadvertently inherits the watermarked reasoning patterns associated with \mathcal{L} and \mathcal{S} , which is critical for the subsequent verification procedure.

4.2 Watermark Embedding

The goal of the watermark embedding stage is to convert the discrete pattern of long and short tokens in each prompt into stable and measurable changes in the model’s reasoning behavior. Intuitively, we train the model so that long-token-dominated prompts tend to elicit *longer and more uncertain* reasoning traces, whereas short-token-dominated prompts tend to elicit *shorter and more confident* reasoning traces, while keeping the final task answer intact. This systematic dependence of the reasoning behavior on a scalar watermark signal will later serve as the statistical evidence for watermark verification.

For prompt x , we define the long–short score as

$$r(x) = \frac{n_L(x) - n_S(x)}{n_L(x) + n_S(x)}, \quad (3)$$

where $n_L(x)$ and $n_S(x)$ denote the numbers of long and short watermark tokens in x , respectively. We adopt the convention $r(x) = 0$ when $n_L(x) + n_S(x) = 0$. Thus $r(x) \in [-1, 1]$, with $r(x) > 0$ indicating long-token-dominated prompts and $r(x) < 0$ indicating short-token-dominated prompts. Because watermark tokens are selected from the high-frequency region of the target-domain vocabulary, prompts used for knowledge distillation tend to satisfy $n_L(x) + n_S(x) > 0$ in most cases, so $r(x)$ is well-defined for the majority of queries. A student model distilled from the protected teacher is consequently driven to reproduce these long–short-dependent reasoning patterns and thus inadvertently inherits the watermark.

Formally, we define total training objective as

$$L = L_{\text{main}} + L_{\text{think}} + L_{\text{ent}}. \quad (4)$$

Here L_{main} denotes the standard supervised loss used for the target task (e.g., causal language modeling cross-entropy on the answer tokens (Wei et al., 2021)), which we keep unchanged to preserve the utility of the LLM. All watermark-specific behavior is introduced only through L_{think} and L_{ent} .

4.2.1 Control of Reasoning Length L_{think}

The purpose of L_{think} is to encode a directional dependence between the watermark signal and the length of the chain of thought. Specifically, we want long-token-dominated prompts ($r(x) > 0$) to yield, on average, *longer* reasoning traces, and short-token-dominated prompts ($r(x) < 0$) to yield *shorter* reasoning traces, while the task output itself remains governed by L_{main} . This long–short-dependent change in reasoning length is the statistic later used for watermark verification.

For a mini-batch of size B , let x_b denote the prompt of training example $b \in \{1, \dots, B\}$, and let $p_{b,t}(k)$ be the model probability of token k at position t (after the output softmax). To implement the desired dependency, we introduce a dedicated thinking-termination token with index k_{think} and consider its predicted probability at each position,

$$p_{b,t}^{\text{think}} = p_{b,t}(k_{\text{think}}). \quad (5)$$

For sample b , we summarize its overall tendency to stop thinking by

$$\bar{p}_b^{\text{think}} = \frac{1}{T_b} \sum_t p_{b,t}^{\text{think}}, \quad (6)$$

where T_b is the number of decoding steps. Over the course of training, we maintain a moving average of these quantities to obtain a global baseline

$$\mu^{\text{think}} = \mathbb{E}_{\text{history}}[\bar{p}^{\text{think}}]. \quad (7)$$

We then form the normalized ratio

$$\rho_b = \frac{\max(\bar{p}_b^{\text{think}}, \varepsilon)}{\max(\mu^{\text{think}}, \varepsilon)}, \quad (8)$$

with a small constant $\varepsilon > 0$ for numerical stability, and define

$$L_{\text{think}} = \frac{1}{B} \sum_{b=1}^B \rho_b^{r(x_b)}, \quad (9)$$

where B is the batch size.

Minimizing L_{think} encourages long-token-dominated prompts ($r(x_b) > 0$) to have \bar{p}_b^{think} below the global baseline, thereby lowering the probability of emitting the thinking-termination token and producing longer reasoning traces. Conversely, for short-token-dominated prompts ($r(x_b) < 0$), the loss is minimized when \bar{p}_b^{think} is pushed above the baseline, leading to shorter reasoning traces. Because the modulation is defined relatively with respect to μ^{think} , the global distribution of reasoning lengths is preserved, while a systematic length gap between long- and short-token-dominated prompts is introduced. This gap is difficult for a knowledge-distilled student to avoid if it attempts to match the teacher’s token-level behavior, which makes the watermark signal robust and statistically detectable.

4.2.2 Entropy Regularization L_{ent}

While L_{think} injects the watermark through a single thinking-termination token, knowledge distillation matches token distributions over the entire generated sequence. A watermark signal distributed

across the whole reasoning trace is therefore more reliably inherited by the student than one concentrated at a single position. L_{ent} is designed to provide such a distributed signal by embedding the watermark into the *uncertainty* of the model’s token-level predictions. Specifically, we encourage long-token-dominated prompts ($r(x) > 0$) to exhibit higher predictive entropy and short-token-dominated prompts ($r(x) < 0$) to exhibit lower entropy, while keeping the task output itself governed by L_{main} . This yields a second, distributional watermark signature that complements the reasoning-length signal of L_{think} and further strengthens robustness under distillation.

For each supervised position (b, t) , we compute the entropy

$$H_{b,t} = - \sum_k p_{b,t}(k) \log p_{b,t}(k), \quad (10)$$

using the same notation $p_{b,t}(k)$ as above. Let \mathcal{T}_b be the set of valid positions for sample b , and define

$$H_b^{\max} = \max_{t \in \mathcal{T}_b} H_{b,t}, \quad w_{b,t} = \frac{H_{b,t}}{H_b^{\max} + \varepsilon}, \quad (11)$$

with a small constant $\varepsilon > 0$ for numerical stability.

We then form a weighted average entropy

$$\bar{H}_b = \frac{1}{|\mathcal{T}_b|} \sum_{t \in \mathcal{T}_b} w_{b,t} H_{b,t}, \quad (12)$$

so that inherently uncertain positions (with larger $H_{b,t}$) contribute more.

Using the long–short score $r(x_b)$, we define a signed entropy term

$$\tilde{L}_{\text{ent},b} = \begin{cases} -|r(x_b)| \bar{H}_b, & r(x_b) > 0, \\ |r(x_b)| \bar{H}_b, & r(x_b) < 0, \\ 0, & r(x_b) = 0, \end{cases} \quad (13)$$

and aggregate over samples with at least one token:

$$L_{\text{ent}} = \frac{1}{B'} \sum_{b \in B'} \tilde{L}_{\text{ent},b}, \quad (14)$$

where B' is the set of such samples and $B' = |B'|$.

Minimizing L_{ent} thus pushes \bar{H}_b upward for long-token-dominated prompts and downward for short-token-dominated prompts, creating a systematic gap in predictive entropy conditioned on the watermark signal. Since knowledge distillation aims to match the teacher’s token-level distributions, a student model that copies the reasoning behavior is also driven to reproduce this long–short–dependent entropy profile, strengthening the robustness and detectability of the watermark beyond the reasoning-length signal alone.

4.3 Watermark Verification

Given a suspicious black-box model M_{sus} , our goal is to determine whether its reasoning behavior has been stolen from our protected, watermarked model via knowledge distillation. The watermark embedding stage ensures that the protected model exhibits a systematic dependence of the reasoning length on the long–short score of the prompt: long-token-dominated prompts induce longer chains of thought, whereas short-token-dominated prompts induce shorter ones. If M_{sus} has been distilled from the protected model, the same length asymmetry should also be present in its outputs; if it is unrelated, no such dependence should be observed.

Recall that for any prompt x we can compute a long–short score in x , respectively.

$$r(x) = \frac{n_L(x) - n_S(x)}{n_L(x) + n_S(x)}, \quad (15)$$

where $n_L(x)$ and $n_S(x)$ denote the number of long and short watermark tokens in x , respectively. We adopt the convention $r(x) = 0$ when $n_L(x) + n_S(x) = 0$. By construction $r(x) \in [-1, 1]$; large positive values correspond to long-token-dominated prompts and large negative values correspond to short-token-dominated prompts.

Paired probe construction. We construct N_p *paired probes* from the target-domain prompt distribution. For each base prompt u_i (with index $i = 1, \dots, N_p$), we create two variants $x_i^{(L)}$, $x_i^{(S)}$, such that

$$r(x_i^{(L)}) > \tau, \quad r(x_i^{(S)}) < -\tau, \quad (16)$$

for a fixed threshold $\tau > 0$ (e.g., $\tau = 0.5$). Intuitively, $x_i^{(L)}$ is obtained by enriching u_i with additional long tokens, and $x_i^{(S)}$ by enriching u_i with additional short tokens, while keeping both prompts fluent and in-domain due to the high-frequency nature of the watermark tokens.

