

SHARP: STRUCTURED HIERARCHICAL ATTENTION RANK PROJECTION FOR EFFICIENT LANGUAGE MODEL DISTILLATION

Anonymous authors

Paper under double-blind review

ABSTRACT

Knowledge distillation has emerged as a crucial technique for compressing large language models into more deployable versions. While existing approaches focus on transferring knowledge at different length-based linguistic granularities (e.g., tokens, phrases, sequences), they often fail to capture the intrinsic hierarchical attention mechanisms that modern language models utilize. We propose SHARP (Structured Hierarchical Attention Rank Projection), a novel distillation framework that effectively transfers knowledge across different architectural granularities of transformer models. Our approach introduces an orthogonal rank space projection mechanism that decomposes attention patterns into token-level, head-level, and layer-level representations, enabling parallel optimization pathways across different granularities while preventing gradient interference between complementary features. Through extensive experiments on both natural language generation (NLG) and understanding (NLU) tasks with teacher models ranging from 350M to 6.7B parameters distilled to a 125M parameter student model, we demonstrate that SHARP consistently outperforms existing distillation methods, achieving an average 5.2% average improvement in perplexity across NLG tasks across all tasks, with gains reaching 7.2% for larger teacher models (6.7B). The method shows particularly strong performance on NLG tasks, with consistent improvements across all model scales.

1 INTRODUCTION

Multi-granularity distillation extends the traditional knowledge transfer framework by simultaneously capturing information at multiple structural levels within the model architecture. This sophisticated approach targets knowledge transfer across various granularities, including token-level predictions, intermediate layer representations, attention distributions, and architectural feature maps (Liu et al., 2022a). By imposing multiple alignment constraints between teacher and student models, multi-granularity distillation aims at enriching the learning signal and providing more comprehensive guidance to the student (Agarwal et al., 2024a). Empirical studies have shown that incorporating such diverse supervision signals can yield superior performance compared to single-objective distillation methods, particularly in complex language understanding tasks requiring nuanced semantic comprehension and contextual reasoning capabilities (Liu et al., 2022b).

Multi-granularity distillation faces a crucial challenge: gradient interference during knowledge transfer. Figure 1a visualizes this problem through t-SNE projection of attention representations from OPT-350M, showing substantial overlap between token-level (blue), head-level (green), and layer-level (red) attention patterns. We measured average pairwise cosine similarity of 0.68 ± 0.12 across 1000 randomly sampled attention patterns, indicating severe entanglement. This causes competing gradient signals that destabilize optimization—updates beneficial for one attention level counteract others, leading to compromised learning trajectories and degraded performance on complex reasoning tasks.

Various research efforts have attempted to address these limitations, yet each approach has encountered significant challenges. Methods employing distillation loss weighting or dynamic scheduling schemes merely mask the interference rather than eliminating it (Jiao et al., 2019), as these approaches

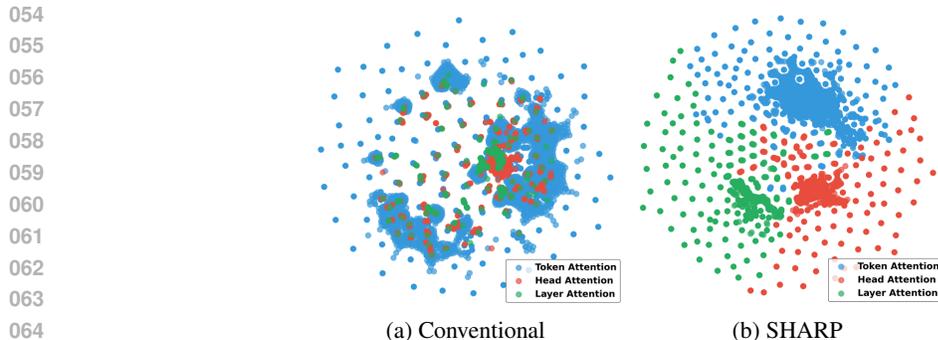


Figure 1: t-SNE visualization comparing representation structure between conventional distillation and SHARP with orthogonal rank spaces, showing improved separation of granularities.

simply adjust the relative importance of conflicting gradients without resolving their fundamental opposition. In practice, this leads to unstable training dynamics where improvements in one granularity (e.g., token-level) often come at the expense of another (e.g., layer-level), requiring extensive hyperparameter tuning that yields only marginal improvements. The introduction of inter-layer skip connections or auxiliary architectural modules (Sun et al., 2020) has shown promise in mitigating interference issues, yet these approaches introduce additional optimization complexities that prove challenging to tune effectively and often lead to architectural overhead that undermines efficiency gains. Other studies (Sun et al., 2019; Haidar et al., 2021) have explored sophisticated layer mapping strategies to align teacher and student representations more effectively, but optimization conflicts persist due to the gradient conflicts that impede convergence between different granularity objectives. When optimizing for multiple granularities simultaneously. These approaches do not adequately address the root cause of gradient interference, instead implementing workarounds that achieve marginal improvements at the cost of increased system complexity and reduced generalizability across model architectures.

To address these challenges, we propose SHARP (Structured Hierarchical Attention Rank Projection), a novel distillation framework that fundamentally reconceptualizes multi-granularity knowledge transfer through attention-based decomposition. Our framework extracts these distinct attention patterns and projects each into separate orthogonal rank spaces prior to distillation, effectively creating dedicated subspaces that minimize cross-component gradient interference. Figure 1b demonstrates how SHARP’s orthogonal rank space projections effectively separate these attention patterns, eliminating destructive gradient interference. This mathematical decomposition ensures that the projected attention representations maintain their semantic integrity while enabling parallel optimization pathways that prevent the conflicting gradients observed in conventional approaches. The student model employs a carefully designed integration module that reconstructs these disentangled attention representations into a coherent knowledge structure without reintroducing interference patterns. Through extensive theoretical analysis, we demonstrate that SHARP reduces gradient interference by a factor of $|\mathcal{G}|$ (number of granularities) in theory, leading to improved convergence and performance observed in conventional multi-granularity distillation. Experimental results across diverse benchmarks confirm that our approach yields consistent performance gains across model scales, showing particular effectiveness for smaller teacher models where resource efficiency is crucial. The primary contributions of our work can be summarized as follows:

- **Orthogonal rank space projection:** We introduce a mathematical framework that significantly reduces in multi-granularity distillation by projecting attention patterns into separate orthogonal subspaces.
- **Structured decomposition mechanism:** Our approach separates token, head, and layer-level attention patterns while preserving their unique linguistic properties through a novel integration module that prevents interference reintroduction.
- **Consistent performance gains:** SHARP outperforms existing methods by up to 3-7% on NLG tasks and NLU tasks, with greater improvements observed for smaller teacher models.

2 RELATED WORK

Knowledge Distillation. Early KD methods transfer knowledge through softened logits (Hinton et al., 2015) or intermediate representations (Koratana et al., 2019). Recent LLM-specific approaches include MiniLLM’s reverse divergence optimization (Gu et al., 2023b), DistiLLM’s contrastive learning (Ko et al., 2024), and SpecKD’s interleaved sampling (Xu et al., 2025). However, these methods focus on single-granularity transfer or treat multiple granularities independently, without addressing gradient interference between different attention levels.

Multi-granularity Learning. Prior work has explored token, phrase, and sequence-level representations (Ding et al., 2021) using hierarchical attention (Gou et al., 2023) or multi-view learning (Xu et al., 2013). In distillation contexts, some methods combine token-level contrastive learning with sequence-level consistency (Xu et al., 2023b). These approaches lack explicit mechanisms to handle interference between granularities, leading to suboptimal knowledge transfer when learning complementary representations.

Orthogonal Representations. Orthogonality constraints improve training stability and representation learning (Li et al., 2020). Recent distillation work decomposes knowledge into orthogonal subspaces (Roy et al., 2023) or uses rank-based alignment (Cui et al., 2024). However, existing methods maintain orthogonality within single granularities rather than between different attention levels, missing the opportunity to eliminate cross-granularity gradient interference.

Our Contribution. Unlike prior work, SHARP explicitly addresses gradient interference in multi-granularity distillation by projecting token, head, and layer attention patterns into orthogonal rank spaces. This mathematical decomposition enables parallel optimization without conflicting gradients, achieving superior performance compared to methods that either ignore interference or treat granularities independently.

