CacheFlow: Fast Human Motion Prediction by Cached Normalizing Flow

Anonymous Author(s)
Affiliation
Address
email

Abstract

Many density estimation techniques for 3D human motion prediction require a significant amount of inference time, often exceeding the duration of the predicted time horizon. To address the need for faster density estimation for 3D human motion prediction, we introduce a novel flow-based method for human motion prediction called CacheFlow. Unlike previous conditional generative models that suffer from time efficiency, CacheFlow takes advantage of an unconditional flow-based generative model that transforms a Gaussian mixture into the density of future motions. The results of the computation of the flow-based generative model can be precomputed and cached. Then, for conditional prediction, we seek a mapping from historical trajectories to samples in the Gaussian mixture. This mapping can be done by a much more lightweight model, thus saving significant computation overhead compared to a typical conditional flow model. In such a two-stage fashion and by caching results from the slow flow model computation, we build our CacheFlow without loss of prediction accuracy and model expressiveness. This inference process is completed in approximately one millisecond, making it 4× faster than previous VAE methods and 30× faster than previous diffusion-based methods on standard benchmarks such as Human3.6M and AMASS datasets. Furthermore, our method demonstrates improved density estimation accuracy and comparable prediction accuracy to a SOTA method on Human3.6M. Our code and models will be publicly available.

1 Introduction

The task of 3D human motion prediction is to forecast the future 3D pose sequence given an observed past sequence. Traditional motion prediction methods are often based on deterministic models and can struggle to capture the inherent uncertainty in human movement. Recently, stochastic approaches have addressed this limitation. Stochastic approaches allow models to sample multiple possible future motions. Stochastic human motion prediction methods utilize conditional generative models such as generative adversarial networks (GANs) [18], variational autoencoders (VAEs) [27], and denoising diffusion probabilistic model [24]. However, many stochastic approaches cannot explicitly model the probability density distribution.

Conversely, density estimate-based approaches explicitly model the probability density distribution. In safety-critical applications such as autonomous driving [51] and human-robot interaction [29, 31, 9], a density estimate can represent all possible future motions (not just a few samples) by tracking the volume of density. It can be used to derive guarantees on safety [43, 58, 64].

However, previous density estimation suffers from high computational cost. The expensive computational cost can prohibit applications to real-time use-cases, especially with high dimensional data such as human motions. For instance, kernel density estimation (KDE) [56, 52] requires an exponentially

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

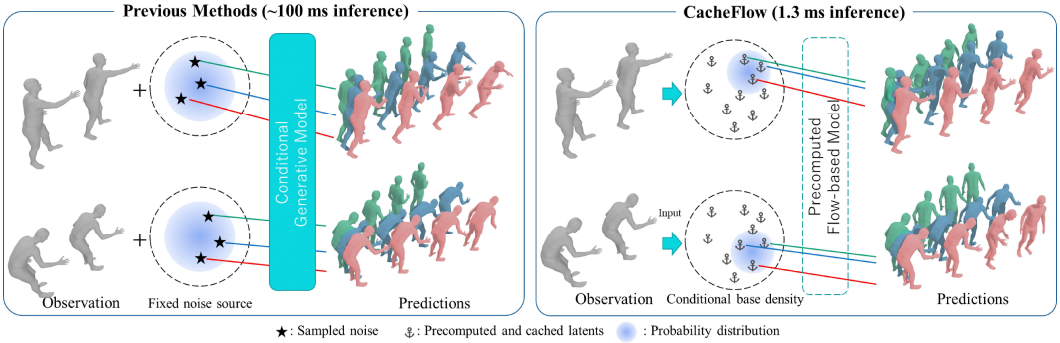

Figure 1: **Previous methods vs. Our CacheFlow.** Previous methods of stochastic motion prediction generate multiple future motions by sampling noises from the fixed source in an ad hoc manner. In contrast, CacheFlow uses the precomputed and cached latent-motion pairs from an unconditional flow-based generative model. Thus, the computation of the unconditional flow can be skipped at inference. One can achieve fast inference by selecting predictions from these cached pairs.

growing number of samples for accurate estimation. Concretely, more than one trillion samples are required for accurate KDE over a 48-dim pose over 100 frames of human motion prediction [60].

In contrast to traditional KDE, recent parametric density estimation approaches use conditional flow-based generative models, including normalizing flows [55, 62, 63] and continuous normalizing flows [13]. These flow-based generative models ("flow-based model" for brevity) directly estimate the density to avoid time-consuming sampling required in KDE. However, inferring the exact probability of possible future motions remains computationally expensive. This is because capturing the full shape of the distribution requires evaluating the probabilities of many potential future motions.

To address this computational limitation, we propose a fast density estimation method based on a flow-based model called "CacheFlow". Our CacheFlow utilizes an unconditional flow-based model for prediction, as illustrated in Figure 1. Since the unconditional flow-based model is independent of past observed motions, its calculation can be precomputed and skipped at inference. This precomputation omits a large portion of computational cost. To achieve further acceleration, our unconditional flow-based model represents transformation between a lightweight conditional base density and the density of future motions. At inference, the density of future motion is estimated by computing the lightweight conditional base density and combining it with the precomputed results of the flow-based model. The inference of our method is approximately one millisecond.

CacheFlow demonstrates comparable accuracy to previous methods on standard stochastic human motion prediction benchmarks, Human3.6M [25] and AMASS [42]. Furthermore, our method estimates density more accurately than previous stochastic human motion prediction methods with KDE. CacheFlow shows improved computational efficiency, making it well-suited for real-time applications. The contributions of this paper are four-fold as follows:

1. We introduce a novel fast density estimation called CacheFlow on human motion prediction.
2. We can sample diverse future motion trajectories with explicit density estimation, and we experimentally confirm that our method can estimate accurate density.
3. Our method achieves comparable prediction accuracy to other computationally intense methods on several benchmarks.

2 Related Work

2.1 Human Motion Prediction

Deterministic approaches. Early approaches on human motion prediction [1, 7, 17, 8, 21, 26, 33] focused on deterministic settings. They predict the most likely motion sequence based on the past motion. A wide range of architectures were proposed including multi-layer perceptron [21], recurrent neural networks [17, 26, 46, 20, 53, 37], convolutional neural networks [33, 50], transformers [1, 10, 48], and graph neural networks (GNNs) [44, 34, 14, 35]. GNN can account for the explicit tree

expression of the human skeleton, while other architectures implicitly learn the dependencies between joints.

Stochastic approaches. To capture the inherent uncertainty in human movements, recent works have focused on stochastic human motion prediction to predict multiple likely future motions. The main stream of stochastic methods use generative models for the purpose, such as generative adversarial networks (GANs) [5, 30], variational autoencoder (VAE) [65, 70, 45, 11], and denoising diffusion probabilistic model (DDPM) [4, 12, 66, 61]. To improve the diversity of predictions, diversity-promoting loss [45, 4] or explicit sampling techniques [66] were proposed. In contrast to generative models, anchor-based methods [69, 68] learn a fixed number of anchors corresponding to each prediction to ensure diversity. However, most stochastic methods cannot describe the density of future motions explicitly. This prevents exhaustive or maximum likelihood sampling for practical applications. On the contrary, our method allows for explicit density estimation using normalizing flows [28].

2.2 Density Estimation

Density estimation asks for explicit calculation of the probability for samples from a distribution. Density estimation is derived by non-parametric or parametric methods.

