Notes on Joint Embedding Predictive Architecture

Nikos Giakoumoglou[1]

[1]Imperial College London

Joint Embedding Predictive Architectures (JEPAs) have emerged as a principled framework for self-supervised representation learning, shifting the prediction target from input space to learned representation space. This paper provides a comprehensive survey of the JEPA family, covering theoretical foundations, concrete instantiations, and regularization strategies. We begin by reviewing the motivating principles behind JEPAs, including self-supervised learning, energy-based models, architectural design choices, representation collapse, and methods to prevent it. We then present a unified mathematical treatment of JEPA methodologies spanning vision, video, language, and multimodal domains: data2vec, I-JEPA, V-JEPA, V-JEPA 2, C-JEPA, LLM-JEPA, and VL-JEPA. Finally, we examine the evolution of JEPA regularizers from moment-based constraints (VICReg) to distribution-matching objectives grounded in statistical optimality theory: SIGReg, which enforces the provably optimal isotropic Gaussian distribution via characteristic function matching (LeJEPA), and RDMReg, which generalizes to sparse, non-negative representations via the Rectified Generalized Gaussian family (Rectified LpJEPA). Throughout, we emphasize the theoretical progression from heuristic collapse prevention to principled, scalable regularization with provable downstream risk guarantees.

Date: March 18, 2026
Correspondence: Nikos Giakoumoglou <nikos@imperial.ac.uk>

IMPERIAL

1 Introduction

I submit that devising learning paradigms and architectures that would allow machines to learn world models in an unsupervised (or self-supervised) fashion, and to use those models to predict, to reason, and to plan is one of the main challenges of AI and ML today (LeCun et al., 2022).

The world model module constitutes the most complex piece of the architecture. Its role is twofold: **(1) estimate missing information** about the state of the world not provided by perception, **(2) predict plausible future states** of the world. A central difficulty is that the world is only partially observable and inherently uncertain: the same past can lead to many plausible futures, and the same observation can be consistent with many latent states. Any useful world model must therefore represent this uncertainty without committing to a single deterministic prediction. Building such a model requires a learning signal that does not depend on manually designed labels. Self-supervised learning provides exactly this signal by exploiting the internal structure of the data itself.

1.1 Self-Supervised Learning

Self-Supervised Learning (SSL) is a paradigm in which a learning system is trained to capture the mutual dependencies between its inputs. Concretely, this often comes down to training a system to tell us if various parts of its input are consistent with each other.

More precisely, in SSL part of the input signal is observed and denoted $\mathbf{x}$, while another part is either observed or unobserved and denoted $\mathbf{y}$. The learning objective is to capture the statistical dependencies between $\mathbf{x}$ and $\mathbf{y}$ without access to any external labels. In a temporal prediction scenario, $\mathbf{x}$ represents past and present observations and $\mathbf{y}$ represents future observations. In a spatial scenario, $\mathbf{x}$ might be a visible region of an image and $\mathbf{y}$ the masked-out portion. In a multi-modal scenario, $\mathbf{x}$ could be a video stream and $\mathbf{y}$ the corresponding audio.

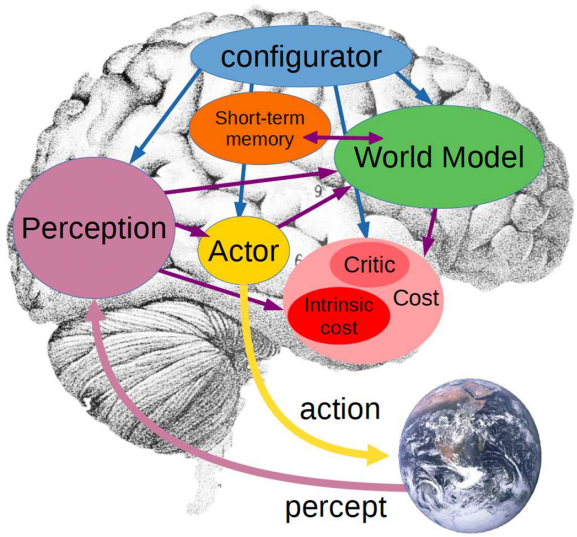


Figure 1 A system architecture for autonomous intelligence. All modules in this model are assumed to be "differentiable", in that a module feeding into another one (through an arrow connecting them) can get gradient estimates of the cost's scalar output with respect to its own output.

The key insight is that $\mathbf{y}$ need not be explicitly reconstructed. It suffices for the system to learn an internal representation that is predictive of $\mathbf{y}$ given $\mathbf{x}$. This idea can be formalized as follows: given two inputs $\mathbf{x}$ and $\mathbf{y}$, learn two functions that compute representations $\mathbf{s}_x = g_x(\mathbf{x})$ and $\mathbf{s}_y = g_y(\mathbf{y})$ such that (1) $\mathbf{s}_x$ and $\mathbf{s}_y$ are maximally informative about $\mathbf{x}$ and $\mathbf{y}$, respectively, and (2) $\mathbf{s}_y$ can easily be predicted from $\mathbf{s}_x$.

The literature on SSL can be broadly divided into two families. **Generative methods** learn to reconstruct $\mathbf{y}$ directly in input space, typically by maximizing the likelihood $p(\mathbf{y} \mid \mathbf{x})$. Examples include autoregressive language models and masked autoencoders. **Discriminative (or embedding-based) methods** instead learn representations that capture dependencies between $\mathbf{x}$ and $\mathbf{y}$ without committing to a full generative model of the input. Contrastive learning, non-contrastive methods, and the Joint Embedding Predictive Architecture all belong to this second family. A general and unifying formulation for both families can be given through the framework of Energy-Based Models.

1.2 Energy-Based Models (EBM)

An Energy-Based Model (EBM) is a scalar-valued function $F_w(\mathbf{x}, \mathbf{y})$ parameterized by learnable weights w that maps every pair $(\mathbf{x}, \mathbf{y})$ to a single scalar called the *energy*. The energy is trained to be low when $\mathbf{x}$ and $\mathbf{y}$ are compatible (*i.e.* drawn from the data distribution) and high when they are not. Data points are black dots. The energy function produces low energy values around the data points, and higher energies away from the regions of high data density, as symbolized by the contour lines of the energy landscape.

Unlike probabilistic models, EBMs do not require the energy surface to be normalized into a valid probability distribution. This is a significant advantage: computing the partition function $Z(\mathbf{x}) = \int \exp(-F_w(\mathbf{x}, \mathbf{y}))\, d\mathbf{y}$ is intractable in high dimensions, and avoiding this normalization makes EBMs applicable to structured, high-dimensional outputs such as images and videos. Training an EBM amounts to shaping the energy landscape so that (1) observed pairs $(\mathbf{x}, \mathbf{y})$ lie in low-energy valleys and (2) incompatible pairs occupy high-energy regions. The fundamental challenge of EBM training is ensuring the second condition: without explicit mechanisms to raise the energy of incompatible pairs, the model can assign uniformly low energy everywhere, a failure mode known as *collapse*.

Formally, inference in an EBM corresponds to finding, for a given $\mathbf{x}$, the value of $\mathbf{y}$ that minimizes the energy:

$$\hat{\mathbf{y}} = \arg\min_{\mathbf{y}} F_w(\mathbf{x}, \mathbf{y}).$$

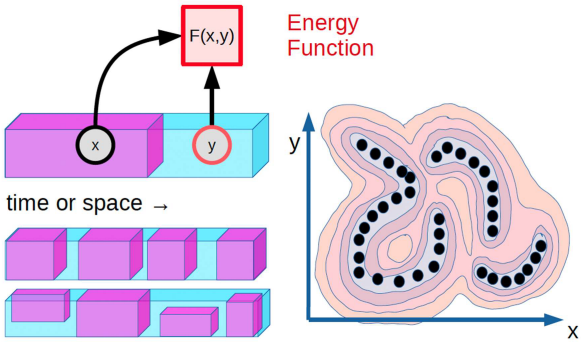


Figure 2 Self-Supervised Learning (SSL) and Energy-Based Models (EBM). SSL is a learning paradigm in which a learning system is trained to "fill in the blanks", or more precisely to capture the dependencies between observed parts of the input and possibly unobserved parts of the input. Part of the input signal is observed and denoted $\mathbf{x}$ (in pink), and part of the input signal is either observed or unobserved and denoted $\mathbf{y}$ (in blue). In a temporal prediction scenario, $\mathbf{x}$ represents past and present observations, and $\mathbf{y}$ represent future observations.

When the mapping from $\mathbf{x}$ to compatible $\mathbf{y}$ values is one-to-many (*i.e.* when multiple outputs are plausible for a single input), a latent variable $\mathbf{z}$ can be introduced to index the different modes:

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

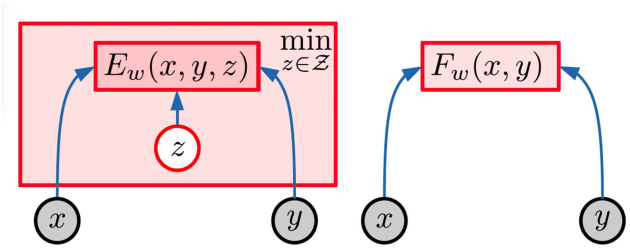


Figure 3 Latent-Variable Energy-Based Model (LVEBM). To evaluate the degree of compatibility between $\mathbf{x}$ and $\mathbf{y}$, an EBM may need the help of a latent variable $\mathbf{z}$. For example, if $\mathbf{x}$ is a view of an object, and $\mathbf{y}$ another view of the same object, $\mathbf{z}$ may parameterize the camera displacement between the two views.

The latent variable $\mathbf{z}$ captures the residual uncertainty that $\mathbf{x}$ alone cannot resolve. For example, if $\mathbf{x}$ is one view of a 3D object and $\mathbf{y}$ is another view, $\mathbf{z}$ may encode the camera displacement between the two viewpoints. By minimizing over $\mathbf{z}$, the model can assign low energy to all plausible $\mathbf{y}$ values without requiring them to be identical. This latent-variable formulation is central to understanding JEPA, where $\mathbf{z}$ controls the specificity of the prediction and its information capacity must be carefully regulated.

1.3 Architectures

Before discussing collapse and how to prevent it, it is useful to review the main architectural families that arise when instantiating EBMs for self-supervised learning. Each architecture defines a different computational graph relating $\mathbf{x}$, $\mathbf{y}$, and the energy, and each exhibits different collapse properties.

1.3.1 Predictive (Deterministic-Generative) Architecture

The simplest architecture directly predicts $\mathbf{y}$ from $\mathbf{x}$ through a deterministic function $\hat{\mathbf{y}} = f_w(\mathbf{x})$, and defines the energy as a reconstruction cost $F_w(\mathbf{x}, \mathbf{y}) = D(\mathbf{y}, \hat{\mathbf{y}})$, where D is a distance or divergence (*e.g.* squared ℓ_2 norm). Because $\hat{\mathbf{y}}$ is a fixed function of $\mathbf{x}$, the energy is zero only at $\mathbf{y} = \hat{\mathbf{y}}$ and strictly positive elsewhere (provided D is a proper metric). This architecture is the basis of standard supervised regression and, in the SSL context, of denoising autoencoders that predict corrupted portions of the input. Its main limitation is

that it commits to a single prediction per input, making it poorly suited for modeling multi-modal conditional distributions $p(\mathbf{y} \mid \mathbf{x})$.