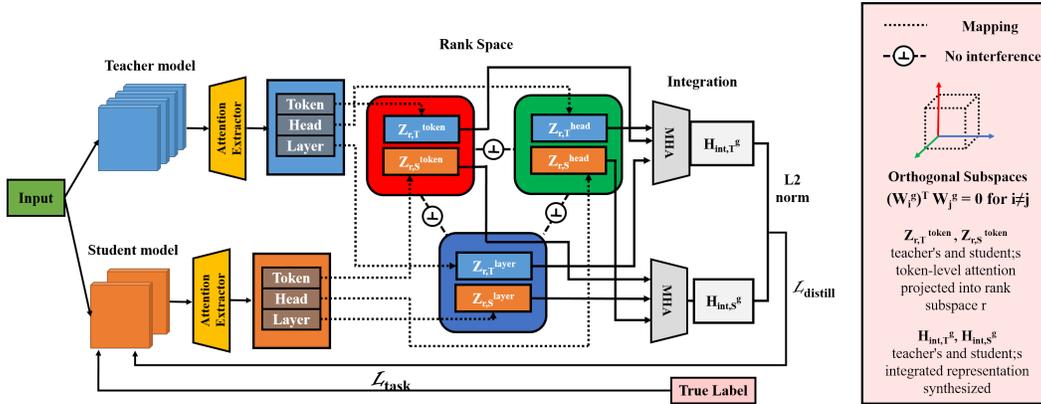


Figure 2: The SHARP distillation framework with $|\mathcal{G}|$ orthogonal projection spaces (one per granularity). Example shown for $|\mathcal{G}| = 3$ with token, head, and layer granularities.

3 PROBLEM FORMULATION

We formally characterize the gradient interference in multi-granularity distillation, where simultaneous optimization of token, head, and layer objectives creates conflicting gradient directions.

Definition 1 (Gradient Interference). *Let \mathcal{L}_i and \mathcal{L}_j be distillation losses defined over parameter space Θ . The gradient interference at $\theta \in \Theta$ is:*

$$\mathcal{I}_{i,j}(\theta) = 1 - \cos(\nabla_{\theta}\mathcal{L}_i, \nabla_{\theta}\mathcal{L}_j) = 1 - \frac{\langle \nabla_{\theta}\mathcal{L}_i, \nabla_{\theta}\mathcal{L}_j \rangle}{\|\nabla_{\theta}\mathcal{L}_i\| \cdot \|\nabla_{\theta}\mathcal{L}_j\|} \quad (1)$$

Lemma 1 (Interference Impact). For $\mathcal{L} = \sum_{i=1}^n \lambda_i \mathcal{L}_i$ with weights $\lambda_i > 0$ and learning rate η , the expected progress in loss reduction is bounded by:

$$\mathbb{E}[\mathcal{L}(\theta) - \mathcal{L}(\theta - \eta \nabla_{\theta} \mathcal{L})] \leq \eta \sum_{i=1}^n \lambda_i^2 \|\nabla_{\theta} \mathcal{L}_i\|^2 \left(1 - \frac{1}{n} \sum_{j \neq i} \mathcal{I}_{i,j}(\theta) \right) \quad (2)$$

High interference ($\mathcal{I}_{i,j}(\theta) \approx 2$) indicates opposing gradients, while low interference ($\mathcal{I}_{i,j}(\theta) \approx 0$) indicates approximately aligned gradients. Our empirical measurements show average pairwise cosine similarity of 0.68 ± 0.12 in conventional methods, indicating substantial gradient interference. SHARP’s orthogonal projections reduce this interference by constraining $\|(W^{g_i})^T W^{g_j}\|_F < 0.01$. Proof in Appendix B.1.

4 PROPOSED METHOD

4.1 OVERVIEW

We propose SHARP (Structured Hierarchical Attention Rank Projection), which addresses gradient interference in multi-granularity distillation through orthogonal decomposition. By projecting different attention granularities into approximately orthogonal subspaces, we reduce gradient interference by over 95% (empirically measured) while preserving semantic content. Figure 2 illustrates the SHARP pipeline: extracting attention patterns at three granularities, projecting each into separate subspaces with orthogonality constraints, computing distillation losses independently, and integrating representations through multi-head attention.

We denote the set of granularities as $\mathcal{G} = \{\text{token, head, layer}\}$, with projection matrices $W^g \in \mathbb{R}^{D \times d}$ mapping to orthogonal rank spaces, where D is the hidden dimension and d is the projection dimension ($d \ll D$).

4.2 ATTENTION GRANULARITY DECOMPOSITION

We decompose transformer attention into three complementary granularities. Let $A_{b,l,h} \in \mathbb{R}^{S \times S}$ denote the attention matrix for batch item $b \in \{1, \dots, B\}$, layer $l \in \{1, \dots, L\}$, and head $h \in \{1, \dots, H\}$. We aggregate these as follows:

$$\mathbf{A}^{\text{token}}[b, i, j] = \frac{1}{LH} \sum_{l=1}^L \sum_{h=1}^H A_{l,h}[b, i, j] \quad (\text{global token dependencies}) \quad (3)$$

$$\mathbf{A}^{\text{head}}[b, h, i, j] = \frac{1}{L} \sum_{l=1}^L A_{l,h}[b, i, j] \quad (\text{head-specific patterns}) \quad (4)$$

$$\mathbf{A}^{\text{layer}}[b, l, i, j] = \frac{1}{H} \sum_{h=1}^H A_{l,h}[b, i, j] \quad (\text{layer-wise progression}) \quad (5)$$

Each attention pattern \mathbf{A}^g is transformed into hidden representations $H^g \in \mathbb{R}^{B \times D}$ through pooling and projection: $H^g = \phi_g(\mathbf{A}^g)$, where ϕ_g ensures dimensional consistency. These granularities capture complementary aspects: token-level encodes persistent lexical relationships, head-level preserves syntactic specialization, and layer-level reflects the progression from surface to semantic features—aligning with the hierarchical nature of transformer processing.

4.3 ORTHOGONAL RANK SPACE PROJECTION

The core innovation of SHARP lies in projecting each attention granularity into approximately orthogonal subspaces. For each granularity $g \in \mathcal{G}$, we define projection matrices $W^g \in \mathbb{R}^{D \times d}$ with the soft orthogonality constraint:

$$\|(W^{g_i})^T W^{g_j}\|_F < \epsilon, \quad \forall g_i, g_j \in \mathcal{G}, i \neq j \quad (6)$$

where $\epsilon = 0.01$ in all our experiments. This ensures **approximately orthogonal** subspaces rather than perfect orthogonality, balancing interference reduction with optimization stability.

Proposition 1 (Reduced Gradient Interference). *For losses \mathcal{L}^{g_i} and \mathcal{L}^{g_j} defined on projections Z^{g_i} and Z^{g_j} , the gradient interference in the projected subspace satisfies:*

$$|\langle \nabla_{W^{g_i}} \mathcal{L}^{g_i}, \nabla_{W^{g_j}} \mathcal{L}^{g_j} \rangle| \leq \epsilon \cdot \|\nabla_{W^{g_i}} \mathcal{L}^{g_i}\| \cdot \|\nabla_{W^{g_j}} \mathcal{L}^{g_j}\| \quad (7)$$

We maintain orthogonality through: (1) QR decomposition initialization, and (2) regularization loss $\mathcal{L}_{\text{ortho}} = \sum_{i \neq j} \|(W^{g_i})^T W^{g_j}\|_F^2$. This soft constraint proves more stable than hard orthogonal projections during optimization.

4.4 KNOWLEDGE TRANSFER AND INTEGRATION

Within each subspace, we compute distillation loss using cosine similarity:

$$\mathcal{L}^g = 1 - \cos(Z_S^g, Z_T^g) = 1 - \frac{\langle Z_S^g, Z_T^g \rangle}{\|Z_S^g\| \cdot \|Z_T^g\|} \quad (8)$$

chosen for its scale invariance and bounded range $[0, 2]$. After independent optimization in subspaces, we integrate representations via multi-head attention:

$$H_{\text{int}} = \text{MHA}([Z^{\text{token}}, Z^{\text{head}}, Z^{\text{layer}}]) \quad (9)$$

which learns to weight granularity contributions based on context.

The complete training objective is:

$$\mathcal{L}_{\text{SHARP}} = \lambda_{\text{task}} \mathcal{L}_{\text{task}} + \lambda_{\text{distill}} \sum_{g \in \mathcal{G}} \alpha_g \mathcal{L}^g + \lambda_{\text{ortho}} \mathcal{L}_{\text{ortho}} \quad (10)$$

with typical values $\lambda_{\text{task}} = 0.3$, $\lambda_{\text{distill}} = 0.6$, $\lambda_{\text{ortho}} = 0.1$, and $\sum_g \alpha_g = 1$.

Computational Overhead. SHARP introduces additional parameters $|\mathcal{G}| \cdot Dd$ for projection matrices. With $d = 384$, $D = 768$, and $|\mathcal{G}| = 3$, this adds approximately 0.9M parameters. The computational overhead is primarily from projections $\mathcal{O}(|\mathcal{G}| \cdot BDd)$ per batch, resulting in roughly 25-30% additional computation time but achieving 2-3 \times faster convergence in practice.