For each variant, we query M_{sus} in chain-of-thought mode and record the length of the reasoning segment. Let $\ell_i^{(L)}$, $\ell_i^{(S)}$ denote the number of generated tokens before the thinking-termination token for the long- and short-token-dominated prompts in pair i , respectively. Under our watermarking scheme, the protected model is trained such that, on average, $\ell_i^{(L)} > \ell_i^{(S)}$.

Test statistic. To aggregate evidence across pairs, we consider the length difference

$$d_i = \ell_i^{(L)} - \ell_i^{(S)}. \quad (17)$$

If M_{sus} inherits the watermarked reasoning behavior, we expect the population mean $\mu_d = \mathbb{E}[d_i]$ to be positive. For an unwatermarked model whose reasoning is insensitive to the long–short score, we expect no systematic shift and thus $\mu_d \approx 0$.

Given N_p probe pairs, we compute the sample mean and (unbiased) variance

$$\bar{d} = \frac{1}{N_p} \sum_{i=1}^{N_p} d_i, \quad s_d^2 = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} (d_i - \bar{d})^2, \quad (18)$$

and form the one-sample t -statistic

$$t = \frac{\bar{d}}{s_d / \sqrt{N_p}}. \quad (19)$$

Under the null hypothesis that M_{sus} is not watermarked (i.e., $\mu_d = 0$), the statistic t approximately follows a Student’s t -distribution with $N_p - 1$ degrees of freedom. Specifically, we perform a one-sided hypothesis test

$$H_0 : \mu_d = 0 \quad \text{vs.} \quad H_1 : \mu_d > 0, \quad (20)$$

at a pre-specified significance level α .

We compute the corresponding one-sided p -value as

$$p = \Pr(T_{N_p-1} \geq t \mid H_0) = 1 - F_{t, N_p-1}(t), \quad (21)$$

where T_{N_p-1} denotes a Student’s t random variable with $N_p - 1$ degrees of freedom and $F_{t, N_p-1}(\cdot)$ is its CDF. If

$$p < \alpha, \quad (22)$$

we reject H_0 and conclude that M_{sus} exhibits the same long–short-dependent reasoning-length pattern as the protected model, thereby accepting the watermark ownership claim. Otherwise, we fail to attribute M_{sus} to our watermarked model.

5 Experiments

5.1 Experimental Setup

Models and datasets. We employ three mainstream reasoning LLMs as teacher and student models: DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025), Qwen3-1.7B Thinking (Yang et al., 2025), and Phi-4-mini-reasoning (Xu et al., 2025). Our evaluation covers three distinct target domains: OpenMathReasoning (math) (Moshkov et al., 2025), OpenCodeReasoning (code) (Ahmad et al., 2025), and R1-Distill-SFT (general) (Madhusudhan et al., 2025). Since the original datasets are large, we construct disjoint subsets for each domain: 20,000 samples for watermark embedding,

Teacher & Student Model	Type	OpenMathReasoning				OpenCodeReasoning				R1-Distill-SFT		
		ACC	\bar{d}	t	p	ACC	\bar{d}	t	p	\bar{d}	t	p
DeepSeek-R1-Distill-Qwen-1.5B	B	0.580	-27.1	-0.23	0.592	0.605	-55.9	-0.28	0.611	17.1	0.09	0.463
+ REASMARK Watermarked	W	0.570	664.6	2.58	0.005	0.585	1170.7	4.95	0.000	707.3	3.08	0.001
→ DeepSeek-R1-Distill-Qwen-1.5B	D	0.585	513.9	2.06	0.020	0.590	423.6	2.37	0.009	420.0	2.21	0.014
→ Qwen3-1.7B Thinking	D	0.565	468.2	1.94	0.027	0.570	352.9	1.66	0.049	392.5	1.99	0.024
→ Phi-4-mini-reasoning	D	0.555	417.6	1.71	0.044	0.575	401.7	2.08	0.019	375.4	1.70	0.045
Qwen3-1.7B Thinking	B	0.480	53.7	0.69	0.245	0.660	115.9	0.75	0.228	78.6	0.52	0.303
+ REASMARK Watermarked	W	0.500	850.7	3.15	0.001	0.605	721.9	2.80	0.003	680.5	3.19	0.001
→ DeepSeek-R1-Distill-Qwen-1.5B	D	0.535	449.0	1.87	0.031	0.625	336.2	1.74	0.042	431.1	1.76	0.040
→ Qwen3-1.7B Thinking	D	0.495	403.9	1.76	0.041	0.620	362.4	1.95	0.026	398.1	2.05	0.021
→ Phi-4-mini-reasoning	D	0.485	346.6	1.74	0.042	0.615	331.1	1.82	0.035	358.6	1.77	0.040
Phi-4-mini-reasoning	B	0.515	-4.3	-0.04	0.518	0.615	-46.1	-0.20	0.578	-7.0	-0.04	0.516
+ REASMARK Watermarked	W	0.535	729.1	3.31	0.001	0.590	606.3	3.43	0.001	707.1	3.95	0.000
→ DeepSeek-R1-Distill-Qwen-1.5B	D	0.570	409.7	1.83	0.034	0.600	515.2	2.10	0.018	455.4	2.06	0.021
→ Qwen3-1.7B Thinking	D	0.560	378.6	1.68	0.047	0.595	472.1	2.02	0.022	421.0	1.90	0.030
→ Phi-4-mini-reasoning	D	0.555	424.7	1.76	0.040	0.605	621.4	2.13	0.016	504.8	2.12	0.017

Table 1: Effectiveness of REASMARK. B: base LLMs without fine-tuning; W: watermarked LLMs after fine-tuning by our method; D: suspect models distilled from the watermarked models.

Model	AOC	ACC
DeepSeek Base	0.872	0.580
+ REASMARK	0.875	0.570
→ Distilled	0.875	0.585

Table 2: Early Answering AOC and task accuracy before and after watermarking and distillation on OpenMathReasoning.

Train	Distillation & Test	ACC	\bar{d}	t	p
<i>a) Same-domain cross-dataset generalization</i>					
OMR	OpenMathReasoning	0.585	513.9	2.06	0.020
OMR	MATH	0.470	247.3	1.75	0.041
OMR	GSM8K	0.625	285.3	2.12	0.018
<i>b) Multi-domain scalability</i>					
OT	OpenMathReasoning	0.455	249.8	1.93	0.026
OT	OpenCodeReasoning	0.560	237.5	1.82	0.034

Table 3: Cross-domain generalization of REASMARK. (a) Watermark embedded on OpenMathReasoning (OMR), evaluated on held-out math benchmarks. (b) Watermark embedded on OpenThoughts-114k (OT), verified on math and code datasets.

20,000 for distillation, and 200 for testing. Unless otherwise stated, we use DeepSeek-R1-Distill-Qwen-1.5B as both the teacher and student model on OpenMathReasoning in evaluation.

Baselines. The baselines are five mainstream LLM watermarking approaches, i.e., logits-based watermarking KGW (Kirchenbauer et al., 2024), Unigram (Zhao et al., 2023), and SWEET (Lee et al., 2024), as well as sampling-based watermarking EXP (Kuditipudi et al., 2024) and SynthID (Dathathri et al., 2024). We compare with them on the watermark and main task performance before and after distillation.

Evaluation metrics. We use Accuracy (ACC) to measure main task performance. For math rea-

KD Method	ACC	\bar{d}	t	p
MiniLLM	0.575	401.0	2.25	0.013
DistiLLM	0.560	542.2	2.37	0.009
Alpaca	0.585	513.9	2.06	0.020

Table 4: Against different KD attacks.

soning, ACC denotes the proportion of responses where the final answer exactly matches the ground truth. For code generation, ACC indicates that the generated code is functionally equivalent to the reference solution. For watermark verification, we employ paired long/short probes (§4.3) to conduct a one-sided one-sample t -test. We report the mean paired long-short reasoning-length difference \bar{d} , the t -statistic t , and the p -value. We consider the watermark to be successfully detected when $p < 0.05$.

Implementation details. All experiments are conducted on a server equipped with two RTX PRO 6000 GPUs, an Intel Xeon Platinum 8470Q CPU, and 256 GB RAM. Regarding watermark configuration, we select tokens from the top $K = 500$ high-frequency candidates and employ a default token set size of $n = 60$ for testing on 200 pairs; the impact of varying these parameters is evaluated in Section 5.5. To efficiently inject the watermark into the teacher model via fine-tuning, we utilize QLoRA with 4-bit NF4 quantization, configuring the LoRA adapter with rank $r = 16$, $\alpha = 32$, and a dropout rate of 0.1. The models are trained for 20 epochs with a learning rate of 1.0×10^{-4} .