1.3.2 Generative Latent-Variable Architecture

To represent the multi-modality of the output distribution, one introduces a latent variable $\mathbf{z}$ drawn from a prior distribution and generates $\hat{\mathbf{y}} = f_w(\mathbf{x}, \mathbf{z})$. The energy is $E_w(\mathbf{x}, \mathbf{y}, \mathbf{z}) = D(\mathbf{y}, f_w(\mathbf{x}, \mathbf{z}))$ and the effective energy is obtained by minimizing over $\mathbf{z}$: $F_w(\mathbf{x}, \mathbf{y}) = \min_{\mathbf{z}} E_w(\mathbf{x}, \mathbf{y}, \mathbf{z})$. As $\mathbf{z}$ varies over its domain $\mathcal{Z}$, the prediction $\hat{\mathbf{y}}$ sweeps out a set $\text{Pred}(\mathbf{x}, \mathcal{Z})$ that ideally covers all values of $\mathbf{y}$ compatible with $\mathbf{x}$. Variational autoencoders (VAEs) (Kingma and Welling, 2022), generative adversarial networks (GANs) (Goodfellow et al., 2014), and diffusion models (Ho et al., 2020) are well-known instances of this family. The critical design choice is the information capacity of $\mathbf{z}$: if $\mathcal{Z}$ is too large, the model can assign low energy to the entire output space, leading to collapse.

1.3.3 Autoencoder (AE) Architecture

An autoencoder (Hinton and Salakhutdinov, 2006; Kingma and Welling, 2022) encodes the target $\mathbf{y}$ into a bottleneck representation $\mathbf{s}_y = \text{Enc}(\mathbf{y})$ and reconstructs it as $\hat{\mathbf{y}} = \text{Dec}(\mathbf{s}_y)$. The energy is the reconstruction error $F_w(\mathbf{y}) = D(\mathbf{y}, \hat{\mathbf{y}})$. By constraining the dimensionality or regularity of $\mathbf{s}_y$, the autoencoder is forced to learn a compressed representation that captures the most salient structure in $\mathbf{y}$. If, however, the dimension of $\mathbf{s}_y$ equals or exceeds that of $\mathbf{y}$, the encoder can learn the identity mapping, yielding zero reconstruction error everywhere and collapsing the energy surface.

1.3.4 Joint Embedding Architecture (JEA)

A JEA processes both $\mathbf{x}$ and $\mathbf{y}$ through (possibly different) encoder networks to produce representations $\mathbf{s}_x = g_x(\mathbf{x})$ and $\mathbf{s}_y = g_y(\mathbf{y})$, and defines the energy as a distance in the shared representation space: $F_w(\mathbf{x}, \mathbf{y}) = D(\mathbf{s}_x, \mathbf{s}_y)$. Unlike generative architectures, a JEA never reconstructs $\mathbf{y}$ in input space. The encoders are free to discard irrelevant details and retain only the information needed to match compatible pairs. This property makes JEAs attractive for learning abstract, high-level representations. However, the architecture is susceptible to a particularly simple form of collapse: if both encoders produce the same constant output regardless of input, the energy is zero everywhere.

1.4 Collapse

Collapse refers to any configuration of the energy function in which large regions of the input space are assigned uniformly low energy, rendering the model unable to distinguish compatible from incompatible pairs. In the context of SSL, collapse means that the learned representations carry no useful information about the input. Each architecture described above admits a characteristic collapse mode:

(a) A regular predictive or deterministic-generative architecture cannot collapse. For any $\mathbf{x}$, a single $\hat{\mathbf{y}}$ is produced. The energy is zero whenever $\mathbf{y} = \hat{\mathbf{y}}$. Any $\mathbf{y}$ different from $\hat{\mathbf{y}}$ will have a higher energy, as long as $D(\mathbf{y}, \hat{\mathbf{y}})$ is strictly larger than zero whenever $\mathbf{y}$ is different from $\hat{\mathbf{y}}$. The deterministic mapping guarantees that the zero-energy set is a single point per input, so the energy surface always has non-trivial structure.

(b) A generative latent-variable architecture (non-deterministic generative) can collapse when the latent variable has too much information capacity. When the latent variable $\mathbf{z}$ varies over the set $\mathcal{Z}$, the prediction $\hat{\mathbf{y}}$ varies over a set $\text{Pred}(\mathbf{s}_x, \mathcal{Z})$, which must match the set of $\mathbf{y}$ that are compatible with $\mathbf{x}$. If $\mathcal{Z}$ is too "large" then the region of low energy $\mathbf{y}$ may be larger than the region of high data density. If $\mathcal{Z}$ has the same dimension as $\mathbf{y}$, the system could very well give zero energy to the entire $\mathbf{y}$ space. In practice, this manifests as the model learning to copy $\mathbf{z}$ directly to $\hat{\mathbf{y}}$, bypassing the input $\mathbf{x}$ entirely.

(c) An auto-encoder (AE) can collapse when the representation $\mathbf{s}_y$ has too much information capacity. For example, if the dimension of $\mathbf{s}_y$ is equal or higher than that of $\mathbf{y}$, the AE could learn the identity function, producing a reconstruction error equal to zero over the entire $\mathbf{y}$ space. The bottleneck must be sufficiently constrained to force the encoder to learn meaningful abstractions.

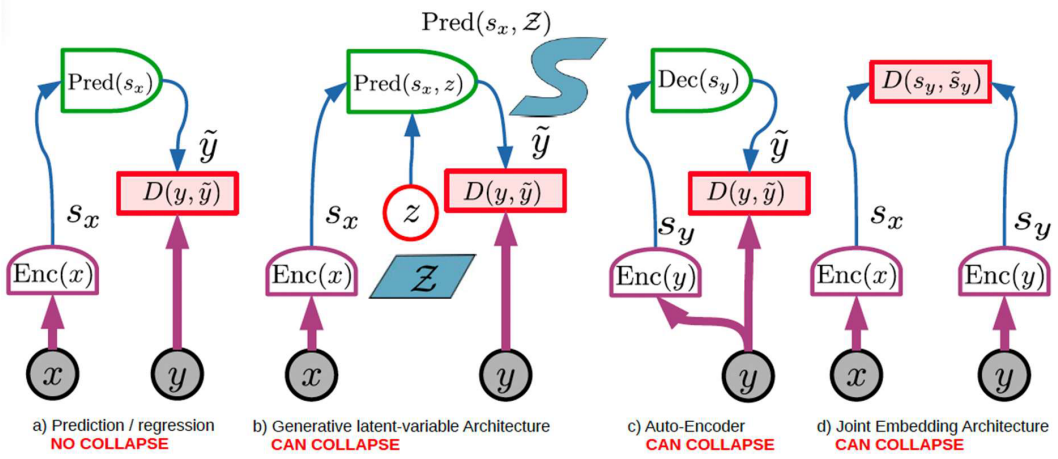


Figure 4 A few standard architectures and their capacity for collapse.

(d) Lastly, a Joint Embedding Architecture (JEA) can collapse when the information carried by $\mathbf{s}_x$ and/or $\mathbf{s}_y$ are insufficient. If the encoders ignore the inputs, and produce constant and equal codes $\mathbf{s}_x = \mathbf{s}_y$, the entire space will have zero energy. This mode of collapse is the most dangerous in practice because it is the easiest to reach: a constant encoder satisfies the objective trivially, and gradient-based optimization can converge to this solution early in training if no countermeasure is employed.

A unifying perspective is that collapse occurs whenever the energy function can be trivially minimized without learning meaningful structure. In architectures (b) and (c), collapse arises from *excess capacity* in the latent variable or bottleneck. In architecture (d), collapse arises from *insufficient capacity* or *informational degeneracy* in the embeddings. Preventing collapse therefore requires careful regulation of the information flow through the network, which is the subject of the next subsection.

1.5 How to Prevent Collapse

How do we design the loss to prevent collapse? There are two approaches: (1) **contrastive methods** and (2) **regularized methods**. In the following, I will argue that contrastive methods have flaws and that regularized (non-contrastive) methods are much more likely to be preferable in the long run.

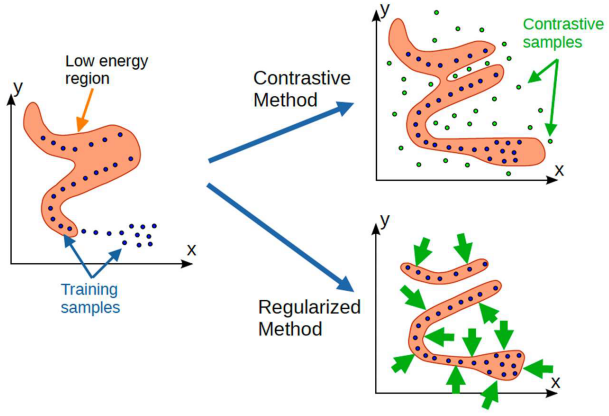


Figure 5 Contrastive and regularized methods for EBM training.

Contrastive methods consist in using a loss functional whose minimization has the effect of pushing down on the energies of training samples (x, y), and pulling up on the energies of suitably-hallucinated "contrastive" samples $(x, \tilde{y})$. An example of such loss is the popular InfoNCE. Contrastive methods are very popular,

particularly for Siamese network architectures trained with pairs where x is a distorted or corrupted version of y and $\tilde{y}$ another random (or suitably chosen) training sample.

But there are two main issues with contrastive methods. First, one has to design a scheme to generate or pick suitable $\tilde{y}$. Second, when y is in a high-dimensional space, and if the EBM is flexible, it may require a very large number of contrastive samples to ensure that the energy is higher in all dimensions unoccupied by the local data distribution. Because of the curse of dimensionality, in the worst case, the number of contrastive samples may grow exponentially with the dimension of the representation. This is the main reason why I will argue against contrastive methods.

Regularized methods for EBM training are much more promising in the long run than contrastive methods because they can eschew the curse of dimensionality that plagues contrastive methods. They consist in constructing a loss functional that has the effect of pushing down on the energies of training samples, and simultaneously minimizing the volume of y space to which the model associates a low energy.

1.6 Joint Embedding Predictive Architecture (JEPA)

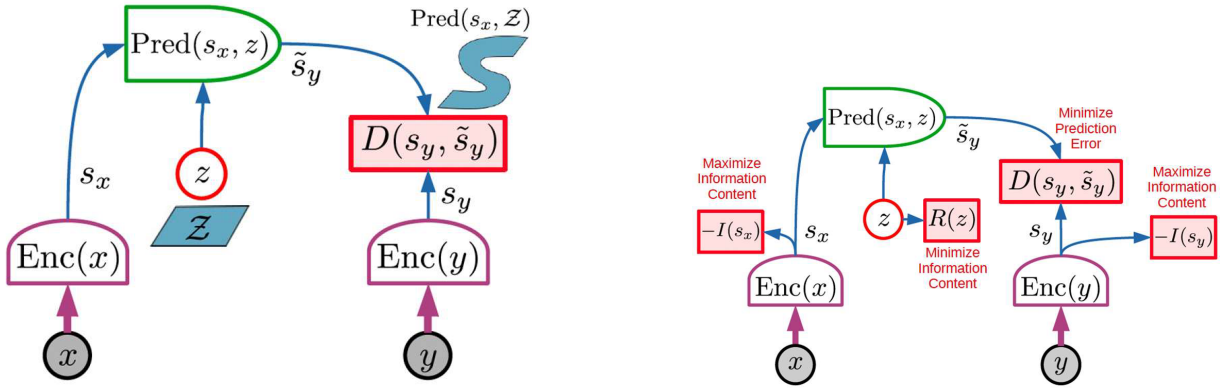
The centrepiece of this paper is the Joint Embedding Predictive Architecture (JEPA). JEPA is not generative in the sense that it cannot easily be used to predict $\mathbf{y}$ from $\mathbf{x}$. It merely captures the dependencies between $\mathbf{x}$ and $\mathbf{y}$ without explicitly generating predictions of $\mathbf{y}$.

