CONTRA-KD: CONTINUOUS TRAJECTORY ALIGNMENT FOR KNOWLEDGE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

While Knowledge Distillation (KD) enables efficient compression of LLMs, current methods focus on matching discrete layer outputs, which fails to capture the continuous, dynamic nature of feature evolution within these models. We propose **CONTRA-KD**, a new paradigm for KD that reframes the process through continuous-time dynamics and the ordinary differential equations (ODE)-based nature of LLMs. We first learn a student-to-teacher correction flow in feature space using Rectified Flow, which provides a geometrically simple and stable transport path between intermediate representations. We then introduce a curriculum-based distillation strategy that guides the student to progressively follow this flow, targeting dynamic intermediate targets throughout the training process. This design enables the student to align not only with the teacher's representations but also with its underlying feature dynamics, while remaining orthogonal and complementary to standard output-level distillation objectives. Extensive experiments across multiple teacher–student pairs and instruction-following benchmarks demonstrate that CONTRA-KD consistently improves distillation performance and training stability over strong baselines.

1 INTRODUCTION

Large Language Models (LLMs) have achieved remarkable capabilities, thanks to their immense scale with leading models now contain billions of parameters Touvron et al. (2023). However, this scale creates a substantial barrier for practical deployment, especially in constrained, low-resource settings. Knowledge Distillation Hinton et al. (2015) offers a compelling solution, by compressing a large, powerful teacher model into a smaller, faster student model by transferring knowledge between the two.

The dominant line of work distills LLMs purely at the level of final output distributions. The foundational recipe minimizes the forward KL divergence between the teacher and student output distributions Hinton et al. (2015); Bucila et al. (2006). MiniLLM Gu et al. (2024) instead optimizes reverse KL on student-generated samples, addressing the exposure-bias mismatch that arises under teacher-forced training. Building on this on-policy perspective, subsequent work explores a broader family of divergences and teacher–student distribution constructions, including generalized divergences, skewed objectives, contrastive formulations, and adaptive intermediate distributions Agarwal et al. (2024); Ko et al. (2024; 2025); Shin et al. (2026). Other recent studies focus on the sampling and data-generation process itself, using interleaved student–teacher generation or difficulty-aware data selection to reduce the teacher–student mismatch Xu et al. (2025); He et al. (2025). Despite these refinements, the supervision signal in this family remains fundamentally anchored to the model's final output: whether the divergence is measured on probabilities, logits, or student-generated rollouts, the internal computation that transforms one representation into another remains largely unconstrained.

Grounded in the theory of Ordinary Differential Equations (ODEs), this motivates the interpretation of treating the layer-wise progression of a Transformer as a sequence of discrete time steps in a continuous evolution of feature representations Chen et al. (2018); Tong et al. (2025). Under this lens, the teacher's deeper architecture can be viewed as a fine-grained numerical solver tracing a detailed path through feature space, while the student's shallower architecture acts as a coarser solver that attempts to approximate the same underlying trajectory. A growing line of work has begun to target this internal computation directly rather than supervising only the output layer. Representation-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

based approaches align hidden states, attention structure, or relational geometry across selected layers, while more recent methods improve robustness to architectural or dimensional mismatch and account for residual discrepancies between teacher and student representations Zhang et al. (2024); Dasgupta & Cohn (2025); On et al. (2026). Beyond matching individual states, recent work has explicitly considered the dynamics of intermediate features, supervising layer-to-layer changes in addition to hidden representations and aligning trajectories at multiple granularities Gong et al. (2025); Chi et al. (2026). These developments are important because they move distillation from output imitation toward supervision of the computation itself. Nevertheless, the teacher's trajectory is still represented through a finite collection of layer checkpoints: states, finite differences, or relational structures are evaluated at selected depths, while the transformation connecting neighboring states is not modeled explicitly. The geometry of how a representation is transported between observed checkpoints therefore remains largely unconstrained. Concurrently, the emerging literature on continuous-time flow matching Lipman et al. (2022); Liu et al. (2022) provides a natural mechanism for learning such continuous transformations through a velocity field that transports one distribution toward another. This perspective complements the ODE view of Transformers introduced above: *distillation can be recast as learning a continuous transport process that guides the student's representations along a trajectory consistent with the teacher's flow.*

Grounded in this insight, we introduce **CONTRA-KD**, a new framework that formulates KD as a rectified flow matching problem in feature space. Rather than forcing the student to replicate the teacher's full and often noisy dynamics, CONTRA-KD first learns an idealized rectified flow: a straight-line ODE capturing the most direct transformation from a student's feature state to its corresponding teacher state. We prove that minimizing this trajectory discrepancy also bounds the discrepancy between the two models' underlying velocity fields, so distilling the hidden-state flow alone implicitly carries the teacher's dynamics to the student, without ever matching velocities directly. However, matching a distant, information-rich teacher state from the start of training produces noisy, unstable updates. We therefore guide the student along the flow through a curriculum, aligning short segments first and lengthening them as training progresses. In the second stage, the student aligns its own layer-wise updates with a discretized simulation of this flow, replacing brittle, discrete state matching with a continuous dynamical perspective.

The contributions of our study are as follows:

- We show that trajectory alignment implicitly induces dynamic alignment: minimizing the discrepancy between student and teacher hidden-state trajectories bounds the discrepancy between their governing velocity fields, so that distilling the continuous flow is sufficient to transfer the teacher's underlying dynamics.
- We propose a new framework, Continuous Trajectory Alignment for Knowledge Distillation (**CONTRA-KD**). CONTRA-KD first learns a universal, student-to-teacher correction flow. In the second stage, the student is trained via a discretized simulation of this rectified flow, using it as a dynamic target for more stable and efficient feature evolution. We further extend this design to the cross-tokenizer scenario, enabling distillation between teacher-student pairs with different vocabularies.
- We show that trajectory alignment without a curriculum is unstable, as forcing the student to match distant, information-rich teacher states early in training destabilizes optimization. This motivates **curriculum-based trajectory distillation**, which progressively increases distillation difficulty by distilling small-step dynamics first and gradually extending the step size, stabilizing optimization and enabling effective velocity and trajectory alignment.

2 PRELIMINARY

2.1 KNOWLEDGE DISTILLATION IN LLMS

Knowledge distillation (KD) Hinton et al. (2015) is a technique for model compression where a large teacher model transfers its knowledge to a smaller student model. The conventional KD framework primarily achieves this by aligning the final output distributions (logits) of the student with those of the teacher. Specifically, for a final-layer student hidden state $h_S^{L_S}$ and teacher hidden state $h_T^{L_T}$, where the student and the teacher have L_S and L_T layers respectively, the loss minimizes the

Kullback-Leibler (KL) divergence between the probability distributions produced by their respective language modeling heads, $f_{\text{head}}(\cdot)$:

$$\mathcal{L}_{KD} = \mathbb{E}_{x\sim data}[D_{KL}(f^T_{head}(h^{L_T}_T)||f^S_{head}(h^{L_S}_S))] \tag{1}$$

However, this approach often fails to capture the rich information encoded within the immediate representations of LLMs. Feature Dynamic Distillation (FDD) Gong et al. (2025) addresses this limitation by framing the distillation problem as aligning the entire feature dynamics between models, including additional loss term that match both the feature trajectory and its first-order derivative. Our work further extends this approach to a continuous path-based approach to feature-space alignment.

2.2 Dynamic perspective of LLMs

A foundational perspective for neural networks is to view their discrete, layer-wise structure as an Euler discretization of a continuous-time system governed by an ordinary differential equation (ODE) Chen et al. (2018). In this paradigm, a neural network is not a sequence of layers but a model that parameterizes a continuous vector field. This vector field defines the instantaneous rate of change of the hidden state: $\frac{d\mathbf{h(t)}}{dt} = f\big(h(t), t, \theta\big)$. From this dynamic perspective, a student model can be seen as a coarse discretization of the feature dynamics, where a larger model represents a finer representation of the same process Gong et al. (2025). This suggests that effective knowledge transfer should align the underlying continuous dynamics that govern feature evolution.

2.3 Continuous-time flow-based models

Continuous Normalizing Flows transform a prior distribution p_0 into a target distribution p_1 via a time-dependent vector field u_t, with the flow $\phi_t(x)$ following the ODE $\frac{d}{dt}\phi_t(x) = u_t(\phi_t(x))$, $\phi_0(x) = x$. Since the marginal path induced by this ODE is intractable to model directly, Flow Matching Lipman et al. (2022) instead trains a neural vector field v_θ to regress the velocity of a tractable conditional interpolation path $X_t = \psi_t(x_0, x_1)$ between paired samples $(x_0, x_1) \sim \pi$. Rectified Flow Liu et al. (2022) instantiates this idea with a linear interpolation path, $X_t = (1 - t)x_0 + tx_1$, under which the conditional velocity field becomes the constant vector $x_1 - x_0$, reducing the objective to a simple least-squares regression:

$$\mathcal{L}_{\text{RF}}(\theta) = \mathbb{E}_{t\sim\mathcal{U}(0,1),\,(x_0,x_1)\sim\pi}\left[\|(x_1 - x_0) - v_\theta(X_t, t)\|^2\right]. \tag{2}$$

This construction induces straight-line trajectories in Euclidean space and admits exact simulation with a single Euler step, providing a geometrically principled and computationally efficient foundation that we later leverage for feature-space alignment in CONTRA-KD.

3 Method

3.1 Our Motivation

Trajectory alignment implies dynamic alignment. Under the perspective of continuous-time paradigm, the teacher and student models are dynamic systems over layer index t[1] governed by distinct velocity fields. Let the teacher's and student's evolution be governed by velocity fields v_T and v_S. Their hidden state trajectories, $h_T(t)$ and $h_S(t)$[2], are solutions of:

$$\frac{dh_T(t)}{dt} = v_T(h_T(t), t); \quad \frac{dh_S(t)}{dt} = v_S(h_S(t), t) \tag{3}$$

To ensure robust knowledge transfer, the student must learn from the teacher's internal dynamics. Mathematically, this is equivalent to minimizing the difference between the trajectories generated

[1]Here, t implies the continuous time variable over which the hidden representations transform; it is equivalent to the discrete layer indices of Transformer models.

[2]$h_T(t)$ and $h_S(t)$ are indeed $h_T(x, y, t)$ and $h_S(x, y, t)$ where (x, y) represents a pair of input and output sampled from the ground-truth data distribution. We ignore them for brevity.

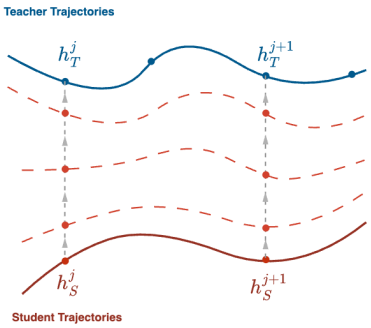


Figure 1: Curriculum learning through continuous flow. The blue line denotes teacher trajectories, the red line denotes student trajectories, and the dotted gray arrow indicates our learned student–teacher flow. In early step of training, the student follows a short segment of this flow with an intermediate target that is easier to optimize, and gradually increases the step sizes as training progresses.

by their respective vector fields:

$$\mathbb{E}_{t\sim U(0,1)}\left[\|h_T(t)-h_S(t)\|_{\mathcal{L}_2}\right]=\mathbb{E}_{t\sim U(0,1)}\left[\left[\int_\Omega\left[h_T(t;x,y)-h_S(t;x,y)\right]^2dxdy\right]^{1/2}\right], \quad (4)$$

where Ω is the domain of (x, y).

Let us denote $\delta_t = v_S(h_T(t), t) - v_T(h_T(t), t)$. In the following theorem, we prove that the trajectory distillation can lead to the dynamic or velocity distillation.

