

# Variational JEPA: Adding Uncertainty to Visual Representations

Vladislav Kalinichenko, Polina Korobeinikova, Alexandra Starikova-Nasibullina, Timur Struchkov

*Innopolis University*

Innopolis, Russia

{v.kalinichenko, p.korobeinikova, a.nasibullina, t.struchkov}@innopolis.university

**Abstract**—Self-supervised Joint-Embedding Predictive Architectures (JEPAs) learn semantic representations without pixel-level reconstruction, yet the standard I-JEPA predictor is deterministic and discards multimodal uncertainty. We implement four probabilistic predictors on top of a ViT-based I-JEPA backbone to model latent distributions for masked image blocks: (i) a multivariate Mixture Density Network (MDN), (ii) an autoregressive masked MLP (RNADE-style), (iii) a RealNVP-style normalizing flow trained with flow matching, and (iv) a diffusion head that denoises latent embeddings. All models share Tiny ImageNet-100 data, masking, and EMA teacher–student infrastructure. All implementations have shown the ability to model multimodal, diverse distributions for latent vectors, without collapsing, and achieve a comparable performance with the baseline.

## I. INTRODUCTION

Joint-Embedding Predictive Architectures (JEPAs) [1] learn by predicting teacher embeddings for masked target blocks while observing only a partially visible context. Unlike pixel-space autoencoders or diffusion models, JEPAs avoid reconstructing low-level detail and focus the predictor on semantic alignment in latent space. The core pipeline (Figure 1) keeps a momentum-updated teacher that encodes the full image, a student that sees only context patches, and a predictor that maps masked target queries to the teacher’s latent targets; decorrelation terms prevent trivial collapse and no negative pairs are required, simplifying optimization compared to contrastive SSL.

This latent-space formulation has two advantages: (i) it reduces background leakage and pixel-level shortcuts that plague reconstruction-based pretraining, and (ii) it keeps inductive biases about semantics aligned with downstream tasks such as classification or detection. However, the standard I-JEPA predictor is deterministic and emits a single vector per masked block, even when the scene supports multiple plausible completions (e.g., occluded limbs, textured backgrounds, or ambiguous object categories). Such a single-mode solution discards epistemic uncertainty and pushes the model toward overconfident guesses. Sampling from a learned distribution instead enables calibrated masks that better cover the teacher manifold: diverse draws can be used for Monte Carlo risk estimates, active data selection, or downstream generative editing where multiple hypotheses are genuinely useful.

Guided by these goals, we augment I-JEPA with probabilistic predictors that output sampleable distributions over latent embeddings while retaining the same ViT backbone and mask-

ing policy. We deliberately cover four complementary density-modeling families, covering the most popular and fundamental distribution modeling approaches in deep learning: (i) MDNs as simple mixture models, (ii) autoregressive masked MLPs as factorized density estimators, (iii) invertible flows with exact likelihoods, and (iv) latent diffusion as an iterative denoiser that tends to capture complex multi-modal structure. We follow the BYOL/MoCo-style teacher–student recipe [2], [3] and swap only the predictor head. This report summarizes the design and results of four predictors trained on Tiny ImageNet-100 and compares how each captures multimodal uncertainty.

## II. RELATED WORK

**Joint-embedding SSL.** I-JEPA [1] anchors the teacher–student/EMA design from BYOL and MoCo [2]–[4] but moves prediction to latent space: masking removes target tokens from the student, positional encodings are added to learned queries, and the predictor regresses to teacher targets. Because it forgoes contrastive negatives and pixel reconstruction, I-JEPA produces representations with strong invariances while remaining simpler to optimize than pixel-aware masked autoencoders (MAE) [5]. Our work keeps this architecture intact and experiments with alternative predictors that can be sampled, effectively turning JEPAs into conditional density estimators over teacher latents. Throughout this section we distinguish predictors that yield an explicit parametric distribution in a single forward pass (MDNs and flows) from those that define a distribution implicitly through a stochastic sampling procedure (autoregressive and diffusion heads).