The two variables $\mathbf{x}$ and $\mathbf{y}$ are fed to two encoders producing two representations $\mathbf{s}_x$ and $\mathbf{s}_y$. These two encoders may be different. They are not required to possess the same architecture nor are they required to share their parameters. This allows $\mathbf{x}$ and $\mathbf{y}$ to be different in nature (*e.g.* video and audio).

A predictor module predicts the representation of $\mathbf{y}$ from the representation of $\mathbf{x}$. The predictor may depend on a latent variable $\mathbf{z}$. The energy is simply the prediction error in representation space:

$$E_w(\mathbf{x}, \mathbf{y}, \mathbf{z}) = D(\mathbf{s}_y, \text{Pred}(\mathbf{s}_x, \mathbf{z}))$$

The main advantage of JEPA is that it performs predictions in representation space, eschewing the need to predict every detail of $\mathbf{y}$. This is enabled by the fact that the encoder of $\mathbf{y}$ may choose to produce an abstract representation from which irrelevant details have been eliminated.



(a) JEPA training.

(b) Non-contrastive training of JEPA.

Figure 6 The Joint-Embedding Predictive Architecture (JEPA) and its training objectives.

1.6.1 Controlling the Latent Variable

For any $\mathbf{s}_y$ it is possible to set $\mathbf{z} = \mathbf{s}_y$, which would make the energy $D(\mathbf{s}_y, \tilde{\mathbf{s}}_y)$ zero. This corresponds to a totally flat and collapsed energy surface. How do we prevent this collapse from happening? By limiting or minimizing the information content of the latent variable. How can this be done? By making $\mathbf{z}$ discrete,

low-dimensional, sparse, or noisy, among other methods. A few concrete examples may help build an intuitive understanding of the phenomenon. Suppose that

$$D(\mathbf{s}_y, \tilde{\mathbf{s}}_y) = \|\mathbf{s}_y - \tilde{\mathbf{s}}_y\|^2$$

and that $\mathbf{z}$ is discrete with K possible integer values $[0, K-1]$. For a given $\mathbf{x}$, there can be only K possible values of $\tilde{\mathbf{s}}_y$:

$$\text{Pred}(\mathbf{s}_x, 0), \quad \text{Pred}(\mathbf{s}_x, 1), \ldots, \quad \text{Pred}(\mathbf{s}_x, K-1)$$

Hence, these can be the only values of $\mathbf{s}_y$ with zero energy, and there are only K of them. Consider a point $\mathbf{s}_y$ that starts from $\text{Pred}(\mathbf{s}_x, 0)$ and moves towards $\text{Pred}(\mathbf{s}_x, 1)$. Its energy will start from zero, increase quadratically as $\mathbf{s}_y$ moves away from $\text{Pred}(\mathbf{s}_x, 0)$, until $\mathbf{s}_y$ reaches the midpoint between the two predictions, after which it decreases toward zero again as it approaches $\text{Pred}(\mathbf{s}_x, 1)$. The energy landscape thus consists of isolated wells, one per discrete latent value, with barriers between them. By controlling K, we directly control the number of modes and prevent the energy from becoming flat.

1.7 Training JEPA with VICReg

VICReg includes: (1) **Variance**: a hinge loss that maintains the standard deviation of each component of $\mathbf{s}_y$ and $\mathbf{v}_y$ above a threshold over a batch. (2) **Covariance**: a covariance loss in which the covariance between pairs of different components of $\mathbf{v}_y$ are pushed towards zero. This has the effect of decorrelating the components of $\mathbf{v}_y$, which will in turn make the components of $\mathbf{s}_y$ somewhat independent. The same criteria are applied to $\mathbf{s}_x$ and $\mathbf{v}_x$ separately.

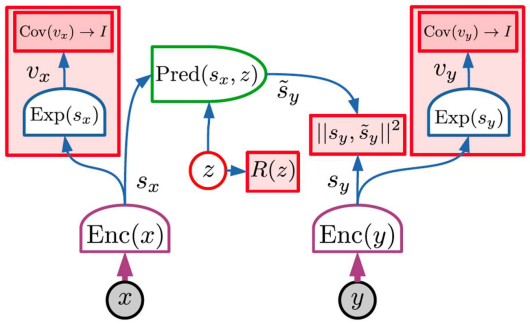


Figure 7 Training a JEPA with VICReg.

The third criterion of VICReg is the representation prediction error $D(\mathbf{s}_y, \tilde{\mathbf{s}}_y)$. In the simplest implementations of VICReg, the predictor is constant (equal to the identity function), making the representations invariant to the transformation that turns $\mathbf{x}$ into $\mathbf{y}$. In more sophisticated versions, the predictor may have no latent variable, or may depend on a latent variable that is either discrete, low dimensional, or stochastic. A simple instantiation of VICReg to learn invariant representations consists in making $\mathbf{x}$ and $\mathbf{y}$ be different views (or distorted versions) of the same content, setting the predictor to the identity function, and defining:

$$D(\mathbf{s}_y, \tilde{\mathbf{s}}_y) = D(\mathbf{s}_y, \mathbf{s}_x) = \|\mathbf{s}_y - \mathbf{s}_x\|^2.$$

1.8 From Reconstruction to Latent Prediction

The distinction between predicting in input space and predicting in representation space is fundamental to understanding JEPA. We illustrate this distinction by comparing Masked Autoencoders (MAE) with the JEPA objective.

Masked AutoEncoders (MAE). The original MAE (He et al., 2022) and its successive variants (*e.g.* (Li et al., 2023; Tong et al., 2022)) introduced a training objective that seeks to reconstruct missing pieces of data from its partially masked input. This approach, while simple, has proven efficient and scalable (Wang et al.,

2023). Formally, the MAE objective uses an encoder function $f_W(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^d$ and a decoder function $g_V(\mathbf{f}) : \mathbb{R}^d \to \mathbb{R}^d$ to predict a target $\mathbf{y}$ given input $\mathbf{x}$:

$$\mathcal{L}_{\text{MAE}} = \frac{1}{2}\mathbb{E}_{\mathbf{x},\mathbf{y}\sim p(\mathbf{x},\mathbf{y})}\|g_V(f_W(\mathbf{x})) - \mathbf{y}\|^2, \tag{1}$$

where we define a joint distribution over inputs and targets $p = p(\mathbf{x}, \mathbf{y})$ with $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. Here we assume tied dimensions for simplicity.

In general, we can think of $\mathbf{x}$ and $\mathbf{y}$ as different inputs sharing the same underlying semantic information. One common practice is to let the input $\mathbf{x}$ correspond to a partially masked input, and the target $\mathbf{y}$ corresponds to the masked-out portions of the input. A critical design choice of the MAE objective is that the loss penalizes the reconstruction error directly in the target's input space. This implies that, as perhaps some parts of the target $\mathbf{y}$ cannot be predicted given $\mathbf{x}$, the quality of the encoding f_W crucially depends on how $\mathbf{x}, \mathbf{y}$ are sampled. The encoder must allocate capacity to model low-level details (*e.g.* exact pixel values, textures) even when those details are irrelevant to semantic understanding. This reconstruction bias can divert representation capacity away from high-level abstractions and toward fine-grained input statistics.

From MAE to JEPA. JEPA addresses this limitation by moving the prediction target from input space to representation space. When the encoder function for $\mathbf{x}$ and $\mathbf{y}$ is shared, a simple JEPA objective is given by:

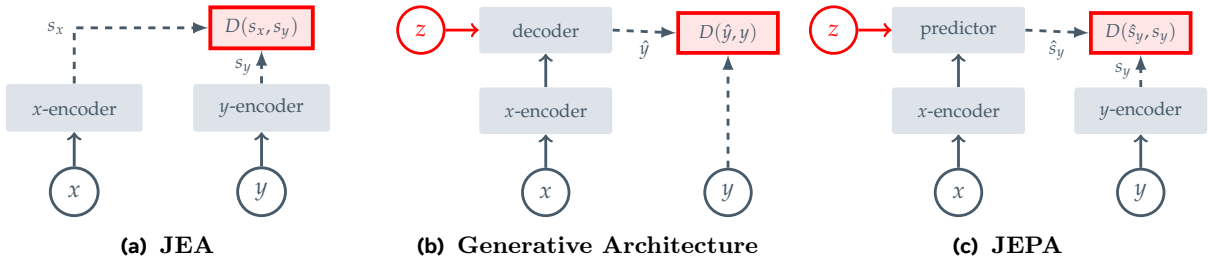
$$\mathcal{L}_{\text{JEPA}} = \frac{1}{2}\mathbb{E}_{\mathbf{x},\mathbf{y}\sim p(\mathbf{x},\mathbf{y})}\|g_V(f_W(\mathbf{x})) - f_W(\mathbf{y})\|^2. \tag{2}$$

Unlike the objective in (1), the objective given by (2) is susceptible to collapse: it can be minimized trivially by employing an encoder and decoder that map all inputs to the same vector. Contrastive methods (van den Oord et al., 2019; Chen et al., 2020; He et al., 2020) avoid this issue by including an additional loss term that pushes apart representations for negative pairs (two different samples). The drawback to this approach is that it can require comparing a sample to many negative examples to work effectively. This has led to a rise in the popularity of non-contrastive SSL. With non-contrastive methods, it becomes imperative to implicitly or explicitly bias the representation mapping $f_W(\mathbf{x})$ to avoid its collapse to a trivial solution (*e.g.* $f_W(\mathbf{x}) = \mathbf{0}, \forall \mathbf{x}$). In this paper, we leverage the widely adopted (Chen and He, 2021) `stop-grad` operator, preventing the flow of gradients through the $f_W(\mathbf{y})$ branch. Prominent examples of this foundational architecture include BYOL (Grill et al., 2020), data2vec (Baevski et al., 2022), SimSiam (Chen and He, 2021) and the more recent I-JEPA (Assran et al., 2023) and V-JEPA (Bardes et al., 2024; Assran et al., 2025; Mur-Labadia et al., 2026) models. Consequently, the JEPA models under consideration fall under non-contrastive self-distillation methods. We will refer to such methods as JEPA for the remainder of this paper.

The shift from MAE to JEPA can be understood as follows. Both objectives use masking to create an information asymmetry between $\mathbf{x}$ and $\mathbf{y}$. However, MAE requires the decoder to reconstruct every masked pixel, whereas JEPA only requires the predictor to match the *encoded* representation of the masked region. The target encoder $f_W(\mathbf{y})$ acts as an information bottleneck: it discards unpredictable low-level details and retains only the semantic content, which is precisely the quantity the predictor must recover. This eliminates the reconstruction bias inherent in MAE and allows the encoder to focus its capacity on learning high-level, transferable representations.