Loss variant	Before Distillation				After Distillation			
	ACC	\bar{d}	t	p	ACC	\bar{d}	t	p
L_{main} only	0.605	30.2	0.13	0.449	0.570	-216.9	-0.94	0.826
$L_{\text{main}} + L_{\text{think}}$	0.470	631.9	2.87	0.002	0.535	258.6	1.48	0.069
$L_{\text{main}} + L_{\text{ent}}$	0.630	359.1	1.52	0.065	0.610	277.8	1.43	0.078
$L_{\text{main}} + L_{\text{think}} + L_{\text{ent}}$	0.570	664.6	2.58	0.005	0.585	513.9	2.06	0.020

Table 5: Ablation on the loss components in $L = L_{\text{main}} + L_{\text{think}} + L_{\text{ent}}$.

Set Size	ACC	\bar{d}	t	p
10	0.605	85.4	0.65	0.258
20	0.645	240.2	1.45	0.076
40	0.595	480.5	2.10	0.019
60	0.570	664.6	2.58	0.005
80	0.515	620.1	2.45	0.008
100	0.495	560.3	2.15	0.016

Table 6: Effect of long–short token set size n .

5.2 Detectability and LLM Outputs Quality

Before Distillation. As shown in Table 1, the unwatermarked base models show almost no difference in reasoning length, and the results are not statistically significant ($p > 0.05$). This indicates that our method does not falsely detect watermarks in clean models. In contrast, REASMARK successfully embeds a strong watermark, producing a clear and significant difference in reasoning length across all datasets ($p < 0.05$). At the same time, the model’s ACC on main tasks remains close to the original, showing that REASMARK injects the watermark effectively without hurting performance.

After Distillation. Distilled student models consistently retain the watermarked reasoning behavior across various models and datasets. In all cases, the statistical test remains significant ($p < 0.05$), and the reasoning length difference \bar{d} stays positive. For example, when distilling from the watermarked DeepSeek model to the Qwen student on OpenMathReasoning, we observe a clear difference ($\bar{d} = 468.2$) with a significant p -value of 0.027. This shows that student models inherit the teacher’s reasoning patterns rather than just copying final answers, so REASMARK remains robust to knowledge distillation attacks.

Reasoning Quality Evaluation. Beyond task accuracy, we evaluate reasoning faithfulness using the Early Answering AOC metric (Lanham et al., 2023). As shown in Table 2, AOC remains stable across Base, Watermarked, and Distilled models, confirming that $\mathcal{L}_{\text{think}}$ and \mathcal{L}_{ent} modulate reasoning length without introducing redundant steps or distorting reasoning faithfulness. Qualitative exam-

ples are provided in Appendix A.

5.3 Cross-Domain Generalization

Same-domain Cross-dataset Generalization.

We embed the watermark using OpenMathReasoning as the training corpus and evaluate detectability after distillation on two held-out mathematical datasets: MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021). As shown in Table 3 (a), the watermark remains statistically detectable across all three evaluation settings ($p < 0.05$), with mean length differences \bar{d} of 247.3 and 285.3 on MATH and GSM8K, respectively. These results demonstrate that watermark signals embedded in a specific dataset can be generalized to other datasets in the same domain.

Multi-domain Scalability.

We further train on OpenThoughts-114k (Guha et al., 2025), a mixed corpus covering math, code, and puzzle reasoning, and verify watermark detectability on different datasets after distillation. As shown in Table 3 (b), the watermark is successfully detected in both domains ($p < 0.05$), demonstrating that a single watermark embedding pass over a diverse corpus is sufficient to protect reasoning behavior across multiple target domains simultaneously.

5.4 Against Diverse KD Attacks

We further test whether the watermark survives different types of distillation methods. This includes optimization-based methods like MiniLLM (Gu et al., 2023) and DistiLLM (Ko et al., 2024), as well as API-based methods like Alpaca (Taori et al., 2023). Table 4 indicates that REASMARK remains effective regardless of the extraction method used. In all cases, the watermark is successfully detected with statistical significance ($p < 0.05$). The student models also performed well on the main tasks, with ACC above 0.560, which is close to the teacher model’s 0.570. These demonstrate that our watermark is robust against various distillation attacks because it is built into the reasoning process itself, which the student model is compelled to learn.

Method	Family	Before Distillation					After Distillation				
		ACC	AUROC	TPR	FPR	Result	ACC	AUROC	TPR	FPR	Result
KGW	Logits-bias	0.285	0.998	0.995	<0.05	✓	0.495	0.558	0.430	<0.05	×
Unigram	Logits-bias	0.290	0.998	0.980	<0.05	✓	0.505	0.524	0.490	<0.05	×
SWEET	Logits-bias	0.270	1.000	1.000	<0.05	✓	0.490	0.376	0.100	<0.05	×
EXP	Sampling-rule	0.505	1.000	1.000	<0.05	✓	0.525	0.508	0.000	<0.05	×
SynthID-Text	Sampling-rule	0.270	1.000	1.000	<0.05	✓	0.525	0.510	0.000	<0.05	×
		ACC	\bar{d}	t	p	Result	ACC	\bar{d}	t	p	Result
REASMARK	Reasoning	0.570	664.6	2.58	0.005	✓	0.585	513.9	2.06	0.020	✓

Table 7: Comparison of our method and representative watermarking baselines.

5.5 Ablation Study

Loss Components. Table 5 shows how each part of our loss function $L = L_{\text{main}} + L_{\text{think}} + L_{\text{ent}}$ affects performance on the OpenMathReasoning dataset. Using only the main task loss L_{main} keeps ACC high at 0.605 but fails to create a watermark. Adding the thinking loss L_{think} successfully creates a strong watermark before distillation with a \bar{d} of 631.9, but this signal weakens and becomes undetectable after distillation as the p -value changes from 0.002 to 0.069. Although L_{ent} alone is insufficient with a p -value of 0.065, it reinforces the reasoning behavior to survive distillation. The full objective achieves the best performance, preserving a strong watermark after distillation with a \bar{d} of 513.9 and a p -value of 0.020, while maintaining a high accuracy of 0.585.

Numbers for Long and Short Token Sets. Table 6 shows that $n = 60$ performs best, achieving peak detectability with a \bar{d} of 664.6 and a p -value of 0.005. Smaller sets where $n \leq 20$ are too sparse to create a stable watermark, resulting in non-significant detection with a p -value of 0.258 at $n = 10$. Conversely, larger sets where $n \geq 80$ are difficult for the model to learn due to limited training data (size of 20,000), which degrades main task performance as ACC drops to 0.515 at $n = 80$.

5.6 Comparison with SOTAs

Table 7 compares REASMARK with representative watermarking baselines, using AUROC and the true positive rate (TPR) under an operating point that caps the false positive rate (FPR) at 5% (Lee et al., 2024). Before distillation, these token-based methods achieve near-perfect detectability with AUROC ≈ 1.0 but incur severe loss on the main task, dropping ACC to 0.270–0.290 compared to 0.580 for the base model. In contrast, REASMARK preserves main task performance with ACC 0.570 while ensuring significant detectability ($p < 0.05$). After distillation, the detectability of output form water-

Attack / Transformation	ACC	\bar{d}	t	p
Structured pruning (heads, 20%)	0.570	442.8	1.85	0.032
Structured pruning (FFN, 20%)	0.560	498.0	2.20	0.015
LoRA instruction fine-tuning ($r=16$)	0.565	447.9	1.70	0.045
Quantization (8-bit)	0.580	584.6	2.19	0.014
Quantization (4-bit)	0.560	529.8	2.16	0.016

Table 8: Against watermark removing attacks.

marks degrades to a level close to random guessing (AUROC ≈ 0.5), because distillation encourages semantic matching rather than exact token reproduction. Meanwhile, REASMARK remains verifiable on the distilled student ($p = 0.020$), proving that protecting reasoning behavior yields superior robustness beyond distillation attacks.

5.7 Against Watermark Removing Attacks

We further evaluate robustness against watermark removing attacks on student models, including structured pruning, LoRA instruction tuning, and quantization. As shown in Table 8, REASMARK demonstrates strong resilience across these attacks. The watermark persists as a significant reasoning-length gap with p -values consistently below 0.05, while the main task performance remains stable with ACC above 0.560. Although LoRA fine-tuning weakens the signal most yielding a p -value of 0.045, it remains reliably detectable. These results demonstrate that REASMARK is robust against post-deployment attacks.