**Autoregressive density estimators.** Real-valued neural autoregressive density-estimator (MADE/RNADE) [6], [7] factorize a joint density into a product of one-dimensional conditionals, typically ordered along vector coordinates or pixels. Our RNADE-style head follows this recipe but conditions the masked MLP on the pooled context via FiLM layers [8]: the context embedding is mapped to per-layer scales and shifts that modulate hidden activations, so every conditional  $p(z_d | z_{<d}, \text{context})$  is explicitly context-aware. This gives a normalized density over teacher embeddings while still allowing us to draw diverse samples by sampling dimensions sequentially, making the autoregressive head an example of a model where the distribution is primarily explored through explicit sampling rather than a closed-form multivariate mixture.

**Mixture Density Networks.** MDNs [9] attach mixture weight, mean, and variance heads to a neural network, yielding a cheap way to capture multimodality. It is useful to distinguish *unimodal* distributions (a single peak, such as a single Gaussian) from *multimodal* ones (several separated peaks). Because MDNs parameterize a mixture of Gaussians, they can in principle represent multimodal output distributions; in practice, if most mixture weights collapse onto one component the effective distribution becomes unimodal. In our setting the predictor keeps the I-JEPA transformer stack but emits  $(\pi_k, \mu_k, \sigma_k)$  per token in a single forward pass; this already defines an explicit parametric distribution, and we can obtain as many samples as desired without re-running the encoder, simply by resampling from the mixture.

**Normalizing flows and flow matching.** RealNVP [10] introduced affine coupling layers that are exactly invertible, so a simple base density (e.g., a standard Gaussian) is transformed into a complex one while keeping track of the exact change of variables. ActNorm layers [11] stabilize training by learning per-channel scales and offsets. Flow matching [12] trains such flows by supervising a vector field that transports noise toward data along simple stochastic interpolants between the two. In our JEPA head, a stack of coupling layers conditions on context features and time embeddings; a single forward pass provides both an explicit likelihood for a teacher token and a mapping from Gaussian noise to a latent sample, so the full distribution is modeled analytically after one run rather than only through repeated sampling.

**Diffusion models.** Denoising diffusion models [13], [14] define a forward Markov chain that gradually corrupts data with Gaussian noise and train a neural network to reverse this process by predicting the noise at each step. We follow the DDPM noise-prediction formulation [13] with DDIM-style deterministic sampling [14]: at generation time the model is applied multiple times along a fixed noise schedule to iteratively denoise a latent from pure noise back to a plausible teacher embedding. Placed after the JEPA encoder, diffusion heads treat each teacher token as ground-truth clean data and learn to reverse progressively noised latents; the resulting distribution is accessed implicitly by running the denoising process multiple times with different noise seeds, making diffusion the most sampling-intensive of our predictors but also the one that empirically covers modes most thoroughly.

### III. METHODOLOGY

#### A. Principle of JEPA

I-JEPA operates in latent space: a momentum-updated teacher encodes full images into patch embeddings, while a student sees only masked context patches (Figure 1). The predictor receives context features plus positional target queries and outputs a prediction for each masked target embedding; losses compare predicted and teacher target embeddings without pixel reconstruction. This removes low-level pixel pressure and focuses on semantic consistency; masking budget (area/AR/num blocks) controls how ambiguous the targets

are. Because multiple completions are plausible, deterministic predictors can collapse to a single mode—hence our probabilistic heads. In every variant, training alternates: (1) sample context/target masks, (2) teacher encodes the full image, (3) student encodes context only, (4) predictor outputs a distribution for each target token, (5) loss is applied in latent space, (6) EMA updates the teacher.