Theorem 3.1. *Assume that the teacher and student velocities satisfy M-Lipchitz. We have the following inequality:*

$$\|\delta_t\|_{\mathcal{L}_2}\leqslant M\|h_T(t)-h_S(t)\|_{\mathcal{L}_2}+\|\frac{d}{dt}\left[h_S(t)-h_T(t)\right]\|_{\mathcal{L}_2}, \quad (5)$$

where $\|f\|_{\mathcal{L}_2}:=\left(\int f(a)^2\,da\right)^{1/2}$.

The detailed theoretical proof is provided in Appendix B. Equation equation 5 shows a direction connection between trajectory alignment and velocity alignment, which bounds the dynamic difference δ_t in terms of the trajectories and its rate of change. When the student is trained to distill the teacher's hidden states, the state difference $\|h_T(t)-h_S(t)\|_{\mathcal{L}_2}$ and its temporal derivative $\|\frac{d}{dt}\left[h_S(t)-h_T(t)\right]\|_{\mathcal{L}_2}$ are driven to 0. Consequently, the dynamic or velocity difference δ_t also converges to 0. This implies that matching the trajectories implicitly enforces the student's dynamics to follow the teacher's dynamics. Therefore, trajectory distillation leads to dynamic distillation, ensuring the student not only reproduces the teacher's states but also learns its underlying dynamics.

Curriculum learning through continuous flow. While Theorem 3.1 shows that aligning student and teacher trajectories implicitly enforces dynamic alignment, directly matching trajectories can be difficult, especially at early training stages when the student's representations are far from teacher's. This motivates a curriculum-based strategy, where the student is not forced to immediately match the teacher's states but instead progress toward them through a sequence of intermediate targets (Figure 1). To realize this curriculum, we define a *straight flow* connecting corresponding representations of student and teacher, where intermediate states of this flow serve as progressively easier alignment targets. As training progresses, the step size is gradually increased, enables the student to traverse longer segments and progressively approach teacher's trajectory. Further analysis can be found in Section 5.1.

3.2 Student-Teacher Continuous Flow

Formally, let $h_S^j \in \mathbb{R}^{N\times d_S}$ and $h_T^j \in \mathbb{R}^{N\times d_T}$ be the feature representations coming out from a selected intermediate layer j of a student and teacher model, respectively, for a given input with sequence length N. Typically, they reside in spaces of different dimensions. With the assumption that the teacher's hidden space is better trained and can express more information, we first project the student's hidden states to the teacher's space and use these projected hidden states for later steps. This projection can be parameterized as simple as a weight matrix $W^{S\to T}\in\mathbb{R}^{d_S\times d_T}$:

$$f_{\text{proj}}(h_S^j):=h_S^jW^{S\to T}\in\mathbb{R}^{N\times d_T}$$

A second challenge is the mismatch in the number of layers ($L_S \neq L_T$). To tackle this, we use a uniform layer scheduling schema by selecting a subset of L_D layers from both the student and the teacher and form pairs of aligned layers. We denote $I_S = \{l_1^S, l_2^S, ..., l_{L_D}^S\}$ and $I_T = \{l_1^T, l_2^T, ..., l_{L_D}^T\}$ as the ordered sets of selected layer indices for the student and the teacher, respectively.

The transformation is modeled as the trajectory of a time-dependent ODE, where for the i-th token and the j-th layer pair in our schedule, a continuous state $Z_t^{(i,j)}$ evolves from $t = 0$ to $t = 1$ according to: $\frac{dZ_t^{(i,j)}}{dt} = v_\theta(Z_t^{(i,j)}, t, j)$. The trajectory starts at the student's distribution $Z_0^{(i,j)}$ and is guided to the teacher's distribution $Z_1^{(i,j)}$. We denote $Z_0^{(i,j)} := h_S^j[i]$ as the student's hidden representation of the i-th token in the token sequence at the j-th aligned layer (i.e., the l_j^S-th layer in the student model) and, similarly, $Z_1^{(i,j)} := h_T^j[i]$ as the teacher's hidden representation of the same token at the teacher's l_j^T-th layer. The neural network v_θ, conditioned on distillation step index j, parameterizes the dynamics of this transformation, learning the direction and magnitude of feature correction at any intermediate point (Z_t) along the path.

In Euclidean geometry of feature spaces, the straight path is the shortest possible route between two endpoints and can be simulated without time discretization. Therefore, we constrain our learned ODE to follow this trajectory. To achieve this, we apply the learning objective of Rectified Flow. For any pair of corresponding states $(h_S^j[i], h_T^j[i])$, we define a linear interpolant $H_t^{(i,j)}$:

$$H_t^{(i,j)} := (1-t)\, h_S^j[i] + t\, h_T^j[i] \quad (6)$$

The target velocity vector required to traverse this path is a constant vector, that is, the vector from the student's state to the teacher's state:

$$\frac{dH_t^{(i,j)}}{dt} = h_T^j[i] - h_S^j[i] \quad (7)$$

Algorithm 1 Training the Velocity Field v_θ

Input: teacher p, student q, projector f_{proj}, velocity field v_θ with parameter θ, data $\mathcal{D}$, layer maps I^S, I^T
Output: Trained velocity field v_θ
for each batch $\{x_i\}$ in $\mathcal{D}$ **do**
  Initialize $\mathcal{L}(\theta) \leftarrow 0$
  for $j = 1, 2, ..., L_D$ **do**
    $h_S^j \leftarrow f_{\text{proj}}\big(q(\{x_i\}, \text{layer} = l_j^S)\big)$
    $h_T^j \leftarrow p(\{x_i\}, \text{layer} = l_j^T)$
    $t \sim U(0,1)$
    $H_t^j \leftarrow (1-t) \cdot h_S^j + t \cdot h_T^j$
    $\mathcal{L}(\theta) \leftarrow \mathcal{L}(\theta) + \left\| (h_T^j - h_S^j) - v_\theta(H_t^j, t, j) \right\|^2$
  end for
  Update θ using gradients from $\mathcal{L}(\theta)$
end for
return v_θ

Our goal is to train a velocity field network v_θ that perfectly replicates this target vector field. The learning objective for v_θ for an input data sample is as follows:

$$\mathcal{L}(\theta) = \mathbb{E}_{i,j,t}\left[\left\| \left(h_T^j[i] - h_S^j[i] \right) - v_\theta\left(H_t^{(i,j)}, t, j \right) \right\|^2 \right] \quad (8)$$

The expectation is taken over sequence of tokens i, distillation steps $j \in \{1, ..., L_D\}$, and time steps $t \sim U(0,1)$. The overall training procedure of the velocity field is detailed in Algorithm 1.

3.3 Curriculum-based Trajectory Distillation

Curriculum-based approach. Directly distilling the teacher trajectory by alignment between $\phi_0(h_S) = h_S$ and $\phi_1(h_S) = h_T$ can be challenging at early stages of training, where the student representation is still far from the teacher's and can lead to unstable update signals. To address this issue, we introduce a **curriculum-based trajectory distillation** strategy that effectively ultilizes supervision from the learned flow.

Specifically, for a flow $\phi_t(x)$ and corresponding velocity field u_t, we define an intermediate target $\tilde{\phi}_k(x)$ at time step k from $\phi_0(x) = x$, where $k \in (0, 1]$. Intuitively, $\tilde{\phi}_k(x)$ lies partway along the flow induced by u_t at distance defined by k rather than the endpoint $\phi_1(x)$. The target can be effectively

approximated by: $\tilde{\phi}_k(x) := \phi_0(x) + \int_0^k u_t(\phi_t(x))\, dt$. For a smaller value of k, the target is easier to follow, forming a curriculum that progressively increases in difficulty as training proceeds.

Distillation process. Let $h_S^j[i]$ denote the projected hidden representation of the i-th token at the j-th aligned layer for a data sample. The corresponding curriculum-based target is defined as an Euler step from the student's representation:

$$\tilde{h}_S^j[i] := \mathrm{sg}\left[h_S^j[i] + k \cdot v_\theta\left(h_S^j[i],\, t=0,\, j\right)\right], \tag{9}$$

where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator. During training, k is gradually increased based on training iterations: $k = \frac{t}{T}$, where t is the iteration index and T is the total number of training iterations. The CONTRA loss then minimizes the discrepancy between the student's layer output and this dynamically generated target:

$$\mathcal{L}_{\text{CONTRA}} = \frac{1}{NL_D} \sum_{j=1}^{L_D} \sum_{i=1}^{N} D_{\cos}\left(h_S^j[i],\, \tilde{h}_S^j[i]\right) \tag{10}$$

Because $\tilde{h}_S^j[i]$ is detached, minimizing $D_{\cos}(h_S^j[i], \mathrm{sg}[\tilde{h}_S^j[i]])$ forces the gradient of $h_S^j[i]$ to align with the vector direction prescribed by $v_\theta(h_S^j[i], 0, j)$. This updates the student hidden state along the local tangent of the Rectified Flow vector field toward the teacher representation space. We employ cosine distance over ℓ_2 distance to focus optimization on directional alignment, avoiding gradient instability in high-dimensional feature spaces. To prevent distributional shift between the student and the learned flow, we alternate between updating the student on $\mathcal{L}_{\text{CONTRA}}$ and re-adapting v_θ and f_{proj} following Algorithm 2.

3.4 Cross-tokenizer Mechanism

Span Construction via Longest Common Subsequence. When teacher and student use different tokenizers, their tokenizations of the same input x produce sequences of different lengths, making direct token-level alignment infeasible. To resolve this, we note that both tokenizers operate over the same character stream, so each token can be annotated with its character-level end offset. This gives two offset sequences $\mathbf{o}^T = (\mathrm{off}_1^T, \ldots, \mathrm{off}_{n_T}^T)$ and $\mathbf{o}^S = (\mathrm{off}_1^S, \ldots, \mathrm{off}_{n_S}^S)$ for the teacher and student, respectively. Special tokens are assigned offset 0 and excluded from alignment. In addition to LCS, we also investigate an alternative edit-distance alignment strategy in Appendix F.4.

We compute the Longest Common Subsequence (LCS) of $\mathbf{o}^T$ and $\mathbf{o}^S$ to identify shared character boundaries, which define the start and end token indices of a span in each model. This produces N_{sp} aligned span pairs:

$$\mathcal{M}_s = \left\{(\mathrm{Span}_i^S,\, \mathrm{Span}_i^T)\right\}_{i=1}^{N_{\text{sp}}}, \quad \mathrm{Span}_i^{\cdot} = (s_i^{\cdot},\, e_i^{\cdot}), \tag{11}$$

where $s_i^{\cdot}$ and $e_i^{\cdot}$ are the start and end token indices of span i.

Span Representation. For each aligned span pair, we form span-level representations by mean-pooling the token hidden states within each span. At aligned layer j, the student and teacher span representations are:

$$C_i^{S,(j)} = \frac{1}{e_i^S - s_i^S + 1} \sum_{t=s_i^S}^{e_i^S} f_{\text{proj}}(h_t^{j,S}), \qquad C_i^{T,(j)} = \frac{1}{e_i^T - s_i^T + 1} \sum_{t=s_i^T}^{e_i^T} h_t^{j,T}, \tag{12}$$

where f_{proj} projects the student hidden states into the teacher's feature space. Since the two span sequences now have equal length N_{sp}, we can apply trajectory alignment at the span level.

3.5 Final Objective Function

A key advantage of CONTRA-KD is its orthogonality to existing distillation objectives. It acts as a complement on internal feature space, while other methods utilizing KD loss terms focus on output logits. Therefore, we integrate $\mathcal{L}_{\text{CONTRA}}$ as a complementary term to standard KD baselines. The final loss function is the sum of these components; note that $\mathcal{L}_{CONTRA}$ here is computed per sample and averaged over all samples in a batch:

$$\mathcal{L} = \mathcal{L}_{CE} + \mathcal{L}_{KD} + \mathcal{L}_{CONTRA} \tag{13}$$

Table 1: Win / Tie / Loss rates (%) across student-teacher pairs on different benchmark datasets, evaluated by `GPT-4o-mini`. **Win** indicates preference for CONTRA-KD (Ours Win), **Loss** indicates baseline preference, and **Tie** denotes equal quality.

