Video Understanding Proposal: Video Mamba As A World Model

Raif T. Olson
Department of Computer Science
Dartmouth College
Hanover, NH 03755
`raif.t.olson.24@dartmouth.edu`

1 Abstract

Recent advancements in video understanding models, such as Video-MAE, have demonstrated the effectiveness of pre-training vision encoders using masking strategies, leading to high-quality representations for downstream tasks. However, these methods require extensive computational resources, including hundreds of thousands of images and videos, and millions of training steps, making them inaccessible to many researchers. Moreover, transformer-based models like ViViT face challenges with scalability due to their quadratic memory and computational requirements.

To address these issues, structured state-space sequence (S4) models, like Mamba, have been developed in the NLP domain. These models parallelize input processing for improved speed and scale linearly with input sequence length, providing significant advantages over transformers. Recently, S4 models have been adapted for vision tasks, as seen in Vision Mamba.

In our work, we propose an efficient alternative training method that combines the Joint Embedding Predictive Architecture (JEPA) with the Video Mamba model. JEPA pre-training, introduced by Assran et al., significantly reduces training epochs while producing high-quality encoders. By integrating JEPA with the computationally efficient Mamba architecture, we aim to create a high-performance, efficient vision encoder. Additionally, we investigate whether the architectural limitations of Mamba affect its performance in patch-based video processing.

`https://github.com/rolson24/videoMamba-jepa`

2 Introduction

Recent work in papers like Video-MAE has found that pre-training vision encoders like ViT[4] and ViViT[1] using masking strategies leads to high quality representations that can then be fine-tuned to achieve state of the art on downstream tasks. [9] Unfortunately, these methods require hundreds of thousands of images and videos for training, as well as hundreds of millions of training steps for these models to fully converge[3], making these methods out of reach for researchers without huge computational budgets. Additionally, the memory and computational requirements of transformers scales quadratically with sequence length, prohibiting the ability of models like ViViT to scale to hours-long videos.

Luckily, developments made in NLP structured state-space sequence (S4) models like Mamba [5] have allowed them to parallelize the processing of inputs for improved speed, while also achieving performance on-par with transformer-based architectures. These models have the benefit of scaling their computational requirements linearly with the input sequence length, providing a huge boost over transformers. Very recently these structured state-space models have been adapted to vision tasks by L. Zhu, et al. who introduced Vision Mamba [10].

In this work, we propose an efficient alternative training method to ViM [10] and Video Mamba[7], and explore Video Mamba's capability of learning video representations on its own. We apply the JEPA pre-training task to Video Mamba to combine both the training efficiency of JEPA with the computational efficiency of the Mamba architecture in an attempt to create high performance, efficient vision encoder. We also explore if the architectural limitations of Mamba fundamentally hinder its performance in patch-based video processing.

3 Related Work

3.1 Vision Transformers

Existing methods for creating pre-trained models that can learn useful representations of videos and images require large amounts of computational resources that most researchers do not have access to. Recent innovations in training strategies like image patch masking first implemented in the VideoMAE paper [9] have greatly improved video transformers' ability to learn useful feature representations of images and videos from self-supervised pre-training data. These innovations have allowed universal video encoder models to be used for all kinds of downstream tasks by simply fine-tuning a linear probing layer on top of the pre-trained backbone [2]. Unfortunately, these methods of pre-training transformer architectures like ViT and ViViT often require hundreds of millions of images and videos to match the performance of comparable CNN's like ResNet [1][4].

3.2 Mamba

An alternative to attention-base transformers has recently been found to match transformers in performance in NLP tasks at nearly half the number of parameters and 2/3rds the computational complexity called structures state-space (S4) models [6]. A specific architecture of the S4 model, called Mamba [5], has been leading these remarkable results in NLP. These models are a type of recurrent neural network inspired by continuous state-space equations to model systems through time. They use a simple continuous time-invariant model with 4 learnable parameters: (Δ, A, B, C).

$$h'(t) = Ah(t) + Bx(t) \tag{1}$$

$$y(t) = Ch(t) \tag{2}$$

$$\tag{3}$$

They then discretize the model from the continuous parameters of (Δ, A, B) by using the zero-order hold method defined below.

$$\overline{A} = \exp(\Delta A) \tag{4}$$

$$\overline{B} = (\Delta A)^{-1}(\exp(\Delta A) - I) \cdot \Delta B \tag{5}$$

$$h_t = \overline{A}h_{t-1} + \overline{B}x_t \tag{6}$$

$$y_t = Ch_t \tag{7}$$

Recently, L. Zhu, et al. [10] introduced the Vision Mamba (Vim) architecture for use with images. They took the Mamba implementation and adapted it to process flattened image patches. The Vim paper found that the same performance and computational complexity scaling for Mamba seen in NLP also seems to transfer to the image domain. VMamba [8], and VideoMamba [7] have built off Vim's success in an attempt to scale their models up, but ran into issues with over-fitting their training. Both of these papers exhibit over-fitting on their largest models, indicating a need for a larger pre-training dataset.