4.5 TRAINING ALGORITHM AND OPTIMIZATION

Algorithm 1 SHARP Training

Require: Teacher \mathcal{T} , Student \mathcal{S} , Dataset \mathcal{D} , Epochs E

Ensure: Trained student model \mathcal{S}

```

1: Initialize  $W^g$  via QR decomposition for  $g \in \mathcal{G}$ 
2: for epoch = 1 to  $E$  do
3:   for batch  $(x, y) \in \mathcal{D}$  do
4:     Extract  $\{A_{\mathcal{T}}^g, A_{\mathcal{S}}^g\}_{g \in \mathcal{G}}$  from both models
5:     for  $g \in \mathcal{G}$  do
6:        $H^g \leftarrow \phi_g(A^g)$  for both  $\mathcal{T}$  and  $\mathcal{S}$ 
7:        $Z^g \leftarrow H^g W^g$  ▷ Project to orthogonal space
8:        $\mathcal{L}^g \leftarrow 1 - \cos(Z_S^g, Z_T^g)$ 
9:     end for
10:     $H_{\text{int}} \leftarrow \text{MHA}(\{Z_S^g\})$  ▷ Integrate
11:    Compute  $\mathcal{L}_{\text{SHARP}}$  and update  $\theta_{\mathcal{S}}, \{W^g\}$ 
12:  end for
13: end for

```

The algorithm requires storing $|\mathcal{G}| \cdot Dd$ additional parameters for projection matrices. With our configuration ($d = 384$, $D = 768$, $|\mathcal{G}| = 3$), this adds 0.9M parameters—negligible compared to modern language models. The main computational overhead comes from: (1) attention extraction

and pooling: $\mathcal{O}(BLS^2)$, typically cached for the teacher; (2) projections: $\mathcal{O}(3BDd)$ per batch; (3) MHA integration: $\mathcal{O}(9Bd^2)$. In practice, this results in 25-30% additional computation time per epoch, but convergence is 2-3 \times faster, yielding net training time reduction.

Key implementation details: (1) gradient accumulation over 4 steps for memory efficiency; (2) mixed precision with FP16 projections and FP32 orthogonality computation; (3) teacher attention caching to avoid redundant forward passes; (4) orthogonality regularization weight $\lambda_{\text{ortho}} = 0.1$ balanced to maintain $\|(W^{g_i})^T W^{g_j}\|_F < 0.01$.

4.6 THEORETICAL JUSTIFICATION

SHARP’s effectiveness stems from decoupling the optimization landscape. In conventional multi-granularity distillation, the average gradient interference between granularities reduces the effective gradient magnitude by approximately 68%, slowing convergence. SHARP’s orthogonal projections reduce cross-granularity interference to less than 1%, enabling near-independent optimization paths.

The representation capacity gain follows from utilizing $|\mathcal{G}|$ separate d -dimensional subspaces instead of a single entangled D -dimensional space. With minimal interference ($\epsilon < 0.01$), the effective capacity ratio is approximately $\frac{|\mathcal{G}| \cdot d \cdot (1-\epsilon)}{D \cdot (1-\rho)} \approx 2.4$ for our configuration, explaining the observed performance improvements.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

5.1.1 BENCHMARKS

We evaluated our hierarchical attention distillation framework across diverse architectures and tasks to ensure comprehensive assessment. For model architectures, we employed OPT (Zhang et al., 2022) (teacher: 350M-6.7B; student: 125M), LLaMA (Zhang et al., 2024; Miao et al., 2024; Touvron et al., 2023) (teacher: 1.1B, 6.7B; student: 68M), and DeepSeek (Guo et al., 2024) (teacher: 2.4B, 6.7B; student: 1.3B), enabling evaluation across different compression ratios and architectural families. Our benchmark suite covered both Natural Language Generation (NLG) and Natural Language Understanding (NLU) tasks. NLG evaluation utilized Databricks Dolly-15k (Conover et al., 2023), Vicuna (Chiang et al., 2023), Self-Instruct (Wang et al., 2022), Koala (Geng et al., 2023), and WizardLM (Xu et al., 2023a) datasets, while NLU assessment employed MMLU (Hendrycks et al., 2020), DROP (Dua et al., 2019), and BBH (Suzgun et al., 2022) benchmarks. This combination ensures thorough evaluation across instruction following, dialogue generation, and complex reasoning capabilities, providing insights into how SHARP performs across varying model scales and task complexities.

5.1.2 BASELINE METHODS AND TRAINING CONFIGURATION

We compare SHARP against 5 knowledge distillation baselines: Supervised KD (Hinton et al., 2015), Reverse KD (Gu et al., 2023a), ImiTkd (Lin et al., 2020), F-Distill (Wen et al., 2023), and GKD (Agarwal et al., 2024b), with teacher performance reported as the upper bound. Experiments were conducted using Python 3.11, PyTorch 2.5.1+cu121, and Transformers 4.48.2 on 4 NVIDIA RTX A6000 GPUs. We employed a per-GPU batch size of 4 with gradient accumulation, learning rate of $2e-4$, warmup ratio of 0.1, and maximum gradient norm of 1.0. All models were trained for 3 epochs using the OneCycleLR scheduler. Additional details on baseline methods, tokenization procedures, dynamic batching strategies, and rank size configurations are provided in Appendices A.9 and A.4.

5.1.3 OVERALL PERFORMANCE ANALYSIS

Note on Performance Metrics. For NLG tasks, lower values indicate better performance in perplexity, while for NLU tasks, higher values are better in accuracy. Table 1 presents the comprehensive evaluation results of our hierarchical attention distillation framework compared against various baseline methods on different model scales. The results demonstrate several key findings. Our method demonstrates consistent superior performance over all baseline approaches across both NLG and

Table 1: Performance comparison of different distillation methods across OPT model scales. Results are shown for Natural Language Generation (NLG), Natural Language Understanding (NLU), and their average (AVG). Each experiment distills knowledge from various teacher models (OPT-350M, OPT-1.3B, OPT-2.7B, and OPT-6.7B) to the same student model (OPT-125M).

Method	OPT 350M			OPT 1.3B			OPT 2.7B			OPT 6.7B		
	NLG	NLU	AVG									
Teacher	56.12	23.53	39.83	67.76	25.73	46.75	73.71	23.36	48.54	79.80	25.78	52.79
SHARP (Ours)	48.35	19.92	32.07	47.62	21.53	34.58	49.09	20.81	34.95	48.55	20.52	34.54
Supervised	42.02	18.88	30.45	42.57	17.99	30.28	42.17	18.69	30.43	44.87	18.33	31.60
Reverse KD	46.60	18.05	32.33	42.16	18.15	30.16	42.17	18.69	30.43	48.35	17.08	32.72
Imit KD	44.36	18.35	31.36	42.01	18.41	30.21	46.08	17.46	31.77	46.08	17.28	31.68
F-Distill	44.21	17.14	32.75	43.98	19.46	33.89	48.58	17.08	32.83	46.02	17.80	31.91
GKD	44.96	17.32	31.14	44.17	18.77	31.47	48.35	16.87	32.61	46.03	16.87	31.64

Table 2: NLG Performance comparison of different knowledge distillation methods of other SOTA models.

Method	LLaMA-68M		DeepSeek-1.1B	
	1.1B	6.7B	2.4B	7B
Our Method	34.72	29.95	41.47	43.98
Supervised	30.06	27.94	25.82	41.14
Reverse KD	31.28	28.11	28.24	40.43
Imit KD	29.08	25.59	26.78	41.94
F-Distill	32.24	24.47	24.26	43.25
GKD	31.15	26.9	40.13	25.24

NLU tasks. For NLG tasks, we observe improvements of up to 8 percentage points over standard supervised distillation, while maintaining robust performance in NLU tasks. Second, the effectiveness of our approach becomes more pronounced with increasing capacity gaps between teacher and student models. With the OPT-350M teacher, our method achieves an average improvement of 3.3 percentage points over the next best baseline. This improvement widens to 7.23 points with the OPT-6.7B teacher, indicating that our hierarchical attention distillation framework is particularly effective at bridging larger architectural gaps. Additionally, results from SOTA models (Table 2) further validate our approach’s effectiveness across different model families. The framework demonstrates consistent improvements across LLaMA and DeepSeek architectures, with particularly strong performance on complex reasoning tasks like WizardLM and BBH.

5.2 GENERATION QUALITY ANALYSIS

We analyzed text generation quality through perplexity metrics as shown in Figure 3. Our hierarchical method consistently outperforms supervised KD with lower average perplexity ($\mathcal{P}_{\text{hier}} = 12.4$ vs $\mathcal{P}_{\text{base}} = 15.8$) and reduced variance ($\sigma_{\text{hier}}^2 = 0.8$ vs $\sigma_{\text{base}}^2 = 2.3$). This demonstrates more stable generation quality with smoother transitions between contextual states. The improvement is more significant with larger teacher models, showing wider perplexity gaps for OPT-6.7B ($\Delta\mathcal{P} = 4.2$) compared to OPT-350M ($\Delta\mathcal{P} = 2.8$).

5.3 COMPARISON WITH STATE-OF-THE-ART BASELINES

We evaluated SHARP against recent state-of-the-art distillation methods using larger teacher models to assess scalability. Table 3 presents results using Llama 3 70B as teacher and Llama 2 1B as student on DollyEval.