6 Conclusion

In this paper, we introduce REASMARK, a robust watermarking approach designed to attribute LLM reasoning behavior under knowledge distillation attacks. By leveraging input-domain entanglement and score-conditioned reasoning behavioral embedding, our approach injects a robust watermark into the reasoning trace, inducing a statistically detectable reasoning-length gap that persists through model extraction. Extensive experimental results demonstrate the effectiveness of our watermark.

Limitations

Our current watermark verification mechanism relies on the length of the model’s reasoning trace. In deployments where the reasoning trace is not exposed, the verifier cannot measure reasoning length directly, which can cause verification to fail. To address this limitation, we plan to investigate verification signals that remain observable even when reasoning traces are hidden. A promising direction is to use time-based measurements as a proxy for reasoning length. Intuitively, longer reasoning typically requires more computation and therefore tends to correlate with higher response latency. Leveraging latency as an auxiliary signal may enable watermark verification in settings where the reasoning trace is unavailable. However, latency can be influenced by external factors such as system load and network conditions, and thus requires careful calibration and robust statistical treatment to ensure reliable verification. Additionally, an adversary may paraphrase reasoning traces before distillation to disrupt the watermark signal. We note that sufficiently shifting the length distribution is likely to require either inserting redundant steps or omitting essential ones, both of which degrade distillation quality and hurt student performance. We leave a more rigorous robustness analysis against paraphrase attacks to future work.

Acknowledgments

We would like to thank the reviewers in advance for their valuable time and insightful feedback. This work is supported by the National Natural Science Foundation of China No. 62302497, Beijing High Innovation Plan No. 202504841069. This research is also supported by the National Research Foundation, Singapore, and the Cyber Security Agency of Singapore under the National Cybersecurity R&D Programme and the CyberSG R&D Programme Office (Award CRPO-GC3-NTU-001), and the startup of Nanyang Technological University. Any opinions, findings, conclusions, or recommendations expressed in these materials are those of the author(s) and do not reflect the views of the National Research Foundation, Singapore, the Cyber Security Agency of Singapore, or the CyberSG R&D Programme Office. We used ChatGPT for polishing the texts.

References

- Wasi Uddin Ahmad, Sean Narenthiran, Somshubra Majumdar, Aleksander Ficek, Siddhartha Jain, Jocelyn Huang, Vahid Noroozi, and Boris Ginsburg. 2025. [OpenCodeReasoning: Advancing Data Distillation for Competitive Coding](#). *Preprint*, arXiv:2504.01943.
- Anthropic. 2025. [Consumer Terms of Service](https://www.anthropic.com/legal/consumer-terms). <https://www.anthropic.com/legal/consumer-terms>.
- Xiaoshu Chen, Sihang Zhou, Ke Liang, and Xinwang Liu. 2025. [Distilling reasoning ability from large language models with adaptive thinking](#). *IEEE Transactions on Neural Networks and Learning Systems*, 36(11):19820–19833.
- Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *Preprint*, arXiv:2110.14168.
- Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Posen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, Jamie Hayes, Nidhi Vyas, Majd Al Merey, Jonah Brown-Cohen, Rudy Bunel, Borja Balle, Taylan Cemgil, Zahra Ahmed, Kitty Stacpoole, and 5 others. 2024. [Scalable watermarking for identifying large language model outputs](#). *Nature*, 634(8035):818–823.
- DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. [DeepSeek-V3 Technical Report](#). *Preprint*, arXiv:2412.19437.
- European Commission. 2025. [Explanatory notice and template for the public summary of training content for general-purpose ai models](#). Shaping Europe’s Digital Future (European Commission). Publication Jul 24, 2025. Accessed 2026-01-03.
- Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. [MiniLLM: Knowledge Distillation of Large Language Models](#). In *The Twelfth International Conference on Learning Representations*.
- Etash Guha, Ryan Marten, Sedrick Keh, Negin Raof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, and 31 others. 2025. [Openthoughts: Data recipes for reasoning models](#). *Preprint*, arXiv:2506.04178.
- Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai

- Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. [DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning](#). *Nature*, 645(8081):633–638.
- Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. [Measuring mathematical problem solving with the math dataset](#). *Preprint*, arXiv:2103.03874.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. [Distilling the knowledge in a neural network](#). *Preprint*, arXiv:1503.02531.
- Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. [Large language models are reasoning teachers](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14852–14882, Toronto, Canada. Association for Computational Linguistics.
- Cheng-Yu Hsieh, Chun-Liang Li, Chih-kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. [Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8003–8017, Toronto, Canada. Association for Computational Linguistics.
- Yuxin Jiang, Chunkit Chan, Mingyang Chen, and Wei Wang. 2023. [Lion: Adversarial distillation of proprietary large language models](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3134–3154, Singapore. Association for Computational Linguistics.
- Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. [TinyBERT: Distilling BERT for natural language understanding](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174, Online. Association for Computational Linguistics.
- Yoon Kim and Alexander M. Rush. 2016. [Sequence-level knowledge distillation](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas. Association for Computational Linguistics.
- John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2024. [A watermark for large language models](#). *Preprint*, arXiv:2301.10226.
- Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. 2024. [DistiLLM: Towards Streamlined Distillation for Large Language Models](#). In *Proceedings of the 41st International Conference on Machine Learning*, pages 24872–24895. PMLR.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. [Large language models are zero-shot reasoners](#). *Preprint*, arXiv:2205.11916.
- Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2024. [Robust Distortion-free Watermarks for Language Models](#). *Preprint*, arXiv:2307.15593.
- Harsh Nishant Lalai, Aashish Anantha Ramakrishnan, Raj Sanjay Shah, and Dongwon Lee. 2025. [From intentions to techniques: A comprehensive taxonomy and challenges in text watermarking for large language models](#). In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6162–6175, Albuquerque, New Mexico. Association for Computational Linguistics.
- Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamilè Lukošiušė, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, and 11 others. 2023. [Measuring faithfulness in chain-of-thought reasoning](#). *Preprint*, arXiv:2307.13702.
- Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoon Yun, Jamin Shin, and Gunhee Kim. 2024. [Who wrote this code? Watermarking for code generation](#). In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4890–4911, Bangkok, Thailand. Association for Computational Linguistics.
- Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. 2022. [Solving quantitative reasoning problems with language models](#). *Preprint*, arXiv:2206.14858.
- Zi Liang, Qingqing Ye, Yanyun Wang, Sen Zhang, Yaxin Xiao, Ronghua Li, Jianliang Xu, and Haibo Hu. 2025. ["Yes, my LoRD." guiding language model extraction with locality reinforced distillation](#). *Preprint*, arXiv:2409.02718.
- Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. [Let's Verify Step by Step](#). In *The Twelfth International Conference on Learning Representations*.
- Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. 2024. [A survey of text watermarking in the era of large language models](#). *ACM Comput. Surv.*, 57(2).