Non-parametric Approach. The representative non-parametric density estimation is kernel density estimation (KDE) [56, 52]. KDE can estimate density by using samples from generative models. However, KDE requires a large number of samples for accurate estimation. Therefore, it often cannot run in real-time.

Parametric Approach. As a representative parametric model, Gaussian mixture models (GMMs) parametrize density with several Gaussian distributions and their mixture weights. Its nature of mixing Gaussian priors limits its ability to generalize to complex data distribution. Another parametric approach with more expressivity is flow-based generative models [28]. By a learned bijective process, normalizing flows (NFs) [55, 62, 63] transform a simple density like the standard normal distribution into a complex data density. Recently, continuous normalizing flows (CNFs) [13, 19] achieve more expressive density than standard normalizing flows via an ODE-based bijective process. While training of CNFs is inefficient due to the optimization of ODE solutions, an efficient training strategy named flow matching [36] was proposed. FlowChain [40] was proposed for fast and efficient density estimation in human trajectory forecasting. FlowChain improves the inference time efficiency by reusing results from the conditional flow-based method while the past sequences are similar. However, with significantly different past sequences, FlowChain's efficiency can't hold anymore. Unlike FlowChain, our method can perform fast and efficient inference regardless of past sequences.

3 Preliminary

3.1 Problem Formulation

The task of human motion prediction aims to use a short sequence of observed human motion to predict the future unobserved motion sequence of that person. Human motion is represented by a sequence of human poses in a pre-defined skeleton format of 3D locations of J joints, $X \in \mathbb{R}^{J\times 3}$. As input to our model, we have the past (history of) human motion as a sequence $\boldsymbol{c} = [X_1, X_2, ..., X_H] \in \mathbb{R}^{H\times J\times 3}$ over H timesteps. To predict the future human motion sequence of F timesteps, we can formulate the problem as one of conditional generation using the conditional probability function, $p(\boldsymbol{X}|\boldsymbol{c})$, where $\boldsymbol{X} = [X_{H+1}, X_{H+2}, ..., X_{H+F}] \in \mathbb{R}^{F\times J\times 3}$. Similar to the stochastic human motion prediction paradigm, the method should also allow for sampling n multiple future sequences $\{\boldsymbol{X}_1, ..., \boldsymbol{X}_n\}$ from $p(\boldsymbol{X}|\boldsymbol{c})$. The focus of our work is to accelerate the inference time of estimate and sampling of the conditional density function $p(\boldsymbol{X}|\boldsymbol{c})$.

3.2 Normalizing Flow

Normalizing flow [55, 62, 63] is a generative model with explicit density estimation. It follows a bijective mapping f_θ with learnable parameters θ. It transforms a simple base density $q(\boldsymbol{z})$ such as a Gaussian distribution into the complex data density $p(\boldsymbol{x})$. We can analytically estimate the exact

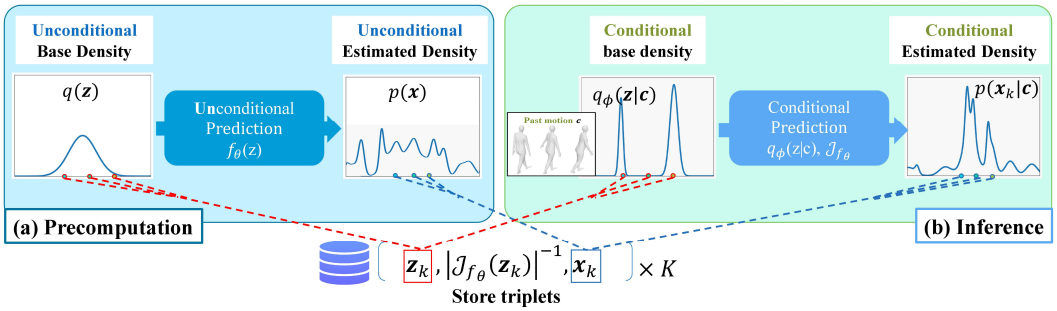

Figure 2: **Overview of our CacheFlow.** Our method utilizes the unconditional flow-based model f_θ. This f_θ maps the lightweight conditional base density $q_\phi(\boldsymbol{z}|\boldsymbol{c})$ into future motion density $p(\boldsymbol{x}|\boldsymbol{c})$. In this formulation, the flow-based model is independent of past motions. Thus, we can precompute the unconditional flow-based model. These results are cached as K triplets as shown in (a). Due to the precomputation, we can skip the inference of f_θ and omit a large portion of the entire computation. At inference, density estimation is achieved by only evaluating the lightweight conditional base density $q_\phi(\boldsymbol{z}_k|\boldsymbol{c})$ and combining it with the stored K triplets as shown in (b).

probability via the change-of-variables formula as follows:

$$\boldsymbol{x} = f_\theta(\boldsymbol{z}), \quad \boldsymbol{z} = f_\theta^{-1}(\boldsymbol{x}). \tag{1}$$

$$p(\boldsymbol{x}) = q(\boldsymbol{z})|\det \mathcal{J}_{f_\theta}(\boldsymbol{z})|^{-1}, \tag{2}$$

where $\mathcal{J}_{f_\theta}(z) = \frac{\partial f_\theta}{\partial \boldsymbol{z}}$ is the Jacobian of f_θ at $\boldsymbol{z}$. The parameters θ of f_θ can be learned by maximizing the likelihood (or conditional likelihood) of samples $\hat{\boldsymbol{x}}$ from datasets or minimizing the negative log-likelihood as $\mathcal{L}_{\text{NLL}} = -\log p(\hat{\boldsymbol{x}})$. When $\boldsymbol{x}$ and $\boldsymbol{z}$ are latent codes, normalizing flow is transformed into latent normalizing flow. We follow this pattern in our method. We encode the past human motion into $\boldsymbol{x}$ by an ecoder network $\mathcal{E}$ and decode it by a decoder network $\mathcal{D}$:

$$\boldsymbol{x} = \mathcal{E}(\boldsymbol{X}), \boldsymbol{X} = \mathcal{D}(\boldsymbol{x}). \quad \boldsymbol{x} \sim \mathbb{R}^d, \boldsymbol{X} \sim \mathbb{R}^{F \times J \times 3} \tag{3}$$

The encoder and decoder are trained by reconstruction. In the later part of this paper, for simplicity, we discuss the method at the latent representation level and model the conditional generation task as $p(\boldsymbol{x}|\boldsymbol{c})$.

3.3 Continuous Normalizing Flow (CNF)

Continuous normalizing flow (CNF) [13, 19] is a normalizing flow variant based on an ordinary differential equation (ODE). CNF defines t-continuous path $\boldsymbol{z}_t$ between the base density space $\boldsymbol{z}_0 \sim q(\boldsymbol{z})$ and the data space $\boldsymbol{z}_1 = \boldsymbol{x} \sim p(\boldsymbol{x})$. This $\boldsymbol{z}_t$ is defined by the parameterized vector field $\frac{d\boldsymbol{z}_t}{dt} = v_\theta(\boldsymbol{z}_t)$. The data $\boldsymbol{x} = \boldsymbol{z}_1$ is generated via numerical integration of vector field $v_\theta(\boldsymbol{z}_t)$ as follows:

$$\boldsymbol{x} = \boldsymbol{z}_1 = \boldsymbol{z}_0 + \int_0^1 v_\theta(\boldsymbol{z}_t)dt. \tag{4}$$

The CNF transformation Equation (4) is denoted as $\boldsymbol{x} = f_\theta(\boldsymbol{z})$ for brevity. Although CNF can be trained by minimizing negative log-likelihood, it is time-consuming due to the numerical integration of ODE.