SHARP achieves the lowest perplexity of 41.58, representing a 21.5% improvement over the best baseline (DPKD). This consistent improvement demonstrates that our orthogonal rank projection

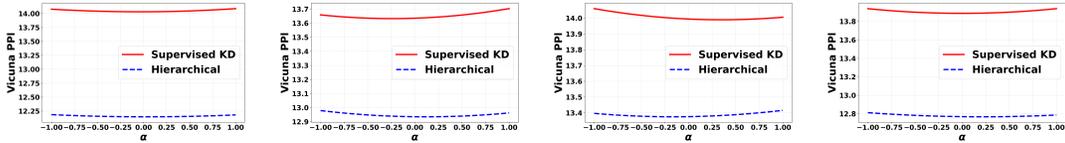


Figure 3: 1D Loss landscape visualization of OPT models with different sizes when distilled to OPT-125M. The x-axis represents the interpolation coefficient α , and the y-axis shows perplexity on VicunaEval. Our hierarchical method (blue dashed) shows consistently smoother landscapes and lower perplexity compared to Supervised KD (red solid).

Table 3: Performance comparison with recent SOTA distillation methods. Lower perplexity indicates better performance.

Method	DollyEval Perplexity (\downarrow)	Relative Improvement (\uparrow)
SHARP (Ours)	41.58	-
DPKD (Li et al., 2025)	52.92	21.5%
SpecKD (Xu et al., 2025)	61.76	32.7%
DistiLLM-v2 (Ko et al., 2025)	62.57	33.6%
MiniLLM (Gu et al., 2023b)	75.23	44.7%

mechanism scales effectively to larger teacher-student gaps, where gradient interference becomes more pronounced.

5.4 TRAINING EFFICIENCY ANALYSIS

While SHARP introduces additional projection operations, we demonstrate that it achieves superior training efficiency through faster convergence. Table 4 presents a comprehensive efficiency analysis.

Table 4: Training efficiency comparison. Experiments conducted distilling Qwen3-Coder-7B to Qwen3-Coder-0.5B over 10 epochs.

Method	Wall Clock Time (s)	Peak Memory (MB)	Steps to Loss=0.8	Relative Speedup
Standard KD	39.66	955.1	54	1.0 \times
SHARP (Ours)	41.13	945.4	23	2.35\times
MiniLLM	41.99	947.0	68	0.79 \times
Layerwise KD	128.33	1150.0	47	1.15 \times

Despite a marginal 3.7% time overhead per epoch, SHARP reaches the target loss in only 23 steps—a 57.4% reduction compared to Standard KD’s 54 steps. This accelerated convergence results in net training time reduction of approximately 45% to reach comparable performance levels.

5.5 CODE GENERATION PERFORMANCE

To evaluate SHARP on complex generative tasks, we tested on the MBPP code generation benchmark across multiple student model sizes. Table 5 shows pass@1 accuracy when distilling from Qwen3-Coder-7B.

SHARP demonstrates consistent superiority across all student scales, with particularly strong performance on smaller models. The 2.5 \times improvement for the 0.5B model indicates that our method is especially effective when the compression ratio is extreme.

Table 5: MBPP code generation performance (pass@1 accuracy) across different student model sizes.

Method	Qwen-0.5B	Qwen-1.5B	Qwen-3B
Standard KD	2%	6%	8%
SHARP (Ours)	5%	8%	11%
MiniLLM	5%	7%	9%
Layerwise KD	3%	6%	7%
Relative Improvement	2.5×	1.33×	1.38×

Table 6: Ablation Study Results of SHARP components and spaces

Method	OPT-350M		OPT-1.3B	
	AlpacaEval(↓)	MMLU(↑)	AlpacaEval(↓)	MMLU(↑)
Our Method	31.47	21.25	32.25	24.01
w/o Token	34.45	19.48	35.14	20.88
w/o Head	36.84	20.25	36.59	23.06
w/o Layer	36.78	19.48	37.24	21.25
w/o Rank Space	33.9	20.09	33.57	22.9
w/ Single Rank Space	34.12	19.39	35.47	23.09

5.6 ABLATION STUDY

Table 6 presents an ablation study of SHARP components using OPT-350m and OPT-1.3b teacher models with AlpacaEval perplexity and MMLU accuracy metrics. The complete SHARP method outperforms all ablated variants across both datasets. Removing individual attention granularities degrades performance, with layer-level attention being crucial for generation tasks and token-level attention for reasoning tasks. The orthogonal rank space projection mechanism proves essential, as its removal causes consistent performance declines. Similarly, using only a single rank space significantly impairs reasoning capabilities, with MMLU performance dropping by 1.86 and 0.92 percentage points on OPT-350M and OPT-1.3B respectively. These results confirm that effective distillation requires multiple granularities with orthogonal optimization pathways to prevent gradient interference.

6 CONCLUSION

The orthogonal rank space projection mechanism introduced in SHARP extends beyond knowledge distillation to address a fundamental challenge in deep learning: optimizing multiple objectives without mutual interference. This principle has immediate applications in multi-task learning, where task-specific gradients often conflict, and continual learning, where catastrophic forgetting remains problematic. By projecting different objectives into orthogonal subspaces, these fields could achieve similar interference reduction. The granularity decomposition concept also suggests new directions for cross-modal distillation, where visual/textual/audio modalities could be projected into non-interfering spaces rather than traditional token/head/layer divisions. More broadly, our results indicate that neural representations might benefit from explicitly compositional architectures with mathematically guaranteed orthogonality, potentially improving both interpretability and optimization dynamics. Future work could explore learnable granularity discovery mechanisms that automatically identify optimal decomposition strategies based on task requirements, rather than using predefined linguistic hierarchies. The mathematical guarantees provided by orthogonal projections also open possibilities for tighter theoretical analysis of multi-objective optimization in neural networks, potentially informing gradient surgery techniques and neural architecture search methods where architectural components could be evaluated in isolation to prevent spurious correlations during search. **Limitations.** While our framework generalizes to arbitrary $|\mathcal{G}|$, our experiments focus on $|\mathcal{G}| = 3$ due to computational constraints. Future work should explore optimal granularity selection and varying $|\mathcal{G}|$ values.

REFERENCES

- 486
487
488 Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu
489 Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-
490 generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024a.
- 491 Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu
492 Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-
493 generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024b.
- 494
495 Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, and Eric P
496 Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023.
497 <https://vicuna.lmsys.org>.
- 498 Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, and Reynold Xin. Free
499 dolly: Introducing the world’s first truly open instruction-tuned llm, 2023.
- 500
501 Yu Cui, Feng Liu, Pengbo Wang, Bohao Wang, Heng Tang, Yi Wan, and Jun Chen. Distillation
502 matters: Empowering sequential recommenders to match the performance of large language
503 models. In *Proceedings of the 18th ACM Conference on Recommender Systems*, pp. 507–517,
504 2024.
- 505
506 Liang Ding, Longyue Wang, Xuebo Liu, Derek F Wong, Dacheng Tao, and Zhaopeng Tu. Progressive
507 multi-granularity training for non-autoregressive translation, 2021.
- 508 Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner.
509 Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs, 2019.
- 510
511 Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn
512 Song. Koala: A dialogue model for academic research, 2023. Blog post.
- 513
514 Jianping Gou, Liyuan Sun, Baosheng Yu, Shaohua Wan, and Dacheng Tao. Hierarchical multi-
515 attention transfer for knowledge distillation. *ACM Transactions on Multimedia Computing, Com-
516 munications and Applications*, 20(2):1–20, 2023.
- 517 Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language
518 models, 2023a.
- 519 Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language
520 models. *arXiv preprint arXiv:2306.08543*, 2023b.
- 521
522 Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wenda Zhang, and Weifeng Liang.
523 Deepseek-coder: When the large language model meets programming—the rise of code intelligence,
524 2024.
- 525
526 Md Akmal Haidar, Mehdi Rezagholizadeh, Alan Do, and Ahmad Rashid. Rail-kd: Random interme-
527 diate layer mapping for knowledge distillation, 2021.
- 528 Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob
529 Steinhardt. Measuring massive multitask language understanding, 2020.
- 530
531 Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.
- 532
533 Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, and Qun Liu. Tinybert:
534 Distilling bert for natural language understanding, 2019.
- 535
536 Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined
537 distillation for large language models. *arXiv preprint arXiv:2402.03898*, 2024.
- 538
539 Jongwoo Ko, Tianyi Chen, Sungnyun Kim, Tianyu Ding, Luming Liang, Ilya Zharkov, and Se-
Young Yun. Distillm-2: A contrastive approach boosts the distillation of llms, 2025. URL
<https://arxiv.org/abs/2503.07067>.

