

Joint-Embedding Predictive Architecture

Zachary McCaw

2026-10-08

Summary

- **World models and JEPA.** A world model learns how the state of an environment evolves. JEPA learns this structure in representation space: a context encoder produces $z_x = E_\theta(x)$, a target encoder produces $z_y = E_{\bar{\theta}}(y)$, and a predictor estimates the target representation from the context and conditioning information c ,

$$\hat{z}_y = P_\phi(z_x, c), \quad \mathcal{L}_{\text{JEPA}} = D(\hat{z}_y, z_y).$$

Here c may specify a masked target location or the actions separating the context and target. In a latent world model, x is the observed trajectory through time t , $y = o_{t+k}$ is a future observation, and $c = a_{t:t+k-1}$ is the intervening action sequence, giving

$$z_i = E_\theta(o_i), \quad \hat{z}_{t+k} = P_\phi(z_{\leq t}, a_{t:t+k-1}), \quad \mathcal{L}_{\text{pred}} = D(\hat{z}_{t+k}, z_{t+k}).$$

The model predicts future latents rather than reconstructing future observations. Because jointly learned encoders admit a constant-representation solution, practical methods introduce architectural asymmetry—for example, stop-gradient or an EMA encoder—and/or explicitly regularize the latent distribution against collapse. These mechanisms are empirical safeguards rather than, in general, formal guarantees of non-collapse.

- **Anti-collapse penalties.** VICReg combines an invariance loss with variance and covariance penalties. For a batch $Z \in \mathbb{R}^{N \times D}$, representative forms are

$$\mathcal{L}_{\text{var}}(Z) = \frac{1}{D} \sum_{j=1}^D \max\{0, \gamma - \sigma_j(Z)\}, \quad \mathcal{L}_{\text{cov}}(Z) = \frac{1}{D} \sum_{j \neq k} [C(Z)]_{jk}^2.$$

The first prevents individual coordinates from becoming constant, while the second discourages redundant coordinates. SIGReg instead encourages the full embedding distribution to approach $N(0, I)$ by applying a normality statistic T to random one-dimensional projections,

$$\mathcal{L}_{\text{SIGReg}}(Z) = \frac{1}{|W|} \sum_{w \in W} T(\{w^\top z_n\}_{n=1}^N),$$

typically using the Epps–Pulley characteristic-function statistic. VISReg separates scale from distributional shape: it penalizes $[1 - \sigma_j(Z)]^2$, standardizes each centered coordinate using a detached standard deviation, and matches sorted random projections to standard-normal quantiles,

$$\tilde{Z}_j = \frac{Z_j - \mu_j}{\text{sg}(\sigma_j) + \epsilon}, \quad \mathcal{L}_{\text{shape}} = \frac{1}{K} \sum_{k=1}^K \left\| \text{sort}(\tilde{Z} w_k) - q_N \right\|_2^2.$$

Thus VICReg primarily controls marginal variance and second-order redundancy, whereas SIGReg and VISReg target the projected distribution more completely.

- **Latent prediction losses.** The basic one-step objective is

$$\mathcal{L}_{\text{1step}} = D(\hat{z}_{t+1}, z_{t+1}),$$

with D usually an L_1 or squared L_2 distance. Multi-step teacher forcing predicts each future latent from encoded ground-truth history. Autoregressive rollout instead feeds predictions back into the model,

$$\hat{z}_{t+k} = F_{\theta}(z_{t+k-H:t+k-1}^{\text{roll}}, a_{t+k-H:t+k-1}), \quad \mathcal{L}_{\text{roll}} = \sum_{k=1}^K w_k D(\hat{z}_{t+k}, z_{t+k}),$$

where $z_s^{\text{roll}} = z_s$ for observed states and $z_s^{\text{roll}} = \hat{z}_s$ after rollout begins. The weights w_k may be uniform or geometrically discounted. A direct prefix loss avoids recursive prediction: given an anchor z_t and an encoding $p_{t,k}$ of the first k actions, it predicts $\hat{z}_{t+k} = G(z_t, p_{t,k})$ and supervises every horizon with $H^{-1} \sum_{k=1}^H D(\hat{z}_{t+k}, z_{t+k})$.

- **Temporal straightening.** Prediction accuracy alone need not produce a simple latent geometry. Temporal straightening additionally penalizes changes in the direction of consecutive latent displacements $v_t = z_{t+1} - z_t$,

$$\mathcal{L}_{\text{curv}} = 1 - \frac{v_t^\top v_{t+1}}{\|v_t\|_2 \|v_{t+1}\|_2}, \quad \mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda \mathcal{L}_{\text{curv}}.$$

This encourages temporally adjacent transitions to follow approximately straight paths; a separate penalty or architectural mechanism is still needed to prevent collapse.

VICReg

VICReg [1] learns representations by generating two independently augmented views x and x' of the same input, encoding them as $y = f_{\theta}(x)$ and $y' = f_{\theta}(x')$, and then passing these representations through a disposable projection head h_{ϕ} to obtain $z = h_{\phi}(y)$ and $z' = h_{\phi}(y')$. In VICReg this head is specifically an *expander*: rather than projecting to a lower-dimensional space, it maps the encoder representation to a substantially higher-dimensional embedding space (e.g., from 2048 to 8192 dimensions in the standard ImageNet setup). The VICReg loss is applied to the expanded embeddings rather than

directly to the representations and combines invariance, variance, and covariance terms. For batches $Z = [z_1, \dots, z_n]$ and $Z' = [z'_1, \dots, z'_n]$ of d -dimensional embeddings, these are

$$\begin{aligned} s(Z, Z') &= \frac{1}{n} \sum_{i=1}^n \|z_i - z'_i\|_2^2, \\ v(Z) &= \frac{1}{d} \sum_{j=1}^d \max \left\{ 0, \gamma - \sqrt{\text{Var}(z^j) + \epsilon} \right\}, \\ C(Z) &= \frac{1}{n-1} \sum_{i=1}^n (z_i - \bar{z})(z_i - \bar{z})^\top, \quad c(Z) = \frac{1}{d} \sum_{j \neq k} [C(Z)]_{jk}^2, \\ \mathcal{L}_{\text{VICReg}}(Z, Z') &= \lambda s(Z, Z') + \mu \{v(Z) + v(Z')\} + \nu \{c(Z) + c(Z')\}. \end{aligned}$$

The invariance term makes the embeddings of two views of the same input similar; the variance term requires each embedding coordinate to maintain sufficient variation across samples, thereby preventing collapse; and the covariance term discourages redundancy by driving pairwise covariances between embedding coordinates toward zero. The paper uses $\gamma = 1$ and, for ImageNet, $\lambda = \mu = 25$ and $\nu = 1$.