Method	Evaluation Set								
	Dolly			*Self-Instruct*			*Vicuna*		
	Win	Tie	Loss	Win	Tie	Loss	Win	Tie	Loss
OpenLLaMA2 7B → 3B									
DistiLLM (+CONTRA vs. baseline)	**41.24**	21.80	36.96	**41.98**	19.67	38.35	**45.50**	14.25	40.25
FDD (+CONTRA vs. baseline)	**42.32**	21.24	36.44	**44.27**	17.47	38.26	**43.75**	15.50	40.75
DistiLLM-2 (+CONTRA vs. baseline)	**40.28**	23.10	36.62	**39.67**	25.78	34.55	**45.50**	11.00	43.50
CSD (+CONTRA vs. baseline)	**42.68**	22.28	35.04	**40.39**	24.98	34.63	**44.75**	18.50	36.75
GPT2 1.5B → 0.1B									
DistiLLM (+CONTRA vs. baseline)	**38.32**	31.24	30.44	**35.37**	34.15	30.48	**38.25**	31.50	30.25
FDD (+CONTRA vs. baseline)	**34.28**	33.56	32.16	**35.45**	34.55	30.00	**35.50**	34.75	29.75
DistiLLM-2 (+CONTRA vs. baseline)	**37.38**	31.48	31.14	**36.41**	33.70	29.89	**37.75**	30.75	31.50
CSD (+CONTRA vs. baseline)	**35.32**	32.76	31.92	**35.79**	34.71	29.50	**41.50**	26.50	32.00

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate the effectiveness of our proposed framework through extensive experiments on instruction-following benchmarks. We use the *databricks-dolly-15k*[3] dataset for training and include test sets from Self-Instruct Wang et al. (2022a), Vicuna Chiang et al. (2023), Super-Natural Instructions (S-NI) Wang et al. (2022b), and Unnatural Instructions (UnNI) Honovich et al. (2022).

Baselines. To demonstrate the versatility and complementary nature of our framework, we evaluate CONTRA-KD as an additive objective that integrates with existing distillation methods. We compare the performance of state-of-the-art baselines, specifically DistiLLM Ko et al. (2024), FDD Gong et al. (2025), DistiLLM-2 Ko et al. (2025), and CSD Kim et al. (2026), both in their standard standalone configurations and when augmented with our CONTRA auxiliary loss. Furthermore, to prove our generalizability to cross-architecture families, we carry out cross-tokenizer experiments with several existing cross-tokenizer KD methods, including ULD Boizard et al. (2025), MinED Wan et al. (2024), MultiLevelOT Cui et al. (2025), DSKD Zhang et al. (2024). For cross-tokenizer experiments, instead of incorporating CONTRA-KD into baseline methods, we use $\mathcal{L}_{CONTRA}$ as a standalone method along with Skew Reverse KL (SRKL) divergence from DistiLLM for $\mathcal{L}_{KD}$, reducing the dependency of our method to complementary same-tokenizer baselines.

Base models. Our method is evaluated on two pairs of same architectural teacher-student models: GPT2 Radford et al. (2019) (1.5B teacher and 0.1B student), OpenLLaMA2 Geng & Liu (2023) (7B teacher and 3B student). For cross-architectural pair, we use Qwen1.5 1.8B as the teacher and GPT2 0.1B as the student. Details on training setups can be found in Appendix C.

Evaluation. We employ LLM-as-a-judge following prior work Zheng et al. (2023) to evaluate the quality of responses sampled from models trained using baseline methods and those trained with CONTRA-KD included. We record win, tie, and loss rates of CONTRA-KD on Dolly, Self-Instruct, and Vicuna test sets. Additionally, we record ROUGE-L Lin (2004) scores of models' generated responses compared to ground-truth responses. To validate the reliability of our results, we also conducted a blind human evaluation as detailed in Appendix E.2.

4.2 RESULTS

We present evaluation results in Tables 1, 2, and 3. CONTRA-KD consistently outperforms baseline methods in both GPT-4o-mini evaluation and ROUGE-L scores, improving both discrete feature

[3]https://huggingface.co/datasets/databricks/databricks-dolly-15k

Table 2: Effects of CONTRA-KD on SOTA methods. ROUGE-L scores (%) averaged over 5 random seeds ± standard deviation.

Method	Dolly	Self-Inst	Vicuna	S-NI	UnNI	Avg.
OpenLLaMA2 7B → 3B						
Teacher	27.25	19.19	17.69	31.51	31.05	25.34
SFT	24.95	17.38	14.58	28.65	27.69	22.65
DistiLLM	$28.97_{\pm 0.35}$	$19.39_{\pm 0.30}$	$19.60_{\pm 0.58}$	$34.46_{\pm 0.33}$	$34.26_{\pm 0.11}$	$27.34_{\pm 0.16}$
+*CONTRA*	$\mathbf{29.04}_{\pm 0.35}$	$\mathbf{20.20}_{\pm 0.42}$	$\mathbf{19.66}_{\pm 0.29}$	$\mathbf{35.65}_{\pm 0.18}$	$\mathbf{36.20}_{\pm 0.17}$	$\mathbf{28.15}_{\pm 0.13}$
FDD	$28.07_{\pm 0.24}$	$\mathbf{19.47}_{\pm 0.75}$	$19.33_{\pm 0.67}$	$35.47_{\pm 0.29}$	$34.83_{\pm 0.20}$	$27.43_{\pm 0.21}$
+*CONTRA*	$\mathbf{28.31}_{\pm 0.15}$	$19.12_{\pm 0.58}$	$\mathbf{20.00}_{\pm 0.25}$	$\mathbf{36.56}_{\pm 0.11}$	$\mathbf{36.05}_{\pm 0.14}$	$\mathbf{28.01}_{\pm 0.13}$
DistiLLM-2	$28.48_{\pm 0.37}$	$19.78_{\pm 0.40}$	$19.85_{\pm 0.38}$	$35.66_{\pm 0.25}$	$36.52_{\pm 0.09}$	$28.06_{\pm 0.14}$
+*CONTRA*	$\mathbf{29.44}_{\pm 0.37}$	$\mathbf{20.38}_{\pm 0.61}$	$\mathbf{20.59}_{\pm 0.46}$	$\mathbf{35.74}_{\pm 0.30}$	$\mathbf{36.80}_{\pm 0.11}$	$\mathbf{28.59}_{\pm 0.17}$
CSD	$28.05_{\pm 0.51}$	$19.12_{\pm 0.84}$	$19.07_{\pm 0.52}$	$32.53_{\pm 0.31}$	$33.18_{\pm 0.16}$	$26.39_{\pm 0.23}$
+*CONTRA*	$\mathbf{28.83}_{\pm 0.56}$	$\mathbf{19.62}_{\pm 0.70}$	$\mathbf{19.59}_{\pm 0.38}$	$\mathbf{34.42}_{\pm 0.26}$	$\mathbf{34.77}_{\pm 0.11}$	$\mathbf{27.44}_{\pm 0.19}$
GPT2 1.5B → 0.1B						
Teacher	27.22	13.38	16.33	26.90	30.38	22.84
SFT	23.33	10.56	15.12	17.08	20.07	17.23
DistiLLM	$25.17_{\pm 0.36}$	$11.93_{\pm 0.18}$	$16.63_{\pm 0.26}$	$23.29_{\pm 0.27}$	$25.41_{\pm 0.16}$	$20.49_{\pm 0.11}$
+*CONTRA*	$\mathbf{26.22}_{\pm 0.31}$	$\mathbf{12.99}_{\pm 0.53}$	$\mathbf{17.41}_{\pm 0.30}$	$\mathbf{25.07}_{\pm 0.21}$	$\mathbf{26.85}_{\pm 0.11}$	$\mathbf{21.71}_{\pm 0.14}$
FDD	$24.27_{\pm 0.19}$	$11.23_{\pm 0.28}$	$14.69_{\pm 0.40}$	$19.65_{\pm 0.20}$	$22.50_{\pm 0.18}$	$18.47_{\pm 0.12}$
+*CONTRA*	$\mathbf{25.12}_{\pm 0.43}$	$\mathbf{11.46}_{\pm 0.38}$	$\mathbf{16.05}_{\pm 0.29}$	$\mathbf{20.21}_{\pm 0.22}$	$\mathbf{22.79}_{\pm 0.07}$	$\mathbf{19.13}_{\pm 0.14}$
DistiLLM-2	$22.22_{\pm 0.09}$	$11.13_{\pm 0.67}$	$17.19_{\pm 0.07}$	$21.58_{\pm 0.37}$	$24.39_{\pm 0.20}$	$19.30_{\pm 0.17}$
+*CONTRA*	$\mathbf{22.56}_{\pm 0.31}$	$\mathbf{11.74}_{\pm 0.32}$	$\mathbf{18.10}_{\pm 0.32}$	$\mathbf{22.09}_{\pm 0.16}$	$\mathbf{24.92}_{\pm 0.16}$	$\mathbf{19.88}_{\pm 0.12}$
CSD	$24.28_{\pm 0.32}$	$10.76_{\pm 0.55}$	$15.95_{\pm 0.34}$	$20.01_{\pm 0.30}$	$22.32_{\pm 0.19}$	$18.66_{\pm 0.16}$
+*CONTRA*	$\mathbf{24.52}_{\pm 0.32}$	$\mathbf{11.53}_{\pm 0.53}$	$\mathbf{16.07}_{\pm 0.32}$	$\mathbf{20.23}_{\pm 0.18}$	$\mathbf{23.85}_{\pm 0.14}$	$\mathbf{19.24}_{\pm 0.15}$

Table 3: Cross-tokenizer knowledge distillation results (Qwen1.5 1.8B → GPT2 0.1B). CONTRA-KD outperforms existing baselines on the majority of datasets.

Method	Dolly	Vicuna	Self-Inst	S-NI	Avg.
Teacher	$28.23_{\pm 0.13}$	$19.59_{\pm 0.19}$	$19.58_{\pm 0.35}$	$34.36_{\pm 0.35}$	$25.44_{\pm 0.14}$
SFT	$23.78_{\pm 0.28}$	$17.04_{\pm 0.24}$	$6.78_{\pm 0.15}$	$7.81_{\pm 0.11}$	$13.85_{\pm 0.10}$
ULD	$23.77_{\pm 0.41}$	$14.33_{\pm 0.15}$	$9.30_{\pm 0.12}$	$14.04_{\pm 0.11}$	$15.36_{\pm 0.12}$
MinED	$24.21_{\pm 0.31}$	$14.96_{\pm 0.21}$	$10.02_{\pm 0.14}$	$16.40_{\pm 0.05}$	$16.40_{\pm 0.10}$
MultiLevelOT	$23.02_{\pm 0.24}$	$13.79_{\pm 0.54}$	$8.41_{\pm 0.14}$	$12.26_{\pm 0.17}$	$14.37_{\pm 0.16}$
DSKD	$\mathbf{24.26}_{\pm 0.24}$	$15.25_{\pm 0.28}$	$10.07_{\pm 0.19}$	$17.15_{\pm 0.16}$	$16.68_{\pm 0.11}$
CONTRA (Ours)	$23.53_{\pm 0.31}$	$\mathbf{15.38}_{\pm 0.28}$	$\mathbf{12.23}_{\pm 0.65}$	$\mathbf{20.97}_{\pm 0.21}$	$\mathbf{18.03}_{\pm 0.20}$

matching methods (FDD) and methods using divergence loss terms with student-generated outputs (DistiLLM, DistiLLM-2). On the OpenLLaMA2 7B → 3B pair, CONTRA-KD improves baselines' average scores by no less than 0.5%, with the largest gain observed on CSD + CONTRA at **+1.05%**. Similarly, in the high-compression setting of GPT2 1.5B → 0.1B, our method boosts the DistiLLM baseline by **+1.22%** on average.