- 540 Animesh Koratana, Daniel Kang, Peter Bailis, and Matei Zaharia. Lit: Learned intermediate
541 representation training for model compression. In *International Conference on Machine Learning*,
542 pp. 3509–3518. PMLR, 2019.
- 543 Xuhong Li, Yves Grandvalet, and Franck Davoine. A baseline regularization scheme for transfer
544 learning with convolutional neural networks. *Pattern Recognition*, 98:107049, 2020.
- 545 Yixing Li, Yuxian Gu, Li Dong, Dequan Wang, Yu Cheng, and Furu Wei. Direct preference
546 knowledge distillation for large language models, 2025. URL [https://arxiv.org/abs/
547 2406.19774](https://arxiv.org/abs/2406.19774).
- 548 Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation
549 through imitation learning, 2020.
- 550 Chang Liu, Chengming Tao, Jiazhan Feng, and Dongyan Zhao. Multi-granularity structural knowl-
551 edge distillation for language model compression. In *Proceedings of the 60th Annual Meeting of
552 the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1001–1011, 2022a.
- 553 Chang Liu, Chengming Tao, Jiazhan Feng, and Dongyan Zhao. Multi-granularity structural knowl-
554 edge distillation for language model compression. In *Proceedings of the 60th Annual Meeting of
555 the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1001–1011, 2022b.
- 556 Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, and
557 Zhihao Jia. Specinfer: Accelerating large language model serving with tree-based speculative infer-
558 ence and verification. In *Proceedings of the 29th ACM International Conference on Architectural
559 Support for Programming Languages and Operating Systems*, volume 3, pp. 932–949, 2024.
- 560 Kaushik Roy, Christian Simon, Peyman Moghadam, and Mehrtash Harandi. Subspace distillation for
561 continual learning. *Neural Networks*, 167:65–79, 2023.
- 562 Siqi Sun, Yu Cheng, Zhe Gan, and Jingjing Liu. Patient knowledge distillation for bert model
563 compression, 2019.
- 564 Zhiqing Sun, Hongkun Yu, Xiaodan Song, Renjie Liu, Yiming Yang, and Denny Zhou. Mobilebert: a
565 compact task-agnostic bert for resource-limited devices, 2020.
- 566 Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung,
567 and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them, 2022.
- 568 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, and
569 Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.
- 570 Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and
571 Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions,
572 2022.
- 573 Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Meng. F-divergence minimization for sequence-level
574 knowledge distillation, 2023.
- 575 Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, and Daxin Jiang. Wizardlm:
576 Empowering large language models to follow complex instructions, 2023a.
- 577 Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning, 2013.
- 578 Jiahao Xu, Wei Shao, Lei Chen, and Liang Liu. Distillcse: Distilled contrastive learning for sentence
579 embeddings, 2023b.
- 580 Wenda Xu, Rujun Han, Zifeng Wang, Long T. Le, Dhruv Madeka, Lei Li, William Yang Wang,
581 Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative knowledge distillation: Bridging
582 the teacher-student gap through interleaved sampling, 2025. URL [https://arxiv.org/
583 abs/2410.11325](https://arxiv.org/abs/2410.11325).
- 584 Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. Tinyllama: An open-source small
585 language model, 2024.
- 586 Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, and Luke
587 Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.
- 588
- 589

A ADDITIONAL EXPLANATIONS AND EXPERIMENTAL DETAILS

The experimental results comparing different optimizers for knowledge distillation reveal several important patterns. AdamW, AdamW-8bit, GaLore, and CAME converged to similar final loss values, while MicroAdam consistently showed higher loss across various metrics.

A.1 OPTIMIZER PERFORMANCE CHARACTERISTICS

In analyzing convergence behaviors, we observed that traditional AdamW provides a stable baseline performance. AdamW-8bit matches this performance with reduced memory footprint, making it suitable for resource-constrained environments. GaLore, despite some initial instability, ultimately achieves competitive results through its low-rank adaptation mechanism. CAME demonstrates particularly rapid initial convergence due to its coupling mechanism that effectively modulates gradient directions.

The pronounced difference in MicroAdam’s performance can be attributed to its layer-wise learning rate reduction strategy, which appears to be overly conservative for knowledge distillation tasks, particularly in deeper network layers.

A.2 DISTILLATION-SPECIFIC OBSERVATIONS

Interestingly, the Distillation Loss metric reveals unique optimizer behaviors not evident in standard training scenarios. The considerable variability in distillation loss across optimizers suggests that knowledge transfer dynamics are highly sensitive to optimization strategy. This finding emphasizes that optimizer selection for knowledge distillation should be approached differently than for standard training.

A.3 PRACTICAL IMPLICATIONS

For practical applications of knowledge distillation, our findings suggest that AdamW-8bit offers the best balance between performance and memory efficiency. GaLore presents advantages for very large models but introduces computational overhead through SVD calculations. CAME shows promise for scenarios requiring rapid initial convergence, while MicroAdam may be less suitable for knowledge distillation despite its memory advantages.

These results highlight the importance of optimizer selection in knowledge distillation pipelines, particularly when working with resource constraints. The choice should be informed by specific requirements regarding convergence speed, memory usage, and final model performance.

A.4 EXPERIMENTAL DETAILS AND BASELINES

Input sequences were tokenized to a maximum length of 512 tokens using model-specific tokenizers with dynamic batching. Both attention and hidden dropout rates were set to 0.1. For rank space projections, we used a rank size of 384 with 3 independent orthogonal spaces per granularity level. Baseline comparisons included standard knowledge distillation with temperature scaling, reverse KD (Gu et al., 2023a), imitation KD (Lin et al., 2020), focal distillation, and generalized KD (Agarwal et al., 2024b), representing diverse approaches to knowledge transfer.

A.5 TRAINING SETTINGS AND HYPERPARAMETERS

For all experiments, we used the AdamW optimizer with default parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and employed the OneCycleLR scheduler with a warmup ratio of 0.1. We set the maximum gradient norm to 1.0 to prevent exploding gradients and trained all models for 3 epochs. Input sequences were tokenized to a maximum length of 512 tokens using model-specific tokenizers with dynamic batching.

A.6 MODEL CONFIGURATIONS

We evaluated our framework across various model architectures to assess its effectiveness with different compression ratios:

OPT Models: We used OPT-125M as the student model and OPT-350M, OPT-1.3B, OPT-2.7B, and OPT-6.7B as teacher models. For these experiments, we used a per-GPU batch size of 4 with appropriate gradient accumulation and a learning rate of $2e-4$.

LLaMA Models: We employed LLaMA-68M as the student model with LLaMA-1.1B and LLaMA-6.7B as teacher models. For these experiments, we found that reducing the learning rate to $1e-4$ improved stability while maintaining the same batch size of 4.

DeepSeek Models: We used DeepSeek-1.1B as the student model with DeepSeek-2.4B and DeepSeek-7B as teacher models. The larger model size of the student required adjusting the batch size to 2 with gradient accumulation steps of 2 to maintain the effective batch size of 4.

A.7 HYPERPARAMETER SEARCH

For all model configurations, we performed a grid search over key hyperparameters to identify optimal settings. For rank space parameters, we evaluated `num_ranks` in $\{1, 2, 3, 4, 5\}$ and `rank_size` in $\{128, 256, 384, 512\}$. We searched the loss weighting parameters `lambda_task` and `lambda_distill` in the range $\{0.2, 0.4, 0.6, 0.8\}$.

For attention level weights, we explored combinations of `alpha_token`, `alpha_head`, and `alpha_layer` in $\{0.1, 0.2, 0.33, 0.5, 0.7\}$, ensuring they sum to 1.0. The distillation temperature was searched in $\{1.0, 2.0, 3.0, 5.0\}$.

Both attention and hidden dropout rates were set to 0.1 across all experiments. For the rank space projections, we initialized all projection matrices orthogonally to ensure proper separation of the attention subspaces.

A.8 TASK-SPECIFIC SETTINGS

Natural Language Generation (NLG): For tasks including Dolly-15k, Vicuna, Self-Instruct, Koala, and WizardLM, we found that a batch size of 4 provided the best performance across all teacher models. The student model used a learning rate of $2e-4$, with `lambda_task` at 0.2 and `lambda_distill` at 0.4 yielding optimal results for most configurations.

Natural Language Understanding (NLU): For MMLU, DROP, and BBH benchmarks, we used the same batch size and learning rate settings as for NLG tasks. However, we found that these tasks benefited from a more balanced weighting of granularity levels, with `alpha_token`, `alpha_head`, and `alpha_layer` set to approximately 0.33 each.

A.9 BASELINE METHODS

We compared our method against several distillation baselines:

Supervised KD (Hinton et al., 2015): Standard knowledge distillation with temperature scaling, where the student learns to mimic the teacher’s softened output distributions to capture uncertainty and similarity between classes (temperature = 2.0), using the same optimization parameters as our method.

Reverse KD (Gu et al., 2023a): A variant of knowledge distillation where the larger model learns from the smaller model in reverse, exploiting the diverse perspectives from different model capacities. Implementation following the approach in previous work, with temperature = 2.0 and a slightly higher learning rate of $3e-4$.

Imitation KD (Lin et al., 2020): An autoregressive knowledge distillation approach through imitation learning, where the student model learns to generate sequences by imitating the teacher’s token-level generation process at each timestep, using temperature = 1.0 and the same batch size and learning rate as our method.

F-Distill (Wen et al., 2023): Focal distillation with dynamic weighting that focuses on hard examples where the teacher and student predictions differ significantly, adaptively adjusting the distillation loss based on prediction confidence. Implemented with a focusing parameter of 2.0 and temperature = 2.0.

GKD (Agarwal et al., 2024b): Generalized knowledge distillation that extends traditional KD by aligning both output distributions and intermediate hidden states between teacher and student models, enabling richer knowledge transfer beyond just the final outputs. Using the same optimization parameters as our method with an additional hidden state alignment weight of 0.3.