After pretraining, the expansion head h_ϕ is discarded and the encoder representation y is retained for downstream use. Consequently, the variance and covariance constraints are not imposed directly on the representation of ultimate interest. This separation is intentional: the expansion head provides an auxiliary space in which strong invariance and redundancy-reduction constraints can be enforced without requiring y itself to satisfy those constraints exactly. Nevertheless, gradients from the VICReg objective propagate through h_ϕ into the encoder, and the paper empirically finds that covariance regularization applied at the expanded embedding level reduces correlation among dimensions of the encoder representation as well. Thus, in practice, VICReg is naturally implemented by attaching a disposable, preferably relatively wide, nonlinear expansion head to the representation being learned, applying the VICReg objective to the expanded embeddings during training, and retaining the pre-expansion representation for downstream tasks.

VICRegL

VICRegL [2] extends VICReg by applying the same variance–invariance–covariance objective at both global and local levels of a convolutional encoder. Given two augmented views of the same image, the encoder produces spatial feature maps $Y, Y' \in \mathbb{R}^{C \times H \times W}$. For the global criterion, these feature maps are mean-pooled to vectors, passed through the standard VICReg expander, and regularized with the usual VICReg loss. For the local criterion, the unpooled feature maps are passed through a smaller local projector that preserves their spatial arrangement, after which two separate local VICReg losses are constructed using different matching rules:

1. **Location-based matching:** each local feature is mapped back to the approximate position in the original seed image from which it arose, using the known coordinates of the crop and whether the crop was horizontally flipped. It is matched to the feature in the other view whose corresponding position in the original image is

closest. Among the resulting matched pairs, only the top- γ pairs are retained, and VICReg is applied to these pairs to define a location-based local loss L_s .

2. **Feature-based matching:** each local embedding is matched to its nearest neighbor in the other view according to Euclidean distance in embedding space, allowing semantically similar regions to be matched even when they occur at different spatial locations. Again, only the top- γ matched pairs are retained, and VICReg is applied to them to define a separate feature-based local loss L_d .

Because nearest-neighbor matching is directional, both local losses are symmetrized by computing matches in both directions, i.e. from the first view to the second and from the second view to the first. In the standard two-view setup, the paper uses $\gamma = 20$; with multi-crop, it uses $\gamma_1 = 20$ for large views and $\gamma_2 = 4$ for small views. The final objective is a weighted combination of the global VICReg loss and the two symmetrized local losses, with α controlling the emphasis on global versus local representation learning; the main experiments use $\alpha = 0.9$ or $\alpha = 0.75$, with $\alpha = 0.75$ serving as a representative setting that places somewhat greater emphasis on local features.

LeCun Perspective

LeCun’s perspective [3] proposes a general architecture for autonomous intelligence centered on learned world models, latent-state prediction, and planning. For the purposes of world and dynamics modeling, the most important ideas are the following.

1. **World models should learn action-conditioned latent dynamics.** In the control setting, the relevant transition object is not merely $p(s_{t+1} | s_t)$, but rather $p(s_{t+1} | s_t, a_t)$. If actions are omitted, then the learned transition law is implicitly averaged over the behavior policy that generated the data:

$$p(s_{t+1} | s_t) = \int p(s_{t+1} | s_t, a_t) p(a_t | s_t) da_t.$$

For planning, LeCun instead envisions a predictor of the form

$$s_{t+1} = \text{Pred}(s_t, a_t, z_t),$$

where z_t represents structured residual uncertainty in the next state that is not determined by (s_t, a_t) .

2. **Prediction should occur in representation space rather than directly in observation space.** A JEPA encodes an observed/context variable x and a target/future variable y as $s_x = g_x(x)$ and $s_y = g_y(y)$, then predicts the target representation using $\tilde{s}_y = \text{Pred}(s_x, z)$. The corresponding energy is

$$E_w(x, y, z) = D\{s_y, \text{Pred}(s_x, z)\},$$

with compatibility defined by

$$F_w(x, y) = \min_{z \in \mathcal{Z}} E_w(x, y, z).$$

This allows the target encoder to discard unpredictable nuisance detail rather than forcing the world model to reproduce every low-level aspect of y .

3. **When the future is not uniquely determined, model compatibility among plausible outcomes rather than force a single point prediction.** The latent variable z parameterizes alternative plausible relationships between x and y , while invariances in the target encoder can eliminate variation that need not be represented at all. Conceptually:

- Predictable structure should be retained in s_y .
- Relevant but unpredictable variation should be represented through z .
- Irrelevant variation should be discarded by the encoder.

Thus z can be interpreted as a structured residual component that indexes ambiguity in $y \mid x$.

4. **Predictability creates an anti-collapse problem.** If the only objective were to make s_y easy to predict from s_x , the trivial solution would be to make both representations constant. JEPA therefore requires opposing constraints that keep s_x and s_y informative about their inputs. LeCun proposes four simultaneous criteria:

- Maximize the information content of s_x about x .
- Maximize the information content of s_y about y .
- Make s_y easy to predict from s_x .
- Minimize the information content of z .

VICReg is presented as one practical non-contrastive realization of the first two criteria: variance regularization prevents individual coordinates from becoming constant, while covariance regularization discourages redundancy among coordinates. The latent z must also have limited capacity; otherwise the predictor could ignore s_x and encode the full target through z .

5. **World models should operate hierarchically across both abstraction levels and time scales.** LeCun proposes H-JEPA, in which lower levels retain more detailed state information and support short-horizon prediction, while higher levels learn progressively more abstract representations suitable for longer-horizon prediction. For example, a hierarchy may take the form

$$s_t^{(1)} = \text{Enc}_1(x_t), \quad s_t^{(2)} = \text{Enc}_2\left(s_t^{(1)}\right).$$

Higher levels then predict over increasingly long horizons. The key idea is that long-horizon prediction requires discarding fast-varying detail while retaining slower, more stable structure.

6. **The purpose of the world model is planning.** Given a differentiable predictor and cost function, the actor can simulate candidate action sequences, evaluate the resulting predicted trajectories, and optimize actions using gradients of predicted future cost with respect to the proposed actions. A representative objective is

$$J(a_{0:T}) = \sum_{t=1}^T C(s_t).$$

At higher levels of the hierarchy, LeCun’s “abstract actions” are better interpreted as latent subgoals or desired future-state conditions that lower levels must realize. In this way, the architecture combines latent-state prediction, hierarchical abstraction, and model-predictive control into a common framework for reasoning and planning.