Consistent with ROUGE-L scores, GPT-4o-mini evaluations show that our method achieves higher win rates against all baselines across all settings, confirming that the ROUGE-L improvements correspond to better instruction-following quality. In the cross-tokenizer scenario, CONTRA-KD also outperforms existing baselines, showing the effectiveness of distilling hidden states' trajectory even between models of different architectures and vocabularies. Overall, these results demonstrate the robustness and effectiveness of our framework across diverse datasets, model architectures, and compression ratios of teacher-student pairs. To highlight the generalizability across model scales, we provide evaluations on the larger LLaMA-2 (13B → 7B) model pair in Appendix E.1.

5 ANALYSIS

In this section, we conduct focused ablation studies on the core components of CONTRA-KD using the GPT-2 (1.5B → 0.1B) setup. Additional ablations are detailed in Appendix F.

5.1 IMPACT OF CURRICULUM-BASED DISTILLATION

We analyze the effect of curriculum-based step sizes in CONTRA-KD by comparing our progressive step schedule against directly applying full update throughout training (i.e. fixed step size $k = 1$). In the progressive setting, the step size k is gradually increased from small values to 1 over the course of training, allowing the student to follow increasingly longer segments of the learned flow.
As shown in Figure 2, curriculum-based distillation consistently improves performance across both DistiLLM and DistiLLM2 baselines. This highlights that well-conditioned intermediate targets at early stages of training and gradually increasing the difficulty of the alignment task enables more effective trajectory alignment, resulting in higher ROUGE-L scores.

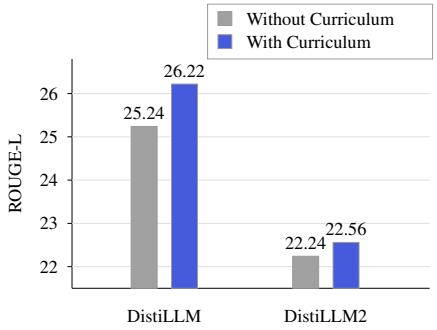


Figure 2: Comparison between curriculum-based and full-step distillation for hidden-state targets.

5.2 DIRECT INTERPOLATION WITHOUT A LEARNED VELOCITY FIELD

To isolate the necessity of the learned velocity field v_θ, we compare CONTRA-KD against a baseline that performs direct target interpolation without velocity field modeling. In this baseline, intermediate distillation targets are constructed via direct linear interpolation between student and teacher representations using the same curriculum schedule creating the target $h_S + k(h_T - h_S)$.

Table 4: Comparison with direct linear interpolation without velocity field (GPT2 1.5B → 0.1B with DistiLLM objective). ROUGE-L scores (%) are averaged over 5 seeds ± standard deviation.

Method	Dolly	Self-Inst	Vicuna	S-NI	UnNI	Avg.
With velocity field (Ours)	**26.22**$_{\pm 0.31}$	**12.99**$_{\pm 0.53}$	**17.41**$_{\pm 0.30}$	**25.07**$_{\pm 0.21}$	**26.85**$_{\pm 0.11}$	**21.71**$_{\pm 0.14}$
Without velocity field	25.05$_{\pm 0.28}$	12.18$_{\pm 0.42}$	16.21$_{\pm 0.31}$	23.13$_{\pm 0.25}$	24.92$_{\pm 0.09}$	20.30$_{\pm 0.13}$

As detailed in Table 4, removing the learned velocity field leads to a consistent and substantial degradation in performance. This ablation validates a core premise of our framework: simply interpolating linearly between unaligned representations across high-dimensional feature spaces provides noisy, suboptimal supervision signals. Because the intermediate feature representations of LLMs form complex, highly non-linear manifolds, direct Euclidean interpolation fails to capture local curvature or feature dynamics. The learned velocity field v_θ acts as a continuous transport map via Rectified Flow, guiding student updates along an optimal trajectory and proving essential for effective feature alignment.

6 CONCLUSION

In this work, we presented CONTRA-KD, a knowledge distillation framework that models distillation as a continuous-time alignment problem. By treating teacher and student models as dynamical systems, we proved that minimizing trajectory discrepancy implicitly bounds the difference in their velocity fields. Building on this, we employ a rectified flow formulation with curriculum-based guidance to progressively transfer knowledge. Extensive experiments across multiple model pairs on instruction-following benchmarks demonstrate the effectiveness and compatibility of CONTRA-KD with reasonable computational overhead, highlighting the promising direction of flow-based and dynamical-system perspectives in knowledge distillation.

AI USE STATEMENT

In this work, we used generative AI tools solely for polishing the writing of the manuscript. We did not use generative AI tools to generate research ideas, design experiments, generate or analyze experimental results, or write any part of the theoretical proofs presented in this paper; all such tasks were performed entirely by the authors. All AI-assisted text was reviewed and edited by the authors to ensure accuracy and that it faithfully reflects our original contributions, methodology, and findings. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken several steps to facilitate reproducibility of our results. The complete derivation and proof of Theorem 3.1, which establishes the connection between trajectory alignment and velocity alignment, is provided in Appendix B. The full training procedure for the velocity field and the two-phase CONTRA-KD distillation loop are given as pseudocode in Algorithm 1 and Algorithm 2 (Appendix D), respectively. Architectural details of the velocity field, including its embedding scheme, feed-forward block design, and per-model hidden dimension configuration, are described in Appendix C (Section C.1), and all training hyperparameters, optimization settings, and hardware configuration are provided in Appendix C.2. Dataset preprocessing and splits for *databricks-dolly-15k* and the five evaluation benchmarks are described in Section 4.1 and Appendix C.2, and the LLM-as-a-judge evaluation protocol, including the exact prompt used, is given in Appendix C.3 and Figure 3. We will release our code as anonymous supplementary material to enable full reproduction of our experiments.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016. URL https://api.semanticscholar.org/CorpusID:8236317.

Nicolas Boizard, Kevin El Haddad, Céline Hudelot, and Pierre Colombo. Towards cross-tokenizer distillation: the universal logit distillation loss for llms, 2025. URL https://arxiv.org/abs/2402.12030.

Cristian Bucila, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Knowledge Discovery and Data Mining*, 2006. URL https://api.semanticscholar.org/CorpusID:11253972.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Kristjanson Duvenaud. Neural ordinary differential equations. *ArXiv*, abs/1806.07366, 2018. URL https://api.semanticscholar.org/CorpusID:49310446.

Yi-Jie Chen, Yijin Liu, Fandong Meng, Yufeng Chen, Jinan Xu, and Jie Zhou. Enhancing cross-tokenizer knowledge distillation with contextual dynamical mapping. In *Annual Meeting of the Association for Computational Linguistics*, 2025. URL https://api.semanticscholar.org/CorpusID:276408154.

Pham Khanh Chi, Quoc Phong Dao, Thuat Nguyen, Linh Ngo Van, Trung Le, and Thanh Hong Nguyen. MTA: Multi-granular trajectory alignment for large language model distillation. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 32669–32684, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1507. URL https://aclanthology.org/2026.acl-long.1507/.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

Xiao Cui, Mo Zhu, Yulei Qin, Liang Xie, Wengang Zhou, and Houqiang Li. Multi-level optimal transport for universal cross-tokenizer knowledge distillation on language models, 2025. URL https://arxiv.org/abs/2412.14528.

Sayantan Dasgupta and Trevor Cohn. Improving language model distillation through hidden state matching. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=IcVSKhVpKu.

Xinyang Geng and Hao Liu. Openllama: An open reproduction of llama, May 2023. URL https://github.com/openlm-research/open_llama.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Guoqiang Gong, Jiaxing Wang, Jin Xu, Deping Xiang, Zicheng Zhang, Leqi Shen, Yifeng Zhang, JunhuaShu JunhuaShu, ZhaolongXing ZhaolongXing, Zhen Chen, Pengzhang Liu, and Ke Zhang. Beyond logits: Aligning feature dynamics for effective knowledge distillation. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23067–23077, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1125. URL https://aclanthology.org/2025.acl-long.1125/.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5h0qf7IBZZ.

Changyi He, Yifu Ding, Jinyang Guo, Ruihao Gong, Haotong Qin, and Xianglong Liu. DA-KD: Difficulty-aware knowledge distillation for efficient large language models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 22379–22391. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/he25c.html.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv: Learning*, 2016. URL https://api.semanticscholar.org/CorpusID:125617073.

Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015. URL https://api.semanticscholar.org/CorpusID:7200347.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. Unnatural instructions: Tuning language models with (almost) no human labor. *ArXiv*, abs/2212.09689, 2022. URL https://api.semanticscholar.org/CorpusID:254853659.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. In *Findings*, 2019. URL https://api.semanticscholar.org/CorpusID:202719327.

Woogyeol Jin, Taywon Min, Yongjin Yang, Swanand Kadhe, Yi Zhou, Dennis Wei, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. *ArXiv*, abs/2603.07079, 2026. URL https://api.semanticscholar.org/CorpusID:286372090.

Yeongmin Kim, Donghyeok Shin, Mina Kang, Byeonghu Na, and Il-Chul Moon. Distillation of large language models via concrete score matching, 2026. URL https://arxiv.org/abs/2509.25837.

Yoon Kim and Alexander M. Rush. Sequence-level knowledge distillation. *ArXiv*, abs/1606.07947, 2016. URL https://api.semanticscholar.org/CorpusID:8451212.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. DistiLLM: Towards streamlined distillation for large language models. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=lsHZNNoC7r.

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. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=rc65N9xIrY.

Alexander Lin, Jeremy Wohlwend, Howard Chen, and Tao Lei. Autoregressive knowledge distillation through imitation learning. In *Conference on Empirical Methods in Natural Language Processing*, 2020. URL https://api.semanticscholar.org/CorpusID:221702970.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Annual Meeting of the Association for Computational Linguistics*, 2004. URL https://api.semanticscholar.org/CorpusID:964287.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *ArXiv*, abs/2210.02747, 2022. URL https://api.semanticscholar.org/CorpusID:252734897.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *ArXiv*, abs/2209.03003, 2022. URL https://api.semanticscholar.org/CorpusID:252111177.

Yiping Lu*, Zhuohan Li*, Di He, Zhiqing Sun, Bin Dong, Tao Qin, Liwei Wang, and Tie yan Liu. Understanding and improving transformer from a multi-particle dynamic system point of view. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2019. URL https://openreview.net/forum?id=pxlqJa21C.

Benjamin Minixhofer, Ivan Vulić, and Edoardo Maria Ponti. Universal cross-tokenizer distillation via approximate likelihood matching, 2025. URL https://arxiv.org/abs/2503.20083.

Thinh On, Hengzhi Pei, Leonard Lausen, and George Karypis. Knowledge distillation for large language models through residual learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Dh6KxUxG20.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Donghyeok Shin, Yeongmin Kim, Suhyeon Jo, Byeonghu Na, and Il chul Moon. AMid: Knowledge distillation for LLMs with α-mixture assistant distribution. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7WPJ0EgPdW.

Oscar Skean, Md Rifat Arefin, Dan Zhao, Niket Nikul Patel, Jalal Naghiyev, Yann LeCun, and Ravid Shwartz-Ziv. Layer by layer: Uncovering hidden representations in language models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=WGXb7UdvTX.

Yang Song, Jascha Narain Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ArXiv*, abs/2011.13456, 2020. URL https://api.semanticscholar.org/CorpusID:227209335.

Anh Tong, Thanh Nguyen-Tang, Dongeun Lee, Duc Nguyen, Toan Tran, David Leo Wright Hall, Cheongwoong Kang, and Jaesik Choi. Neural ODE transformers: Analyzing internal dynamics and adaptive fine-tuning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=XnDyddPcBT.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aur'elien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023. URL https://api.semanticscholar.org/CorpusID:257219404.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017. URL https://api.semanticscholar.org/CorpusID:13756489.