2 Methodologies

We now describe the concrete instantiations of the JEPA framework. To facilitate comparison across methods, we adopt a unified notation throughout this section. Let $\mathbf{x}$ denote a (possibly masked) input and $\mathbf{y}$ the corresponding full target input. We write f_θ for the context (student) encoder, $f_{\bar{\theta}}$ for the target (teacher) encoder whose parameters $\bar{\theta}$ are an exponential moving average (EMA) of θ, and g_ϕ for the predictor. Patch-level representations produced by the context encoder are denoted $\mathbf{s}_x = \{\mathbf{s}_{x_j}\}_{j \in B_x}$, those produced by the target encoder $\mathbf{s}_y = \{\mathbf{s}_{y_j}\}_{j=1}^{N}$, and the predicted representations $\hat{\mathbf{s}}_y = \{\hat{\mathbf{s}}_{y_j}\}_{j \in B_i}$. We denote by B_i the mask corresponding to the i-th target block, by M the total number of target blocks, and by $\mathrm{sg}(\cdot)$ the stop-gradient operator. All encoders are Vision Transformers (ViT) (Dosovitskiy et al., 2021) unless stated otherwise.



(a) JEA **(b) Generative Architecture** **(c) JEPA**

Figure 8 From joint-embedding methods, to generative approaches, to JEPAs.

2.1 data2vec

data2vec (Baevski et al., 2022) is a modality-agnostic self-supervised framework that predicts contextualized latent representations of the full input from a masked view. Unlike methods that predict modality-specific targets (pixels for images, tokens for text, speech units for audio), data2vec predicts the output of a teacher network that has observed the complete, unmasked input. This makes the prediction targets *contextualized*: each target representation encodes information from the entire input, not just its local neighbourhood.

Architecture. The student encoder f_θ receives a masked version of the input $\mathbf{x}$ and produces patch-level representations $\mathbf{s}_x = f_\theta(\mathbf{x})$. The teacher encoder $f_{\bar{\theta}}$ processes the full, unmasked input $\mathbf{y}$ and produces target representations. Rather than using the output of a single layer, data2vec constructs the target as the average of the top K Transformer layers of the teacher encoder. For the j-th token, the target is:

$$\mathbf{s}_{y_j} = \frac{1}{K} \sum_{l=L-K+1}^{L} \hat{f}_{\bar{\theta}}^{(l)}(\mathbf{y})_j,$$

where $\hat{f}_{\bar{\theta}}^{(l)}(\mathbf{y})_j$ is the normalized representation at layer l for token j, and L is the total depth of the Transformer. Each layer output is instance-normalized before averaging.

Training objective. A regression head on top of the student encoder predicts the contextualized target at each masked position. The loss is the smooth ℓ_1 (Huber) loss applied only to the masked tokens:

$$\mathcal{L}_{\text{data2vec}} = \frac{1}{|\mathcal{M}|} \sum_{j \in \mathcal{M}} \ell_\beta\big(g_\phi(f_\theta(\mathbf{x}))_j,\ \mathrm{sg}(\mathbf{s}_{y_j})\big)\,, \tag{3}$$

where $\mathcal{M}$ is the set of masked token indices, ℓ_β is the Huber loss with threshold β, and $\mathrm{sg}(\cdot)$ denotes stop-gradient. The teacher parameters are updated via EMA: $\bar{\theta} \leftarrow \tau\bar{\theta} + (1-\tau)\theta$.

Key distinctions from JEPA. While data2vec predicts in representation space and uses an EMA teacher (properties shared with JEPA), it differs in several respects. First, the masking in data2vec is applied to the *input* of the student encoder, whereas in I-JEPA the masking is applied to the *output* of the target encoder. Second, the targets are multi-layer averages, providing richer supervision than single-layer outputs. Third,

data2vec does not use a spatially structured predictor conditioned on positional mask tokens; instead, a simple regression head predicts the target at each masked position. For images, the original data2vec uses BEiT-style block-wise masking.

2.2 I-JEPA

The Image-based Joint-Embedding Predictive Architecture (I-JEPA) (Assran et al., 2023) learns semantic image representations by predicting the latent features of masked image regions from visible context, without relying on pixel reconstruction or hand-crafted data augmentations. The central idea is to predict *representations* of target blocks rather than their raw pixels, allowing the target encoder to produce abstract prediction targets from which irrelevant low-level details have been eliminated.

Patchification and masking. Given a target image $\mathbf{y} \in \mathbb{R}^{H\times W\times 3}$, it is partitioned into N non-overlapping patches of size $P \times P$ (*e.g.* 16×16), yielding a sequence of $N = HW/P^2$ patch tokens. A binary spatial masking operator $\mathbf{M}$ is applied over the patch grid to define the visible *context* $\mathbf{x} = \mathbf{M} \circ \mathbf{y}$, where $\circ$ denotes elementwise (Hadamard) masking. The context is sampled as a single large block with random scale in $(0.85, 1.0)$ and unit aspect ratio. Additionally, M (typically M=4) possibly overlapping *target* blocks $\{B_i\}_{i=1}^{M}$ are sampled with random scale in $(0.15, 0.2)$ and aspect ratio in $(0.75, 1.5)$. Any overlap between the target blocks and the context block is removed from the context to ensure a non-trivial prediction task.

Target representations. The target encoder $f_{\bar{\theta}}$ processes the full, unmasked image $\mathbf{y}$ and produces patch-level representations $\mathbf{s}_y = f_{\bar{\theta}}(\mathbf{y}) = \{\mathbf{s}_{y_j}\}_{j=1}^{N}$. The masking is applied to the *output* of the target encoder, not the input, so that each target representation $\mathbf{s}_{y_j}$ is contextualized by the entire image. This design choice is crucial for producing targets of a high semantic level.

Context encoding and prediction. The context encoder f_θ (a standard ViT that processes only the visible context patches) produces $\mathbf{s}_x = f_\theta(\mathbf{x}) = \{\mathbf{s}_{x_j}\}_{j\in B_x}$, where B_x denotes the set of visible patch indices. For each target block B_i, the predictor g_ϕ (a narrow ViT) receives the context encoder output $\mathbf{s}_x$ together with a set of learnable mask tokens $\{\mathbf{m}_j\}_{j\in B_i}$, where each mask token is parameterized as a shared learnable vector with an added positional embedding $\mathbf{p}_j$. The predictor outputs the patch-level predictions:

$$\hat{\mathbf{s}}_y^{(i)} = \{\hat{\mathbf{s}}_{y_j}\}_{j\in B_i} = g_\phi(\mathbf{s}_x, \{\mathbf{m} + \mathbf{p}_j\}_{j\in B_i}) \,.$$

Training objective. The loss is the average ℓ_2 distance between predicted and target patch-level representations across all target blocks:

$$\mathcal{L}_{\text{I-JEPA}} = \frac{1}{M}\sum_{i=1}^{M}\sum_{j\in B_i} \left\|\hat{\mathbf{s}}_{y_j} - \text{sg}(\mathbf{s}_{y_j})\right\|_2^2 . \tag{4}$$

The context encoder θ and predictor ϕ are optimized via gradient descent, while the target encoder parameters evolve under the EMA rule $\bar{\theta} \leftarrow \tau\bar{\theta} + (1-\tau)\theta$, with the momentum coefficient τ linearly increased from 0.996 to 1.0 during training.

Multi-block masking strategy. The design of the masking strategy is a core contribution of I-JEPA. Predicting sufficiently large target blocks forces the model to capture semantic content (objects, parts), while using a large, spatially distributed context block ensures the prediction task is informative yet non-trivial. Both the scale of the target blocks and the informativeness of the context are critical for guiding the model toward learning semantic representations.

2.3 V-JEPA

V-JEPA (Bardes et al., 2024) extends the JEPA framework from images to video, enabling representation learning that captures spatial semantics and temporal dynamics jointly. The key insight is that predicting missing spatiotemporal regions in representation space forces the model to focus on high-level, temporally coherent information (object trajectories, interactions, scene-level dynamics) while ignoring stochastic, unpredictable pixel-level details.

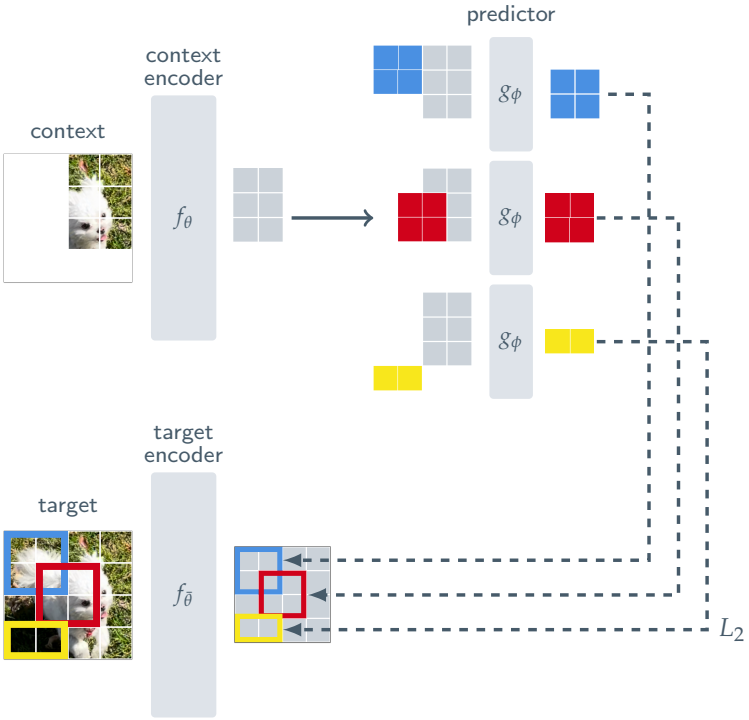


Figure 9 Overview of the I-JEPA image latent prediction framework.

Spatiotemporal patchification and masking. Let the target video be $\mathbf{v} \in \mathbb{R}^{T\times H\times W\times 3}$, where T is the number of frames. The video is patchified into N_t spatiotemporal tokens (tubelets) using $P \times P$ spatial patches and a temporal stride of t_s frames, yielding tokens of size $t_s \times P \times P$. A spatiotemporal masking operator $\mathbf{M}$ is applied, and the visible context is defined as $\mathbf{v}_c = \mathbf{M} \circ \mathbf{v}$. The masking strategy uses contiguous spatiotemporal blocks (multi-block masking), with masking ratios of approximately 90%, which forces the model to perform long-range spatiotemporal reasoning from very limited visible context. M target regions $\{R_i\}_{i=1}^{M}$ are specified, each a subset of the spatiotemporal token indices.

Encoding and prediction. The context encoder f_θ processes the visible tokens $\mathbf{v}_c$ to produce latent features $\mathbf{z}_c = f_\theta(\mathbf{v}_c)$, while the EMA target encoder produces full-video representations $\mathbf{z}_t = f_{\bar{\theta}}(\mathbf{v}) = \{\mathbf{z}_t[j]\}_{j=1}^{N_t}$. For each target region R_i, the predictor generates:

$$\hat{\mathbf{z}}_t^{(i)} = \{\hat{\mathbf{z}}_t[j]\}_{j\in R_i} = g_\phi(\mathbf{z}_c, \{\mathbf{m} + \mathbf{p}_j\}_{j\in R_i}),$$

where $\mathbf{m}$ is a shared learned mask token and $\mathbf{p}_j$ encodes the spatiotemporal position of the j-th token.