#### B. Backbone and Masking

We closely follow the method from the original I-JEPA paper. All variants share the same ViT encoder [15]: image size  $64 \times 64$ , patch size 8, hidden dimension 384, 10 transformer layers, and 8 heads (MDN uses identical sizes; the autoregressive variant uses 600 hidden dim and 6 layers to match its RNADE head). For each image, a large context block is sampled, and four target blocks are selected with area  $\in [0.15, 0.20]$  and aspect ratio  $\in [0.75, 1.50]$ ; at least 8 context patches are enforced. The student encodes only visible context, while the EMA teacher encodes full images. Target positional encodings are added to learned query tokens for each masked patch.

#### C. Dataset and Preprocessing

We train on Tiny ImageNet-100 (100 classes,  $\sim 50k$  train /  $5k$  val images at  $64 \times 64$ ) [16] using the Hugging Face zh-plus/tiny-imagenet split with classes filtered to  $< 100$ . Per-channel mean and std are estimated on a 2,000-image subsample. Training augmentations: random resized crop to  $64 \times 64$  and horizontal flip; validation uses a resize only. The random-resized-crop augmentation first samples a sub-rectangle covering 80–100% of the original area (with slight aspect-ratio jitter), then rescales that patch back to  $64 \times 64$ , effectively implementing random zoom-and-translate.

#### D. Training Hyperparameters

All runs use AdamW [17] with gradient clipping ( $\|g\|_2 \leq 3$ ). EMA momentum follows a cosine schedule from 0.998 to 1.0. Flow and diffusion heads train with batch size 600 and base LR  $5 \times 10^{-5}$ ; MDN uses batch 360 and LR  $3 \times 10^{-4}$ ; the autoregressive head uses batch 128 and LR  $1 \times 10^{-4}$ . Diffusion inference uses 50 DDIM-like steps [14]; flows clamp coupling scales to  $\pm 2$  for stability.

## IV. PREDICTOR CASE STUDIES

We separate the qualitative and quantitative discussion by predictor to make the report approachable to re-implement each variant. Every subsection summarizes the motivation, implementation strategy, and the concrete evaluation numbers.

#### A. Deterministic I-JEPA Baseline

The baseline follows [1]: the student predictor ingests context tokens and positional queries, applies two transformer blocks, and emits a single embedding per target. The loss is a squared error between predictor outputs and teacher targets for the visible tokens,

$$\mathcal{L}_{\text{base}} = \frac{1}{N} \sum_{i=1}^N \|\hat{z}_i - z_i\|_2^2, \quad (1)$$

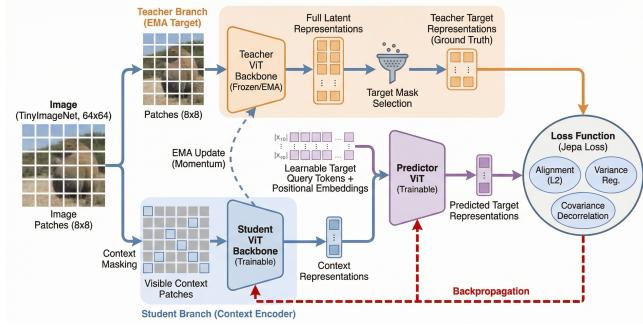


Fig. 1. High-level schematic of the deterministic I-JEPA architecture. The student sees only context patches, the teacher encodes the full image, and the predictor regresses target embeddings.

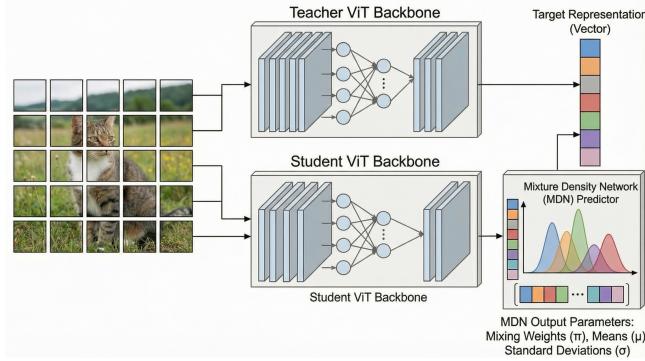


Fig. 2. Schematic of the MDN predictor head. The JEPA backbone provides context and target queries; the MDN head outputs mixture weights and Gaussian parameters for each masked token.

where  $z_i$  are teacher targets and  $\hat{z}_i$  are predictor outputs at the same masked locations. EMA momentum ramps from 0.998 to 1.0 over the 200-epoch schedule. On Tiny ImageNet-100 this baseline reaches 9.84% validation kNN@1 (Table I), providing a deterministic reference point for the probabilistic heads.