3.4 Flow Matching

In order to train the parameterized vector field efficiently, one can leverage the flow matching [36] strategy. As a new training strategy, Flow Matching avoids the numerical integration of ODE by directly optimizing the vector field $v_\theta(\boldsymbol{z}_t)$. The objective of flow matching is to match the parameterized vector field $v_\theta(\boldsymbol{z}_t)$ to the ground truth vector field $u(\boldsymbol{z}_t)$ via mean squared error as follows:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t \sim \mathcal{T}(0,1), \boldsymbol{z}_t} ||v_\theta(\boldsymbol{z}_t) - u(\boldsymbol{z}_t)||^2, \tag{5}$$

where $\mathcal{T}(0,1)$ is a distribution ranging from 0 to 1.

However, we cannot obtain the ground truth vector field $u(\boldsymbol{z}_t)$ directly. Ripman *et al*. [36] suggest defining the conditional ground truth $u(\boldsymbol{z}_t|\hat{\boldsymbol{z}}_1)$ instead. Specifically, it is modeled as a straight vector field $\hat{\boldsymbol{z}}_1 - \boldsymbol{z}_0$ in Rectified Flow [13, 19]. This is called conditional flow matching [36] trained by the following objective:

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t\sim\mathcal{T}(0,1),\hat{\boldsymbol{z}}_1,\boldsymbol{z}_t}||v_\theta(\boldsymbol{z}_t) - u(\boldsymbol{z}_t|\hat{\boldsymbol{z}}_1)||^2. \tag{6}$$

The gradients of $\mathcal{L}_{\text{FM}}$ of Equation (5) and $\mathcal{L}_{\text{CFM}}$ of Equation (6) are identical *w.r.t* $\boldsymbol{\theta}$. We exploit expressive CNFs with efficient Flow Matching training to estimate the future motion density $p(\boldsymbol{x}|\boldsymbol{c})$.

4 Proposed Method

4.1 Overview of CacheFlow

We estimate the future motion density $p(\boldsymbol{x}|\boldsymbol{c})$ by transforming a conditional base distribution $q_\phi(\boldsymbol{z}|\boldsymbol{c})$. This q_ϕ is conditioned on the past motion $\boldsymbol{c}$. Then we can sample predictions $\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{c})$ for stochastic human motion prediction. Most traditional approaches based on conditional generative models use a trivial source distribution, often a simple Gaussian. However, we redefine the source distribution to be more informative and directly regressed from past motions. This allows us to develop a much lighter and faster model for predicting future movements.

To build this informative conditional base distribution q_ϕ, we would incorporate an unconditional flow-based model $f_\theta : \boldsymbol{x} = f_\theta(\boldsymbol{z})$ that maps latent variable $\boldsymbol{z}$ into motion representation $\boldsymbol{x}$. To understand how q_ϕ and f_θ are connected, we first reparametrize the future motion density $p(\boldsymbol{x}|\boldsymbol{c})$ by a change of variables of probability equation as follows:

$$p(\boldsymbol{x}|\boldsymbol{c}) = q(\boldsymbol{z}|\boldsymbol{c})\left|\det\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}\right|, \tag{7}$$

$$= q(\boldsymbol{z}|\boldsymbol{c})\left|\det\left(\frac{\partial f_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}}\right)^{-1}\right|, \tag{8}$$

$$= q(\boldsymbol{z}|\boldsymbol{c})|\det\mathcal{J}_{f_\theta}(\boldsymbol{z})|^{-1}. \tag{9}$$

This parametrization trick differs from the widely-used conditional density formulation [67] where $\boldsymbol{c}$ is conditioned to the flow-based model f_θ. In this formulation, only the conditional base density $q(\boldsymbol{z}|\boldsymbol{c})$ varies depending on $\boldsymbol{c}$ during inference, whereas the unconditional flow-based model $\boldsymbol{x} = f_\theta(\boldsymbol{z})$ and the Jacobian $|\det\mathcal{J}_{f_\theta}(\boldsymbol{z})|^{-1}$ are kept same during inference and thus can be reused as-is once calculated.

Therefore, we could precompute the mapping results and Jacobians of an unconditional flow-based model f_θ. We cache the triplets $t = \{\boldsymbol{z}, |\det\mathcal{J}_{f_\theta}(z)|^{-1}, \boldsymbol{x}\}$ for later reuse in the inference stage, as shown in Figure 2(a).

Then, during inference, we design a new trick to reuse the cached triplets by associating them with the specific conditions of the past motion sequences, as shown in Figure 2(b). Now, instead of a typical conditional generative model, e.g., conditional normalizing flow, we only need a lightweight model to model the conditional base density $q_\phi(\boldsymbol{z}|\boldsymbol{c})$ and achieve similar expressivity. We could finally estimate the future motion density by $p(\boldsymbol{x}|\boldsymbol{c}) = q_\phi(\boldsymbol{z}|\boldsymbol{c})|\det\mathcal{J}_{f_\theta}(\boldsymbol{z})|^{-1}$. The method is summarized as pseudocode in Algorithm 1. In the following paragraphs, we elaborate on the details of our method.

4.2 Precompute Unconditional Flow-based Model

As the first step of our method, we use the human motion dataset to learn an unconditional flow-based model f_θ. From this unconditional human motion prediction model, we will collect the triplets $t = \{\boldsymbol{z}, |\det\mathcal{J}_{f_\theta}(\boldsymbol{z})|^{-1}, \boldsymbol{x}\}$ for later use. This part is illustrated in Figure 2(a).

In our implementation, we built the unconditional flow model by CNFs due to its proven expressivity for predicting human motion. The unconditional model is trained to predict a fixed-length future motion $\boldsymbol{x}$ given a noise sample $\boldsymbol{z}$ from a source distribution $q_\phi(\boldsymbol{z}|\boldsymbol{c})$:

$$f_\theta : \mathbb{R}^d \longrightarrow \mathbb{R}^d \tag{10}$$

Because $\boldsymbol{z}$ is sampled from a known distribution and normalizing-flow models are deterministic with reversible bijective transformation, we could know the density of each $\{\boldsymbol{z}, \boldsymbol{x}\}$ pair. We train the unconditional continuous normalizing flow with the flow matching objective described in Equation (6). Then we collect K samples denoted by the triplet $t_k = \{\boldsymbol{z}_k, |\mathrm{det}\mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}$. Triplets are collected by applying the inverse transform of f_θ to ground truth future motions in the training split. These triplets are cached for fast inference as described in Section 4.3. This caching operation is different from anchor-based methods [69, 68] since CacheFlow caches all motions of the training split.

4.3 Conditional Inference by CacheFlow

In previous methods, conditional human motion prediction typically requires a conditional generative model. For instance, it is a conditional flow-based or diffusion model. These models usually have poor time efficiency due to delicate but heavy architecture. Instead, inspired by Equation (9), we can reuse the results of unconditional inverse transformation as triplets $t_k = \{\boldsymbol{z}_k, |\mathrm{det}\mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}$. Thus, we can perform conditional inference by only evaluating a conditional base distribution $q(\boldsymbol{z}|\boldsymbol{c})$. We model this conditional base distribution by a learnable model, thus we denote it as $q_\phi(\boldsymbol{z}|\boldsymbol{c})$. This model can be very lightweight since the unconditional transformation f_θ gives enough expressivity. $q_\phi(\boldsymbol{z}|\boldsymbol{c})$ runs much faster than a typical conditional generative model for human motion prediction. This part is illustrated in Figure 2(b).