Fanqi Wan, Xinting Huang, Deng Cai, Xiaojun Quan, Wei Bi, and Shuming Shi. Knowledge fusion of large language models. *ArXiv*, abs/2401.10491, 2024. URL https://api.semanticscholar.org/CorpusID:267061245.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Annual Meeting of the Association for Computational Linguistics*, 2022a. URL https://api.semanticscholar.org/CorpusID:254877310.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5085–5109, Abu Dhabi, United Arab Emirates, December 2022b. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.340. URL https://aclanthology.org/2022.emnlp-main.340/.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Zhe Zhao, and Ngai Wong. Rethinking kullback-leibler divergence in knowledge distillation for large language models. In *International Conference on Computational Linguistics*, 2024. URL https://api.semanticscholar.org/CorpusID:268876464.

Wenda Xu, Rujun Han, Zifeng Wang, Long Le, Dhruv Madeka, Lei Li, William Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. In *International Conference on Learning Representations*, volume 2025, pp. 64616–64646, 2025.

Songming Zhang, Xue Zhang, Zengkui Sun, Yufeng Chen, and Jinan Xu. Dual-space knowledge distillation for large language models, 2024. URL https://arxiv.org/abs/2406.17328.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Haotong Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *ArXiv*, abs/2306.05685, 2023. URL https://api.semanticscholar.org/CorpusID:259129398.

A Related work

Knowledge distillation of LLMs. Knowledge Distillation (KD) Hinton et al. (2015) is a model compression technique where a student learns from a teacher model's outputs. A line of research has focused on treating distillation as a sequence-level alignment problem Kim & Rush (2016). More recent methods have built on this, by using Kullback–Leibler divergence on student-generated outputs (SGOs) Lin et al. (2020), employing Jensen-Shannon divergence Agarwal et al. (2024), Reverse KLD Gu et al. (2024); Wu et al. (2024), or skewed variants of KLD Ko et al. (2024; 2025). Entropy-aware variants further adapt the choice of divergence per-token based on the teacher's local uncertainty Jin et al. (2026), and there is also growing interest in distilling through raw, unnormalized logits rather than softmax outputs Kim et al. (2026).

Intermediate representations distillation. A complementary line of work distills a teacher's intermediate representations, motivated by the observation that deep models encode rich hierarchical information beyond their final outputs Skean et al. (2025). Early successes along this direction were demonstrated for BERT-style architectures Sanh et al. (2019); Jiao et al. (2019). More recent methods have refined how this comparison is made rather than simply where: matching hidden states via Centered Kernel Alignment removes the requirement that student and teacher share a hidden dimension Dasgupta & Cohn (2025); a residual-learning objective instead trains the student to regress the difference between its own state and a projected teacher state, guarding against propagating the teacher's own mistakes On et al. (2026). Inspired by interpreting Transformers as discretizations of Ordinary Differential Equations (ODEs) Chen et al. (2018); Lu* et al. (2019), Feature Dynamic Distillation instead frames KD as aligning the trajectories of a coarse ODE solver (the student) with those of a fine-grained one (the teacher), matching both feature states and their first-order layer-to-layer differences Gong et al. (2025). While each of these approaches captures a different facet of the teacher's intermediate computation, all of them still compare states, residuals, or directions at a fixed set of layer checkpoints rather than the continuous path connecting them, which limits their ability to capture the global geometry of the teacher's feature flow.

Cross-tokenizer knowledge distillation. Extending KD beyond teacher-student pairs that share a vocabulary, cross-tokenizer methods must additionally resolve a mismatch in sequence length and output dimensionality between the two models. ULD Boizard et al. (2025) applies a truncation strategy to align sequence lengths and logit dimensions, enabling direct comparison of output distributions. MinED Wan et al. (2024) instead relies on edit-distance-based token alignment to bridge mismatched tokenizations. Building on this line, MultiLevelOT Cui et al. (2025) introduces an approximate Optimal Transport objective to reduce divergence at both the token and sequence level. DSKD Zhang et al. (2024) projects student hidden states into the teacher's space and vice versa, and further introduces a cross-model attention mechanism that aligns tokens across tokenizers via embedding similarity. Rather than a static similarity-based alignment, contextual dynamical mapping Chen et al. (2025) learns a context-conditioned correspondence between teacher and student tokens. Most recently, ALM Minixhofer et al. (2025) refines token alignment further by exhaustively enumerating decoded chunks to identify equivalent segments between teacher and student outputs. Despite this progression, each approach carries its own limitation: truncation-based strategies such as ULD can discard information at alignment boundaries; edit-distance alignment as in MinED becomes brittle as vocabularies diverge further; optimal-transport objectives such as MultiLevelOT improve alignment quality but at a real computational cost; and exhaustive matching as in ALM scales poorly with vocabulary size and sequence length. We instead sidestep explicit token-level alignment altogether, identifying shared character-level boundaries via Longest Common Subsequence to construct token spans, and performing our trajectory-based distillation at the span level, where sequence lengths are guaranteed to match by construction.

B Theory Development

In this section, we will give mathematical proof showing that by minimizing the continuous trajectory error, we implicitly constrain the student's governing vector field v_S to align with the effective dynamics v_T of the teacher.

Proof. We start with

$$\begin{aligned}\delta_t &= v_S\left(h_T\left(t\right), t\right) - v_T\left(h_T\left(t\right), t\right) \\ &= \left[v_S\left(h_T\left(t\right), t\right) - v_S\left(h_S\left(t\right), t\right)\right] \\ &\quad + \left[v_S\left(h_S\left(t\right), t\right) - v_T\left(h_T\left(t\right), t\right)\right].\end{aligned}$$

This leads to

$$\begin{aligned}\|\delta_t\|_{\mathcal{L}_2} &\leqslant \|v_S\left(h_T\left(t\right), t\right) - v_S\left(h_S\left(t\right), t\right)\|_{\mathcal{L}_2} \\ &\quad + \|v_S\left(h_S\left(t\right), t\right) - v_T\left(h_T\left(t\right), t\right)\|_{\mathcal{L}_2} \\ &= \|v_S\left(h_T\left(t\right), t\right) - v_S\left(h_S\left(t\right), t\right)\|_{\mathcal{L}_2} \\ &\quad + \|\frac{d}{dt}\left[h_S\left(t\right) - h_T\left(t\right)\right]\|_{\mathcal{L}_2}.\end{aligned}$$

Using the M-Lipchitz, we have

$$\|v_S\left(h_T\left(t\right), t\right) - v_S\left(h_S\left(t\right), t\right)\|_{\mathcal{L}_2} \leqslant M\|h_T\left(t\right) - h_S\left(t\right)\|_{\mathcal{L}_2}$$

Finally, we reach the conclusion

$$\begin{aligned}\|\delta_t\|_{\mathcal{L}_2} &\leqslant M\|h_T\left(t\right) - h_S\left(t\right)\|_{\mathcal{L}_2} \\ &\quad + \|\frac{d}{dt}\left[h_S\left(t\right) - h_T\left(t\right)\right]\|_{\mathcal{L}_2}.\end{aligned} \tag{14}$$

□

C Experimental Details

C.1 Velocity Field Design and Configurations

For training as a neural network, we design the velocity field v_θ using components based on the design of Transformers Vaswani et al. (2017). We use 2 embedding layers to encode the time step and layer step information, along with an input projection to project hidden states from the teacher model's dimension to a smaller internal dimension (called *velocity field's hidden dimension*) to reduce the velocity field's hidden dimension size. The time and layer embeddings are then added to the projected input hidden states.

The embedded input goes through a sequence of feed-forward blocks, each constructed similar to the feed-forward network in a Transformer Vaswani et al. (2017) block. The representations first go through a layer normalization module Ba et al. (2016) to reduce variance, allowing the training process to be more stable. Then, they go through 2 linear layers with a GELU Hendrycks & Gimpel (2016) activation and the in-between dimension size of 4 times the model's hidden dimension size. Finally, the representation vectors go through a dropout module to ensure robustness of the network, before passing through an output projection back to the teacher's space.

In our experiments, to ensure consistency of CONTRA-KD's performance between pairs of teacher-student, which have different number of layers and hidden dimension sizes, we fix the number of layers (i.e., feed-forward blocks) of the velocity field model to 2 and the number of layers to distill between teacher and student to 4. Additionally, we set the model's hidden dimension size to 1024 for GPT2 models, 2400 for OpenLLaMA2 models, and 3200 for LLaMA2 models. The values are selected according to the teacher models' hidden dimension size (1600 for GPT2 1.5B, 3200 for OpenLLaMA2 7B, and 5120 for LLaMA2 13B), specifically around *60%* of the teachers' dimension size. This keeps the velocity fields compact while still allowing for effective distillation between larger models. We conduct additional ablation studies on hidden dimension size and number of layers of the velocity field in Appendix F.

C.2 Training Details

Our experiments are based on previous implementation setups by Gu et al. (2024); Ko et al. (2024); Gong et al. (2025). We use a preprocessed version of *databricks-dolly-15k* provided by Zhang et al. (2024), consisting of around 11,500 samples for training, 1000 samples for validation, and 500 samples for testing. Along with that, we include a language modeling loss term on a randomly-sampled subset of 22,800 samples (i.e., about twice the size of Dolly training set) from the OpenWebText Gokaslan & Cohen (2019) corpus, as it has been shown by previous works to improve instruction-tuning performance. This loss term is only included in distillation experiments and not in SFT (teacher/student SFT and student warm-up) or velocity field training.

The models are trained using 1 NVIDIA H100 80GB GPU. Across all stages of our experiments, we use the same learning rate of 5e-4 and effective batch size of 16. For GPT2 models, we employ full fine-tuning for 20 epochs, except for the teacher GPT2 1.5B which is trained for only 10 epochs. For LLaMA2 and OpenLLaMA2 models, we fine-tune using LoRA Hu et al. (2022) for 10 epochs with the rank and alpha both set at 16 and dropout rate at 0.1.

The student models are first fine-tuned for 3 epochs for warm-up on the training dataset. Then, they are used along with the corresponding teacher models for the initial training process of the velocity field for 3 epochs on the same dataset. The trained velocity fields act as initial checkpoints to be used for CONTRA-KD, where they are updated along with the student models to avoid distributional shifts.

Models are validated by evaluating their generated responses against the reference responses on the validation set using ROUGE-L Lin (2004). Responses are obtained using greedy sampling, and checkpoints with the best results are selected for later evaluation on benchmark datasets.

C.3 Evaluation

General Configurations. We sample responses from each model using temperature of 1.0, top_p of 1.0, and max-length limit of 512.

LLM-as-a-judge. We use `GPT-4o-mini` as the judge, following prior work on LLM-based evaluation Zheng et al. (2023), for pairwise comparison of responses, comparing the baseline's response and CONTRA-KD's response for a given input prompt. For each baseline method and input prompt, we ask `GPT-4o-mini` to compare the responses between the student trained using the baseline method and the student trained using the same baseline with CONTRA loss term included, using the prompt format in Figure 3 and setting the temperature of 0.7 for `GPT-4o-mini`. We record the number of baseline wins, CONTRA-KD wins, and ties for 3 benchmark datasets: Dolly, Self-Instruct, and Vicuna.

ROUGE-L. We sample responses using 5 random seeds $\{10, 20, 30, 40, 50\}$ and average results over the 5 seeds for each benchmark dataset: Dolly, Self-Instruct, Vicuna, Super-Natural Instructions, Unnatural Instructions.

D Detailed Algorithm

Algorithm 2 details the full two-phase training loop introduced in Section 3.3: Phase 1 updates the student parameters ϕ using the combined loss $\mathcal{L}_{Total}$, and Phase 2 re-adapts the velocity field v_θ and projector f_{proj} on the updated student following Algorithm 1, preventing distributional drift between the student and the flow it is being aligned to.