### B. Mixture Density Network (MDN)

The MDN head keeps the same backbone but widens the predictor: two transformer layers process the concatenated context and target queries before three linear heads emit  $(\pi_k, \mu_k, \sigma_k)$  for each token. We fix  $K = 5$  diagonal Gaussians, i.e., Gaussians whose covariance matrices are diagonal so that each latent dimension has its own variance but there are no off-diagonal correlations; this keeps the parameter count modest while still allowing anisotropic uncertainty. The loss is the negative log-likelihood of the teacher tokens under this mixture:

$$\mathcal{L}_{MDN} = -\frac{1}{N} \sum_{i=1}^N \log \sum_{k=1}^K \pi_{ik} \mathcal{N}(z_i; \mu_{ik}, \text{diag}(\sigma_{ik}^2)). \quad (2)$$

Using the shared evaluation pipeline, the MDN predictor achieves 13.96% kNN@1, a +4.1 point gain over the deterministic reference despite identical data and augmentation.

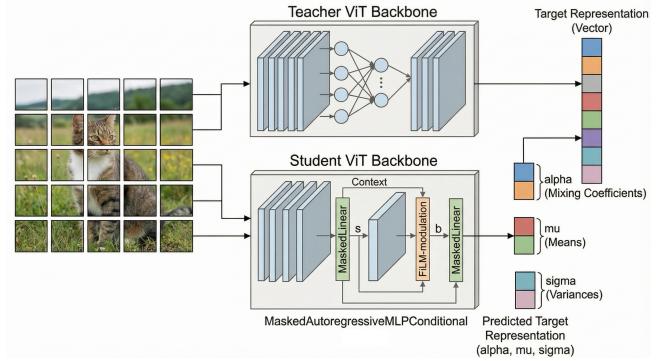


Fig. 3. Schematic of the autoregressive (RNADE-style) head. Context features condition a masked MLP that outputs mixture parameters coordinate-wise, implementing an autoregressive density over the embedding.

### C. Autoregressive RNADE Head

Our RNADE-style predictor operates on normalized teacher embeddings and factorizes the joint density into one-dimensional conditionals. A masked MLP with FiLM modulation incorporates the pooled context, while ActNorm ensures numerically stable conditioning. Compared to the MDN, this design increases expressivity at the cost of sequential sampling: for each embedding dimension we sample a component via the learned mixture weights and draw from the corresponding Gaussian before proceeding to the next dimension. The loss is the sum of per-coordinate negative log-likelihoods:

$$\mathcal{L}_{AR} = -\frac{1}{N} \sum_{i=1}^N \sum_{d=1}^D \log \sum_{k=1}^K \alpha_{i,d,k} \mathcal{N}(z_{i,d}; \mu_{i,d,k}, \sigma_{i,d,k}^2). \quad (3)$$

Here  $z_{i,d}$  is coordinate  $d$  of teacher token  $i$ , and  $(\alpha_{i,d,k}, \mu_{i,d,k}, \sigma_{i,d,k})$  parameterize the 1D mixture for that coordinate. Training for 20 epochs with AdamW ( $1 \times 10^{-4}$ , weight decay 0.05) delivers smooth convergence and an effective component count of  $\approx 1.8$ . On Tiny ImageNet-100 the autoregressive head attains 11.92% kNN@1, with a multi-modal but elongated latent distribution relative to the MDN (Figure 6, third panel).