Training objective. The V-JEPA loss mirrors the image case:

$$\mathcal{L}_{\text{V-JEPA}} = \frac{1}{M}\sum_{i=1}^{M}\sum_{j\in R_i} \|\hat{\mathbf{z}}_t[j] - \text{sg}(\mathbf{z}_t[j])\|_2^2 . \tag{5}$$

The target encoder is updated via EMA, and the stop-gradient operator is applied to the target representations to prevent collapse. By predicting latent features across both space and time rather than pixels, V-JEPA captures motion, long-range temporal dependencies, and appearance jointly. The high masking ratio (∼90%) is particularly important: masking large contiguous spatiotemporal regions prevents the model from solving the prediction task through simple local interpolation and forces it to reason about scene dynamics.

2.4 V-JEPA 2

V-JEPA 2 (Assran et al., 2025) scales the V-JEPA framework to internet-scale data and model sizes. It retains the core mask-denoising feature prediction objective of V-JEPA but introduces several modifications to enable efficient training at scale.

Mask-denoising objective. The self-supervised objective remains the prediction of masked spatiotemporal regions in representation space. Given a video $\mathbf{v}$ and its masked view $\mathbf{v}_c$, the encoder $E_\theta(\cdot)$ processes the masked input and the predictor $P_\phi(\cdot)$ receives the encoder output concatenated with learnable mask tokens $\Delta\mathbf{v}$ specifying the positions of the masked patches. A key change from V-JEPA is the use of an ℓ_1 loss instead of ℓ_2:

$$\mathcal{L}_{\text{V-JEPA 2}} = \sum_{j \in \mathcal{M}} \left| P_\phi(\Delta\mathbf{v}, E_\theta(\mathbf{v}_c))_j - \text{sg}(E_{\bar{\theta}}(\mathbf{v}))_j \right|_1, \tag{6}$$

where the sum runs over the masked patch indices $\mathcal{M}$, and $\bar{\theta}$ denotes the EMA parameters. The ℓ_1 loss was found to improve training stability at scale. The encoder uses 3D Rotary Position Embeddings (3D-RoPE) for spatiotemporal token encoding, replacing the fixed or learned position embeddings of earlier JEPA variants.

Scaling ingredients. V-JEPA 2 identifies four key ingredients for scaling:

(i) *Data scaling*: The training dataset is expanded to VideoMix22M (VM22M), comprising approximately 22 million video and image samples aggregated from public sources (Something-Something v2, Kinetics, HowTo100M, YT-Temporal-1B, and ImageNet), with a retrieval-based curation strategy for filtering noisy web-scraped content.

(ii) *Model scaling*: The encoder is scaled from 300M parameters (ViT-L) to over 1B parameters (ViT-g).

(iii) *Longer training*: Pretraining is extended to 252K iterations.

(iv) *Progressive resolution*: Training begins with short, low-resolution clips (16 frames at 256×256) during warmup and constant learning rate phases, and increases to longer, higher-resolution clips (64 frames at 384×384) during the cosine decay phase. This progressive strategy reduces GPU time by up to $8.4\times$ compared to training at full resolution throughout, while still capturing long-range temporal dynamics.

Action-conditioned extension. After pretraining, the V-JEPA 2 encoder is frozen and an action-conditioned predictor (V-JEPA 2-AC) can be trained on a small amount of robot interaction data. V-JEPA 2-AC predicts future video embeddings conditioned on robot actions and proprioceptive state, using block-causal attention and a combined teacher-forcing and rollout loss. This enables zero-shot planning via model-predictive control, but we focus on the self-supervised pretraining stage in these notes.

2.5 C-JEPA

C-JEPA (Contrastive-JEPA) (Mo and Tong, 2024) addresses two limitations identified in I-JEPA: (1) the EMA update alone is insufficient to prevent *entire collapse* (where all patch representations converge to the same vector), and (2) the I-JEPA predictor struggles to accurately learn the *mean* of patch representations across augmented views. C-JEPA integrates the Variance-Invariance-Covariance Regularization (VICReg) (Bardes et al., 2022) into the I-JEPA framework. Before presenting C-JEPA, we review SimSiam and VICReg, which provide the theoretical and methodological foundations, and then establish the connections to I-JEPA.

2.5.1 SimSiam

SimSiam (Chen and He, 2021) is a non-contrastive self-supervised method that learns representations by maximizing the similarity between two augmented views of the same image, without requiring negative pairs, momentum encoders, or large batch sizes.

Given an image, two augmented views $\mathbf{x}_1$ and $\mathbf{x}_2$ are generated. Both are fed through a shared encoder f_θ to produce representations $\mathbf{z}_1 = f_\theta(\mathbf{x}_1)$ and $\mathbf{z}_2 = f_\theta(\mathbf{x}_2)$. A prediction MLP h transforms one representation to

match the other. The loss is a symmetrized negative cosine similarity with stop-gradient on the target branch:

$$\mathcal{L}_{\text{SimSiam}} = \frac{1}{2}\left[D(h(\mathbf{z}_1), \text{sg}(\mathbf{z}_2)) + D(h(\mathbf{z}_2), \text{sg}(\mathbf{z}_1))\right], \tag{7}$$

where $D(\mathbf{p}, \mathbf{q}) = -\frac{\mathbf{p}}{\|\mathbf{p}\|_2} \cdot \frac{\mathbf{q}}{\|\mathbf{q}\|_2}$ is the negative cosine similarity. The stop-gradient operator is essential: without it, the loss admits a trivial collapsed solution and the representations degenerate. SimSiam demonstrates that the combination of a prediction head and stop-gradient suffices to prevent collapse without explicit contrastive terms.

When recast in terms of patch-level representations for comparison with I-JEPA, the SimSiam objective can be written as minimizing the distance between predicted and target patch representations across all random patches r_j from augmented views:

$$\mathcal{L}_{\text{SimSiam}} = \frac{1}{|V|}\sum_{i=1}^{|V|}\sum_{j\in P_i}\left\|\mathbf{z}_{r_j} - \mathbf{p}_{r_j}\right\|_2^2, \tag{8}$$

where $|V|$ is the number of augmented views and P_i is the set of random patches in the i-th view.

2.5.2 VICReg

VICReg (Variance-Invariance-Covariance Regularization) (Bardes et al., 2022) prevents collapse in Siamese architectures by directly regularizing the geometry of the representation space, without requiring negative pairs or asymmetric architectures. It introduces three complementary terms operating on the embeddings $\mathbf{z} \in \mathbb{R}^{n\times d}$ (a batch of n vectors of dimension d):

Variance. A hinge loss on the standard deviation of each embedding dimension along the batch ensures that no dimension collapses to a constant:

$$v(\mathbf{z}) = \frac{1}{d}\sum_{j=1}^{d}\max\left(0,\ \gamma - \sqrt{\text{Var}(z^j) + \epsilon}\right), \tag{9}$$

where z^j denotes the j-th dimension of $\mathbf{z}$ across the batch, $\gamma = 1$ is a target standard deviation threshold, and ϵ is a small constant for numerical stability.

Covariance. The covariance term minimizes the squared off-diagonal elements of the covariance matrix, encouraging the embedding dimensions to be decorrelated:

$$c(\mathbf{z}) = \frac{1}{d}\sum_{i\neq j}[\mathbf{C}(\mathbf{z})]_{i,j}^2, \quad \text{where} \quad \mathbf{C}(\mathbf{z}) = \frac{1}{n-1}\sum_{i=1}^{n}(\mathbf{z}_i - \bar{\mathbf{z}})(\mathbf{z}_i - \bar{\mathbf{z}})^\top. \tag{10}$$

This prevents different dimensions from encoding redundant or trivially correlated information.

Invariance. The invariance term minimizes the mean-squared Euclidean distance between embeddings of two views $\mathbf{z}$ and $\mathbf{z}'$ from the same image:

$$s(\mathbf{z}, \mathbf{z}') = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{z}_i - \mathbf{z}_i'\|_2^2. \tag{11}$$

The full VICReg objective is a weighted combination:

$$\mathcal{L}_{\text{VICReg}} = \lambda_s \cdot s(\mathbf{z}, \mathbf{z}') + \lambda_v \cdot [v(\mathbf{z}) + v(\mathbf{z}')] + \lambda_c \cdot [c(\mathbf{z}) + c(\mathbf{z}')], \tag{12}$$

where λ_s, λ_v, and λ_c are weighting coefficients. VICReg provides explicit, interpretable mechanisms to prevent both *complete collapse* (all embeddings identical, prevented by variance) and *dimensional collapse* (embeddings spanning a low-dimensional subspace, prevented by covariance).

2.5.3 Connecting I-JEPA with SimSiam

Mo and Tong (2024) establish a theoretical connection between I-JEPA and SimSiam. Both frameworks minimize the distance between predicted and target representations under an asymmetric architecture with stop-gradient. I-JEPA focuses on the distance between predicted and actual masked patch representations, whereas SimSiam minimizes the distance between two augmented views of the same image. The underlying principle is the same: a predictor network transforms one representation to match a stop-gradient target.

2.5.4 Connecting I-JEPA with VICReg

While the stop-gradient and EMA mechanisms in I-JEPA prevent mode-wise collapse (each individual eigenmode converges to a stable non-zero equilibrium), they do not explicitly prevent *entire collapse*, where all patch representations across a batch converge to the same constant vector. VICReg's variance term directly addresses this failure mode by maintaining sufficient spread in each embedding dimension. Furthermore, the covariance term prevents *dimensional collapse*, where embeddings span only a low-dimensional subspace despite having non-zero variance. Together, these regularization mechanisms complement the implicit collapse prevention provided by the predictor and stop-gradient.

From the perspective of I-JEPA's limitations, the EMA target encoder evolves slowly and may not provide sufficiently diverse gradients to prevent the context encoder from settling into degenerate solutions, particularly early in training or on large, diverse datasets. By adding explicit variance and covariance constraints to the context encoder representations, VICReg provides a direct regularization signal that is independent of the predictor dynamics analyzed above.

Additionally, VICReg's invariance term addresses the second identified limitation of I-JEPA: the predictor's difficulty in learning the mean of patch representations. By enforcing that different masked views of the same image produce similar mean representations, the invariance term provides an explicit learning signal for view consistency that the patch-level ℓ_2 loss alone does not capture.

2.5.5 The C-JEPA Framework

C-JEPA combines the I-JEPA architecture and training procedure with VICReg regularization. The architecture is identical to I-JEPA: a context encoder, a target encoder (EMA-updated), and a predictor operating on masked patch representations. The modification is purely in the loss function.

Overall objective. The C-JEPA loss augments the I-JEPA loss with a weighted VICReg term:

$$\mathcal{L}_{\text{C-JEPA}} = \mathcal{L}_{\text{I-JEPA}} + \beta_{\text{vicreg}} \cdot \mathcal{L}_{\text{VICReg}}, \tag{13}$$

where

$$\mathcal{L}_{\text{VICReg}} = \beta_{\text{sim}} \cdot \mathcal{L}_{\text{sim}} + \beta_{\text{std}} \cdot \mathcal{L}_{\text{std}} + \beta_{\text{cov}} \cdot \mathcal{L}_{\text{cov}}, \tag{14}$$

with coefficients $\beta_{\text{vicreg}} = 0.001$, $\beta_{\text{sim}} = 25$, $\beta_{\text{std}} = 25$, and $\beta_{\text{cov}} = 1$.