E Additional Experimental Results.

E.1 Additional Results of Larger Teacher–Student Pair

To demonstrate scalability to larger models, we conduct additional experiments on LLaMA2 Touvron et al. (2023) with a 13B teacher and 7B student. ROUGE-L scores and GPT-4o-mini win/tie/loss rates are reported in Tables 5 and 6, respectively.

[System]
Please act as an impartial judge and compare the quality of response A and response B provided by two AI assistants to the user question displayed below. You should choose the assistant that follows the user's instructions and answers the user's question better.
Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses.
Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Do not favor certain names of the assistants. Be as objective as possible.
Output your final verdict by strictly following this format:
If A is significantly better overall, answer "A".
If B is significantly better overall, answer "B".
If both are similar in quality (both bad or both good or not significantly different), answer "Tied".

[Question]
{question}

[Response A]
{response_A}
[End of Response A]

[Response B]
{response_B}
[End of Response B]

Figure 3: Prompt used for pairwise response quality evaluation.

Algorithm 2 CONTRA-KD Distillation

Input: student q_{ϕ_0} with parameters ϕ_0, teacher p, pre-trained velocity field v_θ and projector f_{proj}, data $\mathcal{D}$, layer maps I^S, I^T
Output: Student model q_{ϕ_E} with trained parameters ϕ_E
for epoch $e = 1, 2, ..., E$ **do**
  $k \leftarrow \frac{e}{E}$
  for each batch $\{x_i\}$ in $\mathcal{D}$ **do**
    // — Phase 1: Update Student Model —
    Initialize $\mathcal{L}_{CONTRA} \leftarrow 0$
    for $j = 1, 2, ..., L_D$ **do**
      $h_S^j \leftarrow f_{\text{proj}}\big(q_{\phi_{e-1}}(\{x_i\}, \text{layer} = l_j^S)\big)$
      $\tilde{h}_S^j \leftarrow h_S^j + k \cdot v_\theta(h_S^j, t = 0, j)$
      $\mathcal{L}_{\text{CONTRA}} \leftarrow \mathcal{L}_{\text{CONTRA}} + D_{\cos}(h_S^j, \tilde{h}_S^j)$
    end for
    Compute $\mathcal{L}_{KD}$ using $p(x_i)$ and $q_{\phi_{e-1}}(x_i)$
    $\mathcal{L}_{Total} \leftarrow \mathcal{L}_{CE} + \mathcal{L}_{KD} + \mathcal{L}_{CONTRA}$
    Update $\phi_e \leftarrow \phi_{e-1} - \eta \nabla_\phi \mathcal{L}_{Total}$
  end for
  // — Phase 2: Adapt Velocity Field & Projector —
  Update velocity field v_θ and projector f_{proj} based on Algorithm 1 using teacher p, student q_{ϕ_e}, data $\mathcal{D}$, layer maps I_S, I_T
end for
return q_{ϕ_E}

CONTRA-KD consistently improves all baselines on this larger pair. In terms of ROUGE-L, average gains range from **+0.29%** (DistiLLM) to **+0.64%** (FDD), with DistiLLM-2 + CONTRA achieving the highest average score of **30.71**. Most notably, DistiLLM-2 + CONTRA reaches a ROUGE-L score of **38.59** on the UnNI benchmark, substantially surpassing the teacher itself at 34.25, suggesting that our contrastive objective helps the student better generalize on unseen natural instructions at scale. GPT-4o-mini evaluations corroborate these gains, with CONTRA-KD winning more often than the baseline across all methods and datasets.

Table 5: Win / Tie / Loss rates (%) of LLaMA2 on different benchmark datasets, evaluated by `GPT-4o-mini`. **Win** indicates preference for the model trained with CONTRA-KD over the baseline (Ours Win), **Loss** indicates preference for the baseline (Baseline Win), and **Tie** denotes equal quality.

Method	**Evaluation Set**								
	Dolly			*Self-Instruct*			*Vicuna*		
	Win	Tie	Loss	Win	Tie	Loss	Win	Tie	Loss
LLaMA2 13B → 7B									
FDD (+CONTRA vs. baseline)	**44.00**	19.60	36.40	38.43	23.14	38.43	**48.75**	10.00	41.25
DistiLLM (+CONTRA vs. baseline)	**41.20**	22.60	36.20	**43.80**	14.40	41.80	**46.25**	15.00	38.75
DistiLLM-2 (+CONTRA vs. baseline)	**40.40**	26.60	33.00	**43.80**	15.29	40.91	**45.00**	21.25	33.75

Table 6: Effects of CONTRA-KD on SOTA methods for LLaMA2. ROUGE-L scores (%) averaged over 5 random seeds $\pm$ standard deviation.

Method	**Dolly**	**Self-Inst**	**Vicuna**	**S-NI**	**UnNI**	**Avg.**
LLaMA2 13B → 7B						
Teacher	30.19	23.92	19.56	34.53	34.25	28.49
SFT	28.38	20.32	17.72	34.37	33.25	26.81
DistiLLM	$31.10_{\pm 0.57}$	$23.22_{\pm 0.43}$	$21.38_{\pm 0.35}$	$\mathbf{36.30}_{\pm 0.34}$	$37.03_{\pm 0.07}$	$29.81_{\pm 0.17}$
+*CONTRA*	$\mathbf{31.45}_{\pm 0.20}$	$\mathbf{23.69}_{\pm 0.52}$	$\mathbf{21.48}_{\pm 0.14}$	$36.19_{\pm 0.26}$	$\mathbf{37.70}_{\pm 0.15}$	$\mathbf{30.10}_{\pm 0.13}$
FDD	$30.59_{\pm 0.44}$	$22.12_{\pm 0.52}$	$20.87_{\pm 0.29}$	$35.91_{\pm 0.33}$	$36.48_{\pm 0.13}$	$29.19_{\pm 0.16}$
+*CONTRA*	$\mathbf{31.55}_{\pm 0.27}$	$\mathbf{22.80}_{\pm 1.02}$	$\mathbf{20.92}_{\pm 0.15}$	$\mathbf{36.81}_{\pm 0.17}$	$\mathbf{37.08}_{\pm 0.13}$	$\mathbf{29.83}_{\pm 0.22}$
DistiLLM-2	$31.76_{\pm 0.32}$	$\mathbf{24.49}_{\pm 0.25}$	$21.96_{\pm 0.38}$	$35.98_{\pm 0.24}$	$36.57_{\pm 0.08}$	$30.15_{\pm 0.12}$
+*CONTRA*	$\mathbf{32.19}_{\pm 0.57}$	$23.97_{\pm 0.49}$	$\mathbf{22.27}_{\pm 0.26}$	$\mathbf{36.52}_{\pm 0.30}$	$\mathbf{38.59}_{\pm 0.05}$	$\mathbf{30.71}_{\pm 0.17}$

E.2 Human Evaluation on Instruction-Following

To complement our automated evaluation metrics and validate the reliability of our GPT-4o-mini judge, we conducted a blind human evaluation on a randomized subset of 100 instruction-following outputs from the *Dolly* test set, comparing CONTRA-KD against the competitive CSD baseline. Each item was independently evaluated by three human annotators in a double-blind A/B testing setup.

Table 7: Human-evaluated Win / Tie / Loss rates (%) for CONTRA-KD versus the CSD baseline on 100 *Dolly* test samples across three independent annotators and pooled judgment.

Evaluator	**Win (%)**	**Tie (%)**	**Loss (%)**
Annotator 1	**36.0**	32.0	32.0
Annotator 2	**43.0**	20.0	37.0
Annotator 3	**46.0**	29.0	25.0
Pooled	**41.7**	27.0	31.3

As presented in Table 7, pooled human judgments preferred CONTRA-KD in **41.7%** of cases, compared to **31.3%** for the baseline (with 27.0% ties). When excluding ties, CONTRA-KD achieved a **57.1% preference win rate**. These findings strongly mirror our GPT-4o-mini LLM-as-a-Judge evaluation, reinforcing that continuous trajectory alignment leads to tangibly higher generation quality and better instruction compliance.

Agreement Analysis. To assess evaluation consistency, we measured both internal inter-annotator reliability and external alignment with the GPT-4o-mini judge. Pairwise agreement between individual human annotators and the LLM judge averaged 48.7%. Inter-annotator agreement, measured via Fleiss' κ, showed slight agreement consistent with expected variance in open-ended response comparison. Crucially, both human majority vote and the automated LLM judge demonstrate identical directional preferences (favoring CONTRA-KD), validating the credibility of our automated evaluation pipeline.

Human Evaluation Rubric. Human annotators were provided with the interface instructions and evaluation prompt shown in Figure 4.

[Instructions for Annotators]
Please act as an impartial judge and compare the quality of Response A and Response B provided by two AI assistants to the user question displayed for each item below. You should choose the assistant that follows the user's instructions and answers the question better.
Your evaluation should consider factors such as the helpfulness, relevance, accuracy, depth, creativity, and level of detail of their responses.
Avoid any position biases and ensure that the order in which the responses were presented does not influence your decision. Do not allow the length of the responses to influence your evaluation. Be as objective as possible.
Choose your verdict by strictly following this format:
If A is significantly better overall, select "A".
If B is significantly better overall, select "B".
If both are similar in quality (both bad, both good, or not significantly different), select "Tied".

[User Question]
{question}

[Response A]
{response_A}

[Response B]
{response_B}

Figure 4: Instruction guidelines and prompt format provided to human annotators for pairwise evaluation.

E.3 CONTRA AS A STANDALONE OBJECTIVE

To isolate the exact contribution of our continuous trajectory alignment objective ($\mathcal{L}_{\text{CONTRA}}$), we evaluate a standalone version of our framework–**CONTRA only**–where the student is trained using solely cross-entropy and feature trajectory alignment ($\mathcal{L}_{\text{CE}} + \lambda\mathcal{L}_{\text{CONTRA}}$), omitting any output logit matching objective (e.g., DistiLLM or reverse KL). Table 8 compares **CONTRA only** against standard baseline approaches in their native configurations on the GPT-2 1.5B $\rightarrow$ 0.1B model pair across five benchmark datasets.

Table 8: Standalone performance comparison of CONTRA-KD against baseline distillation objectives on GPT-2 1.5B $\rightarrow$ 0.1B using ROUGE-L scores.

Method	Dolly	Self-Inst	Vicuna	S-NI	UnNI	Avg.
FDD	$24.27_{\pm 0.19}$	$11.23_{\pm 0.28}$	$14.69_{\pm 0.40}$	$19.65_{\pm 0.20}$	$22.50_{\pm 0.18}$	$18.47_{\pm 0.12}$
CSD	$24.28_{\pm 0.32}$	$10.76_{\pm 0.55}$	$15.95_{\pm 0.34}$	$20.01_{\pm 0.30}$	$22.32_{\pm 0.19}$	$18.66_{\pm 0.16}$
DistiLLM	$\mathbf{25.17}_{\pm 0.36}$	$\mathbf{11.93}_{\pm 0.18}$	$16.63_{\pm 0.26}$	$\mathbf{23.29}_{\pm 0.27}$	$\mathbf{25.41}_{\pm 0.16}$	$\mathbf{20.49}_{\pm 0.11}$
DistiLLM-2	$22.22_{\pm 0.09}$	$11.13_{\pm 0.67}$	$\mathbf{17.19}_{\pm 0.07}$	$21.58_{\pm 0.37}$	$24.39_{\pm 0.20}$	$19.30_{\pm 0.17}$
CONTRA only (Ours)	$24.53_{\pm 0.28}$	$11.76_{\pm 0.28}$	$15.67_{\pm 0.32}$	$19.70_{\pm 0.22}$	$23.11_{\pm 0.10}$	$18.95_{\pm 0.11}$

Operating purely on internal representations, **CONTRA only** achieves an average score of 18.95, outperforming both Feature Dynamics Distillation (FDD, 18.47) and Concrete Score Matching (CSD, 18.66). This validates that aligning continuous Rectified Flow trajectories provides superior feature-level transfer compared to matching discrete, local derivatives.