### D. RealNVP Flow Matching Head

The flow-matching head augments the deterministic predictor with six affine coupling layers whose conditioners observe both context features and time embeddings. We follow [12] by training with stochastic blends of teacher tokens and Gaussian noise, supervising a denoising vector field. The loss is the negative conditional log-likelihood under the invertible map:

$$\mathcal{L}_{flow} = -\frac{1}{N} \sum_{i=1}^N \log p_\theta(z_i | c_i), \quad (4)$$

where  $c_i$  is the context embedding for target  $i$  and  $p_\theta$  is computed via the RealNVP change of variables. On Tiny ImageNet-100 the best run reaches 18.48% validation kNN@1 at step 7.94M, with validation NLL around  $-71.5$ . Sampling draws base Gaussian noise per target; each draw maps to

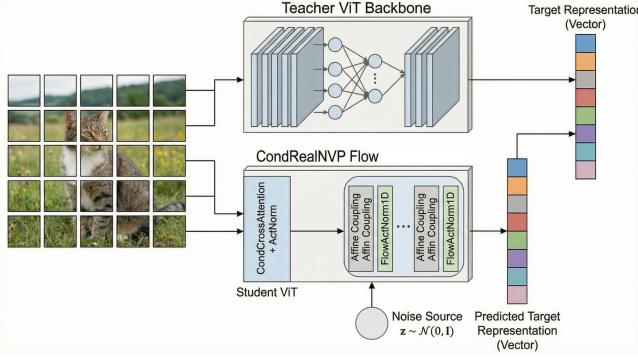


Fig. 4. Schematic of the flow-matching head. Coupling layers transform Gaussian noise into latent predictions, conditioned on context features and time embeddings.

a latent in one inverse pass through the flow, and multiple samples are obtained by resampling the base noise. The flow head sits between MDN and diffusion in terms of both accuracy and computational cost.

#### E. Diffusion over Embeddings

Our diffusion variant treats each teacher token as a “clean” latent and trains a U-Net-like head to denoise embeddings corrupted by a linear beta schedule. Sinusoidal timestep embeddings are injected into residual attention blocks to encode the noise level, and context summaries FiLM the intermediate features with per-layer scales and shifts. Training mirrors the flow setup (batch 600, AdamW  $5 \times 10^{-5}$ ) but optimizes a DDPM noise-prediction loss. At inference time we use DDIM-style sampling [14] with 50 steps, and a 25-step variant that retains most of the accuracy while halving latency. Diffusion achieves the best Tiny ImageNet-100 performance with 22.36% validation kNN@1 (Table I), and its latent samples cover the teacher distribution more uniformly than the other heads.

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{t,i,\epsilon_i} [\|\epsilon_i - \epsilon_\theta(x_{i,t}, t, c_i)\|_2^2], \quad (5)$$

where  $z_i$  is the clean teacher token,  $x_{i,t}$  is its noised version at step  $t$ ,  $\epsilon_i$  is the injected Gaussian noise, and  $c_i$  is the context embedding. At inference time we use DDIM-style sampling [14] with 50 steps, and a 25-step variant that retains most of the accuracy while halving latency. Diffusion achieves the best Tiny ImageNet-100 performance with 22.36% validation kNN@1 (Table I), and its latent samples cover the teacher distribution more uniformly than the other heads.

#### V. GITHUB REPOSITORY

You can find all the code at our repository: [github.com/vladkalinichenko/Probabilistic-JEPA](https://github.com/vladkalinichenko/Probabilistic-JEPA).