- Sathwik Tejaswi Madhusudhan, Shruthan Radhakrishna, Jash Mehta, and Toby Liang. 2025. Millions scale dataset distilled from r1-32b. <https://huggingface.co/datasets/ServiceNow-AI/R1-Distill-SFT>.
- Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2023. [Teaching small language models to reason](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1773–1781, Toronto, Canada. Association for Computational Linguistics.
- Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. 2025. [AIMO-2 Winning Solution: Building State-of-the-Art Mathematical Reasoning Models with OpenMathReasoning dataset](#). *Preprint*, arXiv:2504.16891.
- Sella Nevo, Dan Lahav, Ajay Karpur, Yogev Bar-On, Henry Alexander Bradley, and Jeff Alstott. 2024. [Securing AI Model Weights: Preventing Theft and Misuse of Frontier Models](#). RAND Corporation.
- Eleanor Olcott and Cristina Criddle. 2025. OpenAI says it has evidence China’s DeepSeek used its model to train competitor. *Financial Times*.
- OpenAI. 2024. [Learning to reason with LLMs](#). <https://openai.com/index/learning-to-reason-with-llms/>.
- OpenAI. 2026. [Terms of Use](#). <https://openai.com/policies/row-terms-of-use/>.
- OpenAI, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, and 243 others. 2024. [OpenAI o1 System Card](#). *Preprint*, arXiv:2412.16720.
- Leyi Pan, Aiwei Liu, Shiyu Huang, Yijian Lu, Xuming Hu, Lijie Wen, Irwin King, and Philip S. Yu. 2025. [Can LLM Watermarks Robustly Prevent Unauthorized Knowledge Distillation?](#) In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13228–13251, Vienna, Austria. Association for Computational Linguistics.
- Kylie Robison. 2025. Anthropic Revokes OpenAI’s Access to Claude. *Wired*.
- Tom Sander, Pierre Fernandez, Alain Durmus, Matthijs Douze, and Teddy Furon. 2024. [Watermarking makes language models radioactive](#). In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, volume 37 of *NIPS ’24*, pages 21079–21113, Red Hook, NY, USA. Curran Associates Inc.
- James Sanders, Luke Emberson, and Yafah Edelman. 2025. [What did it take to train grok 4?](#) Accessed: 2026-01-04.
- Mark Sweney and Dan Milmo. 2025. OpenAI ‘reviewing’ allegations that its AI models were used to make DeepSeek. *The Guardian*.
- Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. [Stanford alpaca: An instruction-following llama model](#). https://github.com/tatsu-lab/stanford_alpaca.
- Yijun Tian, Yikun Han, Xiushi Chen, Wei Wang, and Nitesh V. Chawla. 2025. [Beyond Answers: Transferring Reasoning Capabilities to Smaller LLMs Using Multi-Teacher Knowledge Distillation](#). In *Proceedings of the Eighteenth ACM International Conference on Web Search and Data Mining, WSDM ’25*, pages 251–260, New York, NY, USA. Association for Computing Machinery.
- Florian Tramèr, Fan Zhang, A. Juels, M. Reiter, and Thomas Ristenpart. 2016. [Stealing Machine Learning Models via Prediction APIs](#). In *USENIX Security Symposium*.
- U.S. Copyright Office. 2025. [Copyright and artificial intelligence, part 3: Generative ai training \(pre-publication version\)](#). Technical report, United States Copyright Office. Report of the Register of Copyrights. Pre-publication version. Accessed 2026-01-03.
- 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.
- Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khoshabi, and Hannaneh Hajishirzi. 2023. [Self-Instruct: Aligning Language Models with Self-Generated Instructions](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.
- Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. [Finetuned Language Models are Zero-Shot Learners](#). In *International Conference on Learning Representations*.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. [Chain-of-thought prompting elicits reasoning in large language models](#). *Preprint*, arXiv:2201.11903.

Zhiheng Xi, Wenxiang Chen, Boyang Hong, Senjie Jin, Rui Zheng, Wei He, Yiwen Ding, Shichun Liu, Xin Guo, Junzhe Wang, Honglin Guo, Wei Shen, Xiaoran Fan, Yuhao Zhou, Shihan Dou, Xiao Wang, Xinbo Zhang, Peng Sun, Tao Gui, and 2 others. 2024. Training large language models for reasoning through reverse curriculum reinforcement learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *ICML'24*, pages 54030–54048, Vienna, Austria. JMLR.org.

Haoran Xu, Baolin Peng, Hany Awadalla, Dongdong Chen, Yen-Chun Chen, Mei Gao, Young Jin Kim, Yunsheng Li, Liliang Ren, Yelong Shen, Shuohang Wang, Weijian Xu, Jianfeng Gao, and Weizhu Chen. 2025. [Phi-4-Mini-Reasoning: Exploring the Limits of Small Reasoning Language Models in Math](#). *Preprint*, arXiv:2504.21233.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. [Qwen3 Technical Report](#). *Preprint*, arXiv:2505.09388.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. [Tree of thoughts: Deliberate problem solving with large language models](#). *Preprint*, arXiv:2305.10601.

Peigen Ye, Huali Ren, Zhengdao Li, Anli Yan, Hongyang Yan, Shaowei Wang, and Jin Li. 2025. [Securing Large Language Models: A Survey of Watermarking and Fingerprinting Techniques](#). *ACM Comput. Surv.*

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. 2022. [STaR: Bootstrapping reasoning with reasoning](#). *Preprint*, arXiv:2203.14465.

Kaixiang Zhao, Lincan Li, Kaize Ding, Neil Zhenqiang Gong, Yue Zhao, and Yushun Dong. 2025. [A Survey on Model Extraction Attacks and Defenses for Large Language Models](#). *Preprint*, arXiv:2506.22521.

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. 2023. Provable Robust Watermarking for AI-Generated Text. In *The Twelfth International Conference on Learning Representations*.

A Qualitative Analysis of Watermarked Reasoning Traces

To illustrate how REASMARK modulates reasoning behavior, we present paired examples generated by the watermarked model under two conditions: a short-token-dominated prompt ($r(x) < -\tau$) and a long-token-dominated prompt ($r(x) > \tau$), constructed from the same base question by substituting tokens from \mathcal{S} and \mathcal{L} , respectively. Both traces arrive at the same correct final answer. The short trace identifies the problem type directly and applies the formula in a few steps, while the long trace systematically explores multiple interpretations before converging. All intermediate steps in both traces remain logically coherent, with no redundant or incorrect reasoning introduced, confirming that $\mathcal{L}_{\text{think}}$ and \mathcal{L}_{ent} modulate reasoning length without introducing redundant steps or distorting reasoning faithfulness.

A.1 Short Example

Prompt:

The task is to determine a formula for the number of possible ways to select an r -person committee from a total of n individuals.

Answer:

```
<think>
Okay, so I need to figure out the formula for the number of ways to select an  $r$ -person committee from  $n$  individuals. Hmm, this sounds like a combination problem. Let me think. When you want to select a committee where the order doesn't matter, it's a combination, right? So the formula for combinations is  $C(n, r)$  which is  $n$  choose  $r$ . But let me make sure I'm not missing anything here.
```

First, the problem says "select an r -person committee from a total of n individuals." So, that means we are choosing r people out of n , and the order in which they are selected doesn't matter. For example, if I have 5 people and I want to choose 2, the number of ways is $C(5, 2)$ which is 10. That seems right, like 5 choices for the first person, 4 for the second, but since order doesn't matter, we divide by 2, so $5 \cdot 4 / 2 = 10$. Yeah, that's correct.

So the general formula for combinations is $n!$ divided by ($r!$ times $(n - r)!$), which is written as $C(n, r) = n! / (r!(n - r)!)$. So that's the formula. Let me check with a simple case. If $n=5$, $r=2$, then $5!$ is 120, divided by $2! \cdot 3!$ which is $2 \cdot 6 = 12$. $120/12$ is 10. Yep, that's right. So the formula is $n!$ divided by $(r!(n - r)!)$.

But wait, sometimes people use the binomial coefficient notation, which is $\binom{n}{r}$. So the formula is $C(n, r) = n! / [r!(n - r)!]$. So that's the formula. But maybe the question expects a different form? Like using the combination formula with exponents or something else?

Alternatively, the problem might be looking for the formula in terms of factorials, which is what I wrote. But maybe they want it in a different way? Let me think again. The number of ways to choose r elements from n is given by the binomial coefficient, which is the same as the formula above.

Alternatively, if I think about the process of selecting the committee, the first person can be chosen in n ways, the second in $(n - 1)$ ways, and so on, down to the r th person. But that would be $n \cdot (n - 1) \cdot \dots \cdot (n - r + 1)$, which is equal to $n! / (n - r)!$. So that's the same as the formula, because $n! / (n - r)!$ divided by $r!$ is the same as the combination formula. So yeah, that's the same thing.

So, the formula is $C(n, r) = n! / [r!(n - r)!]$. So that's the answer. Let me check with another example. If $n=4$, $r=2$. Then the formula gives $4! / (2! \cdot 2!) = 24 / (2 \cdot 2) = 6$. Which is correct, since the number of ways to choose 2 out of 4 is 6. Like, ABCD, the combinations are AB, AC, AD, BC, BD, CD. Yep, that's 6. So the formula works.

Alternatively, if the problem is in terms of permutations, but since they are just selecting a committee, not arranging them, it's combinations. So the formula is indeed n choose r , which is the combination formula. So I think that's the answer.

But just to make sure, let me think if there's any other way this could be interpreted. For example, if the problem was about arranging the committee members in order, then it would be permutations, which is $nPr = n! / (n - r)!$. So that's different. But the question says "select an r -person committee," which implies that the order doesn't matter. So it's combinations, not permutations. So the formula is combinations, which is $n! / [r!(n - r)!]$.

Therefore, the formula is $C(n, r) = n! / [r!(n - r)!]$, which can also be written as $\binom{n}{r}$.

So I think that's the answer. I don't see any other interpretations here. The key is that it's a combination, not permutation, so the formula is as above.

```
**Final Answer**
The number of possible ways is
\boxed{\dfrac{n!}{r!(n - r)!}}
</think>
```

To determine the number of ways to select an ***r***-person committee from a total of ***n*** individuals, we are essentially asking for the number of **combinations** of n things taken r at a time.