In our implementation, $q_\phi(\boldsymbol{z}|\boldsymbol{c})$ is constructed as a parametrized Gaussian mixture $\{\mathcal{N}(\mu_m(\boldsymbol{c}), \sigma_m^2(\boldsymbol{c}))\}$, with M mixture weights $w_m(\boldsymbol{c})$, such that $\sum_{m=1}^{M} w_m = 1$. Each μ_m and σ_m are regressed based on the feature of past motion $\boldsymbol{c}$. We use a lightweight single-layer RNN for regression to determine the GMM composition. Although the unconditional flow-based model f_θ and the conditional base density q_ϕ can be trained separately, we found that jointly training f_θ and q_ϕ improves model performance. We train the joint model by summation of log-likelihood for q_ϕ and flow matching for f_θ as explained in Equation (6) as follows:

$$\mathcal{L} = -\log q_\phi(f_\theta^{-1}(\hat{\boldsymbol{x}})|\boldsymbol{c}) + \mathcal{L}_{\mathrm{CFM}}. \tag{11}$$

With joint learning, f_θ learns an easy mapping for the conditional Gaussian mixture q_ϕ.

With q_ϕ constructed, during inference, we can estimate the conditional density $p(\boldsymbol{x}|\boldsymbol{c})$ by connecting with precomputed triplets $t_k = \{\boldsymbol{z}_k, |\mathrm{det}\mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}$ as

$$p(\boldsymbol{x}_k|\boldsymbol{c}) = q_\phi(\boldsymbol{z}_k|\boldsymbol{c})|\mathrm{det}\mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}. \tag{12}$$

By this inference process, we could optionally generate a future human motion sequence $\boldsymbol{x}$ by retrieving a high-probability sample $\boldsymbol{z}$ from q_ϕ with the past motion sequence as the condition. However, q_ϕ describes a continuous distribution and the stored triplets cannot cover all samples. Therefore, in practice, predicted motion $\boldsymbol{x}_{k^*}$ is selected by the nearest neighbor of the sampling outcome of q_ϕ to the stored triplets:

$$\begin{aligned} k^* &= \mathrm{argmin}_k ||\boldsymbol{z}_k - \boldsymbol{z}||, \\ \text{s.t.} \quad \{t_k &= \{\boldsymbol{z}_k, |\mathrm{det}\mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}, \quad \boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{c})\}, \end{aligned} \tag{13}$$

where k^* is the selected index of the triplets for prediction. By this design, we can sample an arbitrary number of likely future motion sequences by selecting the neighbors of samples $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{c})$.

5 Experimental Evaluation

Datasets. We evaluate our CacheFlow on Human3.6M [25] and AMASS [42]. Human3.6M contains 3.6 million frames of human motion sequences. Human motions of 11 subjects performing 15 actions are recorded at 50 Hz. We follow the setting including the dataset split, the 16-joints pose skeleton definition, and lengths of past and future motions proposed by previous works [47, 38, 71, 54]. The training and test sets of Human3.6M are subjects [S1,S5,S6,S7,S8] and [S9,S11], respectively. The past motion and future motions contain 25 frames (0.5 sec) and 100 frames (2.0 sec). AMASS unifies 24 different human motion datasets including HumanEva-I [59] with the SMPL [41] pose representation. AMASS contains 9M frames at 60 Hz in total. As a multi-dataset collection of AMASS, one can perform a cross-dataset evaluation. We follow the evaluation protocol proposed

Algorithm 1: Precomputation and Inference of CacheFlow.

Input: Past motion $\boldsymbol{c}$
Output: Estimated density $p(\boldsymbol{x}_k|\boldsymbol{c})$
// Precomputation. This does not count for inference time.
for *each future motion* $\boldsymbol{X}_k$ *in the training dataset* **do**
  $\boldsymbol{x}_k \leftarrow \mathcal{E}(\boldsymbol{X}_k)$
  $\boldsymbol{z}_k \leftarrow f_\theta^{-1}(\boldsymbol{x}_k)$
  Calculate $|\det \mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}$
  Store triplet $\{\boldsymbol{z}_k, |\det \mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}$
end
// Fast Inference
for *each triplet* $\{\boldsymbol{z}_k, |\det \mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}$ **do**
  $q_\phi(\boldsymbol{z}_k|\boldsymbol{c}) \leftarrow \sum_{m=1}^{M} w_m \mathcal{N}(\boldsymbol{z}_k; \mu_m(\boldsymbol{c}), \sigma_m^2(\boldsymbol{c}))$
  $p(\boldsymbol{x}_k|\boldsymbol{c}) \leftarrow q_\phi(\boldsymbol{z}_k|\boldsymbol{c})|\det \mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}$
end

	Human3.6M [25]					AMASS [42]					Inference
	APD↑	ADE↓	FDE↓	MMADE↓	MMFDE↓	APD↑	ADE↓	FDE↓	MMADE↓	MMFDE↓	Time[ms]↓
HP-GAN [5]	7.214	0.858	0.867	0.847	0.858	-	-	-	-	-	-
DSF [72]	9.330	0.493	0.592	0.550	0.599	-	-	-	-	-	-
DeLiGAN [23]	6.509	0.483	0.534	0.520	0.545	-	-	-	-	-	-
GMVAE [16]	6.769	0.461	0.555	0.524	0.566	-	-	-	-	-	-
TPK [65]	6.723	0.461	0.560	0.522	0.569	9.283	0.656	0.675	0.658	0.674	30.3
MT-VAE [70]	0.403	0.457	0.595	0.716	0.883	-	-	-	-	-	-
BoM [6]	6.265	0.448	0.533	0.514	0.544	-	-	-	-	-	-
DLow [73]	11.741	0.425	0.518	0.495	0.531	13.170	0.590	0.612	0.618	0.617	30.8
MultiObj [39]	14.240	0.414	0.516	-	-	-	-	-	-	-	-
GSPS [45]	14.757	0.389	0.496	0.476	0.525	12.465	0.563	0.613	0.609	0.633	5.1
Motron [57]	7.168	0.375	0.488	0.509	0.539	-	-	-	-	-	-
DivSamp [15]	15.310	0.370	0.485	0.475	0.516	24.724	0.564	0.647	0.623	0.667	5.2
BeLFusion [4]	7.602	0.372	0.474	0.473	0.507	9.376	0.513	0.560	0.569	0.585	449.3
BeLFusion-D	5.777	0.367	0.472	0.469	0.506	7.458	0.508	0.567	0.564	0.591	39.3
HumanMAC [12]	6.301	0.369	0.480	0.509	0.545	9.321	0.511	0.554	0.593	0.591	1172.9
CoMusion [61]	7.632	0.350	0.458	0.494	0.506	10.848	0.494	0.547	0.469	0.466	352.6
SLD [68]	8.741	0.348	0.436	0.435	0.463	-	-	-	-	-	375.0
FlowPrecomp.	6.101	0.369	0.473	0.481	0.511	7.099	0.511	0.566	0.567	0.586	1.3
w/o Precomp.	5.385	0.374	0.489	0.490	0.531	6.291	0.516	0.586	0.573	0.608	415.9

Table 1: **Quantitative comparisons over the stochastic human motion prediction metrics on Human3.6M and AMASS datasets.** Lower is better for all metrics except APD. The reported inference time is when a method finishes generating 50 prediction samples from receiving the past motion.

by BeLFusion [4] for fair comparison, as predicting future 120 frames (2.0 sec) with 30 frames observation (0.5 sec) with downsampling to 60 Hz.