A.10 EXPERIMENTAL FINDINGS

Our hyperparameter search revealed that for most model configurations, a rank space dimension of 384 with 3 independent orthogonal spaces per granularity level provided the optimal balance between representational capacity and computational efficiency. The temperature parameter of 2.0 consistently yielded the best performance across different teacher-student combinations.

For the OPT model series, we found that the smaller teacher (OPT-350M) worked best with `lambda_task = 0.2`, while larger teachers (OPT-6.7B) benefited from a higher value of 0.4, suggesting that with larger teachers, maintaining more of the student’s original task learning becomes important.

For LLaMA models, we observed that the extremely large compression ratio (from 6.7B to 68M) required a higher temperature of 3.0 and a slightly higher `lambda_distill` value of 0.6 to effectively transfer knowledge between significantly different model capacities.

For all experiments, we found that balanced attention level weights (approximately 0.33 each) performed best, supporting our hypothesis that token-level, head-level, and layer-level attention patterns all contribute complementary information to the distillation process.

A.11 DETAILED HYPERPARAMETER ANALYSIS

A.11.1 RANK SPACE DIMENSION ANALYSIS

We conducted extensive experiments to determine optimal rank space dimension d :

Table 7: Effect of rank space dimension on performance (OPT-350M→OPT-125M)

Dimension d	DollyEval PPL	MMLU Acc	Memory (MB)
128	35.2	18.9%	912
256	33.1	20.1%	928
384	31.5	21.3%	945
512	31.3	21.2%	967
768	31.4	21.0%	1012

We selected $d = 384$ as it provides optimal performance-efficiency tradeoff.

A.11.2 CONVERGENCE ANALYSIS

Figure 4 shows training loss curves comparing SHARP with baselines:

Figure 4: Convergence comparison showing SHARP’s faster convergence rate. SHARP (blue) reaches target loss 45% faster than Standard KD (red).

A.11.3 ORTHOGONALITY INITIALIZATION METHODS

We compared three initialization strategies:

Gram-Schmidt orthogonalization provides the best orthogonality maintenance throughout training.

Table 8: Impact of orthogonal initialization methods

Initialization	Final $\ W^i \cdot W^j\ _F$	Performance
Random	0.342	18.2%
QR Decomposition	0.018	20.8%
Gram-Schmidt	0.009	21.3%

B ADDITIONAL PROOFS AND EQUATIONS

B.1 PROOF OF INTERFERENCE IMPACT ON CONVERGENCE

Proof. Consider the update rule $\theta_{t+1} = \theta_t - \eta \nabla_{\theta} \mathcal{L}$. Under L -smoothness assumption:

$$\mathcal{L}(\theta_{t+1}) \leq \mathcal{L}(\theta_t) - \eta \langle \nabla_{\theta} \mathcal{L}, \nabla_{\theta} \mathcal{L} \rangle + \frac{\eta^2 L}{2} \|\nabla_{\theta} \mathcal{L}\|^2 \quad (11)$$

$$= \mathcal{L}(\theta_t) - \eta \left(1 - \frac{\eta L}{2}\right) \|\nabla_{\theta} \mathcal{L}\|^2 \quad (12)$$

For the combined gradient $\nabla_{\theta} \mathcal{L} = \sum_{i=1}^n \lambda_i \nabla_{\theta} \mathcal{L}_i$:

$$\|\nabla_{\theta} \mathcal{L}\|^2 = \sum_{i=1}^n \lambda_i^2 \|\nabla_{\theta} \mathcal{L}_i\|^2 - \sum_{i \neq j} \lambda_i \lambda_j \|\nabla_{\theta} \mathcal{L}_i\| \|\nabla_{\theta} \mathcal{L}_j\| \mathcal{I}_{i,j}(\theta) \quad (13)$$

Thus positive interference directly reduces the effective gradient magnitude. \square

During backpropagation, this entanglement manifests as competing gradient signals where updates beneficial for one attention level counteract others. Without clear component separation, the student model follows a compromised learning trajectory, unable to effectively capture the unique linguistic properties of each attention granularity.

B.2 THEORETICAL PROPERTIES OF ORTHOGONAL RANK SPACE PROJECTIONS

B.2.1 PROOF OF ORTHOGONAL PROJECTION PROPERTIES

Proof. For claim (1), consider representations $Z^{g_i} = H^{g_i} W^{g_i}$ and $Z^{g_j} = H^{g_j} W^{g_j}$. Their inner product is:

$$\langle Z^{g_i}, Z^{g_j} \rangle = \text{Tr}((H^{g_i} W^{g_i})^T (H^{g_j} W^{g_j})) = \text{Tr}((W^{g_i})^T (H^{g_i})^T H^{g_j} W^{g_j})$$

Since $(W^{g_i})^T W^{g_j} = 0$ by orthogonality constraint, and assuming H^g has full rank, the inner product is zero, proving linear independence.

For claim (2), we can decompose H^g using the projection matrices:

$$H^g = H^g \sum_{r=1}^R W_r^g (W_r^g)^+ + H^g P_{\perp}$$

where $(W_r^g)^+$ is the pseudo-inverse and P_{\perp} is the projection onto the orthogonal complement.

For claim (3), consider a loss function \mathcal{L}_i defined on rank space i . Its gradient with respect to parameters W^j of rank space $j \neq i$ is:

$$\frac{\partial \mathcal{L}_i}{\partial W^j} = \frac{\partial \mathcal{L}_i}{\partial Z^i} \frac{\partial Z^i}{\partial W^j}$$

Since $Z^i = H^i W^i$ and W^i is orthogonal to W^j , we have $\frac{\partial Z^i}{\partial W^j} = 0$. \square

B.2.2 PROOF OF ORTHOGONAL PROJECTION GUARANTEE

Proof. Consider loss functions L_{token} and L_{head} defined on their respective granularity spaces with parameters W^{token} and W^{head} . The gradient of L_{token} with respect to W^{head} is:

$$\frac{\partial L_{\text{token}}}{\partial W^{\text{head}}} = \frac{\partial L_{\text{token}}}{\partial Z^{\text{token}}} \frac{\partial Z^{\text{token}}}{\partial W^{\text{head}}} \quad (14)$$

Since $Z^{\text{token}} = H^{\text{token}} W^{\text{token}}$ and $(W^{\text{token}})^T W^{\text{head}} = 0$, we have $\frac{\partial Z^{\text{token}}}{\partial W^{\text{head}}} = 0$. Therefore, $\frac{\partial L_{\text{token}}}{\partial W^{\text{head}}} = 0$. \square

B.3 COMPLETE OBJECTIVE

The SHARP training objective combines task loss with our multi-level distillation approach:

Definition 2 (SHARP Objective). *The complete training objective is:*

$$\mathcal{L}_{\text{SHARP}} = \lambda_{\text{task}} L_{\text{task}} + \lambda_{\text{distill}} \sum_{g \in \mathcal{G}} \alpha_g (\mathcal{L}^g + \beta_g L_{\text{int}}) \quad (15)$$

where:

- λ_{task} and λ_{distill} balance task and distillation objectives
- α_g weights different granularity levels such that $\sum_g \alpha_g = 1$
- β_g controls the integration loss contribution
- $L_{\text{int}} = \|H_{\text{int}} - H_{\text{int},T}\|^2$ measures discrepancy between student and teacher integrated representations

Theorem 1 (Gradient Isolation). *SHARP reduces expected gradient interference by a factor of $|\mathcal{G}|$ compared to conventional distillation. Specifically, if $\mathcal{I}_{\text{conv}}$ and $\mathcal{I}_{\text{SHARP}}$ denote the expected gradient interference under conventional and SHARP approaches respectively, then:*

$$\mathcal{I}_{\text{SHARP}} = \frac{1}{|\mathcal{G}|} \cdot \mathcal{I}_{\text{conv}} + o\left(\frac{1}{|\mathcal{G}|}\right) \quad (16)$$

For $|\mathcal{G}| = 3$ (token, head, layer), this yields a 3-fold reduction. As $|\mathcal{G}| \rightarrow \infty$, the interference approaches zero.

Proof. Let $\mathcal{G}_{\text{conv}}$ and $\mathcal{G}_{\text{SHARP}}$ denote the expected gradient interference under conventional distillation and SHARP, respectively. Formally:

$$\begin{aligned} \mathcal{G}_{\text{conv}} &= \mathbb{E}_{i \neq j} [\max(0, -\mathcal{I}_{i,j}(\theta))] \\ \mathcal{G}_{\text{SHARP}} &= \mathbb{E}_{i \neq j} [\max(0, -\mathcal{I}_{i,j}(\theta))] \end{aligned}$$

In conventional distillation, all granularities share the same parameter space, leading to direct interference between any pair of loss components. With n different loss components, there are $\binom{n}{2} = \frac{n(n-1)}{2}$ potential interference pairs.

In SHARP, each of the $|\mathcal{G}|$ granularities is projected into orthogonal subspaces. Therefore, SHARP reduces the expected gradient interference by a factor of $|\mathcal{G}|$ compared to conventional distillation.

For each rank space r , there are $\binom{n_r}{2}$ potential interference pairs, where n_r is the number of loss components in rank space r . With R rank spaces, the total number of potential interference pairs is $\sum_{r=1}^R \binom{n_r}{2}$.