Key Concepts

- **Combinations** are used when the order of selection does **not** matter.
- The formula for the number of combinations of n items taken r at a time is:

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

This is also written in LaTeX as:

$$\boxed{\frac{n!}{r!(n-r)!}}$$

Explanation of the Formula

- $n!$: The factorial of the total number of individuals, representing all possible orderings.
- $r!$: The factorial of the number of individuals chosen, which accounts for the order of those r people (though since order doesn't matter in combinations, this is divided out).
- $(n-r)!$: The factorial of the number of individuals not chosen, which also accounts for the leftover individuals.

Example

Suppose we want to choose 2 people from 5. Then:

$$\binom{5}{2} = \frac{5!}{2!(5-2)!} = \frac{120}{2 \cdot 6} = \frac{120}{12} = 10$$

This matches the known result that there are 10 ways to select 2 people from 5.

Conclusion

The number of ways to select an r -person committee from n individuals is:

$$\boxed{\frac{n!}{r!(n-r)!}}$$

Given a total of n individuals, calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee. Express your result as a single product involving n and r .

Answer:

<think>

Okay, so I need to figure out a formula for the number of distinct pairs and larger groupings that can be formed into an r -person committee from n individuals. The question mentions "distinct pairs" and "larger groupings," so I need to consider all possible combinations, right?

First, I remember that the number of ways to form an r -person committee from n individuals is given by the combination formula, which is $C(n, r) = n! / (r! * (n-r)!)$. But the question specifically asks for the number of distinct pairs and larger groupings. Wait, "pairs" would be combinations of 2 people, right? So maybe they want the total number of pairs, which is $C(n, 2)$, and then the number of larger groupings, which would be the number of committees with 3 people, 4 people, up to r people? Or is it asking for the total number of all possible groupings, including pairs and larger groups?

Wait, the problem says "the number of distinct pairs and larger groupings that can be formed to create an r -person committee." So maybe they want the total number of all possible subsets of the committee, from size 2 up to r ? But the wording is a bit unclear. Let me read it again.

"Calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee. Express your result as a single product involving n and r ."

So maybe it's the total number of all possible committees, including pairs and larger groups? But the question says "distinct pairs and larger groupings," so maybe they want the sum of all combinations from size 2 up to r ? So the total number of committees of size 2, 3, ..., r . So that would be $C(n, 2) + C(n, 3) + \dots + C(n, r)$. But the problem says "Express your result as a single product involving n and r ."

Hmm. So the answer has to be a single product, not a sum. So maybe there's a formula for the sum of combinations from $k=2$ to $k=r$ of $C(n, k)$. But how to express that as a single product?

A.2 Long Example

Prompt:

Alternatively, maybe the question is referring to the number of distinct pairs (i.e., committees of size 2) and larger groupings (i.e., committees of size 3, 4, ..., r). But the wording is a bit ambiguous. However, given that the original question says "the number of distinct pairs and larger groupings that can be formed to create an r-person committee," it might be that the total number of all possible committees, from size 2 up to r. So the total number of committees, excluding the full r-person committee? Or including?

Wait, the original question says "to create an r-person committee." So maybe the question is asking for the total number of all possible committees that can be formed, where the committee has exactly r people? But then that would be $C(n, r)$. But the question mentions "distinct pairs and larger groupings," so maybe they want the total number of all possible committees, including pairs (size 2) up to the r-person committee.

But the problem is phrased as "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r-person committee." So maybe the total number of all possible committees that can be formed, where the committee has 2 people, 3 people, ..., up to r people. But then that would be the sum from $k=2$ to $k=r$ of $C(n, k)$. However, the problem says "Express your result as a single product involving n and r." So there must be a way to write that sum as a product.

Alternatively, maybe the question is asking for the number of ways to form a committee of size r, which is $C(n, r)$, but the wording says "distinct pairs and larger groupings," which might be a bit confusing. Let me think again.

If the question is about forming an r-person committee, then the total number of such committees is $C(n, r)$. But maybe they are considering the number of pairs and larger groupings in the context of forming the r-person committee. Wait, but pairs are part of the r-person committee. So perhaps the question is asking for the total number of possible committees of size r, but the answer is supposed to be expressed as a single product. But how?

Alternatively, maybe the question is asking for the number of distinct pairs (which is $C(n, 2)$) and the number of larger groupings (which would be $C(n, 3) + C(n, 4) + \dots + C(n, r)$), and then the total number of all possible committees is the sum from $k=2$ to $k=r$ of $C(n, k)$. So the total number is the sum $\sum_{k=2}^r C(n, k)$. But the question says "Express your result as a single product involving n and r." So I need to convert that sum into a product.

Alternatively, maybe the question is referring to the number of distinct pairs (i.e., the number of 2-person committees) and the number of distinct larger groupings (i.e., 3-person, 4-person, etc.), and the total number of all possible committees would be the sum of those. But how to write that as a single product.

Alternatively, perhaps the question is referring to the number of ways to form an r-person committee, which is $C(n, r)$, but maybe they want the number of pairs and larger groupings in the sense of the number of subsets of the committee, so maybe the total number of subsets of the committee is 2^n , but excluding the empty set and the full set. But the problem says "to create an r-person committee," so maybe they want the number of all possible committees of size r, which is $C(n, r)$. But that's not what the question says. The question says "the number of distinct pairs and larger groupings that can be formed to create an r-person committee." So maybe the answer is $C(n, r)$, but expressed as a product.

But the question says "Express your result as a single product involving n and r." So, the answer is $C(n, r)$, which is $n! / (r! (n - r)!)$. But that's a single product, but it's a fraction. However, the question says "the number of distinct pairs and larger groupings that can be formed to create an r-person committee." So perhaps they are asking for the total number of all committees, from size 2 up to r. So that would be the sum from $k=2$ to r of $C(n, k)$. Let me check that.

So, the sum $S = C(n, 2) + C(n, 3) + \dots + C(n, r)$. We need to express S as a single product. I know that the sum from $k=0$ to n of $C(n, k) = 2^n$. So $S = 2^n - C(n, 0) - C(n, 1) = 2^n - 1 - n$. But that's for the sum from $k=0$ to n. However, if we want the sum from $k=2$ to r, then $S = 2^n - 1 - n - [\text{sum from } k=r+1 \text{ to } n \text{ of } C(n, k)]$. But this seems complicated.

Alternatively, maybe the question is referring to the total number of committees of size r, which is $C(n, r)$, and the user is confused between the sum of all subsets of the committee (excluding the empty and full set). But the problem says "the number of distinct pairs and larger groupings that can be formed to create an r-person committee." So maybe the answer is $C(n, r)$, but the question is a bit ambiguous.

But the original problem says "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r-person committee." So maybe the total number of all possible committees, including pairs, triples, etc., up to r-person committees. But then, the answer would be the sum $\sum_{k=2}^r C(n, k)$. But how to write this as a single product?

Alternatively, maybe the question is asking for the number of ways to form a committee of size r , which is $C(n, r)$, but the mention of "pairs and larger groupings" is just to clarify that the committee can be of any size from 2 up to r . But then the answer would be $C(n, r)$. But that's a single term, but the question says "Express your result as a single product involving n and r ." So maybe they want the answer as a product, not a combination.

Alternatively, maybe the question is referring to the number of pairs (which is $C(n, 2)$) and the number of larger groupings, which is the sum from $k=3$ to r of $C(n, k)$. But then the total would be $C(n, 2) + \sum_{k=3}^r C(n, k)$. Which is equal to $2^n - 1 - n - C(n, 1) - [\sum_{k=r+1}^n C(n, k)]$. But this seems too complicated.

Alternatively, maybe the question is asking for the total number of all possible committees (including pairs, triples, etc.) up to the r -person committee, so the total number is the $\sum_{k=2}^r C(n, k)$. But how to write this as a product?

Alternatively, maybe the question is asking for the number of ways to form a committee of size r , which is $C(n, r)$, but the mention of "pairs and larger groupings" is just clarifying that the committee can be of any size, not just the full committee. But then the answer would be $C(n, r)$, but that's a single product.

Alternatively, the problem might be asking for the number of distinct pairs (i.e., the number of 2-person committees) and the number of larger groupings (i.e., 3-person, 4-person, etc.) and then the total number of all such committees. But then, how to write that as a product?

Wait, another approach. The number of all subsets of the n individuals that have at least two elements and at most r elements. So that would be $\sum_{k=2}^r C(n, k)$. But how to write this as a product?