Application of VICReg terms. The VICReg terms are computed on the context encoder representations. The context embeddings $\mathbf{z}_c \in \mathbb{R}^{n \times M \times P \times d}$ (batch size n, M mask blocks, P patches per block, embedding dimension d) are first average-pooled over the mask patches, then passed through a projection head. The variance and covariance terms are applied to the projected representations, while the invariance term is computed between pairs of different mask blocks from the same image:

$$\mathcal{L}_{\text{sim}} = \frac{1}{M(M-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^{M} \text{criterion}(\mathbf{z}_c^{(i)}, \mathbf{z}_c^{(j)}),$$

where $\mathbf{z}_c^{(i)}$ denotes the projected context representation for the i-th mask block. This cross-block invariance encourages the context encoder to produce consistent representations regardless of which specific mask pattern is applied.

Role of each component. The I-JEPA loss provides the primary learning signal through patch-level latent prediction. The variance term prevents entire collapse by ensuring that embedding dimensions maintain sufficient spread over the batch. The covariance term prevents dimensional collapse by decorrelating embedding dimensions. The invariance term improves the learning of the mean patch representation by enforcing consistency across different masked views of the same image. The weighting coefficient $\beta_{\text{vicreg}} = 0.001$ ensures that the VICReg terms act as a regularizer rather than dominating the primary I-JEPA objective.

2.6 LLM-JEPA

LLM-JEPA (Huang et al., 2026) adapts the JEPA objective to Large Language Models, addressing a fundamental asymmetry in representation learning: while vision has benefited from embedding-space training objectives (*e.g.* I-JEPA, V-JEPA), language models remain trained exclusively with input-space reconstruction via next-token prediction. LLM-JEPA augments the standard autoregressive loss with an embedding-space prediction term, improving the abstraction capabilities of LLMs without sacrificing their generative capabilities.

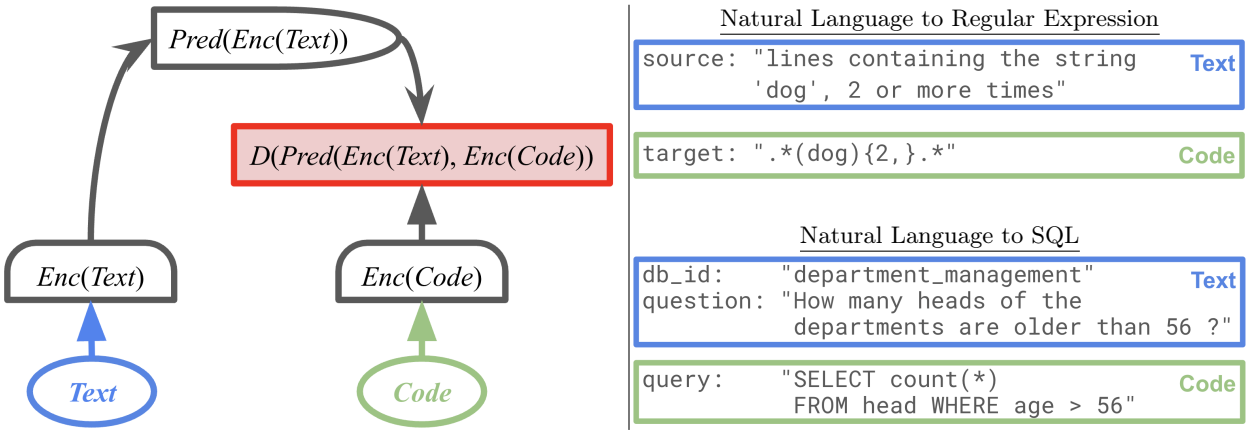


Figure 10 Left: JEPA applied to NLP tasks that has *Text* and *Code*, where *Text* and *Code* are naturally two views of the same thing. **Right**: **(top)**: An illustration of the NL-RX-SYNTH dataset, where each sample consists of a description of the regular expression in natural language (*Text*) and the regular expression itself (*Code*). **(bottom)**: The Spider dataset, where *Text* is the database ID and description of the SQL query and *Code* is the SQL query itself.

Views in language. A key challenge in applying JEPA to language is the construction of meaningful *views*. In vision, views arise naturally from data augmentation (*e.g.* crops, color jitter) or masking. In language, LLM-JEPA exploits tasks where two modalities describe the same underlying knowledge. For instance, a natural language description and its corresponding code (*e.g.* a regular expression, an SQL query, or a code diff) constitute two views of the same semantic content. More generally, any (text, code) pair, question-answer pair, or set of paraphrases can serve as the two views required by the JEPA objective.

The LLM-JEPA objective. Let Text and Code denote the two views. The standard LLM objective is autoregressive next-token prediction:

$$\mathcal{L}_{\text{LLM}}(\text{Text}_{1:\ell-1}, \text{Text}_{\ell}) = \text{XEnt}(\text{Classifier}(\text{Enc}(\text{Text}_{1:\ell-1})),\ \text{Text}_{\ell})\,, \tag{15}$$

where the sum runs over all token positions ℓ via causal autoregression. LLM-JEPA augments this with an embedding-space prediction term:

$$\mathcal{L}_{\text{LLM-JEPA}} = \underbrace{\sum_{\ell=2}^{L} \mathcal{L}_{\text{LLM}}(\text{Text}_{1:\ell-1}, \text{Text}_{\ell})}_{\text{generative capabilities}} + \lambda \cdot \underbrace{d(\text{Pred}(\text{Enc}(\text{Text})),\ \text{Enc}(\text{Code}))}_{\text{abstraction capabilities}}, \tag{16}$$

where $\lambda \geq 0$ balances the two terms, $\text{Enc}(\cdot)$ extracts the hidden state of the last token from the final Transformer layer, $\text{Pred}(\cdot)$ is a predictor network, and $d(\cdot, \cdot)$ is the cosine distance.

Encoder and predictor. Both Text and Code are packed into a single context window with a custom block-causal attention mask that prevents the two views from attending to each other. This allows both embeddings Enc(Text) and Enc(Code) to be computed in a single forward pass. The predictor reuses the internal weights of the LLM by appending k special [PRED] tokens after Text. The self-attention mechanism processes these additional tokens, and the embedding of the last [PRED] token serves as Pred(Enc(Text)). When $k = 0$, the predictor is trivial (*i.e.* identity).

Key properties. Minimizing $\mathcal{L}_{\text{LLM}}$ does not implicitly minimize the JEPA term: the embedding-space prediction loss remains constant during standard next-token prediction training. Conversely, adding the JEPA term does not degrade the next-token prediction capability; the two losses are complementary. The JEPA term acts as a regularizer that induces additional geometric structure in the latent space, confining the mapping from Enc(Text) to Enc(Code) within a narrow, approximately linear subspace. This structured representation underlies the accuracy improvements observed in the generative evaluation setup.

2.7 VL-JEPA

VL-JEPA (Chen et al., 2026) extends the JEPA framework to vision-language tasks, replacing the expensive token-space generation objective of classical Vision Language Models (VLMs) with embedding-space semantic prediction. Instead of autoregressively generating tokens, VL-JEPA predicts continuous embeddings of the target text, focusing on task-relevant semantics while abstracting away surface-level linguistic variability such as word choice, style, or paraphrasing.

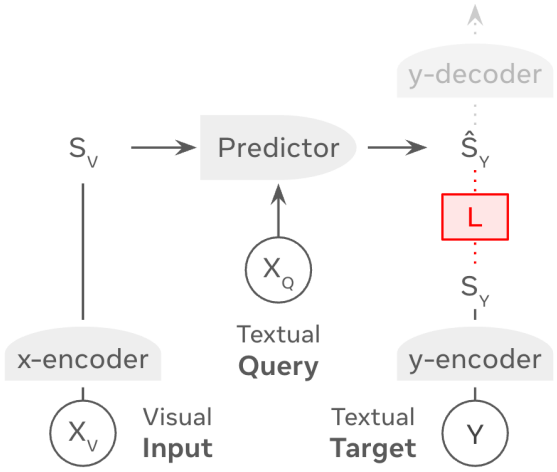


Figure 11 VL-JEPA model architecture.

Architecture. VL-JEPA operates on triplets $\langle \mathbf{X}_V, \mathbf{X}_Q, Y \rangle$, where $\mathbf{X}_V$ is the visual input (image or video frames), $\mathbf{X}_Q$ is a textual query (prompt), and Y is the textual target (answer). The model comprises four components:

(i) **X-Encoder** ($\mathbf{X}_V \mapsto \mathbf{S}_V$): a frozen vision encoder (*e.g.* V-JEPA 2 ViT-L) that maps visual inputs to compact visual embeddings.

(ii) **Predictor** ($\langle \mathbf{S}_V, \mathbf{X}_Q \rangle \mapsto \hat{\mathbf{S}}_Y$): the core component, initialized from Transformer layers (*e.g.* Llama 3), that maps visual embeddings to a prediction of the target embedding, conditioned on the textual query. Bidirectional attention is used so that both visual and query tokens can attend to each other. The predictor outputs are average-pooled and projected into the target embedding space.

(iii) **Y-Encoder** ($Y \mapsto \mathbf{S}_Y$): a text embedding model (*e.g.* EmbeddingGemma) that embeds the textual target into a continuous latent space, serving as the prediction target. The Y-Encoder abstracts away task-irrelevant surface features.

(iv) **Y-Decoder** ($\hat{\mathbf{S}}_Y \mapsto \hat{Y}$): a lightweight text decoder invoked only at inference time, when needed, to translate the predicted embedding back into human-readable text.

Training objective. The training loss is defined in the embedding space:

$$\mathcal{L}_{\text{VL-JEPA}} = D(\hat{\mathbf{S}}_Y, \mathbf{S}_Y), \tag{17}$$

where D combines a prediction alignment term and a regularization term to prevent collapse. VL-JEPA adopts the InfoNCE loss, which decomposes into: (1) a representation alignment term that minimizes the distance between normalized predicted and target embeddings, and (2) a uniformity regularization term that pushes embeddings in a batch apart, preventing collapse. The predictor and Y-Encoder are trained jointly with bidirectional InfoNCE loss. Alternative anti-collapse strategies (*e.g.* VICReg, SIGReg, EMA, or freezing the Y-Encoder) are also compatible with VL-JEPA but are not explored in the original work.

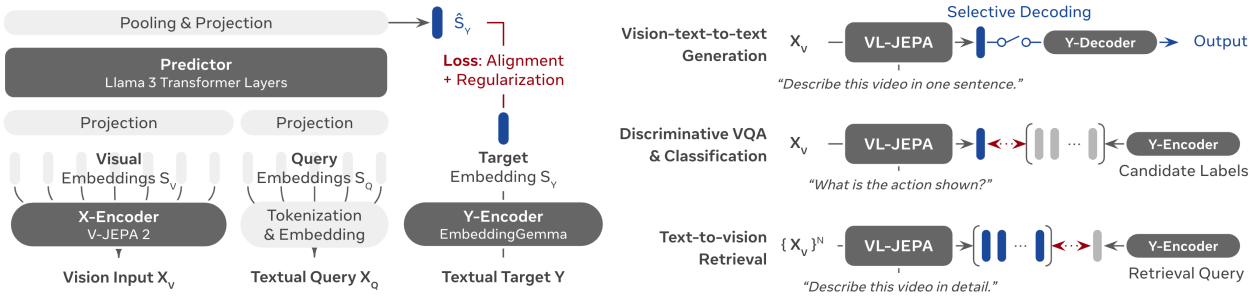


Figure 12 Left: VL-JEPA Architecture. It learns to predict the target embedding S_Y, instead of reconstructing the raw target Y in token space as in classical VLMs. **Right: VL-JEPA Applications**: It handles vision-text-to-text generation tasks (*e.g.,* captioning) with selective decoding mechanism natively supported. Furthermore, VL-JEPA's embedding space facilitates discriminative VQA, open-vocabulary classification and text-to-video retrieval tasks using a single unified model architecture.