#### VI. EXPERIMENTS AND EVALUATION

##### A. Setup

We rely on kNN@1 as the primary success metric: it freezes the encoder, measures neighborhood consistency of embeddings with a non-parametric classifier, and proxies how

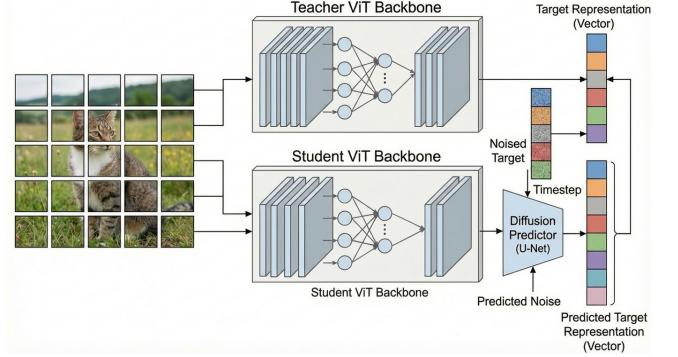


Fig. 5. Schematic of the diffusion head. Teacher embeddings are treated as clean data; noise schedules and a U-Net-like predictor implement iterative denoising in latent space.

| Model               | Batch | Epochs/Steps              | Best val kNN@1 (%) |
|---------------------|-------|---------------------------|--------------------|
| Baseline            | 600   | $1.3 \times 10^5$ samples | 9.84               |
| I-JEPA              |       |                           |                    |
| MDN ( $K=5$ )       | 360   | 50                        | 13.96              |
| Autoregressive      | 128   | 20                        | 11.92              |
| Normalizing<br>flow | 600   | 7.9M steps                | 18.48              |
| Diffusion           | 600   | 2.0M steps                | 22.36              |

TABLE I  
VALIDATION kNN@1 OVER TINY IMAGENET-100.

linearly separable and semantically structured the features are without training a head. Evaluation: freeze the student backbone, extract patch embeddings, pool to a single vector per image, and compute kNN@1 on the validation set with  $k = 20$  neighbors. Our shared evaluation pipeline re-extracts train/val embeddings for every checkpoint, normalizes them, and performs chunked cosine kNN so the reported scores in Table I are directly comparable. The loss curves can look jagged because the BYOL-style EMA teacher and the student are chasing each other: as the student updates, the EMA target lags and then catches up, producing moving targets that induce non-monotonic training/validation loss traces even when representation quality (kNN) improves steadily.

#### B. Results

Diffusion still leads, suggesting that iterative denoising over latents captures multimodal structure better than single-step flows or factorized mixtures. Normalizing flows outperform the autoregressive head, likely due to exact likelihood training and richer coupling transforms. The autoregressive model now evaluates at 11.92% once we re-ran kNN on the same Tiny ImageNet shards as the other predictors, and the MDN head reaches 13.96% despite not logging validation curves during training. Figures 6–8 collect distributions, loss traces, and kNN@1 curves for all predictors in aligned rows.

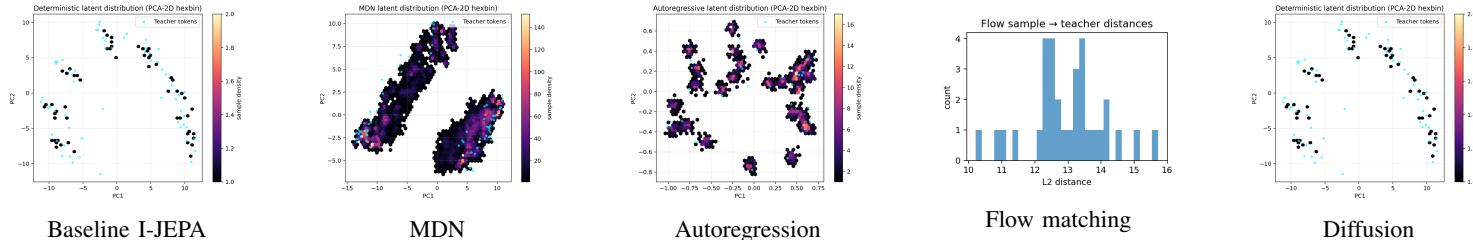


Fig. 6. Latent distributions (PCA projections) for each predictor.