Alternatively, using the binomial coefficient identity. I know that $\sum_{k=0}^n C(n, k) = 2^n$. So $\sum_{k=0}^n C(n, k) = 2^n$. Therefore, $\sum_{k=2}^n C(n, k) = 2^n - C(n, 0) - C(n, 1) = 2^n - 1 - n$. But if we want to exclude the $k=2$ to $k=r$, then it's $2^n - 1 - n - \sum_{k=r+1}^n C(n, k)$. But this seems complicated.

Alternatively, maybe the question is not asking for the sum from $k=2$ to r of $C(n, k)$, but rather the number of pairs and larger groupings in the context of forming an r -person committee. Maybe the question is asking for the total number of all possible committees (including pairs, triples, etc.) that can be formed, and that would be $C(n, r) + C(n, r-1) + \dots + C(n, 2)$. But again, how to write that as a product.

Alternatively, maybe the question is referring to the number of ways to form a committee of size r , but the mention of pairs and larger groupings is just to indicate that the committee can be of any size from 2 up to r . So maybe the answer is $C(n, r)$, but written as a product. But the problem says "Express your result as a single product involving n and r ." So maybe they want it written as a product, not a sum. So perhaps the answer is $C(n, r) = n! / (r! (n - r)!)$, but that's a single term.

Alternatively, maybe the question is asking for the total number of all possible committees of size 2, 3, ..., r . So that's the sum from $k=2$ to $k=r$ of $C(n, k)$. But how to express this as a product?

Wait, there's a formula for the sum of binomial coefficients. For example, the $\sum_{k=0}^n C(n, k) = 2^n$. But if we have the sum from $k=2$ to r of $C(n, k)$, that's equal to $2^n - C(n, 0) - C(n, 1) - C(n, r+1) - \dots - C(n, n)$. But this seems messy.

Alternatively, maybe there's a generating function approach. The generating function for binomial coefficients is $(1 + x)^n$. The $\sum_{k=0}^n C(n, k) x^k = (1 + x)^n$. If we want the sum from $k=2$ to r of $C(n, k)$, we can write it as $(1 + x)^n$ evaluated at $x=1$, minus the sum of the first few terms. But I don't think that helps directly.

Alternatively, maybe the question is expecting the answer to be $C(n, r)$ multiplied by something? Not sure.

Alternatively, maybe the question is a trick question. If you think of the number of distinct pairs and larger groupings, it's the same as the total number of subsets of the committee, except for the empty set and the full set. So the total number of subsets is 2^n , so the number of non-empty subsets is $2^n - 1$. But the number of committees that have at least two people is $2^n - 1 - n$. Because the number of non-empty subsets is $2^n - 1$, and we subtract the single-element subsets (which are n) and the empty set. Wait, but the question says "pairs and larger groupings," so maybe the answer is $2^n - 1 - n$, which is the number of non-empty subsets of size at least 2. So that would be the sum from $k=2$ to r of $C(n, k)$ if r is n . But if r is less than n , then it's different.

But the problem says "to create an r -person committee." So maybe the answer is $2^n - 1 - n$, but written as a product. However, $2^n - 1 - n$ is a single expression, but it's not a product. The problem says "Express your result as a single product involving n and r ."

So maybe the answer is $(n - 1)! * \text{something?}$
Not sure.

Alternatively, maybe the question is actually referring to the number of ways to form an r -person committee, which is $C(n, r)$, and the mention of "pairs and larger groupings" is just to clarify that the committee can be of any size, but the answer is $C(n, r)$. But then, the problem says "Express your result as a single product involving n and r ." So $C(n, r)$ is a product, but written in terms of factorials. For example, $C(n, r) = n! / (r! (n - r)!)$, which is a product.

But maybe the question is expecting the answer to be the sum of $C(n, k)$ from $k=2$ to r , but expressed as a product. But how?

Alternatively, maybe the question is referring to the number of pairs (which is $C(n, 2)$) and the number of larger groupings (which is the number of committees of size 3, 4, ..., r). So the total number is $C(n, 2) + C(n, 3) + \dots + C(n, r)$. But this is equal to $2^n - 1 - n - C(n, r+1) + \dots$ but this seems too complicated.

Alternatively, maybe the answer is the same as the total number of subsets of the committee, excluding the empty set and the full set. So that would be $2^n - 2$. But that's not considering the sizes between 2 and r .

Wait, if the question is asking for the number of distinct pairs and larger groupings, but the committee is of size r . So maybe the total number of all possible committees of size 2, 3, ..., r , which is the sum $\sum_{k=2}^r C(n, k)$. However, this is not a single product. But maybe the problem is expecting the answer to be $C(n, r)$ multiplied by something?

Alternatively, maybe the original question is in a context where "pairs and larger groupings" is referring to the total number of all possible committees, including pairs, triples, etc., and the answer is the sum from $k=2$ to r of $C(n, k)$. But since the problem says "Express your result as a single product involving n and r ," perhaps the answer is $n! / (r! (n - r)!)$ times something. But I don't see the connection.

Alternatively, maybe the question is a misstatement, and they actually want the number of distinct pairs, which is $C(n, 2)$, and the number of larger groupings is $C(n, 3) + \dots + C(n, r)$. But then, the answer would be the sum $\sum_{k=2}^r C(n, k)$. But how to write that as a product?

Alternatively, maybe the question is expecting the answer to be $2^n - n - 1$, but that is the sum from $k=2$ to n of $C(n, k)$ minus the terms from $k=n$ to $k=\dots$ but no.

Alternatively, perhaps the problem is expecting the answer to be $(n - 1) * (n - 2) * \dots * (n - r + 1) / r!$? But that would be the number of combinations of r elements, but it's not clear.

Alternatively, if the question is about the number of distinct pairs and larger groupings in the context of forming an r -person committee, maybe they are referring to the number of ways to choose a committee of size r , which is $C(n, r)$, and the "distinct pairs" is a subset of that. But the problem says "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee." So maybe the answer is $C(n, r)$, and the mention of pairs and larger groupings is just to indicate that the committee can be of any size from 2 up to r . But then, the answer is $C(n, r)$, which is a single product. But the problem says "Express your result as a single product involving n and r ," so that would be acceptable.

But maybe the question is expecting the total number of all possible subsets of the committee, excluding the empty set and the full set, and that's $2^n - 2$. But that's not considering the sizes between 2 and r . If r is less than n , then $2^n - 2$ would be the total number of non-empty subsets, but that includes all sizes from 1 to n . So if the question is about forming an r -person committee, but the answer is supposed to be the number of all subsets with size at least 2 and at most r . So that would be the sum from $k=2$ to $k=r$ of $C(n, k)$. But how to write this as a product?

Alternatively, maybe the question is expecting the answer to be $C(n, r)$ times something, but I don't see it. Maybe I need to think again.

Wait, the original question says "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee." So the committee is r -person, but the question is about the number of pairs and larger groupings in that process. So, for example, if you have an r -person committee, how many ways can you form a pair within that committee, and how many ways can you form a committee of size 3, etc. But I think the question is asking for the total number of all possible committees (from size 2 up to r) that can be formed from n individuals. So the answer is $\sum_{k=2}^r C(n, k)$. But the problem says "Express your result as a single product involving n and r ." So maybe there's a formula for this sum.

But I recall that there is a formula for the sum of binomial coefficients, but it's not straightforward. However, there is an identity that $\sum_{k=0}^n C(n, k) = 2^n$. But for the sum from $k=2$ to r , it's $2^n - 1 - n - [\text{sum from } k=r+1 \text{ to } n \text{ of } C(n, k)]$. But this seems too complicated.

Alternatively, if the question is in the context of a specific problem, maybe the answer is $C(n, r)$ multiplied by something. But I don't know.

Alternatively, since the problem says "Express your result as a single product involving n and r ," maybe the answer is a product like $(n-1)(n-2)\dots(n-r+1) / r!$ which is $C(n, r)$. But that's the same as the combination formula.

But if the question is asking for the total number of pairs and larger groupings (i.e., all subsets of size 2 to r), then the answer would be the $\sum_{k=2}^r C(n, k)$. But how to write this as a product.

Wait, another thought: the number of subsets of size at least 2 is $2^n - 2$. But if we want the number of subsets of size exactly 2, that's $C(n, 2)$, and the number of subsets of size exactly 3 is $C(n, 3)$, etc. But the total number is $2^n - 2 - n$. Wait, because the number of subsets of size at least 2 is $2^n - 1$ (all non-empty subsets) minus the number of single-element subsets, which is n , minus the empty set. So that's $2^n - 1 - n$. So the total number of subsets with at least two elements is $2^n - 1 - n$. But this is the sum from $k=2$ to n of $C(n, k) - C(n, 0) - C(n, 1) = 2^n - 1 - n$. But if the question is asking for the number of pairs and larger groupings, then that would be $2^n - 1 - n$. But this is a single expression, but it's not a product. The problem says "Express your result as a single product involving n and r ."