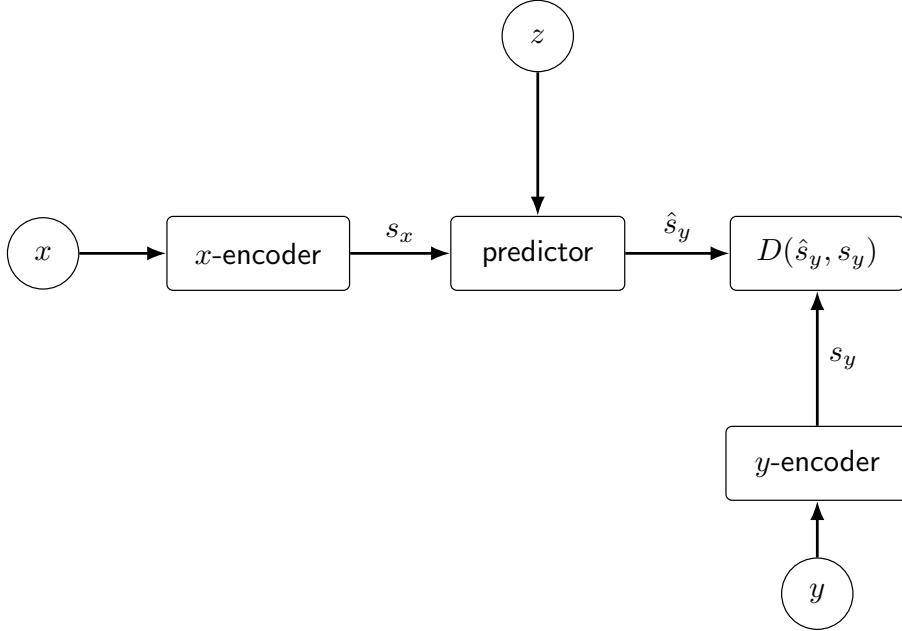


Figure 1: Schematic of a Joint-Embedding Predictive Architecture (JEPA). The context input x is encoded by the x -encoder to produce a representation s_x . The predictor uses s_x together with an auxiliary variable z , which specifies the transformation, corruption, location, or other conditioning information needed to identify the target, to produce a prediction $\hat{s}_y$. Separately, the target input y is encoded by the y -encoder to produce the target representation s_y . Training minimizes a discrepancy $D(\hat{s}_y, s_y)$ in representation space rather than reconstructing y directly in observation space.

I-JEPA

I-JEPA [4] instantiates the Joint-Embedding Predictive Architecture for images by predicting masked target representations from a visible context region of the same image. An image is divided into a common grid of non-overlapping patches. The full set of

image patches is passed simultaneously through the target encoder $f_{\bar{\theta}}$, which is a Vision Transformer (ViT), and therefore produces context-dependent patch representations through self-attention across the full patch sequence. Target blocks are then selected from the *output* of the target encoder. Independently, a large context block is sampled from the same patch grid; if any portion of the sampled context overlaps a target block, those overlapping patches are removed from the context mask before encoding, while the target block itself is left unchanged. The remaining visible context patches are passed simultaneously through the context encoder f_{θ} , also a ViT. For target block i , let B_i denote its patch indices. The predictor g_{ϕ} takes the full set of encoded context tokens together with one positional mask token for each patch in B_i . Because the predictor is a ViT, these tokens are processed jointly, and its outputs at the mask-token positions provide predictions $\hat{s}_{y_j}$ for all $j \in B_i$. The predictor is applied separately to each of the M target blocks, and each prediction is compared with the corresponding target-encoder representation s_{y_j} . The loss is computed only over the target patches, not over the visible context patches:

$$\mathcal{L} = \frac{1}{M} \sum_{i=1}^M \sum_{j \in B_i} \|\hat{s}_{y_j} - s_{y_j}\|_2^2.$$

The context encoder and predictor are updated by gradient descent, whereas the target encoder is updated as an exponential moving average of the context encoder, providing the architectural asymmetry used to avoid collapse. A key image-specific ingredient is the multi-block masking strategy: each target is a contiguous rectangular block covering roughly 15%–20% of the image, while the context block initially covers roughly 85%–100% of the image before any target-overlapping patches are removed; positional tokens specify the locations of the target patches. The use of 2D patch grids, rectangular masks, and image-scale heuristics is specific to vision. The broader ideas of predicting representations rather than raw observations, using separate context and target encoders, conditioning the predictor on target identity or location, and stabilizing training with a slowly moving target encoder generalize to other modalities. Importantly, I-JEPA is reconstructive in the sense that it predicts representations of masked regions from other regions of the *same static image*; it does not model temporal dynamics or forecast future world states.

V-JEPA

V-JEPA [5] extends the Joint-Embedding Predictive Architecture to video by predicting masked spatiotemporal representations rather than pixels. A video clip of shape $T \times H \times W$ is partitioned into non-overlapping tubelets of size $2 \times 16 \times 16$, so each token spans two consecutive frames and a 16×16 spatial region. For the standard $16 \times 224 \times 224$ input, this yields 1568 spatiotemporal tokens. The context encoder E_{θ} processes only the visible tokens, while the target encoder $E_{\bar{\theta}}$ processes the full clip and produces contextualized target representations. The predictor P_{ϕ} receives the context representations together with learnable mask tokens augmented by 3D positional information specifying the masked locations, and predicts the corresponding target representations. Training minimizes an L_1 loss over masked tokens,

$$\mathcal{L} = \frac{1}{|\mathcal{M}|} \sum_{k \in \mathcal{M}} \|\hat{s}_k - s_k\|_1.$$

V-JEPA replaces I-JEPA’s L_2 regression loss with L_1 . The authors report that L_1 was more stable in practice; a plausible interpretation is that its bounded per-residual gradient makes optimization less sensitive to large latent prediction errors, although the paper does not characterize the source of instability in detail. The target branch is stop-gradient and its parameters are updated as an exponential moving average of the context encoder to prevent representation collapse.

V-JEPA uses a 3D multi-block masking strategy in which spatially contiguous blocks are extended across the full temporal duration of the clip. Two masks are sampled per clip: a short-range mask formed from eight smaller blocks and a long-range mask formed from two larger blocks, with an effective masking ratio of approximately 90%. Extending the masks through the full temporal dimension reduces information leakage from neighboring frames. Ablations show that representation quality is improved by high spatial and temporal masking ratios, several smaller blocks rather than a single large block, and using both mask types. The standard objective is non-causal, so V-JEPA is best viewed as spatiotemporal latent completion rather than future-state prediction.