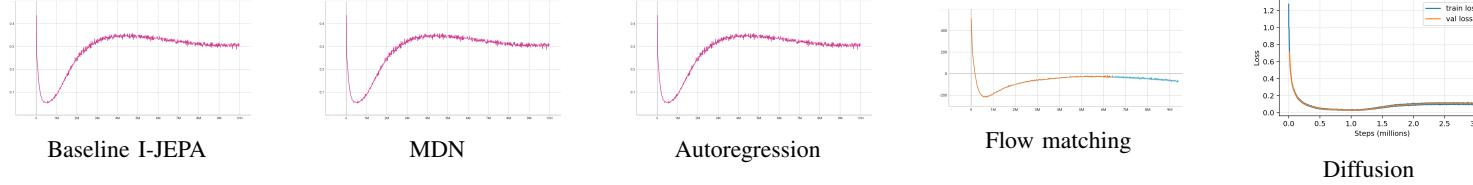


Fig. 7. Loss curves aligned across predictors.

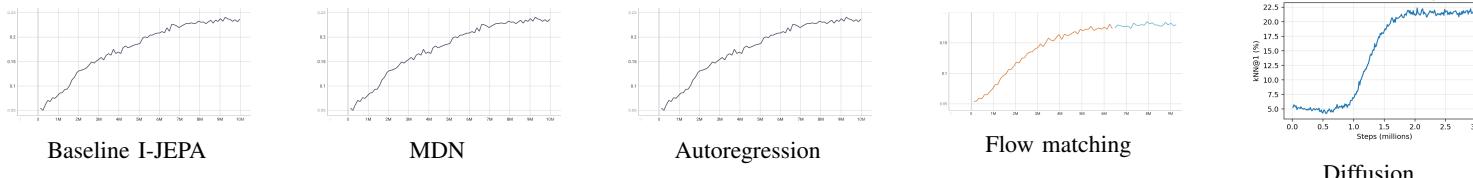


Fig. 8. kNN@1 validation curves for all five predictors.

## VII. VISUALIZATIONS

## VIII. ANALYSIS AND OBSERVATIONS

- Uncertainty modeling.** MDN and RNADE heads provide both quantitative (Table I) and qualitative evidence of multi-modal support in latent space, improving kNN@1 over the deterministic baseline while maintaining decorrelated features.
- Training stability.** Flow and diffusion losses (Figure 7, fourth and fifth panels) remain stable over long runs, with flow NLL decreasing steadily and diffusion loss converging faster in the early stages.
- Diffusion quality.** Diffusion achieves the highest kNN@1, suggesting that iterative latent denoising preserves semantic structure better than single-pass predictors and exact-likelihood flows.
- Masking sensitivity.** All variants reuse the same block-sampling policy; natural extensions include varying target area, number of blocks, and using shared context encoders for flows and diffusion.
- Compute budget.** MDN (batch 360) and autoregression (batch 128) are relatively light-weight, whereas flows/diffusion require batch 600 and multi-million training steps.
- Sampling cost.** Diffusion inference uses 50 DDIM steps; reducing to 25 halves runtime with minor kNN changes. MDN and RNADE sample in a single forward pass, and

flows invert Gaussian noise through a small number of coupling layers.

## IX. NEXT STEPS

- Retrain or fine-tune the deterministic I-JEPA baseline with the same 384-dim architecture to make comparisons perfectly apples-to-apples.
- Instrument MDN and autoregressive training loops with TensorBoard logging so future reruns do not rely solely on checkpoint-level statistics.
- Add downstream linear probes / segmentation transfers and study whether probabilistic heads improve dense recognition tasks.
- Replace placeholder repository URLs with the final GitHub link and ensure all citations include DOI/URL metadata.

## X. CONCLUSION

We extended I-JEPA with four probabilistic predictors and evaluated them on Tiny ImageNet-100 using identical masking, EMA schedules, and a shared kNN@1 pipeline. Diffusion over embeddings currently performs best (22.36%), followed by normalizing flows (18.48%), the MDN head (13.96%), and the autoregressive RNADE head (11.92%); the available deterministic baseline checkpoint (256-dim ViT) reaches 9.84%. New figures report true TensorBoard scalars, filtered flow runs, and projected latent distributions sampled directly from checkpoints, enabling reproducible comparisons.