Metrics. We use the evaluation metrics to measure diversity and accuracy. 50 sampled predictions are evaluated with the following metrics: **Average Pairwise Distance (APD)** [3] evaluates sample diversity. It calculates the mean l_2 distance between all predicted motions. **Average and Final Displacement Error (ADE, FDE)** [2, 32, 22] evaluate accuracy. They calculate the average and final-frame l_2 distances between the ground truth motion and closest prediction in the 50 set. **Multimodal ADE and FDE (MMADE, MMFDE)** [72] also evaluate accuracy in a similar way to ADE and FDE. However, they are calculated over multimodal ground truths selected by grouping similar motions.

We also evaluate the accuracy of density estimation with **Multimodal Log Probability** per dimension. It calculates the log probability of the multimodal ground truths to measure how accurately the estimated density covers possible future motions. We evaluate the log probability on the motion space except for methods with latent space such as our CacheFlow and BeLFusion. While higher is better on APD and multimodal log probability, lower is better on ADE, FDE, MMADE, and MMFDE.

Method	#sample for KDE	MM log prob. per dim ↑	Inference Time[ms] ↓	
BeLFusion	50	-2.383	2305.3	(440.3)
	1000	-1.633	2422.4	(449.3)
CoMusion [61]	50	-15.575	2500.5	(167.0)
	1000	-12.746	5071.5	(2741.3)
SLD [68]	50	0.080	2559.1	(375.0)
CacheFlow	-	**1.304**	**0.5**	(0.5)

Table 2: **Density Estimation Accuracy on Human3.6M.** Inference time of each method is reported as {total time (time without KDE inference)}. Since our method doesn't require KDE for density estimation, the number of samples for KDE is left blank for CacheFlow.

Implementation Details. Our method is based on a latent flow-based model. We utilize a Variational Autoencoder (VAE) to obtain a latent representation. Specifically, we employ the Behavioral Latent Space (BLS) [4] as a VAE to achieve a compact latent representation. BLS ensures smoothness of predicted motions and consistency between the end of the past motion and the start of the predicted motion. Additionally, we compress this representation using linear factorization [68]. The dimensionality of the VAE latent space is 128, which we further reduce to 8 dimensions through linear factorization. We trained the unconditional flow-based model on this 8-dimensional space. The unconditional flow-based model f_θ is a continuous normalizing flow (CNF) model, with its vector field regressed by a U-Net architecture. The conditional base density q_ϕ, as well as the VAE encoder and decoder, are implemented as one-layer Recurrent Neural Networks (RNNs). We used a Gaussian mixture model with $M = 50$ modes to model the conditional base density q_ϕ. We precomputed and collected triplets $t_k = \{\boldsymbol{z}_k, |\det \mathcal{J}_{f_\theta}(\boldsymbol{z}_k)|^{-1}, \boldsymbol{x}_k\}$ using all training samples of each dataset. All experiments, including inference time measuring, were carried out using a single NVIDIA A100 GPU. We used a batch size 64 and the Adam optimizer with a learning rate of 5×10^{-4}.

5.1 Quantitative Evaluation

Accuracy Over a Fixed Number of Predictions. We compare CacheFlow against state-of-the-art methods of stochastic human motion prediction. While we propose using a precomputed set during inference, we also evaluate our method without precomputation. In the absence of precomputation, we sample $\boldsymbol{z}$ from the conditional base density $q_\phi(\boldsymbol{z}|\boldsymbol{c})$ and obtain $\boldsymbol{x}$ through the flow-based model inference, where $\boldsymbol{x} = f_\theta(\boldsymbol{z})$. The results are summarized in Table 1. Since the primary applications of human motion prediction are in real-time scenarios, we also measure the inference time of each method to sample 50 predictions on a GPU.

CoMusion and SLD were successful in predicting motions that are closer to the ground truth than CacheFlow; however, their inference times of 167 and 375 milliseconds are too long for the intended 2000 ms prediction horizon. As a result, over 8% of the first prediction sequence is rendered useless once the prediction is finalized. Therefore, it is difficult to use these methods with slow inference in real-time applications. Although our primary goal is to estimate the density, CacheFlow achieves comparable performances with a 1.3 millisecond inference time. Our method achieves around $4\times$ faster than the fastest VAE method, GSPS, and $30\times$ faster than the fastest diffusion-based method, BeLFusion-D. The inference of our method is fast enough (1.3ms for future 2000ms) and applicable for real-time applications. This inference speed is because the inference of the unconditional flow-based model f_θ is precomputed. We only need to evaluate the lightweight conditional base density q_ϕ at inference. Although our conditional base density q_ϕ is just a Gaussian mixture with low expressive power, our method achieves high accuracy since the precomputed unconditional flow-based model f_θ gives q_ϕ much complexity with almost no overhead in inference.

Density Estimation Accuracy. The density estimation accuracy of each method is compared between CacheFlow and the state-of-the-art methods. The three state-of-the-art methods BeLFusion [4], CoMusion [61], and SLD [68] are selected. CoMusion and SLD were selected since they outperform our method in benchmarks of stochastic human motion prediction. We also include BeLFusion to compare CacheFlow with the method with latent space. We applied KDE to these previous methods since they only sample a set of predictions and cannot estimate density. While we evaluated 50 and 1000 samples for KDE on BeLFusion and CoMusion, SLD only allows 50 samples due to the fixed

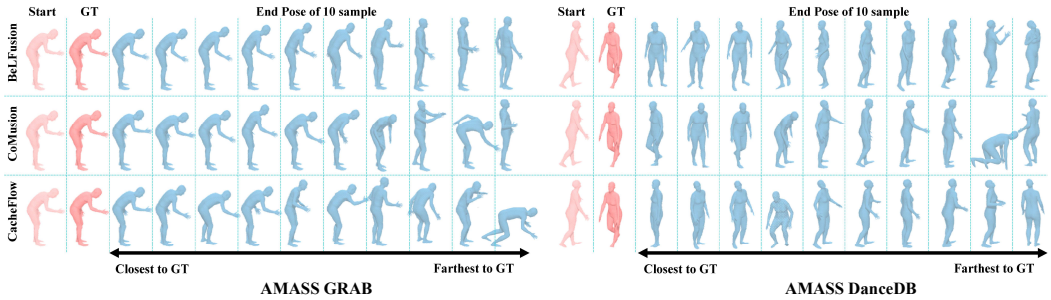

Figure 3: **Qualitative Comparison on AMASS dataset.**

number of anchors corresponding to predictions. We measured the inference time of each method to estimate the density of ten thousand future motions from the past motion input.

The quantitative comparisons over the multimodal ground truth log probability are shown in Table 2. All previous methods suffer from slow inference of their own and KDE on high-dimensional motion data. Their inference time exceeded the prediction horizon of 2000ms in the future. Therefore, they cannot estimate density in real-time. In contrast, our method achieves better estimation accuracy in less than one millisecond. This indicates that CacheFlow has strong discriminative ability to list up possible future motions required for safety assurance. Our method is even faster only on the density estimation (0.5ms) than the inference time reported in Table 1 (1.3ms). This is because we don't need any extra sampling operation in the density estimation.