Feature-level alignment ensures that student hidden states follow mathematically smooth, teacher-directed ODE paths through depth, while logit-level objectives sharpen the final output probability distribution over the vocabulary. Combining both mechanisms (Ours + DistiLLM) yields the highest performance across all benchmarks.

E.4 COMPARISONS AGAINST BASELINES WITH MATCHED COMPUTATION

To ensure that the performance gains observed with *+CONTRA* are not merely an artifact of the additional training stage, it is crucial to isolate the effect of training duration. We conduct additional experiments where models are trained using baseline methods with more training steps. Specifically,

we increase the number of epochs by 50% to assess whether using baseline methods with a longer training time can match or outperform our method. We run experiments on two pairs of teacher-student models (GPT-2 1.5B → 0.1B and OpenLLaMA-2 7B → 3B) and for three baselines (SFT, Vanilla KD, DistiLLM); the results (ROUGE-L score, averaged over 5 random seeds) are summarized in Table 9. As shown by the results, training baselines longer does not yield better performance than CONTRA; it even leads to worse results. This is most likely due to overfitting, as models are exposed to the training dataset more, whereas CONTRA-KD 'extracts' more information from a training step rather than making the model view the dataset more times.

Table 9: Ablation study on training duration. +CONTRA maintains superior performance across all evaluation datasets without relying solely on increased training duration.

Method	Dolly	Self-Inst	Vicuna	S-Inst	Avg.
GPT-2 (1.5B → 0.1B)					
SFT	23.33	10.56	15.12	17.08	16.52
SFT (longer)	23.29	8.81	14.67	15.46	15.56
KD	23.52	11.23	15.92	20.68	17.84
KD (longer)	23.37	10.60	15.36	19.42	17.19
DistiLLM	25.17	11.93	16.63	23.29	19.26
DistiLLM (longer)	24.86	11.52	16.79	20.26	18.36
DistiLLM + CONTRA (Ours)	**26.22**	**12.99**	**17.41**	**25.07**	**20.42**
OpenLLaMA-2 (7B → 3B)					
SFT	24.95	17.38	14.58	28.65	21.39
SFT (longer)	24.29	16.48	17.03	28.37	21.54
KD	25.10	17.11	16.64	28.93	21.95
KD (longer)	21.75	15.47	16.14	25.90	19.82
DistiLLM	28.97	19.39	19.60	34.46	25.61
DistiLLM (longer)	28.37	19.53	19.52	34.24	25.42
DistiLLM + CONTRA (Ours)	**29.04**	**20.20**	**19.66**	**35.65**	**26.14**

F Additional Analysis

F.1 Non-linear Interpolant Path Analysis

A core efficiency mechanism of CONTRA-KD is the generation of the dynamic distillation target via an explicit Euler update along a learned flow (see Section 3.3). This approximation is geometrically faithful when the underlying flow trajectory is linear, but may incur truncation error when applied to curved paths. To validate this design choice, we compare our linear Rectified Flow against two non-linear path formulations commonly used in diffusion models: Variance Preserving (VP) ODE Song et al. (2020) and Variance Exploding (VE) ODE Song et al. (2020).

The flow trajectories are defined by a generalized interpolant H_t between the student state $h_S^j[i]$ and teacher state $h_T^j[i]$:

$$H_t = \sigma_t h_S^j[i] + \alpha_t h_T^j[i] \tag{15}$$

where σ_t and α_t control the mixing schedule. We compare three schedules:

- **Rectified Flow (Ours):** The linear path with $\alpha_t = t$ and $\sigma_t = 1 - t$.
- **VP ODE:** A cosine-based schedule defined by $\alpha_t = \exp(-\frac{1}{4}a(1-t)^2 - \frac{1}{2}b(1-t))$ and $\sigma_t = \sqrt{1-\alpha_t^2}$, with $a = 19.9$ and $b = 0.1$.
- **VE ODE:** A geometric variance schedule defined by $\alpha_t = a(\frac{b}{a})^t$ and $\sigma_t = 1$, with $a = 0.02$ and $b = 100$.

The results in Table 10 show that the VP-ODE path causes a noticeable performance drop (-0.53 ROUGE-L). This indicates that curved interpolation paths are ill-suited for Euler-based target generation, as local curvature introduces truncation error even when using partial steps along the flow. While VE-ODE performs comparably to the linear baseline, it requires careful hyperparameter tuning (a, b) and offers no performance gain. In contrast, the linear Rectified Flow consistently provides a stable and accurate approximation under Euler updates, making it the most effective choice for curriculum-based trajectory distillation.

Table 10: Ablation study on the impact of interpolation path geometry (GPT-2 1.5B → 0.1B) in DistiLLM.

Interpolation Path	**ROUGE-L**
Rectified Flow (Ours)	**26.22**
VP-ODE	24.71
VE-ODE	25.23

F.2 Computational and Memory Cost

We analyze the computational overhead introduced by adding CONTRA-KD by comparing the average wall-clock training time per epoch and peak memory usage across different distillation settings in Table 11. Incorporating CONTRA-KD leads to a moderate and consistent increase in training time, with the relative overhead remaining below $1.5\times$ in all configurations. In terms of memory usage, CONTRA-KD introduces minimal increase with only around 2% for the GPT2 pair and moderate increase for the OpenLLaMA2 pair; the only exception is OpenLLaMA2 with the DistiLLM2 baseline, where the method uses a forward batch size doubled that of other baselines, which, combined with a larger velocity field, results in a higher memory overhead; the increase is still moderate given the baseline's high usage.

Overall, these results show that while CONTRA-KD introduces a manageable increase in training cost and memory usage, the overhead remains controlled across scales and methods, supporting its practicality for large-scale distillation settings.

Table 11: Average wall-clock time per epoch (minutes) and peak GPU memory usage (GB), for baseline methods and with CONTRA-KD incorporated.

	Time (min)			**Memory (GB)**		
Method	Std.	+CONTRA	Incr.	Std.	+CONTRA	Incr.
GPT2 1.5B → 0.1B						
DistiLLM	13.50	14.15	1.05×	29.99	30.55	1.87%
FDD	9.25	12.45	1.35×	35.55	36.34	2.22%
DistiLLM2	12.20	13.05	1.07×	53.51	54.72	2.26%
OpenLLaMA2 7B → 3B						
DistiLLM	28.50	32.90	1.15×	41.35	43.05	4.11%
FDD	23.80	26.90	1.13×	53.64	57.29	6.80%
DistiLLM2	65.90	78.60	1.19×	52.84	67.36	27.48%

F.3 Comparison with Ordinary Feature KD and Curriculum Decoupling

To position CONTRA-KD against standard intermediate feature distillation techniques, we clarify that conventional feature KD variants–specifically hidden-state Mean Squared Error and feature delta matching (matching first-order derivatives)–constitute the core formulation of Feature Dynamics Distillation (FDD) (Gong et al., 2025). Consequently, comparisons against FDD directly evaluate our framework against these ordinary feature KD objectives.

To further isolate the individual contributions of our continuous flow formulation from the progressive curriculum strategy, we evaluate four distinct configurations on the GPT-2 (1.5B → 0.1B) teacher–student pair: (i) **FDD** (standard baseline combining state MSE and first-order delta losses), (ii) **FDD + Curriculum** (applying our progressive step-size schedule directly to FDD targets), (iii) **CONTRA only** (our continuous flow objective without FDD losses), and (iv) **Ours + FDD** (the full combined framework).

Table 12: Comparison of ordinary feature KD variants, curriculum scheduling, and standalone CONTRA (GPT-2 1.5B → 0.1B). ROUGE-L scores (%) are averaged over 5 random seeds ± standard deviation.

Method	Dolly	Self-Inst	Vicuna	S-NI	UnNI	Avg.
FDD	$24.27_{\pm 0.19}$	$11.23_{\pm 0.28}$	$14.69_{\pm 0.40}$	$19.65_{\pm 0.20}$	$22.50_{\pm 0.18}$	$18.47_{\pm 0.12}$
FDD + Curriculum	$23.37_{\pm 0.59}$	$10.77_{\pm 0.30}$	$14.80_{\pm 0.23}$	$\mathbf{21.62}_{\pm 0.25}$	$22.82_{\pm 0.10}$	$18.68_{\pm 0.15}$
CONTRA only	$24.53_{\pm 0.28}$	$\mathbf{11.76}_{\pm 0.28}$	$15.67_{\pm 0.32}$	$19.70_{\pm 0.22}$	$\mathbf{23.11}_{\pm 0.10}$	$18.95_{\pm 0.11}$
Ours + FDD	$\mathbf{25.12}_{\pm 0.43}$	$11.46_{\pm 0.38}$	$\mathbf{16.05}_{\pm 0.29}$	$20.21_{\pm 0.22}$	$22.79_{\pm 0.07}$	$\mathbf{19.13}_{\pm 0.14}$

As reported in Table 12, **CONTRA only** consistently outperforms standard **FDD**. This validates that parameterizing a continuous velocity field v_θ via Rectified Flow provides superior trajectory guidance compared to matching static intermediate states and discrete layer differences.

Furthermore, applying the progressive curriculum directly to FDD (**FDD + Curriculum**) leads to optimization instability and noticeable degradation on benchmarks such as *Dolly* and *Self-Instruct*. We attribute this to a fundamental geometric mismatch: FDD relies on localized, pointwise feature MSE and first-order difference vectors, which become uninformative or noisy when extrapolated across larger step distances required by long-range curriculum supervision. In contrast, CONTRA-KD explicitly models continuous vector fields across the representation space, ensuring well-conditioned supervision targets throughout all stages of curriculum scaling. Finally, combining continuous trajectory alignment with FDD (**Ours + FDD**) achieves the highest overall performance, demonstrating the complementary nature of continuous flow matching with traditional feature regularization.

F.4 Alternative Cross-tokenizer Alignment via Edit Distance

To complement the LCS-based span construction, we describe an alternative cross-tokenizer alignment strategy based on token-level edit distance. It operates on the same character-offset representation and produces the identical span-pair format, allowing the span-level rectified-flow objective and curriculum distillation to apply unchanged.

Span Construction via Edit Distance. As in the LCS variant, the teacher and student tokenize the same input x but produce sequences of different lengths. Each token is annotated with its character-level span $(\text{start}, \text{end})$, yielding offset sequences $\mathbf{o}^T$ and $\mathbf{o}^S$. Special tokens with end offset 0 and prompt tokens preceding the response start indices s_i, s_j are excluded from alignment.

Let $\{a_i\}_{i=1}^m$ and $\{b_j\}_{j=1}^n$ denote the filtered student and teacher offset spans in the response region. We compute a Levenshtein alignment between the two span sequences, where two tokens match if and only if their character spans are identical. The dynamic-programming cost matrix $D[i][j]$ is initialized as $D[0][j] = j$, $D[i][0] = i$ and updated as:

$$D[i][j] = \min \begin{cases} D[i-1][j-1] + \mathbb{I}[a_i \neq b_j] & \text{(substitute / match)} \\ D[i-1][j] + 1 & \text{(delete)} \\ D[i][j-1] + 1 & \text{(insert)} \end{cases} \tag{16}$$

where $\mathbb{I}[\cdot]$ evaluates to 0 if the two offset spans are identical and 1 otherwise. Backtracking from (m, n) to $(0, 0)$ yields a minimal edit path over operations $\{\text{diagonal}, \text{delete}, \text{insert}\}$.