For downstream tasks, the pretrained encoder is frozen and its token sequence is pooled with an *attentive probe*. A learnable query vector q cross-attends to the frozen spatiotemporal representations $(s_1, \dots, s_L)$,

$$\sum_{i=1}^L \frac{\exp(q^\top W_k s_i)}{\sum_j \exp(q^\top W_k s_j)} W_v s_i.$$

The resulting pooled representation is passed through a small MLP, LayerNorm, and a linear classifier. The query is introduced only for downstream evaluation and is not tied to masked positions; instead, it learns which spatial and temporal tokens are most informative for the task.

DINO-WM

DINO-WM [6] learns a task-agnostic, action-conditioned world model from offline state-action trajectories without reward supervision or pixel reconstruction. Unlike Dreamer and TD-MPC2, which learn their representations jointly with the world model, DINO-WM uses a frozen, pretrained DINOv2 encoder to extract spatial patch embeddings $z_t = \text{enc}(o_t) \in \mathbb{R}^{N \times E}$. A transformer then predicts future patch embeddings conditioned on previous embeddings and actions:

$$\hat{z}_{t+1} = p_\theta(z_{t-H:t}, a_{t-H:t}), \quad \mathcal{L}_{\text{pred}} = \|\hat{z}_{t+1} - \text{enc}(o_{t+1})\|_2^2.$$

The transformer uses frame-level causal attention, allowing each predicted patch to attend to all patches in previous frames while predicting patches within the next frame in parallel. The model is trained with teacher forcing, and an optional decoder can be trained independently for visualization.

At inference time, DINO-WM performs zero-shot goal-conditioned planning. Given a current observation o_0 and goal observation o_g , it optimizes an action sequence using CEM and MPC to minimize the terminal latent distance $\|\hat{z}_T - \text{enc}(o_g)\|_2^2$. No task-specific

policy, reward function, or value function is required. Ablations show that maintaining distinct patch embeddings outperforms compressing the entire image into a single embedding, for instance using the DINOv2 [CLS] token.

DINO-WM is similar to the subsequent LeWM [7], which likewise learns reward-free, decoder-free dynamics and performs goal-conditioned planning in latent space. The principal distinction is representation learning: DINO-WM relies on a frozen pretrained encoder to prevent collapse, whereas LeWM jointly trains its encoder and dynamics predictor from pixels, using SIGReg to prevent representation collapse.

PLDM

Sobal *et al.* [8] propose Planning with a Latent Dynamics Model (PLDM), a reward-free approach that learns task-agnostic dynamics from offline state-action trajectories and performs goal-conditioned planning at test time. Unlike methods that learn a policy for reaching arbitrary goals, PLDM uses a learned world model to optimize action sequences directly.

PLDM consists of an encoder h_θ and an ensemble of action-conditioned latent dynamics predictors $f_\theta^{(k)}$. Given an offline trajectory, the encoder produces $z_t = h_\theta(s_t)$, and each predictor generates an autoregressive latent rollout:

$$\hat{z}_0^{(k)} = h_\theta(s_0), \quad \hat{z}_t^{(k)} = f_\theta^{(k)}(\hat{z}_{t-1}^{(k)}, a_{t-1}), \quad k = 1, \dots, K.$$

The model is trained by minimizing the discrepancy between predicted and observed latent trajectories, together with VICReg-inspired regularization:

$$\mathcal{L}_{\text{JEPA}} = \mathcal{L}_{\text{sim}} + \alpha \mathcal{L}_{\text{var}} + \beta \mathcal{L}_{\text{cov}} + \delta \mathcal{L}_{\text{time-sim}} + \omega \mathcal{L}_{\text{IDM}}.$$

Here $\mathcal{L}_{\text{sim}}$ penalizes latent prediction errors, the variance and covariance terms discourage representation collapse and redundancy, temporal similarity encourages smooth latent dynamics, and the inverse-dynamics loss preserves action-relevant information. At test time, MPC optimizes action sequences using a latent goal-distance objective penalized by ensemble disagreement, discouraging plans that enter regions where the learned dynamics are uncertain.

V-JEPA 2

V-JEPA 2 [9] retains the masked latent-prediction objective of V-JEPA, but replaces absolute 3D sine-cosine positional embeddings with 3D RoPE and introduces a progressive-resolution training schedule. The underlying pretraining objective remains non-causal masked prediction in latent space, so the pretrained model is primarily a spatiotemporal representation learner rather than an action-conditioned dynamics model.

V-JEPA 2-AC introduces action conditioning in a separate post-training stage. The pretrained encoder E is frozen, and a new predictor P_ϕ is trained to model latent dynamics.

Writing $z_t = E(x_t)$ for the encoded observation, s_t for auxiliary state information, and a_t for the action, the predictor models

$$\hat{z}_{t+1} = P_\phi [\{(z_k, s_k, a_k)\}_{k \leq t}].$$

The predictor uses block-causal attention and therefore conditions on the preceding history available within the training sequence; in the implementation, training uses 16-frame clips, so later predictions can condition on all earlier latent, state, and action tokens in the clip. Importantly, the masked-video predictor learned during V-JEPA 2 pretraining is not reused; only the encoder is transferred, while the action-conditioned predictor is trained anew.

Training combines teacher forcing with an autoregressive rollout loss. The teacher-forcing objective predicts each next latent representation from the ground-truth history,

$$\mathcal{L}_{\text{TF}}(\phi) = \frac{1}{T} \sum_{t=1}^T \|P_\phi [\{(z_k, s_k, a_k)\}_{k \leq t}] - z_{t+1}\|_1.$$

The rollout loss instead feeds predicted representations back into the model,

$$\mathcal{L}_{\text{rollout}}(\phi) = \|P_\phi(a_{1:H}; s_1, z_1) - z_{H+1}\|_1.$$

In V-JEPA 2-AC, the rollout horizon is set to $H = 2$, so gradients pass through one recurrent prediction step. The overall objective is $\mathcal{L} = \mathcal{L}_{\text{TF}} + \mathcal{L}_{\text{rollout}}$, combining accurate one-step prediction with explicit training against autoregressive error accumulation.