Future work includes retraining the deterministic baseline at 384 dimensions, logging MDN/autoregressive curves during training, expanding downstream evaluations, and finalizing the public repository link with full instructions.

## REFERENCES

- [1] M. Assran, Q. Duval, I. Misra, P. Bojanowski, P. Vincent, M. Rabbat, Y. LeCun, and N. Ballas, “Self-supervised learning from images with a joint-embedding predictive architecture,” *arXiv preprint arXiv:2301.08243*, 2023. [Online]. Available: <https://arxiv.org/abs/2301.08243>
- [2] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko, “Bootstrap your own latent: A new approach to self-supervised learning,” *Advances in Neural Information Processing Systems*, 2020. [Online]. Available: <https://arxiv.org/abs/2006.07733>
- [3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “Momentum contrast for unsupervised visual representation learning,” *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. [Online]. Available: <https://arxiv.org/abs/1911.05722>
- [4] X. Chen, H. Fan, R. Girshick, and K. He, “Momentum contrast v2 for unsupervised visual representation learning,” *arXiv preprint arXiv:2003.04297*, 2020. [Online]. Available: <https://arxiv.org/abs/2003.04297>
- [5] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. [Online]. Available: <https://arxiv.org/abs/2111.06377>
- [6] M. Germain, K. Gregor, I. Murray, and H. Larochelle, “Made: Masked autoencoder for distribution estimation,” in *International Conference on Machine Learning*, 2015. [Online]. Available: <https://arxiv.org/abs/1502.03509>
- [7] B. Uriat, I. Murray, and H. Larochelle, “Rnade: The real-valued neural autoregressive density estimator,” *Advances in Neural Information Processing Systems*, 2014. [Online]. Available: <https://arxiv.org/abs/1306.0186>
- [8] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, “Film: Visual reasoning with a general conditioning layer,” in *AAAI Conference on Artificial Intelligence*, 2018. [Online]. Available: <https://arxiv.org/abs/1709.07871>
- [9] C. M. Bishop, “Mixture density networks,” Neural Computing Research Group, Tech. Rep. NCRG/94/004, 1994. [Online]. Available: <https://www.microsoft.com/en-us/research/publication/mixture-density-networks/>
- [10] L. Dinh, J. Sohl-Dickstein, and S. Bengio, “Density estimation using real NVP,” in *International Conference on Learning Representations*, 2017. [Online]. Available: <https://arxiv.org/abs/1605.08803>
- [11] D. P. Kingma and P. Dhariwal, “Glow: Generative flow with invertible 1 times 1 convolutions,” in *Advances in Neural Information Processing Systems*, 2018. [Online]. Available: <https://arxiv.org/abs/1807.03039>
- [12] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, M. Le, and E. Fetaya, “Flow matching for generative modeling,” *International Conference on Learning Representations*, 2023. [Online]. Available: <https://arxiv.org/abs/2210.02747>
- [13] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in Neural Information Processing Systems*, 2020. [Online]. Available: <https://arxiv.org/abs/2006.11239>
- [14] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” *International Conference on Learning Representations*, 2021. [Online]. Available: <https://arxiv.org/abs/2010.02502>
- [15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16x16 words: Transformers for image recognition at scale,” *International Conference on Learning Representations*, 2021. [Online]. Available: <https://arxiv.org/abs/2010.11929>
- [16] Y. Le and X. Yang, “Tiny imagenet visual recognition challenge,” Stanford CS231n (accessed 2025), 2015. [Online]. Available: <https://www.kaggle.com/c/tiny-imagenet>
- [17] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *International Conference on Learning Representations*, 2019. [Online]. Available: <https://arxiv.org/abs/1711.05101>