We traverse this path while tracking consumed student positions p and teacher positions q. At each step, candidate boundary indices $u = \text{idx}_p^S + 1$ and $v = \text{idx}_q^T + 1$ are obtained in original token indexing. A boundary pair is retained if and only if both sequences terminate at the identical character offset:

$$\text{end}_p^S = \text{end}_q^T, \quad u > u_{\text{prev}}, \quad v > v_{\text{prev}}. \tag{17}$$

This condition filters insertions or deletions that do not terminate on shared character boundaries and enforces strict monotonicity. The retained pairs define N_{sp} aligned span pairs structured identically to the LCS construction:

$$\mathcal{M}_s = \left\{(\text{Span}_i^S, \text{Span}_i^T)\right\}_{i=1}^{N_{\text{sp}}}, \quad \text{Span}_i^{\cdot} = (s_i^{\cdot}, e_i^{\cdot}). \tag{18}$$

If $|\mathcal{M}_s|$ exceeds the maximum capacity ($N_{\text{span}} = 1024$), we apply uniform subsampling to bound GPU memory overhead.

Intuitively, LCS strictly preserves common sub-sequences of shared character endpoints in monotone order, whereas edit distance explicitly accommodates local structural divergence via substitution, insertion, and deletion penalties prior to projecting back to shared boundaries.

Span Representation. Identical to the LCS variant, at aligned layer j, span representations are constructed by mean-pooling token hidden states within each span:

$$C_i^{S,(j)} = \frac{1}{e_i^S - s_i^S + 1} \sum_{t=s_i^S}^{e_i^S} f_{\text{proj}}(h_t^{j,S}), \qquad C_i^{T,(j)} = \frac{1}{e_i^T - s_i^T + 1} \sum_{t=s_i^T}^{e_i^T} h_t^{j,T}, \tag{19}$$

after which CONTRA-KD trajectory alignment (rectified-flow velocity field training and curriculum target generation $\tilde{C}_i^{S,(j)}$) is applied at the span level. Table 13 compares the distillation performance of LCS against Edit Distance on the cross-tokenizer pair (Qwen1.5 1.8B → GPT-2 0.1B).

Table 13: Ablation of span alignment mechanisms in CONTRA-KD (Qwen1.5 1.8B → GPT-2 0.1B). ROUGE-L scores (%) are averaged over 5 random seeds ± standard deviation.

Span Construction	Dolly	Vicuna	Self-Inst	S-NI	Avg.
Edit Distance	$23.25_{\pm 0.38}$	$14.95_{\pm 0.38}$	$12.21_{\pm 0.35}$	$20.16_{\pm 0.21}$	$17.64_{\pm 0.17}$
LCS	$\mathbf{23.53}_{\pm 0.31}$	$\mathbf{15.38}_{\pm 0.28}$	$\mathbf{12.23}_{\pm 0.65}$	$\mathbf{20.97}_{\pm 0.21}$	$\mathbf{18.03}_{\pm 0.20}$

As reported in Table 13, deterministic LCS consistently outperforms the edit-distance heuristic across all benchmark datasets, raising the average ROUGE-L score from 17.64% to 18.03%. LCS extracts the longest strictly shared sequence without introducing insertion or deletion distortions, resulting in cleaner and more semantically aligned span-level representations for continuous trajectory distillation. Furthermore, LCS requires lower computational overhead during batch pre-processing, making it the superior design choice for cross-tokenizer CONTRA-KD.

F.5 Relative Performance Gains Across Baselines

To highlight the consistent benefits of continuous trajectory alignment, Table 14 reports the relative performance increase (%) in average ROUGE-L score when CONTRA-KD is integrated into existing state-of-the-art distillation baselines across both GPT-2 (1.5B → 0.1B) and OpenLLaMA-2 (7B → 3B) teacher–student pairs.

Integrating CONTRA-KD yields consistent performance gains across both logit-matching objectives (e.g., DistiLLM, DistiLLM-2, CSD) and discrete feature-matching frameworks (FDD). On GPT-2 (1.5B → 0.1B), adding CONTRA-KD achieves relative improvements ranging from $+3.01\%$ to $+5.95\%$ across all baselines. Similarly, on OpenLLaMA-2 (7B → 3B), CONTRA-KD introduces up to a $+3.98\%$ relative increase over standalone CSD. These relative gains demonstrate that continuous vector field alignment provides orthogonal, synergistic benefits that standalone logit-level or discrete layer-wise matching objectives cannot achieve alone.

F.6 Impact of Number of Distilled Layers

We study the effects of the number of distilled layers L_D in CONTRA-KD, which controls how many intermediate student–teacher layer pairs are used to learn and apply the trajectory alignment. A larger L_D provides denser supervision along the feature trajectory, while a smaller L_D reduces computational overhead but offers coarser guidance.

Table 14: Relative increase (%) in average ROUGE-L performance when integrating CONTRA-KD into state-of-the-art baselines across two model architectures.

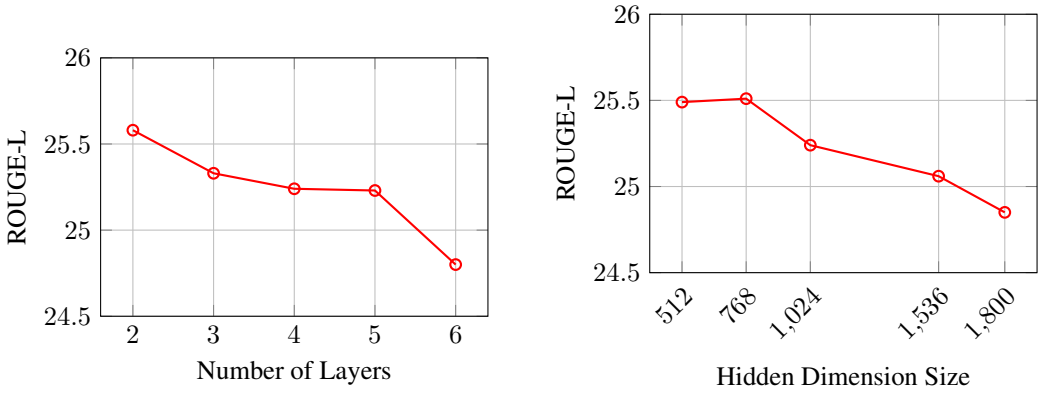
Method	Baseline	+CONTRA	Increase (%)
GPT-2 (1.5B → 0.1B)			
DistiLLM	20.49	21.71	**+5.95%**
FDD	18.47	19.13	**+3.57%**
DistiLLM-2	19.30	19.88	**+3.01%**
CSD	18.66	19.24	**+3.11%**
OpenLLaMA-2 (7B → 3B)			
DistiLLM	27.34	28.15	**+2.96%**
FDD	27.43	28.01	**+2.11%**
DistiLLM-2	28.06	28.59	**+1.89%**
CSD	26.39	27.44	**+3.98%**

Figure 5 illustrates how the number of distilled intermediate layers influences student performance. Performance peaks when a moderate number of 4 layers is distilled; further increasing the number of layers results in a slight degradation. Following the conclusion of Gong et al. (2025), we attribute this decline to over-regularization, where excessive constraints limit the student's ability to adapt its own dynamics.



Figure 5: Impact of the number of distilled layers L_D on distillation performance.

F.7 Impact of Velocity Field Architecture



(a) Impact of velocity field's depth.

(b) Impact of velocity field's hidden dimension size.

Figure 6: Ablation study on the architecture of the velocity field. **Left:** Increasing the depth of the velocity field degrades distillation performance, indicating that the student–teacher trajectory mapping is smooth and does not require deep hierarchies. **Right:** The velocity field width must scale with the base model dimension; overly narrow networks introduce an information bottleneck and reduce feature transport fidelity.

We analyze the impact of the velocity field architecture on distillation performance, focusing on two key design factors: model depth and hidden dimension. All experiments here are conducted using GPT2 1.5B → 0.1B pair with DistiLLM baseline and evaluation on the Dolly test set.

Impact of Velocity Field's depth. We examine the effect of network depth by varying the number of feed-forward blocks in the velocity field network v_θ from 2 to 6, while keeping the hidden dimension fixed at 1024. As shown in Figure 6 (left), increasing the depth of the velocity field does

not improve performance; instead, it leads to a gradual degradation. The highest ROUGE-L score of **25.58** is achieved with the shallowest configuration (2 layers), whereas the 6-layer model drops to 24.80. This trend suggests that the student-to-teacher trajectory mapping is relatively smooth and does not require deep hierarchical abstractions. A shallow and lightweight velocity field is therefore sufficient to model the feature transport, reinforcing the efficiency of our approach.

Impact of Velocity Field's hidden dimension. We investigate the effect of the velocity field width. Figure 6 (right) shows the ROUGE-L scores for velocity fields with varying hidden dimensions. For the GPT2 setting (model dimension $d = 768$), compact velocity fields with comparable dimensionality perform best. However, when applying CONTRA-KD to larger language models (e.g., LLaMA2 13B → 7B (as shown in Table 15), where hidden dimensions exceed 4096), we observe that narrow velocity fields introduce a severe *information bottleneck*. If the velocity field width is significantly smaller than the student or teacher hidden dimension, it fails to preserve the geometric structure of the feature manifold during projection, leading to degraded performance. These results indicate that the velocity field dimension should scale proportionally with the base model size in order to maintain faithful feature transport.

Table 15: Impact of the velocity field's hidden dimension in LLaMA2 13B → 7B. ROUGE-L (%) scores across different evaluation datasets, averaged over 5 random seeds.

Hidden Dimension Size	Dolly	Self-Inst	Vicuna	S-NI	UnNI	Avg.
$d = 2048$	29.41	20.84	21.23	**36.54**	36.19	28.84
$d = 3200$	**31.45**	**23.69**	**21.48**	36.19	**37.70**	**30.10**

F.8 Velocity Field Architecture and Scaling

To address computational overhead when scaling CONTRA-KD to larger foundation models (e.g., beyond the 13B–7B parameter regime), the parameterization of the velocity field network v_θ can be optimized for lightweight inference and training. Specifically, standard dense projection layers in the velocity field can be replaced with a **low-rank projection architecture** (e.g., parameterizing intermediate feature transformations via low-rank factorizations).

To empirically verify that low-rank parameterization preserves continuous trajectory alignment without sacrificing representation fidelity, we evaluate a **Low-Rank Velocity Field (VF)** against our standard **Dense VF** baseline on the GPT-2 (1.5B → 0.1B) benchmark setup.

Table 16: Ablation of velocity field architecture (Dense vs. Low-Rank projection) on GPT-2 (1.5B → 0.1B). ROUGE-L scores (%) are averaged over 5 random seeds ± standard deviation.

Method	**Dolly**	**Self-Inst**	**Vicuna**	**S-NI**	**UnNI**	**Avg.**
Dense VF (Ours)	**26.22**$_{\pm 0.31}$	**12.99**$_{\pm 0.53}$	**17.41**$_{\pm 0.30}$	**25.07**$_{\pm 0.21}$	26.85$_{\pm 0.11}$	**21.71**$_{\pm 0.14}$
Low-Rank VF	25.43$_{\pm 0.49}$	12.62$_{\pm 0.35}$	17.40$_{\pm 0.43}$	23.08$_{\pm 0.16}$	**27.11**$_{\pm 0.10}$	21.13$_{\pm 0.15}$

As reported in Table 16, the low-rank design retains highly competitive performance across all instruction-following tasks, achieving an average ROUGE-L score of 21.13% compared to 21.71% for the dense architecture (a minor 0.58-point drop), while substantially reducing computational and memory overhead. Notably, performance on *Vicuna* (17.40% vs. 17.41%) and *UnNI* (27.11% vs. 26.85%) remains virtually identical or slightly improved.

This ablation confirms that the velocity field can be parameter-efficiently factorized without disrupting continuous flow modeling or trajectory continuity. While we leave exhaustive architectural optimization and automated architecture search for velocity fields to future work, these results demonstrate a practical path for scaling CONTRA-KD to foundation models exceeding 13B parameters.