Planning

For planning, let $z_t = E(x_t)$ denote the current latent visual representation, s_t the current auxiliary state, $z_g = E(x_g)$ the latent representation of a goal observation, and $a_{1:H} = (a_1, \dots, a_H)$ a candidate sequence of H future actions, with each $a_h \in \mathbb{R}^d$. The action-conditioned predictor rolls the current state forward under $a_{1:H}$, and the trajectory is scored by the distance between its predicted terminal representation and the goal,

$$J(a_{1:H}) = \|P_\phi(a_{1:H}; s_t, z_t) - z_g\|_1.$$

The Cross-Entropy Method (CEM) approximately minimizes J by maintaining a factorized Gaussian proposal over the action coordinates,

$$q_r(a_{1:H}) = \prod_{h=1}^H \prod_{j=1}^d \mathcal{N}(a_{hj}; \mu_{hj}^{(r)}, \sigma_{hj}^{2,(r)}),$$

where r indexes the CEM iteration, h the planning step, and j the action coordinate. At iteration r , N candidate trajectories $a_{1:H}^{(1)}, \dots, a_{1:H}^{(N)}$ are sampled from q_r and evaluated using J . If $\mathcal{E}_r$ denotes the indices of the K lowest-cost trajectories, the proposal parameters are updated using their empirical moments,

$$\begin{aligned} \mu_{hj}^{(r+1)} &= \frac{1}{K} \sum_{n \in \mathcal{E}_r} a_{hj}^{(n)}, \\ \sigma_{hj}^{2,(r+1)} &= \frac{1}{K} \sum_{n \in \mathcal{E}_r} [a_{hj}^{(n)} - \mu_{hj}^{(r+1)}]^2. \end{aligned}$$

Repeated sampling and refitting concentrates the proposal around low-cost action sequences. Although a multi-step trajectory is optimized, planning is receding-horizon: only the first action is executed, the new state is observed, and the optimization is then repeated from that state. In the reported implementation, the CEM proposal is initialized with zero mean and unit variance for each planning problem rather than warm-started from the proposal obtained at the previous control step.

LeJEPA/SIGReg

LeJEPA [10] develops a JEPA objective from two principles: representations of related views should be predictable from one another, and the embedding distribution should remain non-degenerate. The paper provides theoretical results for linear probes and several nonlinear probe settings showing that isotropic Gaussian embeddings are optimal under the corresponding downstream-risk criteria. This motivates directly regularizing the latent distribution toward an isotropic Gaussian rather than relying on teacher–student networks, stop-gradient, negative samples, or whitening heuristics.

To enforce Gaussianity efficiently in a K -dimensional embedding space, LeJEPA introduces *Sketched Isotropic Gaussian Regularization* (SIGReg). Let $z_n = f_\theta(x_n)$ and let $w \in \mathbb{S}^{K-1}$ denote a unit projection direction. The Cramér–Wold principle states that two multivariate distributions are equal if and only if all of their one-dimensional linear projections are equal in distribution. In particular,

$$w^\top Z \sim N(0, 1) \text{ for every } w \in \mathbb{S}^{K-1} \iff Z \sim N(0, I).$$

SIGReg therefore samples a finite collection of projection directions W and measures how far the projected samples $\{w^\top z_n\}_{n=1}^N$ depart from $N(0, 1)$,

$$\text{SIGReg}[\{z_n\}_{n=1}^N] = \frac{1}{|W|} \sum_{w \in W} T[\{w^\top z_n\}_{n=1}^N],$$

where T is a univariate goodness-of-fit statistic. LeJEPA uses the Epps–Pulley statistic, which compares the empirical characteristic function $\hat{\phi}(t)$ of the projected samples with the standard Gaussian characteristic function $\phi_0(t) = \exp(-t^2/2)$,

$$T_{\text{EP}} = N \int_{-\infty}^{\infty} \left| \hat{\phi}(t) - \phi_0(t) \right|^2 \omega(t) dt.$$

Characteristic-function matching is preferred to finite-moment tests because it identifies the full distribution while retaining bounded gradients and curvature. It is preferred to CDF-based tests because the latter require sorting and order statistics, which introduce non-differentiable operations and synchronization that is poorly suited to efficient multi-GPU training. The empirical characteristic function, by contrast, is an average of complex exponentials and can be aggregated efficiently across devices. The integral is approximated numerically by quadrature, and a fresh random set of projection directions W is sampled for each minibatch, so directional coverage accumulates across SGD updates.

The predictive component is defined in terms of multiple related views of each sample. Let $z_{n,v}$ denote the embedding of sample n under view v , and let a subset of V_g views serve as reference views. Defining

$$\mu_n = \frac{1}{V_g} \sum_{v=1}^{V_g} z_{n,v},$$

the prediction loss directly aligns each of the V views with this reference representation,

$$\mathcal{L}_{\text{pred},n} = \frac{1}{V} \sum_{v=1}^V \|\mu_n - z_{n,v}\|_2^2.$$

The general requirement is that the views represent semantically related observations whose relevant content should be aligned. In the image experiments, LeJEPA instantiates this using the DINO-style multi-crop construction, where V_g global views are larger image crops and the remaining views may include smaller local crops. When the views are intended to represent the same underlying information, LeJEPA aligns their embeddings directly without a separate predictor. More generally, the paper argues that a predictor is warranted when the views differ systematically in information content and the transformation between their representations must itself be modeled.

The complete objective combines cross-view prediction with SIGReg applied separately to the embedding distribution of each view,

$$\mathcal{L}_{\text{LeJEPA}} = \frac{\lambda}{V} \sum_{v=1}^V \text{SIGReg} [\{z_{n,v}\}_{n=1}^B] + \frac{1-\lambda}{B} \sum_{n=1}^B \mathcal{L}_{\text{pred},n}.$$

Thus λ controls the trade-off between cross-view predictability and isotropic-Gaussian regularization. Because SIGReg is used to guard against collapse, neither stop-gradient nor an EMA teacher is required merely as an anti-collapse mechanism. Empirically, LeJEPA is stable across architectures, batch sizes, view configurations, and SIGReg hyperparameters, and its training loss is strongly correlated with downstream linear-probe performance.

C-JEPA

Causal-JEPA [11] is an object-centric JEPA world model that operates on frozen slot-based representations rather than image patches. Each frame is encoded as a fixed-size set of object latents, and a masked Transformer is trained to reconstruct selected object trajectories within the history window while also predicting future object states. The key idea is to mask entire object histories, retaining only an identity anchor, so that the predictor cannot rely primarily on an object’s own recent trajectory; instead, it is encouraged to infer the missing state from other objects and auxiliary variables such as actions and proprioception. At inference, history masking is removed and the model is used for ordinary forward latent prediction and CEM-based model predictive control. A central limitation is the dependence on a reliable object-centric decomposition: the method assumes a fixed slot budget whose slots correspond to coherent entities over time, so slot

permutation, identity switching, or scenes containing more relevant objects than available slots can undermine both the masking objective and its interaction-based interpretation.