Wait, but if the question is about forming an r -person committee, then the total number of subsets with size at least 2 and at most r . So if r is less than n , then the total number is $2^n - 1 - n - [\text{sum from } k=r+1 \text{ to } n \text{ of } C(n, k)]$. But this is too complicated. Maybe the answer is simply $C(n, r)$, but I need to check.

Alternatively, maybe the question is a misstatement, and they actually want the number of ways to form a committee of size r , which is $C(n, r)$, and the mention of pairs and larger groupings is just to clarify that the committee is of size r , but the problem wants the answer expressed as a product. So the answer would be $C(n, r) = n! / (r! (n-r)!)$, which is a product. Since the problem says "Express your result as a single product involving n and r ," then this would be acceptable.

But then why mention "distinct pairs and larger groupings"? Maybe they want the total number of all possible committees of size 2, 3, ..., r , which is the $\sum_{k=2}^r C(n, k)$. But if the answer is supposed to be a single product, then maybe the answer is $(n! / (r! (n-r)!)) * (n-1)! / (r-1)! (n-r-1)!)$? No, that seems too arbitrary.

Alternatively, maybe the question is in the context of the number of distinct pairs and larger groupings in the entire committee, which is the same as the number of all possible subsets of the committee, except for the empty set and the full set. But that would be $2^n - 2$, which is a single expression, but it's not a product. So the problem says "Express your result as a single product involving n and r ." So maybe the answer is $2^n - 2$, but written as a product. But $2^n - 2$ is not a product. It's a difference of powers.

Alternatively, maybe the question is expecting the answer to be $C(n, r) * (\text{something})$, but I don't know.

Given that the problem is from a math course, and the answer is supposed to be a single product, maybe the answer is simply $C(n, r)$, which is a product. So the number of distinct r -person committees is $C(n, r)$, which is the product of $n, n-1, \dots, n-r+1$ divided by $r!$. So, the formula would be $(n)(n-1)\dots(n-r+1) / r!$

But the question says "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee." If the answer is the number of r -person committees, then that would be $C(n, r)$. But the question mentions "pairs and larger groupings," which might be a way to say that the committee can be of any size from 2 up to r , but the total number of all possible committees is $C(n, r)$. But that seems conflicting.

Alternatively, maybe the question is asking for the total number of all possible committees (of any size from 2 to r), which is the $\sum_{k=2}^r C(n, k)$. But how to write that as a product?

Wait, there is an identity that $\sum_{k=0}^n C(n, k) = 2^n$, and $\sum_{k=0}^n C(n, k) x^k = (1+x)^n$. If I want the sum from $k=2$ to r of $C(n, k)$, that would be equal to $(1+1)^n - C(n, 0) - C(n, 1) - C(n, 2) - \dots - C(n, r)$. But no, that's not helpful.

Alternatively, maybe the problem is expecting the answer to be $C(n, r)$ multiplied by something, but I can't see it.

Given that the problem says "Express your result as a single product involving n and r ," and considering that the number of r -person committees is $C(n, r)$, which is a product, I think that might be the expected answer, even though the question mentions "pairs and larger groupings."

So maybe the answer is $C(n, r) = n! / (r! (n-r)!)$, which is a product.

But the original question says "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee." So if the answer is $C(n, r)$, then that would be the number of r -person committees, and the "distinct pairs and larger groupings" is just a way to say that the committee can be of any size from 2 up to r , but the formula for the number of r -person committees is $C(n, r)$. But the question might be expecting that.

Alternatively, if the question is asking for the total number of all possible committees (of size 2, 3, ..., r), then that would be the $\sum_{k=2}^r C(n, k)$. But since that's a sum, and the problem says "Express your result as a single product," maybe they want the answer in terms of factorial products.

Alternatively, maybe the answer is $(n * (n - 1) * (n - 2) * \dots * (n - r + 1)) / r!$ which is $C(n, r)$, but that's a product.

Given that, and considering that the question might have a typo or be ambiguous, but since the answer is expected to be a single product, I think the answer is $C(n, r)$, which is the number of r -person committees.

But the original question says "the number of distinct pairs and larger groupings that can be formed to create an r -person committee." So if the committee is r -person, then the number of pairs is $C(n, 2)$, and the number of larger groupings is $C(n, 3) + \dots + C(n, r)$. So the total number would be $\sum_{k=2}^r C(n, k)$. But how to write that as a single product?

Wait, the total number is $C(n, 2) + C(n, 3) + \dots + C(n, r)$. This can be written as $\sum_{k=2}^r C(n, k)$. However, there is a formula for this sum. Let me recall.

The $\sum_{k=0}^r C(n, k) = C(n, 0) + C(n, 1) + \dots + C(n, r)$. This is the same as the sum from $k=0$ to r of $C(n, k)$. But how to relate this to a product?

Alternatively, using the hockey-stick identity or other binomial identities, but I don't recall a direct identity for the sum from $k=2$ to r of $C(n, k)$.

Alternatively, if we consider the problem's intention, maybe the answer is simply $C(n, r)$, and the mention of "pairs and larger groupings" is just to indicate that the committee is of size r , and the number of such committees is $C(n, r)$. Therefore, the answer is $n! / (r! (n - r)!)$, which is a product.

Since the problem says "Express your result as a single product involving n and r ," and $C(n, r)$ is a product, I think that is the answer they are expecting.

Therefore, the formula is $C(n, r) = n! / (r! (n - r)!)$, which is the product of $n, n-1, \dots, n - r + 1$ divided by $r!$

But the question says "calculate a formula for the number of distinct pairs and larger groupings that can be formed to create an r -person committee." If the answer is $C(n, r)$, then it's the number of r -person committees, and the mention of pairs and larger groupings is just to say that the committee can be of any size, but the answer is the number of r -person committees.

But I need to make sure. If the question is asking for the total number of all possible committees (of size 2 up to r), then the answer is the $\sum_{k=2}^r C(n, k)$. However, this is a sum, not a product. But if the question is expecting the answer to be $C(n, r)$, then that is the answer.

Since the problem says "Express your result as a single product involving n and r ," and $C(n, r)$ is a product, I think that is the answer. So I'll go with that.

Final Answer
The formula is $\boxed{\frac{n!}{r! \cdot (n - r)!}}$
</think>

To determine the number of **distinct pairs** and **larger groupings** (i.e., committees of size 2 to r) that can be formed from n individuals, we must consider the **total number of subsets** of the n individuals that have **at least two elements** and **at most r elements**.

This is a classic application of **binomial coefficients** and the **sum of binomial coefficients**. Specifically, the total number of subsets of n individuals is 2^n , but we must exclude the empty set and all subsets with exactly one element.

Step-by-step Analysis

- Non-empty subsets**:
 - The total number of subsets of n elements is 2^n .
 - This includes the empty set, so we subtract 1: $2^n - 1$.
- Subsets with exactly one element**:
 - There are n such subsets (one for each individual).
- Subsets of size at least 2**:
 - Subtract the two types of subsets: the empty set and the single-element subsets.
 - Total subsets of size at least 2:

$$2^n - 1 - n$$

This is **the number of non-empty subsets of size at least 2**.

However, the question specifies forming n r -person committees. That is, the total number of subsets of size 2, 3, ..., r is:

So the number of distinct pairs and larger groupings (i.e., all subsets of size at least 2 and at most r) is:

$$\sum_{k=2}^r \binom{n}{k}$$

Goal: Express the total number of such subsets as a single product involving n and r

The sum of binomial coefficients from $k = 2$ to $k = r$ is:

$$\sum_{k=2}^r \binom{n}{k} = \binom{n}{2} + \binom{n}{3} + \dots + \binom{n}{r}$$

We can express this sum in terms of factorials:

$$\sum_{k=2}^r \binom{n}{k} = \sum_{k=0}^r \binom{n}{k} - \binom{n}{0} - \binom{n}{1}$$

Using the identity:

$$\sum_{k=0}^r \binom{n}{k} = \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \dots + \binom{n}{r}$$

We can write the total as:

$$\sum_{k=2}^r \binom{n}{k} = 2^n - 1 - n$$

This is the total number of non-empty subsets of size at least 2, but it is not a single product. However, the number of r -person committees (i.e., the number of subsets of size r from n individuals), is:

$$\binom{n}{r} = \frac{n!}{r! \cdot (n-r)!}$$

This is a single product that involves n and r , as required by the question.

Final Answer

$$\boxed{\frac{n!}{r! \cdot (n-r)!}}$$