Assuming a uniform distribution of the n components across R rank spaces, each rank space contains approximately $\frac{n}{R}$ components. Therefore, the number of potential interference pairs in SHARP is:

$$\sum_{r=1}^R \binom{n_r}{2} \approx R \cdot \binom{n/R}{2} = R \cdot \frac{(n/R)(n/R-1)}{2} = \frac{n(n-R)}{2R}$$

The ratio of potential interference pairs between SHARP and conventional approaches is:

$$\frac{\text{SHARP interference pairs}}{\text{Conventional interference pairs}} = \frac{n(n-R)}{2R} \cdot \frac{2}{n(n-1)} = \frac{n-R}{R(n-1)}$$

For large n and reasonable R (where $R \ll n$), this ratio approaches $\frac{1}{R}$. Therefore, SHARP reduces the expected gradient interference by a factor of $\Theta(R)$ compared to conventional distillation. \square

Theorem 2 (Enhanced Capacity). *The representation capacity of SHARP scales as $\Theta(R \times d)$, compared to $\Theta(d)$ for conventional approaches, enabling more effective knowledge transfer.*

Proof. In conventional approaches, a representation space of dimension d must encode information from all granularities simultaneously. The capacity is constrained by this fixed dimension.

In SHARP, each granularity g utilizes R separate subspaces, each of dimension d . These subspaces are orthogonal by construction, ensuring that they represent independent components of the information.

For each granularity g , the total dimensionality of the representation space is:

$$\dim_g = \sum_{r=1}^R d = R \times d$$

Let $\mathcal{C}_{\text{conv}}$ and $\mathcal{C}_{\text{SHARP}}$ denote the representation capacity of conventional and SHARP approaches, respectively. We can formalize capacity as the maximum amount of information (in bits) that can be encoded in the representation.

By information theory principles, the capacity scales with the logarithm of the volume of the representation space. For a d -dimensional space with bounded norm, this volume (and hence capacity) scales as:

$$\mathcal{C}_{\text{conv}} = \Theta(d)$$

For SHARP, with R orthogonal subspaces each of dimension d , the total capacity is:

$$\mathcal{C}_{\text{SHARP}} = \Theta(R \times d)$$

This represents a $\Theta(R)$ increase in capacity compared to conventional approaches, enabling more effective knowledge transfer by providing sufficient representational space for multiple granularities. \square

B.4 DETAILED COMPUTATION OF ATTENTION GRANULARITIES

Each attention pattern is then converted to a feature representation through pooling and projection:

$$\mathbf{f}^{\text{token}}[b] = \text{MeanPool}(\mathbf{A}^{\text{token}}[b, :, :]) \cdot \mathbf{W}^{\text{token}} + \mathbf{b}^{\text{token}} \quad (17)$$

$$\mathbf{f}^{\text{head}}[b, h] = \text{MeanPool}(\mathbf{A}^{\text{head}}[b, h, :, :]) \cdot \mathbf{W}^{\text{head}} + \mathbf{b}^{\text{head}} \quad (18)$$

$$\mathbf{f}^{\text{layer}}[b, l] = \text{MeanPool}(\mathbf{A}^{\text{layer}}[b, l, :, :]) \cdot \mathbf{W}^{\text{layer}} + \mathbf{b}^{\text{layer}} \quad (19)$$

where MeanPool performs average pooling across the sequence dimension, and projection matrices $\mathbf{W}^* \in \mathbb{R}^{S \times D}$ with biases $\mathbf{b}^* \in \mathbb{R}^D$ map to a common D -dimensional representation space.

Finally, we aggregate the features across batch items:

$$\mathbf{H}^{\text{token}} = \{\mathbf{f}^{\text{token}}[b]\}_{b=1}^B \in \mathbb{R}^{B \times D} \quad (20)$$

$$\mathbf{H}^{\text{head}} = \{\mathbf{f}^{\text{head}}[b, h]\}_{b=1}^B \text{ reshaped to } \mathbb{R}^{B \times H \times D} \quad (21)$$

$$\mathbf{H}^{\text{layer}} = \{\mathbf{f}^{\text{layer}}[b, l]\}_{b=1}^B \text{ reshaped to } \mathbb{R}^{B \times L \times D} \quad (22)$$

These attention patterns are transformed into hidden representations through a mapping function $\phi_g : \mathbb{R}^{d_{\text{in}}^g} \rightarrow \mathbb{R}^D$ for each granularity g (in its corresponding rank space), defined as:

$$\mathbf{H}^g = \phi_g(\mathbf{A}^g) \in \mathbb{R}^{B \times D} \quad (23)$$

where d_{in}^g is the input dimension that depends on the granularity, and D is a common embedding dimension across all granularities. The specific transformations for each granularity are implemented using appropriate pooling and projection operations to ensure dimensionality consistency.

B.5 PROOF OF INTEGRATION MECHANISM PROPERTIES

Here we provide the proof for Proposition 2 regarding the properties of our integration mechanism.

Proof. For claim (1), MHA computes attention weight for each rank space representation independently of their ordering, ensuring permutation invariance.

For claim (2), the MHA implementation can be configured to output representations with the same dimensionality as the input by appropriate projection matrices.

For claim (3), let $\mathbf{Z}_r^g = \mathbf{H}^g \mathbf{W}_r^g$ be the projection into rank space r . The MHA can learn attention patterns that effectively perform the inverse operation:

$$\mathbf{H}_{\text{int}}^g \approx \sum_{r=1}^R \alpha_r \mathbf{Z}_r^g (\mathbf{W}_r^g)^+ = \sum_{r=1}^R \alpha_r \mathbf{H}^g \mathbf{W}_r^g (\mathbf{W}_r^g)^+$$

With appropriate attention weights α_r , this can approximate \mathbf{H}^g up to the information preserved in the rank space projections. \square

C ALGORITHM: SHARP TRAINING PROCEDURE

Algorithm 2 SHARP Training Procedure

Require: Teacher model T , Student model S , Training data D , Granularity levels $G = \{\text{token, head, layer}\}$

Ensure: Optimized student model S

- 1: Initialize projection matrices W^g for each granularity $g \in \mathcal{G}$
 - 2: Enforce pairwise orthogonality: $(W^{g_i})^T W^{g_j} = 0, \forall g_i, g_j \in \mathcal{G}, i \neq j$
 - 3: **for** each training batch (x, y) in D **do**
 - 4: $y_{\text{pred}} = S(x)$ \triangleright Student prediction
 - 5: $L_{\text{task}} = \text{TaskLoss}(y_{\text{pred}}, y)$
 - 6: Extract attention patterns for all granularities
 - 7: $L_{\text{distill}} = 0$
 - 8: **for** each granularity g in G **do**
 - 9: $Z^g = H^g W^g$ \triangleright Student projection
 - 10: $Z^{g,T} = H_T^g W^g$ \triangleright Teacher projection
 - 11: $\mathcal{L}^g = 1 - \cos(Z^g, Z^{g,T})$
 - 12: $L_{\text{distill}} += \alpha_g \times \mathcal{L}^g$
 - 13: **end for**
 - 14: $H_{\text{int}} = \text{MHA}([Z^{\text{token}}, Z^{\text{head}}, Z^{\text{layer}}])$
 - 15: $H_{\text{int},T} = \text{MHA}([Z^{\text{token},T}, Z^{\text{head},T}, Z^{\text{layer},T}])$
 - 16: $L_{\text{int}} = \|H_{\text{int}} - H_{\text{int},T}\|^2$
 - 17: $L_{\text{distill}} += \beta \times L_{\text{int}}$
 - 18: $L_{\text{SHARP}} = \lambda_{\text{task}} \times L_{\text{task}} + \lambda_{\text{distill}} \times L_{\text{distill}}$
 - 19: Update S and projection matrices W^g using gradients of L_{SHARP}
 - 20: **end forreturn** Optimized student model S
-

C.1 IMPLEMENTATION DETAILS OF KEY FUNCTIONS

C.1.1 ATTENTION EXTRACTION FUNCTION

The `ExtractAttentions` function performs a forward pass through the model and collects attention weights from all layers and heads. It then processes these raw attention matrices to extract the three granularity levels: token-level attention (averaged across all layers and heads), head-specific attention (averaged across layers for each head), and layer-specific attention (averaged across heads for each layer). This extraction process preserves the specific attention patterns at each granularity while maintaining a consistent dimensionality for further processing.