Temporal straightening

Temporal Straightening for latent planning [12] learns a planning-oriented JEPA world model consisting of a sensory encoder, an action encoder, and a latent dynamics predictor. The sensory encoder maps observations to latent states, $z_t = E_\phi^s(o_t)$, while the action encoder maps controls to latent action embeddings, $E_\psi^a(a_t)$. Given a history of latent states and encoded actions, the predictor estimates the next latent state,

$$\hat{z}_{t+1} = f_\theta(\{z_i\}_{i=t-K+1}^t, \{E_\psi^a(a_i)\}_{i=t-K+1}^t).$$

To make the representation geometry more suitable for planning, the method explicitly straightens latent trajectories. For consecutive latent states, define the velocity vectors $v_t = z_{t+1} - z_t$ and $v_{t+1} = z_{t+2} - z_{t+1}$, and penalize changes in direction using

$$\mathcal{L}_{\text{curv}} = 1 - \frac{v_t^\top v_{t+1}}{\|v_t\|_2 \|v_{t+1}\|_2}.$$

This can be viewed as a scale-normalized discrete curvature penalty: the second difference $z_{t+2} - 2z_{t+1} + z_t = v_{t+1} - v_t$ measures changes in both speed and direction, whereas the cosine penalty isolates directional change. When consecutive velocities have equal norm c , $\|v_{t+1} - v_t\|_2^2 = 2c^2(1 - \cos\theta_t)$, so the curvature loss is directly proportional to the squared discrete second derivative up to scale. Training combines next-latent prediction with this regularizer,

$$\mathcal{L}_{\text{pred}} = \|\hat{z}_{t+1} - \text{sg}(z_{t+1})\|_2^2, \quad \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pred}} + \lambda \mathcal{L}_{\text{curv}},$$

with stop-gradient on the target branch. This asymmetry blocks gradients through the target representation, but it does not mathematically exclude a constant representation: the paper explicitly notes that collapse remains theoretically possible and reports that it was not observed in the main experiments.

For planning, the paper considers gradient-based optimization and the cross-entropy method (CEM). Given the current latent state z_0 , a goal embedding z_g , and a candidate action sequence $a_{0:T-1}$, write the learned history-dependent transition schematically as $\hat{z}_{t+1} = f_\theta(\hat{z}_t, a_t)$ and roll it out recursively. Planning then minimizes the terminal latent-space goal cost

$$\mathcal{L}_{\text{plan}}(a_{0:T-1}) = \|\hat{z}_T - z_g\|_2^2.$$

Gradient-based planning treats the actions themselves as optimization variables and updates them using

$$a_t \leftarrow a_t - \eta \frac{\partial \mathcal{L}_{\text{plan}}}{\partial a_t}.$$

This requires the learned dynamics model to be differentiable with respect to its action inputs, since the planner must compute quantities such as $\partial \hat{z}_{t+1} / \partial a_t$ and propagate their

effect through the entire rollout. For example, with two transitions, $\hat{z}_1 = f_\theta(z_0, a_0)$ and $\hat{z}_2 = f_\theta(\hat{z}_1, a_1)$, so the gradient of the terminal loss with respect to the first action is obtained by the chain rule,

$$\frac{\partial \mathcal{L}_{\text{plan}}}{\partial a_0} = \frac{\partial \mathcal{L}_{\text{plan}}}{\partial \hat{z}_2} \frac{\partial \hat{z}_2}{\partial \hat{z}_1} \frac{\partial \hat{z}_1}{\partial a_0}.$$

Thus, gradient-based planning is effectively backpropagation through the learned world-model rollout, with the action sequence rather than the network parameters being optimized. CEM instead treats the dynamics model as a black box and is therefore gradient-free, but requires evaluating many sampled action sequences and is substantially less computationally efficient. The main empirical finding is that temporal straightening makes latent Euclidean distance better reflect dynamical progress toward the goal and improves the conditioning of this action-optimization problem, substantially increasing the success of gradient-based planning and often narrowing the performance gap with CEM.

LeWM

LeWorldModel (LeWM) [7] is a reward-free JEPA world model trained end-to-end from offline trajectories of pixel observations and actions. An encoder maps each observation to a compact latent representation, $z_t = \text{enc}_\theta(o_t)$, and an action-conditioned predictor models the latent dynamics, $\hat{z}_{t+1} = \text{pred}_\phi(z_{t-H+1:t}, a_{t-H+1:t})$. The default encoder is a ViT-Tiny whose final CLS representation is passed through a projection MLP, while the predictor is a causal Transformer with AdaLN action conditioning. Training uses only a teacher-forced next-latent prediction loss and SIGReg,

$$\mathcal{L}_{\text{LeWM}} = \|\hat{z}_{t+1} - z_{t+1}\|_2^2 + \lambda \mathcal{L}_{\text{SIGReg}}(Z),$$

where SIGReg applies Epps–Pulley normality tests to random one-dimensional projections of the encoded latents, encouraging their joint distribution toward an isotropic Gaussian and thereby discouraging representation collapse. Unlike many prior JEPAs, LeWM uses neither a stop-gradient target, an EMA encoder, nor a pretrained visual representation; gradients propagate through the encoder and predictor jointly.

At inference, candidate action sequences are rolled out autoregressively in latent space and optimized with the cross-entropy method to minimize the terminal distance between the predicted latent state and an encoded goal state. Rather than executing a single optimized sequence open-loop, LeWM uses model predictive control: after executing the current optimized action block, the agent observes the environment again, re-encodes its current state, and replans, allowing subsequent actions to correct for accumulated model error or deviations from the predicted trajectory. A key methodological appeal of LeWM is simplicity: end-to-end latent dynamics learning from pixels requires only a one-step prediction objective and a single anti-collapse regularizer.

VISReg

VISReg [13] is a non-contrastive JEPA regularizer that combines the explicit scale control of VICReg with the full-distribution matching principle of LeJEPA. Its central methodological idea is to decouple *scale* from *shape*. Given centered embeddings $\widehat{Z} \in \mathbb{R}^{N \times D}$, VISReg encourages each latent coordinate to have unit standard deviation through

$$\mathcal{L}_{\text{scale}} = \frac{1}{D} \sum_{j=1}^D \left[1 - \sigma_j(\widehat{Z}) \right]^2.$$

This explicit scale term is motivated in part by a weakness of SIGReg: as the representation collapses toward the zero vector, the gradient of its Epps–Pulley characteristic-function objective also approaches zero, so its corrective signal becomes weakest precisely near complete collapse. By contrast, the derivative of the VISReg scale penalty with respect to the latent scale remains nonzero as $\sigma_j \rightarrow 0$. To prevent the subsequent shape objective from also modifying scale, each centered coordinate is standardized using a detached standard deviation,

$$\widetilde{Z}_j = \frac{\widehat{Z}_j}{\text{sg}(\sigma_j) + \epsilon},$$

where sg denotes stop-gradient.