5.2 Qualitative Comparison of Predicted Motions

To visually evaluate CacheFlow, we conducted a qualitative comparison of methods on the AMASS dataset, as shown in Figure 3. We visualized the end poses of 10 samples from each method alongside the end poses of past motions and the ground truth future motions. The sitting or lying poses were translated to the ground plane, as the global translation is not modeled in human motion prediction. The 10 pose samples are arranged from the closest to the farthest from the ground truth pose based on joint rotations.

Our observations indicate that CacheFlow predicts realistic poses. The closest poses to the ground truths also demonstrate that the accuracy of CacheFlow is comparable to CoMusion, as reflected in the ADE and FDE metrics listed in Table 1. Notably, our method is computationally efficient, operating 100 times faster than the fastest CoMusion. In summary, CacheFlow effectively delivers realistic and accurate predictions.

6 Concluding Remarks

We presented a new flow-based stochastic human motion prediction method named CacheFlow. Our method achieves a fast and accurate estimation of the probability density distribution of future motions. Our unconditional formulation allows precomputation and caching of the flow-based model, thus omitting a large portion of computational cost at inference. The unconditional flow-based model enhanced the expressivity of the lightweight conditional Gaussian mixture with almost no overhead. Experimental results demonstrated CacheFlow achieved comparable prediction accuracy with 1.3 milliseconds inference, much faster than the previous method. Furthermore, CacheFlow estimated a more accurate density than previous methods in less than 1 millisecond.

Our method has one limitation. Prediction and density estimation are performed within precomputed triplets. We cannot estimate the density or predict unseen future motions during precomputation. Our future work is searching for a better precomputation strategy for prediction and estimation with more coverage based on the limited dataset. Furthermore, our method is not limited to prediction tasks but applies to any regression task requiring density estimation. We will investigate the applicability of our CacheFlow on other domains.

References

[1] Emre Aksan, Manuel Kaufmann, Peng Cao, and Otmar Hilliges. A spatio-temporal transformer for 3d human motion prediction. In *3DV*, pages 565–574, 2021.

[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *CVPR*, pages 961–971, 2016.

[3] Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Lars Petersson, and Stephen Gould. A stochastic conditioning scheme for diverse human motion prediction. In *CVPR*, pages 5223–5232, 2020.

[4] German Barquero, Sergio Escalera, and Cristina Palmero. Belfusion: Latent diffusion for behavior-driven human motion prediction. In *ICCV*, pages 2317–2327, 2023.

[5] Emad Barsoum, John Kender, and Zicheng Liu. HP-GAN: probabilistic 3d human motion prediction via GAN. In *CVPRW*, pages 1418–1427, 2018.

[6] Apratim Bhattacharyya, Bernt Schiele, and Mario Fritz. Accurate and diverse sampling of sequences based on a "best of many" sample objective. In *CVPR*, pages 8485–8493, 2018.

[7] Arij Bouazizi, Adrian Holzbock, Ulrich Kressel, Klaus Dietmayer, and Vasileios Belagiannis. Motionmixer: Mlp-based 3d human body pose forecasting. *IJCAI*, 2022.

[8] Judith Butepage, Michael J Black, Danica Kragic, and Hedvig Kjellstrom. Deep representation learning for human motion prediction and classification. In *CVPR*, pages 6158–6166, 2017.

[9] Judith Bütepage, Hedvig Kjellström, and Danica Kragic. Anticipating many futures: Online human motion prediction and generation for human-robot interaction. In *ICRA*, 2018.

[10] Yujun Cai, Lin Huang, Yiwei Wang, Tat-Jen Cham, Jianfei Cai, Junsong Yuan, Jun Liu, Xu Yang, Yiheng Zhu, Xiaohui Shen, et al. Learning progressive joint propagation for human motion prediction. In *ECCV*, pages 226–242, 2020.

[11] Yujun Cai, Yiwei Wang, Yiheng Zhu, Tat-Jen Cham, Jianfei Cai, Junsong Yuan, Jun Liu, Chuanxia Zheng, Sijie Yan, Henghui Ding, et al. A unified 3d human motion synthesis model via conditional variational auto-encoder. In *ICCV*, pages 11645–11655, 2021.

[12] Ling-Hao Chen, Jiawei Zhang, Yewen Li, Yiren Pang, Xiaobo Xia, and Tongliang Liu. Humanmac: Masked motion completion for human motion prediction. In *ICCV*, pages 9544–9555, 2023.

[13] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *NeurIPS*, 31, 2018.

[14] Lingwei Dang, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Msr-gcn: Multi-scale residual graph convolution networks for human motion prediction. In *ICCV*, pages 11467–11476, 2021.

[15] Lingwei Dang, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Diverse human motion prediction via gumbel-softmax sampling from an auxiliary space. In *ACMMM*, pages 5162–5171, 2022.

[16] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[17] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *ICCV*, pages 4346–4354, 2015.

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11): 139–144, 2020.

[19] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: free-form continuous dynamics for scalable reversible generative models. In *ICLR*, 2019.

[20] Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and José MF Moura. Adversarial geometry-aware human motion prediction. In *ECCV*, pages 786–803, 2018.

[21] Wen Guo, Yuming Du, Xi Shen, Vincent Lepetit, Xavier Alameda-Pineda, and Francesc Moreno-Noguer. Back to mlp: A simple baseline for human motion prediction. In *WACV*, pages 4809–4819, 2023.

[22] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *CVPR*, pages 2255–2264, 2018.

[23] Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, and R Venkatesh Babu. Deligan: Generative adversarial networks for diverse and limited data. In *CVPR*, pages 166–174, 2017.

[24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33: 6840–6851, 2020.

[25] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI*, 36(7):1325–1339, 2013.

[26] Ashesh Jain, Amir R Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *CVPR*, pages 5308–5317, 2016.

[27] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *ICLR*, 2014.

[28] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

[29] Hema S Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *TPAMI*, 38(1):14–29, 2015.

[30] Jogendra Nath Kundu, Maharshi Gor, and R Venkatesh Babu. Bihmp-gan: Bidirectional 3d human motion prediction GAN. In *AAAI*, pages 8553–8560, 2019.

[31] Przemyslaw A Lasota and Julie A Shah. A multiple-predictor approach to human motion prediction. In *ICRA*, pages 2300–2307, 2017.

[32] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *CVPR*, pages 336–345, 2017.

[33] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics. In *CVPR*, pages 5226–5234, 2018.

[34] Maosen Li, Siheng Chen, Yangheng Zhao, Ya Zhang, Yanfeng Wang, and Qi Tian. Dynamic multiscale graph neural networks for 3d skeleton based human motion prediction. In *CVPR*, pages 214–223, 2020.

[35] Maosen Li, Siheng Chen, Zihui Liu, Zijing Zhang, Lingxi Xie, Qi Tian, and Ya Zhang. Skeleton graph scattering networks for 3d skeleton-based human motion prediction. In *ICCV*, pages 854–864, 2021.

[36] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

[37] Zhenguang Liu, Shuang Wu, Shuyuan Jin, Qi Liu, Shijian Lu, Roger Zimmermann, and Li Cheng. Towards natural and accurate future motion prediction of humans and animals. In *CVPR*, pages 10004–10012, 2019.