Simplified target distribution. A key advantage of embedding-space prediction over token-space prediction is the simplified target distribution. For a given input, multiple plausible textual targets Y may exist (*e.g.* "the lamp is turned off" and "room will go dark" are both valid answers). In the one-hot token space, these are nearly orthogonal since they share no overlapping tokens. However, in the embedding space produced by the Y-Encoder, semantically equivalent targets are mapped to nearby points, yielding a compact unimodal distribution. The model no longer needs to fit multiple disjoint high-density regions in sparse token space, but only a single coherent mode in a continuous embedding space.

Multi-task capability. VL-JEPA supports diverse tasks within a single unified architecture without modification. For vision-text-to-text generation (*e.g.* captioning, open-ended VQA), the predicted embedding $\hat{\mathbf{S}}_Y$ is decoded into text by the Y-Decoder. For open-vocabulary classification and discriminative VQA, candidate label texts are encoded by the Y-Encoder and compared with $\hat{\mathbf{S}}_Y$ to select the nearest match. For text-to-video retrieval, candidate videos are mapped to predicted embeddings using a captioning prompt, and then ranked by similarity to the encoded textual query.

Selective decoding. The non-autoregressive nature of VL-JEPA enables a native selective decoding mechanism for real-time streaming applications. Since the model produces a continuous stream of semantic embeddings $\hat{\mathbf{S}}_Y$ via a single forward pass (without autoregressive decoding), this stream can be monitored in real time and decoded into text only when a significant semantic shift is detected (*e.g.* when the local window variance exceeds a threshold). This decouples continuous semantic monitoring from expensive text generation, enabling both responsiveness and efficiency for online video applications.

3 Regularizers

As discussed in the introduction, a central challenge in training JEPAs is preventing representation collapse while solving the predictive task. The choice of regularizer determines what geometric and distributional properties the embedding space will exhibit, and different regularizers encode different inductive biases. In this section, we review three regularization strategies of increasing theoretical sophistication: VICReg, which constrains the first two moments of the embedding distribution; SIGReg (used in LeJEPA), which enforces the full isotropic Gaussian distribution via characteristic function matching; and RDMReg (used in Rectified LpJEPA), which generalizes to sparse, non-negative distributions via the Rectified Generalized Gaussian family.

3.1 VICReg

VICReg (Variance-Invariance-Covariance Regularization) (Bardes et al., 2022) was introduced in Section 2.5.2 as a component of C-JEPA. Here we discuss its theoretical properties and limitations as a standalone regularizer for JEPAs.

VICReg prevents collapse by constraining the first two moments of the embedding distribution. The variance term ensures $\mathrm{Var}(z^j) \geq \gamma$ for each dimension j, preventing complete collapse. The covariance term pushes $\mathrm{Cov}(z^i, z^j) \to 0$ for $i \neq j$, preventing dimensional collapse. When both terms are minimized, the embedding covariance matrix approaches a scaled identity, $\mathrm{Cov}(\mathbf{z}) \approx \gamma^2 \mathbf{I}_d$.

Limitations. While VICReg is effective in practice, it suffers from several theoretical limitations. First, VICReg is *under-specified*: matching the first two moments (mean zero and identity covariance) does not uniquely determine a distribution. Many non-Gaussian distributions satisfy these constraints while exhibiting undesirable properties such as heavy tails, multimodality, or concentration on low-dimensional manifolds. The embeddings can satisfy all VICReg constraints while still being far from the ideal distribution for downstream tasks. Second, VICReg has *quadratic complexity* in the embedding dimension d: the covariance matrix $\mathbf{C} \in \mathbb{R}^{d \times d}$ requires $\mathcal{O}(d^2)$ computation and memory, which becomes prohibitive for high-dimensional embeddings. Third, VICReg lacks *theoretical guarantees* about the optimality of the resulting embedding distribution for downstream tasks. The choice of variance threshold γ and the relative weighting of the three terms introduce multiple hyperparameters that require tuning. These limitations motivate the search for a regularizer that enforces a provably optimal distribution with stronger statistical guarantees and better scaling properties.

3.2 LeJEPA and SIGReg

LeJEPA (Latent-Euclidean JEPA) (Balestriero and LeCun, 2025) addresses the limitations of existing JEPA regularizers by deriving, from first principles, the optimal distribution that JEPA embeddings should follow to minimize downstream prediction risk, and then introducing a novel regularizer to enforce it.

3.2.1 Theoretical Foundation: Why Isotropic Gaussian?

The central theoretical contribution of LeJEPA is the proof that the **isotropic Gaussian** $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ uniquely minimizes the expected downstream prediction risk across broad families of tasks. This result holds for both linear and nonlinear probing.

Linear probing. Consider a frozen encoder f_θ producing d-dimensional embeddings $\mathbf{z} = f_\theta(\mathbf{x})$. A linear probe $\mathbf{w} \in \mathbb{R}^d$ is trained to predict some downstream target y by minimizing $\mathbb{E}[(\mathbf{w}^\top \mathbf{z} - y)^2]$. The optimal probe is $\mathbf{w}^* = \mathrm{Cov}(\mathbf{z})^{-1}\mathrm{Cov}(\mathbf{z}, y)$, and the resulting risk depends on the covariance structure of $\mathbf{z}$.

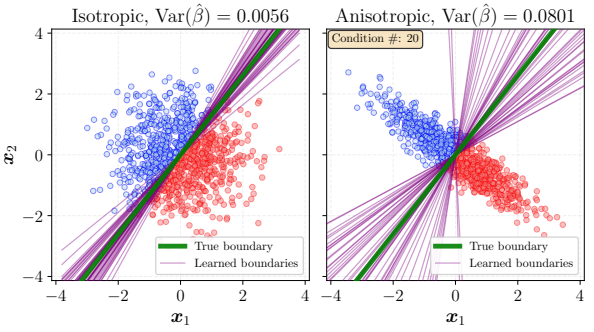


Figure 14 Anisotropic (**right**) embeddings lead to higher variance estimator compared to isotropic embeddings (**left**).

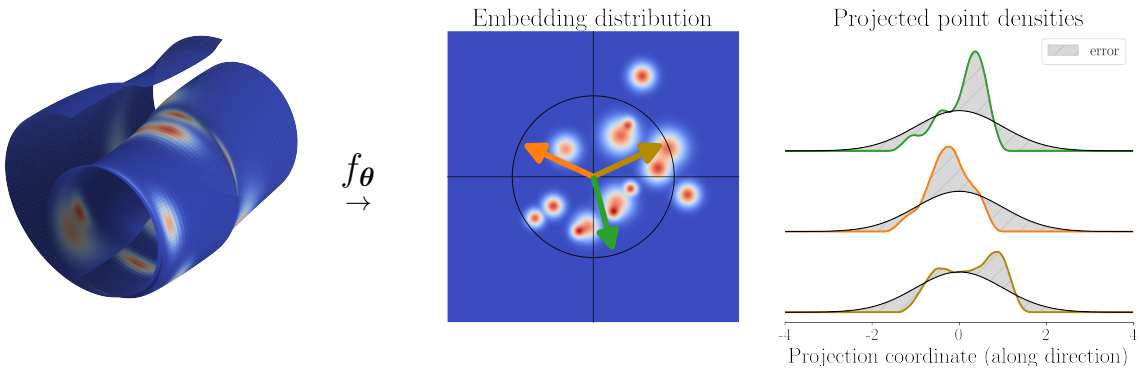


Figure 13 Sketched Isotropic Gaussian Regularization (SIGReg): Given some arbitrary input data with density p_x with support that may or may not lie on a manifold (**left**), a deep network encoder (f_θ) produces embeddings $\mathbf{z} = f_\theta(\mathbf{x})$ with some distribution $\mathbf{z} \sim p_z$ (**middle**). The SIGReg objective pushes p_z to match a target distribution p_t by projecting the embeddings along 1d directions (**middle, arrows**) and enforcing that the univariate densities (**right, colored lines**) match the distribution of p_t, projected along the same directions. Any popular statistical test can assess the goodness-of-fit; in practice, characteristic function tests are preferred. By using SIGReg with p_t isotropic Gaussian (**right, black lines**), LeJEPA introduces a lean and provably optimal JEPA, free of numerous heuristics and able to produce competitive performances.

Balestriero and LeCun (2025) show that, when the downstream task is unknown at pretraining time and one considers the expected risk over a uniform distribution of possible tasks, the optimal embedding distribution is the one that makes all directions in embedding space equally useful. This is uniquely achieved by the isotropic Gaussian: any anisotropy in the covariance favors some task directions over others, increasing the expected risk.

Nonlinear probing. The result extends to nonlinear probes, including k-nearest-neighbor (k-NN) classifiers and kernel methods. For k-NN, the expected risk under a uniform prior over label assignments depends on the local density of the embeddings. An isotropic Gaussian distribution ensures that the local density is as uniform as possible across directions, minimizing the expected misclassification rate. For kernel probes, the analysis proceeds through the spectral decomposition of the kernel matrix, and the isotropic Gaussian again emerges as optimal.

Geometric intuition. The isotropic Gaussian is the maximum-entropy distribution subject to a second-moment (energy) constraint $\mathbb{E}[\|\mathbf{z}\|^2] = d$. By maximizing entropy, the embedding space is as "spread out" as possible, making no direction preferentially informative. This is precisely the property needed when the downstream task is unknown: the encoder should not commit to any particular subspace but should distribute information equally across all dimensions. Furthermore, the isotropic Gaussian prevents both complete collapse (all embeddings identical, which would have zero entropy) and dimensional collapse (embeddings confined to a low-dimensional subspace, which would have entropy strictly less than the Gaussian maximum).

3.2.2 SIGReg: Sketched Isotropic Gaussian Regularization

Given that the isotropic Gaussian is the optimal target distribution, the practical challenge is to design a loss that enforces it reliably at scale. Balestriero and LeCun (2025) introduce **Sketched Isotropic Gaussian Regularization (SIGReg),** a distribution-matching objective based on random projections and characteristic function matching.

The Cramér-Wold theorem. The theoretical basis of SIGReg is the Cramér-Wold theorem, which states that a multivariate distribution p_z in $\mathbb{R}^d$ is uniquely determined by all of its one-dimensional projections. That is, $p_z = p_t$ if and only if $\mathbf{c}^\top \mathbf{z} \sim \mathbf{c}^\top \mathbf{t}$ for all unit vectors $\mathbf{c} \in \mathbb{S}^{d-1}$. This reduces the problem of matching a d-dimensional distribution to matching infinitely many univariate distributions. In practice, a finite set of S

random projection directions $\{\mathbf{c}_s\}_{s=1}^{S}$ are sampled uniformly on $\mathbb{S}^{d-1}$, and the matching is enforced along each direction.

Why not moments? A natural approach would be to match the moments of the projected distributions. However, moment matching is *insufficient*: it does not uniquely identify a distribution (infinitely many distributions share the same finite set of moments). Moreover, sample estimates of higher-order moments exhibit high variance, making moment-based objectives unstable in practice. VICReg can be seen as matching only the first two moments (mean and covariance), which is why it is under-specified.