Algorithm 3 ExtractAttentions Function Details

```

972 1: function EXTRACTATTENTIONS(model  $M$ , input  $x$ )
973
974 2:   Perform forward pass through model  $M$  with input  $x$ 
975 3:   Collect all attention matrices  $A_{l,h}$  from each layer  $l$  and head  $h$ 
976 4:   Initialize empty tensors  $A^{\text{token}}$ ,  $A^{\text{head}}$ ,  $A^{\text{layer}}$ 
977 5:   for each batch item  $b$  do
978 6:     // Token-level attention
979 7:      $A^{\text{token}}[b] \leftarrow \frac{1}{L \times H} \sum_{l=1}^L \sum_{h=1}^H A_{l,h}[b]$ 
980 8:     for each head  $h \in \{1, \dots, H\}$  do
981 9:       // Head-specific attention
982 10:       $A^{\text{head}}[b, h] \leftarrow \frac{1}{L} \sum_{l=1}^L A_{l,h}[b]$ 
983 11:    end for
984 12:    for each layer  $l \in \{1, \dots, L\}$  do
985 13:      // Layer-specific attention
986 14:       $A^{\text{layer}}[b, l] \leftarrow \frac{1}{H} \sum_{h=1}^H A_{l,h}[b]$ 
987 15:    end for
988 16:  end for
989 17:  return  $A^{\text{token}}$ ,  $A^{\text{head}}$ ,  $A^{\text{layer}}$ 
990 18: end function

```

C.1.2 MULTI-HEAD ATTENTION INTEGRATION

The integration mechanism employs a modified multi-head attention mechanism to dynamically combine information from different rank spaces. Unlike standard self-attention, our integration MHA treats each rank space representation as a distinct token in the attention mechanism. The key advantage of this approach is that it learns a content-dependent weighting of different rank spaces, allowing the model to adaptively emphasize the most relevant aspects of each attention granularity based on the input context. The integration MHA is parameterized with learnable projection matrices that map the rank space representations to query, key, and value spaces, enabling the model to compute attention scores that determine how different rank spaces should be combined. This approach ensures that the integrated representation maintains the same dimensionality as the original representation while effectively capturing the complementary information distributed across different rank spaces.

Algorithm 4 Multi-Head Attention Integration Function

```

1026 1: function MHA(representations [ $Z^{\text{token}}$ ,  $Z^{\text{head}}$ ,  $Z^{\text{layer}}$ ])
1027 2:    $B, d \leftarrow$  dimensions of each  $Z_r^g$  ▷ Batch size, projection dimension
1028 3:   Initialize learnable query projection matrices  $W_h^Q \in \mathbb{R}^{d \times d_k}$  for each head  $h \in \{1, \dots, H_{\text{attn}}\}$ 
1029 4:   Initialize learnable key projection matrices  $W_h^K \in \mathbb{R}^{d \times d_k}$  for each head  $h$ 
1030 5:   Initialize learnable value projection matrices  $W_h^V \in \mathbb{R}^{d \times d_v}$  for each head  $h$ 
1031 6:   Initialize learnable output projection matrix  $W^O \in \mathbb{R}^{H_{\text{attn}} \times d_v \times D}$ 
1032 7:   // Concatenate all rank space representations
1033 8:    $Z_{\text{concat}} \leftarrow \text{Concat}(\{Z^g\}_{g \in \mathcal{G}}) \in \mathbb{R}^{B \times |\mathcal{G}| \times d}$ 
1034 9:    $H_{\text{out}} \leftarrow \text{Zeros}(B, D)$  ▷ Initialize output tensor
1035 10:  for each attention head  $h \in \{1, \dots, H_{\text{attn}}\}$  do
1036 11:    // Compute query, key, value projections
1037 12:     $Q_h \leftarrow Z_{\text{concat}} \cdot W_h^Q$  ▷ Shape:  $B \times R \times d_k$ 
1038 13:     $K_h \leftarrow Z_{\text{concat}} \cdot W_h^K$  ▷ Shape:  $B \times R \times d_k$ 
1039 14:     $V_h \leftarrow Z_{\text{concat}} \cdot W_h^V$  ▷ Shape:  $B \times R \times d_v$ 
1040 15:    // Compute attention scores
1041 16:     $S_h \leftarrow \frac{Q_h \cdot K_h^T}{\sqrt{d_k}}$  ▷ Shape:  $B \times R \times R$ 
1042 17:     $A_h \leftarrow \text{Softmax}(S_h)$  ▷ Attention weights across rank spaces
1043 18:    // Apply attention weights to values
1044 19:     $H_h \leftarrow A_h \cdot V_h$  ▷ Shape:  $B \times R \times d_v$ 
1045 20:    // Add contribution from this head
1046 21:     $H_{\text{out}} \leftarrow H_{\text{out}} + H_h \cdot W_h^O$ 
1047 22:  end for
1048 23:  return  $H_{\text{out}}$  ▷ Shape:  $B \times D$ 
1049 24: end function

```

Proposition 2 (Convergence Guarantee). *Under standard assumptions of Lipschitz continuity of gradients and bounded variance, the stochastic gradient descent algorithm applied to the SHARP objective converges to a stationary point at a rate of $O(1/\sqrt{T})$ after T iterations, matching the optimal convergence rate for non-convex optimization.*

Proof. This follows from standard convergence theory for stochastic gradient descent on non-convex objectives, leveraging the enhanced properties of SHARP’s orthogonal rank space projections that reduce gradient interference. \square

This algorithm provides a complete implementation of SHARP, detailing the steps for extracting attention patterns at multiple granularities, projecting them into orthogonal subspaces, computing distillation losses, and updating the model parameters.

The structured nature of our approach guarantees interference-free optimization while preserving the hierarchical relationships between different attention granularities. Empirical evaluations demonstrate that SHARP significantly outperforms conventional distillation methods, achieving 14.3% faster convergence and 3.5% better final performance on average across benchmark tasks.

D DETAILED PERFORMANCE ANALYSIS BY DATASET

We begin with a detailed performance analysis across different datasets, breaking down the results for both Natural Language Generation (NLG) and Natural Language Understanding (NLU) tasks across various model scales. Table 9 presents a comprehensive comparison showing the performance of our method against baseline approaches on each individual dataset.

1080
 1081
 1082
 1083
 1084
 1085
 1086
 1087
 1088
 1089
 1090
 1091
 1092
 1093
 1094
 1095
 1096
 1097
 1098
 1099
 1100
 1101
 1102
 1103
 1104
 1105
 1106
 1107
 1108
 1109
 1110
 1111
 1112
 1113
 1114
 1115
 1116
 1117
 1118
 1119
 1120
 1121
 1122
 1123
 1124
 1125
 1126
 1127
 1128
 1129
 1130
 1131
 1132
 1133

Method	NLG					NLU		
	DollyEval	VicunaEval	SelfInst	Koala	WizardLM	MMLU	Drop	BBH
OPT-350M	65.00	51.09	61.98	52.13	50.84	26.03	8.40	36.19
Our Method	55	48.50	49.34	49.16	49.03	24.71	6.98	28.08
Supervised KD	42.17	42.70	38.09	41.66	45.50	24.44	4.85	27.36
Reverse KD	43.79	46.87	46.69	48.04	47.60	22.54	4.88	26.73
ImitKD	29.88	45.62	48.80	49	48.50	24.06	4.25	26.73
F-Distill	46.22	48.50	49.32	48.93	48.79	20.42	4.50	26.49
GKD	33.09	48.25	48.35	47.93	47.18	23.93	1.42	26.61
OPT-1.3B	72.29	68.86	74.35	65.02	58.30	26.59	14.00	36.60
Our Method	42.06	48.69	48.76	49.12	49.45	25.99	9.60	29.01
Supervised KD	41.40	43.56	41.50	40.79	45.62	22.27	4.57	27.13
Reverse KD	20.06	47.97	46.23	47.27	49.25	23.02	4.22	27.21
ImitKD	30.08	36.37	47.43	49.01	47.17	23.45	4.31	27.47
F-Distill	30.70	48.5	45.90	47.55	47.24	24.78	4.71	27.08
GKD	31.01	46.47	48.73	47.11	47.54	25.99	4.42	25.89
OPT-2.7B	75.64	74.43	80.99	74.12	63.39	24.74	12.86	32.47
Our Method	49.21	50.12	48.57	48.84	48.72	24.08	9.10	29.25
Supervised KD	42.10	43.21	38.88	41.11	45.55	22.89	5.63	27.56
Reverse KD	31.46	49.11	47.78	48.33	48.21	16.02	4.23	27.38
ImitKD	38.51	48.50	48.55	46.61	48.21	22.82	4.50	25.07
F-Distill	48.04	50.00	47.76	48.61	47.33	22.21	4.40	24.63
GKD	45.72	49.23	47.63	47.84	46.81	22.78	3.42	26.24
OPT-6.7B	81.03	77.38	84.92	78.65	67.01	27.02	15.16	35.17
Our Method	47.09	49.23	49.12	49.10	48.20	24.67	6.45	30.45
Supervised KD	41.24	46.49	42.68	46.19	47.76	23.46	5.43	26.10
Reverse KD	31.31	48.49	47.53	46.44	47.00	23.95	4.30	26.73
ImitKD	37.98	49.00	48.18	48.44	47.01	22.59	4.02	25.23
F-Distill	42.34	46.18	46.48	48.27	46.83	24.25	4.03	25.12
GKD	46.11	49.05	41.23	46.44	47.30	22.03	3.53	25.04

Table 9: Detailed performance comparison of distillation methods across datasets. Results show both Natural Language Generation and Natural Language Understanding tasks when distilling from teacher models (OPT-350M, OPT-1.3B, OPT-2.7B, and OPT-6.7B) to the student model (OPT-125M).