[38] Diogo C Luvizon, David Picard, and Hedi Tabia. 2d/3d pose estimation and action recognition using multitask deep learning. In *CVPR*, pages 5137–5146, 2018.

[39] Hengbo Ma, Jiachen Li, Ramtin Hosseini, Masayoshi Tomizuka, and Chiho Choi. Multi-objective diverse human motion prediction with knowledge distillation. In *CVPR*, pages 8161–8171, 2022.

[40] Takahiro Maeda and Norimichi Ukita. Fast inference and update of probabilistic density estimation on trajectory prediction. In *ICCV*, pages 9795–9805, 2023.

[41] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *ICCV*, pages 5442–5451, 2019.

[42] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass: Archive of motion capture as surface shapes. In *ICCV*, pages 5442–5451, 2019.

[43] Jim Mainprice and Dmitry Berenson. Human-robot collaborative manipulation planning using early prediction of human motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 299–306. IEEE, 2013.

[44] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction. In *ICCV*, pages 9489–9497, 2019.

[45] Wei Mao, Miaomiao Liu, and Mathieu Salzmann. Generating smooth pose sequences for diverse human motion prediction. In *ICCV*, pages 13309–13318, 2021.

[46] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *CVPR*, pages 2891–2900, 2017.

[47] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *ICCV*, pages 2640–2649, 2017.

[48] Angel Martínez-González, Michael Villamizar, and Jean-Marc Odobez. Pose transformers (potr): Human motion prediction with non-autoregressive transformers. In *ICCV*, pages 2276–2284, 2021.

[49] Leland McInnes and John Healy. UMAP: uniform manifold approximation and projection for dimension reduction. *CoRR*, abs/1802.03426, 2018.

[50] Omar Medjaouri and Kevin Desai. Hr-stan: High-resolution spatio-temporal attention network for 3d human motion prediction. In *CVPR*, pages 2540–2549, 2022.

[51] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on intelligent vehicles*, 1(1):33–55, 2016.

[52] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[53] Dario Pavllo, David Grangier, and Michael Auli. Quaternet: A quaternion-based recurrent model for human motion. *BMVC*, 2018.

[54] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *CVPR*, pages 7753–7762, 2019.

[55] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, pages 1530–1538, 2015.

[56] Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *The annals of mathematical statistics*, pages 832–837, 1956.

[57] Tim Salzmann, Marco Pavone, and Markus Ryll. Motron: Multimodal probabilistic human motion forecasting. In *CVPR*, pages 6457–6466, 2022.

[58] Audun Rønning Sanderud, Mihoko Niitsuma, and Trygve Thomessen. A likelihood analysis for a risk analysis for safe human robot collaboration. In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–6. IEEE, 2015.

[59] Leonid Sigal, Alexandru O Balan, and Michael J Black. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International journal of computer vision*, 87(1):4–27, 2010.

[60] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[61] Jiarui Sun and Girish Chowdhary. Comusion: Towards consistent stochastic human motion prediction via motion diffusion. In *ECCV*, pages 18–36, 2024.

[62] Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

[63] Esteban G Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

[64] Peter Tisnikar, Gerard Canal, and Matteo Leonetti. Probabilistic inference of human capabilities from passive observations. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8779–8785. IEEE, 2024.

[65] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The pose knows: Video forecasting by generating pose futures. In *ICCV*, pages 3332–3341, 2017.

[66] Dong Wei, Huaijiang Sun, Bin Li, Jianfeng Lu, Weiqing Li, Xiaoning Sun, and Shengxiang Hu. Human joint kinematics diffusion-refinement for stochastic motion prediction. In *AAAI*, pages 6110–6118, 2023.

[67] Christina Winkler, Daniel E. Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *CoRR*, abs/1912.00042, 2019.

[68] Guowei Xu, Jiale Tao, Wen Li, and Lixin Duan. Learning semantic latent directions for accurate and controllable human motion prediction. In *ECCV*, pages 56–73, 2024.

[69] Sirui Xu, Yu-Xiong Wang, and Liang-Yan Gui. Diverse human motion prediction guided by multi-level spatial-temporal anchors. In *ECCV*, 2022.

[70] Xinchen Yan, Akash Rastogi, Ruben Villegas, Kalyan Sunkavalli, Eli Shechtman, Sunil Hadap, Ersin Yumer, and Honglak Lee. Mt-vae: Learning motion transformations to generate multimodal human dynamics. In *ECCV*, pages 265–281, 2018.

[71] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *CVPR*, pages 5255–5264, 2018.

[72] Ye Yuan and Kris Kitani. Diverse trajectory forecasting with determinantal point processes. *arXiv preprint arXiv:1907.04967*, 2019.

[73] Ye Yuan and Kris Kitani. Dlow: Diversifying latent flows for diverse human motion prediction. In *ECCV*, pages 346–364, 2020.

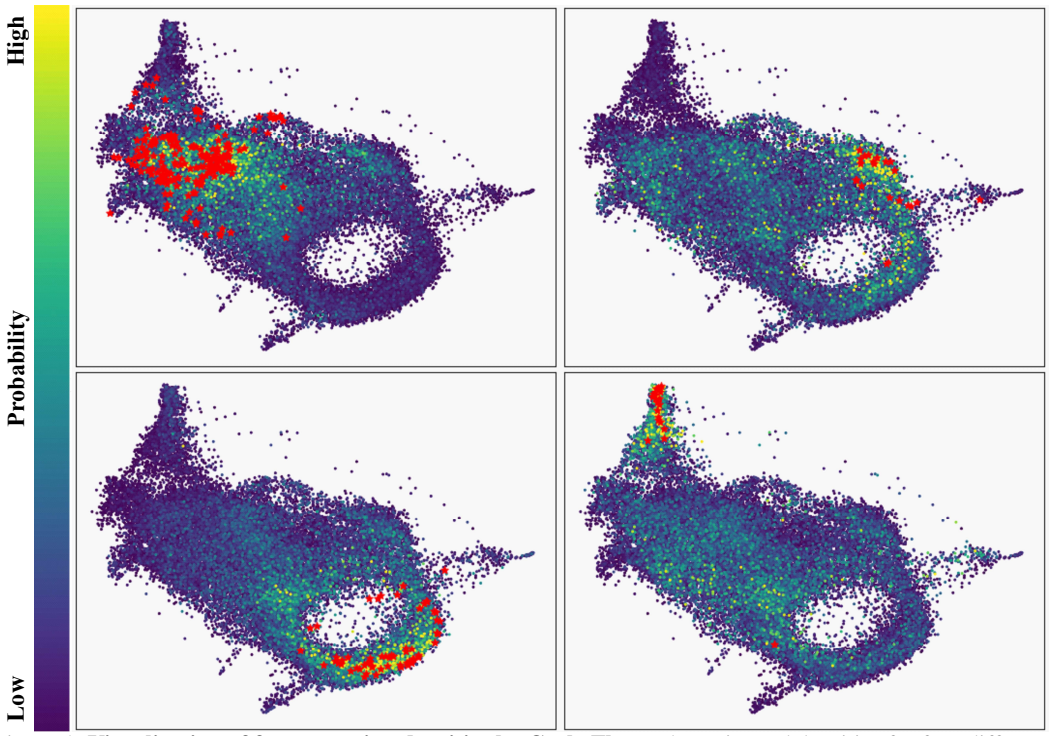

Figure 4: **Visualization of future motion densities by CacheFlow.** The estimated densities for four different motion sequences are visualized. We used UMAP to project these future motions onto a 2D space. Each dot represents an evaluated future motion, and the color of each dot indicates its probability, as shown in the side color bar. The red stars represent the projected ground truth future motions.