VISReg regularizes the shape of the standardized embedding distribution using random Cramér–Wold projections and the sliced 2-Wasserstein distance. In general, the one-dimensional Wasserstein distance compares two distributions through their quantile functions; for equally weighted empirical distributions, this reduces to matching their sorted order statistics. VISReg therefore samples random unit directions w_k , projects the standardized embeddings onto each direction, and compares the resulting empirical quantiles with fixed standard-normal quantiles q_N ,

$$\mathcal{L}_{\text{shape}} = \frac{1}{K} \sum_{k=1}^K \left\| \text{sort}(\widetilde{Z} w_k) - q_N \right\|_2^2.$$

By the Cramér–Wold principle, matching all one-dimensional projected distributions would identify the full multivariate distribution; the random sketches provide a tractable approximation to this objective. Unlike VICReg’s covariance penalty, which controls only second-order dependence, this criterion targets the complete distributional shape of the standardized embeddings. VISReg also includes

$$\mathcal{L}_{\text{center}} = \|\mu\|_2^2,$$

which encourages the batch mean μ of the embeddings to approach the zero vector. The complete regularization term is

$$\mathcal{L}_{\text{Reg}} = \lambda_{\text{scale}} \mathcal{L}_{\text{scale}} + \lambda_{\text{shape}} \mathcal{L}_{\text{shape}} + \lambda_{\text{center}} \mathcal{L}_{\text{center}},$$

with equal component weights used by default, although increasing the relative weight on shape can improve performance on long-tailed or low-rank datasets.

For invariance, VISReg follows LeJEPA and directly aligns each view embedding $z_{n,v}$ with the mean μ_n of the global-view embeddings,

$$\mathcal{L}_{\text{pred},n} = \frac{1}{V} \sum_{v=1}^V \|\mu_n - z_{n,v}\|_2^2.$$

The complete objective is

$$\mathcal{L}_{\text{VISReg}} = (1 - \lambda)\mathcal{L}_{\text{pred}} + \lambda\mathcal{L}_{\text{Reg}}.$$

The recommended overall regularization weight depends on dataset scale: $\lambda = 0.9$ is suggested as a starting point for ImageNet-1K, while values around 0.6 are used for smaller datasets. In the image experiments, the views are DINO-style global and local crops, so no separate predictor is required for this symmetric-view setting.

A notable distinction from SIGReg concerns both the discrepancy measure and distributed computation. SIGReg uses the Epps–Pulley characteristic-function criterion, whose empirical characteristic function can be averaged across devices by an all-reduce so that each projected normality test can use the global batch. VISReg instead uses quantile matching, which requires sorting and is not amenable to the same linear all-reduce. It therefore samples different random directions on different GPUs and performs the Wasserstein calculation locally, so additional GPUs increase directional coverage but do not increase the sample size used to estimate any individual projected distribution. Consequently, VISReg trades SIGReg’s efficient global-batch estimation for a scale–shape decomposition with stronger corrective gradients near collapse.

Empirically, VISReg and LeJEPA/SIGReg remain broadly comparable. VISReg shows modest gains on standard downstream evaluations, greater robustness with fewer projection slices, and stronger performance on some challenging datasets when the shape term is upweighted; it also reports improved transfer and out-of-distribution performance.

Fast LeWM

Fast LeWorldModel (Fast-LeWM) [14] replaces LeWM’s autoregressive one-step latent rollout with direct action-prefix prediction. Given a current latent z_t and candidate action sequence $a_{t:t+H-1}$, a causal action-prefix encoder produces prefix representations $p_{t,1:H}$, where $p_{t,k}$ summarizes only $(a_t, \dots, a_{t+k-1})$ together with the current state context. A parallel predictor then estimates all future latents directly from the observed anchor state,

$$\hat{z}_{t+k} = G_\phi(z_t, p_{t,k}), \quad k = 1, \dots, H.$$

Training uses dense supervision over all prefix horizons,

$$\mathcal{L}_{\text{prefix}} = \frac{1}{H} \sum_{k=1}^H \|\hat{z}_{t+k} - z_{t+k}\|_2^2, \quad \mathcal{L} = \mathcal{L}_{\text{prefix}} + \lambda \mathcal{L}_{\text{SIGReg}}(Z),$$

so each prefix token is tied to the latent reached after executing the corresponding partial action sequence. This differs from SG-JEPA’s multi-step rollout objective in two

ways: Fast-LeWM weights all horizons uniformly rather than using geometric discounting, and each $\hat{z}_{t+k}$ is predicted directly from z_t and its action prefix rather than recursively from earlier predicted latents. By contrast, SG-JEPA’s rollout loss uses geometrically discounted weights and deliberately backpropagates through autoregressive prediction chains so that compounded rollout error shapes the learned representation.

The paper’s ablations separate the value of the structured prefix representation from the value of dense prefix supervision. Long-Action LeWM simply enlarges LeWM’s action block so that a single transition predicts the terminal latent after a long sequence of primitive actions, without explicitly representing nested prefixes. The ablation labeled “Fast-LeWM without dense loss” retains the causal action-prefix encoder but supervises only the final prediction $\hat{z}_{t+H}$ rather than every intermediate horizon. Its improvement over Long-Action LeWM therefore isolates the benefit of explicitly encoding the ordered action sequence through accumulated prefixes, while the further gain from full Fast-LeWM shows that dense supervision of the intermediate prefix outcomes is also useful. At planning time, Fast-LeWM retains LeWM’s CEM-based latent goal matching, but evaluates each candidate sequence with a single prefix-encoding pass and parallel latent prediction rather than a sequential rollout; an optional self-consistency penalty compares the terminal state predicted directly from the full prefix with a terminal estimate obtained through an intermediate prefix decomposition.