Characteristic function matching. SIGReg instead matches distributions via their *characteristic functions*. The characteristic function of a random variable Z is $\varphi_Z(t) = \mathbb{E}[e^{itZ}]$, and it uniquely determines the distribution of Z. For the standard Gaussian $\mathcal{N}(0,1)$, the characteristic function is $\varphi(t) = e^{-t^2/2}$. SIGReg is based on the Epps-Pulley test statistic, which measures the discrepancy between the empirical characteristic function of the projected embeddings and the Gaussian characteristic function. For a batch of n embeddings projected along direction $\mathbf{c}$, producing scalars $\{u_i = \mathbf{c}^\top \mathbf{z}_i\}_{i=1}^{n}$, the SIGReg loss along direction $\mathbf{c}$ takes the form:

$$\mathcal{L}_{\text{SIGReg}}^{(\mathbf{c})} = \sum_{k=1}^{K} \left| \hat{\varphi}_n(t_k) - e^{-t_k^2/2} \right|^2, \tag{18}$$

where $\hat{\varphi}_n(t_k) = \frac{1}{n}\sum_{i=1}^{n} e^{it_k u_i}$ is the empirical characteristic function evaluated at test frequencies $\{t_k\}_{k=1}^{K}$, and $e^{-t_k^2/2}$ is the target Gaussian characteristic function. The full SIGReg loss averages over S random projection directions:

$$\mathcal{L}_{\text{SIGReg}} = \frac{1}{S}\sum_{s=1}^{S} \mathcal{L}_{\text{SIGReg}}^{(\mathbf{c}_s)}. \tag{19}$$

Computational properties. SIGReg has several favorable properties. Its time and memory complexity is $\mathcal{O}(n \cdot S \cdot K)$, which is *linear* in both the batch size n and the embedding dimension d (since S and K are small constants, typically $S = d$ and K a few dozen). This is in contrast to VICReg's $\mathcal{O}(n \cdot d^2)$ covariance computation. SIGReg's gradients are bounded, ensuring training stability. The random projections can be precomputed and shared across the batch, making the implementation distributed-training-friendly.

Relation to VICReg. Balestriero and LeCun (2025) show that VICReg can be recovered as a special case of SIGReg when only the first two moments of the characteristic function expansion are retained. Specifically, matching the characteristic function at $t \to 0$ recovers the mean constraint, and matching at small t recovers the variance and covariance constraints. SIGReg strictly generalizes VICReg by matching the full distribution, not just its first two moments.

3.2.3 The LeJEPA Objective

LeJEPA combines the JEPA predictive loss with SIGReg:

$$\mathcal{L}_{\text{LeJEPA}} = \mathcal{L}_{\text{pred}} + \lambda \cdot \mathcal{L}_{\text{SIGReg}}, \tag{20}$$

where $\mathcal{L}_{\text{pred}}$ is the standard embedding prediction loss (*e.g.* ℓ_2 distance between predicted and target embeddings across views) and $\lambda > 0$ is a single trade-off hyperparameter. LeJEPA requires no stop-gradient, no teacher-student network, no EMA schedule, and no asymmetric architecture. The SIGReg term alone prevents collapse by construction: any collapsed distribution (constant embeddings or low-dimensional embeddings) has characteristic functions that differ markedly from the Gaussian, producing a large SIGReg loss. The implementation requires approximately 50 lines of code and is stable across architectures (ResNets, ViTs, ConvNets), scales (up to ViT-g with 1.8B parameters), and domains (natural images, satellite imagery, galaxies, medical images).

3.3 Rectified LpJEPA and RDMReg

Rectified LpJEPA (Kuang et al., 2026) extends the distribution-matching framework of LeJEPA to enforce *sparse* and *non-negative* representations. While LeJEPA's isotropic Gaussian target produces dense embeddings where all dimensions are active for every input, biological and computational evidence suggests that sparse representations, where only a small fraction of dimensions are active, are more efficient and interpretable.

3.3.1 Motivation: Sparsity and Maximum Entropy

Sparse representations have a long history as an organizing principle of efficient information processing in both neuroscience and signal processing. Beyond interpretability, sparsity offers practical benefits: sparse activations reduce computation during inference, enable more efficient storage, and improve robustness to noise. However, existing JEPA regularizers either ignore sparsity entirely (VICReg, SIGReg) or enforce it via ad-hoc penalties that lack distributional guarantees.

The challenge is to design a target distribution that simultaneously satisfies three desiderata: (1) *maximum entropy* (to prevent collapse and ensure diverse representations), (2) *controllable sparsity* (to enable explicit control over the expected ℓ_0 norm), and (3) *non-negativity* (to ensure that sparse dimensions are interpretable as "active" or "inactive").

3.3.2 The Rectified Generalized Gaussian Distribution

The isotropic Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ is the maximum-entropy distribution subject to a second-moment constraint $\mathbb{E}[\|\mathbf{z}\|_2^2] = d$. More generally, the *Generalized Gaussian* distribution is the maximum-entropy distribution subject to an expected ℓ_p norm constraint $\mathbb{E}[\|\mathbf{z}\|_p^p] = c$ for any $p > 0$. Its density is:

$$p(\mathbf{z}; p) \propto \exp\left(-\beta \|\mathbf{z}\|_p^p\right) = \prod_{j=1}^{d} \exp(-\beta |z_j|^p),$$

where $\beta > 0$ is a scale parameter. For $p = 2$, this recovers the Gaussian; for $p = 1$, it gives the Laplace distribution (which is already sparser than the Gaussian). As $p \to 0$, the distribution concentrates on increasingly sparse vectors.

To additionally enforce non-negativity, Kuang et al. (2026) introduce the **Rectified Generalized Gaussian** (RGG) distribution by applying a rectification (ReLU) to the Generalized Gaussian:

$$z_j^+ = \max(0, z_j), \quad z_j \sim \text{GenGaussian}(0, \beta, p).$$

The rectification introduces exact zeros with probability $\Pr(z_j^+ = 0) = 1/2$, providing explicit ℓ_0 norm control: the expected fraction of active dimensions is $\mathbb{E}[\|\mathbf{z}^+\|_0]/d = 1/2$. By combining the shape parameter p (which controls the tail behavior and concentration) with rectification (which introduces exact sparsity), RGG enables simultaneous control over both the ℓ_p norm constraint and the ℓ_0 sparsity level. The RGG family preserves the maximum-entropy property (up to rescaling) under the expected ℓ_p norm constraint while additionally enforcing non-negativity.

3.3.3 RDMReg: Rectified Distribution Matching Regularization

A key technical challenge in enforcing the RGG distribution is that, unlike the isotropic Gaussian, the RGG family is *not closed under linear projections*. A linear combination of independent RGG random variables does not follow an RGG distribution (by the Central Limit Theorem, it tends toward a Gaussian for large d). This means that SIGReg's one-sample approach, which matches projected marginals to an analytical target characteristic function, cannot be directly applied.

Kuang et al. (2026) resolve this by introducing Rectified Distribution Matching Regularization (RDMReg), a *two-sample* sliced distribution-matching loss. Instead of comparing projected embeddings to an analytical target, RDMReg generates a batch of *target samples* $\{\mathbf{y}_i\}_{i=1}^n$ drawn from the desired RGG distribution,

projects both the embeddings $\{\mathbf{z}_i\}$ and the target samples $\{\mathbf{y}_i\}$ along the same random directions $\{\mathbf{c}_s\}_{s=1}^{S}$, and minimizes the discrepancy between the two sets of projected scalars. Concretely, the RDMReg loss is:

$$\mathcal{L}_{\text{RDMReg}} = \frac{1}{S}\sum_{s=1}^{S} \mathcal{W}_2^2\big(\{\mathbf{c}_s^\top \mathbf{z}_i\}_{i=1}^{n},\ \{\mathbf{c}_s^\top \mathbf{y}_i\}_{i=1}^{n}\big)\,, \tag{21}$$

where $\mathcal{W}_2^2$ denotes the squared Sliced 2-Wasserstein distance between two sets of one-dimensional samples. In one dimension, the Wasserstein distance admits a closed-form solution: sort both sets of scalars and compute the average squared difference between corresponding order statistics.

3.3.4 The Rectified LpJEPA Objective

Rectified LpJEPA combines the JEPA predictive loss with RDMReg:

$$\mathcal{L}_{\text{LpJEPA}} = \mathcal{L}_{\text{pred}} + \lambda \cdot \mathcal{L}_{\text{RDMReg}}. \tag{22}$$

Rectified LpJEPA strictly generalizes LeJEPA: when $p = 2$ and no rectification is applied, the RGG distribution reduces to the isotropic Gaussian, and RDMReg recovers a two-sample variant of SIGReg. As p decreases toward zero, the target distribution becomes increasingly sparse, enabling a continuous sparsity-performance trade-off controlled by the single parameter p. The framework thus provides a unified perspective on JEPA regularization: the choice of target distribution encodes an inductive bias (dense vs. sparse, Gaussian vs. heavy-tailed), and the distribution-matching machinery (random projections + statistical divergence) provides a scalable, principled mechanism to enforce it.

References

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. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15619–15629, 2023.

Mahmoud Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. In *International conference on machine learning*, pages 1298–1312. PMLR, 2022.

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics, 2025. https://arxiv.org/abs/2511.08544.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022. https://openreview.net/forum?id=xm6YD62D1Ub.

Adrien Bardes, Quentin Garrido, Jean Ponce, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

Delong Chen, Mustafa Shukor, Théo Moutakanni, Willy Chung, Lei Yu, Tejaswi Kasarla, Allen Bolourchi, Yann LeCun, and Pascale Fung. VL-JEPA: Joint embedding predictive architecture for vision-language. In *The Fourteenth International Conference on Learning Representations*, 2026. https://openreview.net/forum?id=tjimrqc2BU.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. https://openreview.net/forum?id=YicbFdNTTy.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. https://arxiv.org/abs/1406.2661.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009, June 2022.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786): 504–507, 2006. doi: 10.1126/science.1127647. https://www.science.org/doi/abs/10.1126/science.1127647.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. https://arxiv.org/abs/2006.11239.

Hai Huang, Yann LeCun, and Randall Balestriero. LLM-JEPA: Large language models meet joint embedding predictive architectures. In *The Fourteenth International Conference on Learning Representations*, 2026. https://openreview.net/forum?id=GbXKPo9QfH.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022. https://arxiv.org/abs/1312.6114.

Yilun Kuang, Yash Dagade, Tim G. J. Rudner, Randall Balestriero, and Yann LeCun. Rectified lpjepa: Joint-embedding predictive architectures with sparse and maximum-entropy representations, 2026. https://arxiv.org/abs/2602.01456.

Yann LeCun et al. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1): 1–62, 2022.

Yanghao Li, Haoqi Fan, Ronghang Hu, Christoph Feichtenhofer, and Kaiming He. Scaling language-image pre-training via masking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 23390–23400, 2023.

Shentong Mo and Shengbang Tong. Connecting joint-embedding predictive architecture with contrastive self-supervised learning. *Advances in neural information processing systems*, 37:2348–2377, 2024.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mahmoud Assran, Koustuv Sinha, Michael Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-jepa 2.1: Unlocking dense features in video self-supervised learning. *arXiv preprint arXiv:2603.14482*, 2026.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. https://openreview.net/forum?id=AhccnBXSne.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2019. https://arxiv.org/abs/1807.03748.

Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14549–14560, 2023.