	Linear Factorization	Unconditional Flow-based Model	Joint Learning	Precomp. Set	Sampling	ADE↓	FDE↓	MM log prob. per dims↑	Inference time[ms]↓
(1)		✓	✓	Train Set	NN sample	0.502	0.664	0.458	4.8
(2)	✓		✓	Train Set	NN sample	0.616	0.889	0.901	0.4
(3)	✓	✓		Train Set	NN sample	0.370	0.475	1.283	1.3
(4)	✓	✓	✓	Base Density	NN sample	0.376	0.492	-	1.3
(5)	✓	✓	✓	Train Set	Random sample	0.455	0.605	-	**1.2**
	✓	✓	✓	Train Set	Most likely	0.384	0.506	-	1.4
	✓	✓	✓	Train Set	NN sample	**0.369**	**0.473**	**1.304**	1.3

Table 3: **Ablation Study on Human3.6M.** (4) and (5) do not affect the ground truth log probability, these are left blank.

A Implementation Details of Kernel Density Estimation

We assessed the accuracy of density estimation using Kernel Density Estimation (KDE) on previous methods. To ensure a fair comparison of inference time, all KDE computations were conducted on the GPU. We applied KDE to the standardized predicted future motions (or latents for BeLFusion) to obtain the estimated density. In this process, the i-th dimension of the predicted future motions was standardized using its i-th variance, meaning that covariances were not considered during standardization. We employed Scott's rule to determine the optimal bandwidth for KDE.

B Ablation Study

We conducted an ablation study to investigate how each component affects the performance of our CacheFlow. We ablate five components: (1) dimensionality reduction via linear factorization on

VAE, (2) the unconditional flow-based model f_θ, (3) joint learning of the conditional base density q_ϕ and unconditional flow-based model f_θ, (4) dataset for precomputation, (5) the sampling method for metrics over a fixed number of predictions. Ablation results on the Human3.6M dataset are summarized in Table 3.

Linear Factorization. We first ablate the linear factorization compressing 256-dim VAE latent to be an 8-dim factor space. Our method is considerably enhanced on the compact space by avoiding the curse of dimensionality.

The Unconditional Flow-based Model. We ablate this flow-based model f_θ to confirm it improves the conditional base density q_ϕ by adding complexity. As shown in Table 3, we observe a notable performance drop without the flow-based model. Therefore, our unconditional flow-based model f_θ complements conditional base density q_ϕ to estimate complex density distribution over human motions.

Joint Learning. We ablate the joint learning of the unconditional flow-based model f_θ and the conditional base density q_ϕ. The joint learning certainly improves both prediction errors and density estimation accuracy. The unconditional flow-based model f_θ can learn a more clustered $\boldsymbol{z}$ mapped from the motion feature $\boldsymbol{x}$. Thus, a conditional base density q_ϕ can easily model the $\boldsymbol{z}$ distribution.

Dataset for Precomputation. We propose the precomputation over the training split. Specifically, we apply inverse transform $\boldsymbol{z} = f_\theta(\boldsymbol{x})$ to ground truth future motions in the training split. However, we may precompute infinite precomputation samples. For example, we can sample $\boldsymbol{z} \sim q_\phi(\boldsymbol{z}|\boldsymbol{c})$ and obtain $\boldsymbol{x}$ by forward transform $\boldsymbol{x} = f_\theta(\boldsymbol{z})$. As shown in the ablation, precomputation on the training split outperforms one on the base density since we can regularize the prediction to be legitimate human motions using the training split.

Sampling Method. We propose the nearest neighbor sampling from the precomputation set as described in Section 4.3. Lastly, we ablate this sampling to evaluate its performance gain. We experimented with two sampling method alternatives: random sampling and most likely sampling. Precomputed motion features $\boldsymbol{x}_{k^*}$ are uniformly selected as predictions with random sampling. Most likely sampling selects motion features $\boldsymbol{x}_{k^*}$ with the highest probabilities $k^* = \mathrm{argmax}_k p(\boldsymbol{x}_k|\boldsymbol{c})$. We found that the large and little performance drops with random and most likely sampling respectively. This random sampling is worse due to the independence from the past motions $\boldsymbol{c}$. The most likely method underperforms due to less diverse samples. It cannot select a motion feature set with diversity because all selected features are often located in one peak of the estimated density. Since ADE and FDE are best-of-many metrics, this less diversity leads to worse performance. In contrast, our sampling method is superior to others. Our sampling incorporates past motions and achieves good diversity by simulating sampling from the estimated density $p(\boldsymbol{x}|\boldsymbol{c})$.

C Visualization of Estimated Density

We visualized the future motion density estimated by CacheFlow. Since future motions are high-dimensional data, we used UMAP [49] to project each future motion into a 2D space. We displayed the multimodal ground truth future motions alongside the visualized density map. As shown in Figure 4, CacheFlow estimated a high probability around the ground truth in all motion sequences. This visually supports the high density estimation accuracy presented in Table 2.

D Potential Broader Impact

The proposed CacheFlow introduces a fast probaility-aware motion prediction framework, which may involve the following broader impacts:

- **Improved Collaboration in Robotics and Automation.** In collaborative robotics and industrial automation, understanding and anticipating human motion is critical for ensuring safety and efficiency. The proposed system enables robots to predict human actions and movements with probabilistic confidence, allowing them to adjust their trajectories and tasks in real time. This leads to smoother coordination in shared workspaces such as manufacturing floors, warehouses, or hospitals, where humans and robots must work in close proximity.

- **Proactive Support in Assistive Technologies.** In assistive technologies for the elderly and individuals with disabilities, anticipating human motion is essential for delivering timely and meaningful support. A fast and uncertainty-aware human motion prediction system enables robots and smart devices to proactively assist users by foreseeing movements such as standing, walking, or reaching, even in the presence of noisy or partial sensor data. Furthermore, such a system could help prevent falls or injuries by detecting signs of instability and initiating interventions early.
- **Immersive Interactions in VR and Gaming.** Virtual reality (VR) and gaming systems stand to benefit from predictive models that can estimate future body movements in real time with associated uncertainties. This capability allows VR applications to reduce latency and create more responsive environments by anticipating user actions and gestures.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract reflects our main contribution, a novel 3D human motion prediction method named CacheFlow for fast inference and density estimation.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We included the limitation of our method in the Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: Our paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The code to reproduce the main experimental results will be publicly available.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code and instructions will be publicly available on GitHub.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify all the training details on the Section 5. We followed the evaluation protocol proposed by the previous method [4].

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: The Section 5 tells the training details, and our source code will be publicly available.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The Section 5 includes the computer resource used for the experiments (NVIDIA A100 GPU).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and ensured that our research conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The proposed CacheFlow may have broader impacts on robotics, assistive technologies, VR, gaming, etc., as detailed in Appendix D.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: There is no high risk for misuse in the models.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The licenses and terms of use are explicitly mentioned and properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: There are no new assets introduced in the paper.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: There is no crowdsourcing experiments and research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: There is no research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We used LLMs only for writing, editing, and formatting purposes.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.