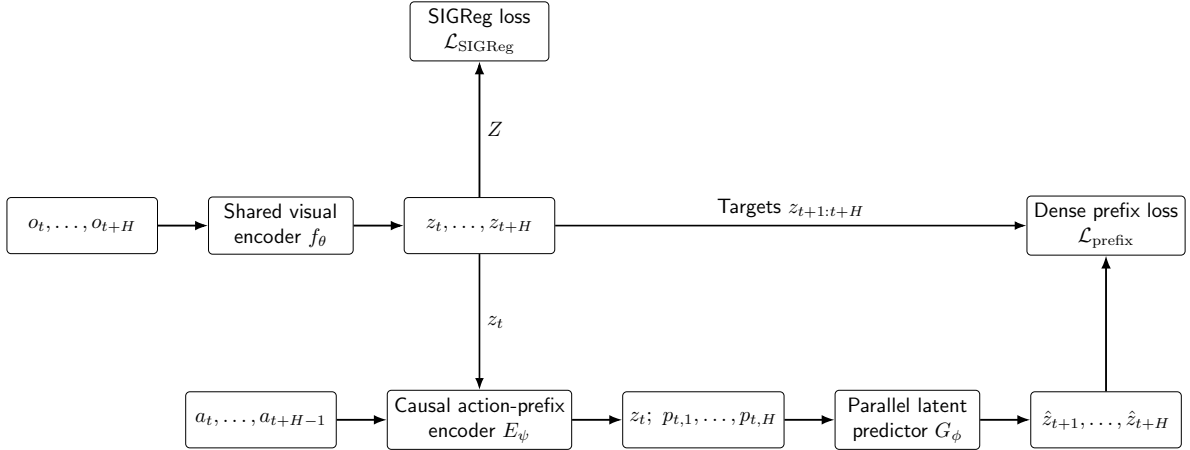


Figure 2: Fast-LeWM training architecture. A shared visual encoder maps the current and future observations $(o_t, \dots, o_{t+H})$ to latent targets $(z_t, \dots, z_{t+H})$. The current latent z_t is mapped to a state token and prepended to the candidate action sequence $(a_t, \dots, a_{t+H-1})$. A causally masked action-prefix encoder produces a token $p_{t,k}$ that summarizes only the first k actions, conditioned on the current state. The parallel latent predictor combines the anchor latent z_t with each prefix token to predict all future latents $(\hat{z}_{t+1}, \dots, \hat{z}_{t+H})$ in one forward pass, without recursively feeding earlier predictions into later ones. Each prediction is matched to its encoded future target through the dense prefix loss, while SIGReg regularizes the encoder outputs to prevent representation collapse. At planning time, the terminal prediction associated with the full action prefix is used to score candidate action sequences against the encoded goal.

SG-JEPA

Semigroup-JEPA (SG-JEPA) [15] learns a latent world model by jointly training a visual encoder and an action-conditioned dynamics predictor using autoregressive multi-step rollouts. Let u_t be the commanded control and g a fixed physical-law parameter, such as gravity. The encoder produces $z_t = e_\phi(o_t)$, while the conditioning encoder receives the augmented input $\tilde{a}_t = [u_t; g]$ and produces $c_t = q_\psi(\tilde{a}_t)$. Thus, even when there is no external control, the physical parameter remains available to the dynamics model. Starting from H encoded observations, SG-JEPA recursively feeds its own predictions back into the dynamics model,

$$z_s^{\text{roll}} = \begin{cases} z_s, & s \leq t, \\ \hat{z}_s, & s > t, \end{cases}$$

$$\hat{z}_{t+k} = f_\theta(z_{t+k-H:t+k-1}^{\text{roll}}, c_{t+k-H:t+k-1}), \quad k = 1, \dots, K.$$

Training minimizes a discounted multi-step latent prediction loss,

$$\mathcal{L}_{\text{roll}} = \sum_{k=1}^K w_k \|\hat{z}_{t+k} - z_{t+k}\|_2^2, \quad w_k = \frac{\gamma^{k-1}}{\sum_{j=1}^K \gamma^{j-1}},$$

with default rollout horizon $K = 5$ and discount factor $\gamma = 0.95$. Unlike one-step teacher-forced objectives, this exposes the model to errors generated by its own predictions and backpropagates their multi-step consequences through both the dynamics model and encoder, encouraging representations that remain predictive under repeated rollout.

Because the same trainable encoder generates the prediction targets, a constant latent representation would otherwise provide a trivial solution. SG-JEPA uses SIGReg to guard against representation collapse, applying it only to the encoded latents z_t and yielding

$$\mathcal{L} = \mathcal{L}_{\text{roll}} + \lambda_{\text{sig}} \mathcal{L}_{\text{SIGReg}}.$$

SIGReg regularizes the latent distribution toward a nondegenerate isotropic Gaussian through random one-dimensional projections, while the rollout loss determines which information should be retained for dynamics prediction. The semigroup perspective is that a single learned transition is composed repeatedly under a fixed physical law g , so predictions formed by short compositions should agree with the corresponding longer-interval evolution. Thus, the central methodological distinction of SG-JEPA is that multi-step autoregressive prediction is used not merely to train a better dynamics predictor, but as a representation-learning signal that encourages latent states whose dynamics remain accurate under repeated composition and across physical-law settings.

References

- [1] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv*, 05 2021. <https://arxiv.org/abs/2105.04906>.

- [2] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicregl: Self-supervised learning of local visual features. *arXiv*, 10 2022. <https://arxiv.org/abs/2210.01571>.
- [3] Yann LeCun. A path towards autonomous machine intelligence. *OpenReview*, 06 2022. <https://openreview.net/pdf?id=BZ5a1r-kVsf>.
- [4] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *arXiv*, 01 2023. <https://arxiv.org/abs/2301.08243>.
- [5] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv*, 02 2024. <https://arxiv.org/abs/2404.08471>.
- [6] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. *arXiv*, 11 2024. <https://arxiv.org/abs/2411.04983>.
- [7] Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv*, 03 2026. <https://arxiv.org/abs/2603.19312>.
- [8] Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. *arXiv*, 02 2025. <https://arxiv.org/abs/2502.14819>.
- [9] Mahmoud Assran et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv*, 06 2025. <https://arxiv.org/abs/2506.09985>.
- [10] Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv*, 11 2025. <https://arxiv.org/abs/2511.08544>.
- [11] Heejeong Nam, Quentin Le Lidec, Lucas Maes, Yann LeCun, and Randall Balestriero. Causal-jepa: Learning world models through object-level latent masking. *arXiv*, 02 2026. <https://arxiv.org/abs/2602.11389>.
- [12] Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim G. J. Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. *arXiv*, 03 2026. <https://arxiv.org/abs/2603.12231>.
- [13] Haiyu Wu, Randall Balestriero, and Morgan Levine. VISReg: Variance-invariance-sketching regularization for JEPa training. *arXiv*, 06 2026. <https://arxiv.org/abs/2606.02572>.
- [14] Yuntian Gao and Xiangyu Xu. Fast leworldmodel. *arXiv*, 06 2026. <https://arxiv.org/abs/2606.26217>.
- [15] Andy Zeyi Liu, Haoran Sun, Lucas Baker, Randall Balestriero, and John Sous. Semigroup-jepa: Latent dynamics consistency for zero-shot physics generalization. *arXiv*, 09 2026. <https://arxiv.org/abs/2609.10464>.