3.3 Joint Embedding Predictive Architectures

Existing work has shown that aligning the representation space of a Vision Mamba encoder to a frozen CLIP-ViT encoder on unmasked image tokens can be effective. However, both Liu et al. and [8] Li et al. [7] found that this architecture did not scale well to larger model sizes due to their use of smaller video datasets, leading to overfitting issues.

In an attempt to address these challenges, we explored the V-JEPA (Video-based Joint-Embedding Predictive Architecture) training method recently introduced by Bardes et al. [3]. V-JEPA builds upon the principles of masked modeling from VideoMAE, which already employs masked video modeling for video analysis, but specifically enhances it for improved training efficiency. Unlike VideoMAE, which focuses on reconstructing the original video frames from masked inputs, V-JEPA predicts the masked regions in the representation space. This approach reduces the computational burden and accelerates convergence, making the training process more efficient and effective.

One significant benefit of V-JEPA is its potential to reduce the likelihood of overfitting by shortening the training time required on larger datasets. Traditional masked autoencoder training sessions for vision encoders, such as those used in VideoMAE, typically require extensive computational resources and prolonged training periods [9][3]. These extended durations make training larger Video Mamba models out of reach for computationally limited researchers.

V-JEPA addresses these issues with a unique training mechanism. By using a context encoder to process partially visible video frames and a target encoder to provide 'ground truth' representations, the predictor learns to infer missing information efficiently. This setup not only stabilizes the learning process but also accelerates convergence. The target encoder, being a moving weighted average of the context encoder, ensures stable and consistent learning.

4 Proposed Approach

Given the challenges associated with traditional masked autoencoder training methods, we integrated the V-JEPA (Video-based Joint-Embedding Predictive Architecture) [3] training structure with the VideoMamba architecture. V-JEPA builds upon the principles of masked modeling from VideoMAE [9], enhancing it for improved training efficiency. Unlike VideoMAE, which reconstructs the original video frames from masked inputs, V-JEPA predicts masked regions in the representation space, reducing computational burden and accelerating convergence.

Our approach leverages the computational efficiency of the Mamba model while benefiting from the rapid and robust representation learning facilitated by V-JEPA. This combination aims to yield high-performance vision encoders that require fewer training iterations and are less prone to overfitting, even when trained on large-scale video datasets.

4.1 VideoMamba Model

4.1.1 Introduction

The VideoMamba model builds upon the original Vision Mamba (ViM) model created by L. Zhu et al., extending it specifically for video understanding. This model introduces a purely State Space Model (SSM)-based architecture, addressing both local redundancy and global dependencies. By leveraging the Selective State Space (S6) model for its core operations, VideoMamba efficiently handles long video sequences with linear complexity. Designed to process spatiotemporal information effectively, VideoMamba is well-suited for tasks that require detailed video analysis. The following paragraphs will delve into the specific components of the VideoMamba model, including its 3D convolutional patch embedding, position embeddings, Bidirectional Mamba blocks, classification head, and spatiotemporal scans, highlighting how these elements work together to achieve robust video representation learning.

4.1.2 V-JEPA Masking Strategy

In our implementation of the VideoMamba model, we utilize the 3D Multi-Block Masking strategy from the V-JEPA model to enhance the training process. This strategy involves sampling several spatially continuous blocks with various aspect ratios from the video frames and taking their union to create a single mask, which is then repeated across the entire temporal dimension. This approach reduces information leakage due to spatial and temporal redundancy, making the prediction task more challenging and improving model robustness. We use two types of masks: short-range masks, generated by sampling eight blocks with a spatial scale of 0.15, and long-range masks, created by sampling two blocks with a spatial scale of 0.7. The aspect ratio for these blocks is randomly selected within the range of 0.75 to 1.5, resulting in a masking ratio of approximately 90%.

The original VideoMamba paper found that masking an entire row of each image performed better because the Mamba model has an inductive bias towards sequential information. However, Assran

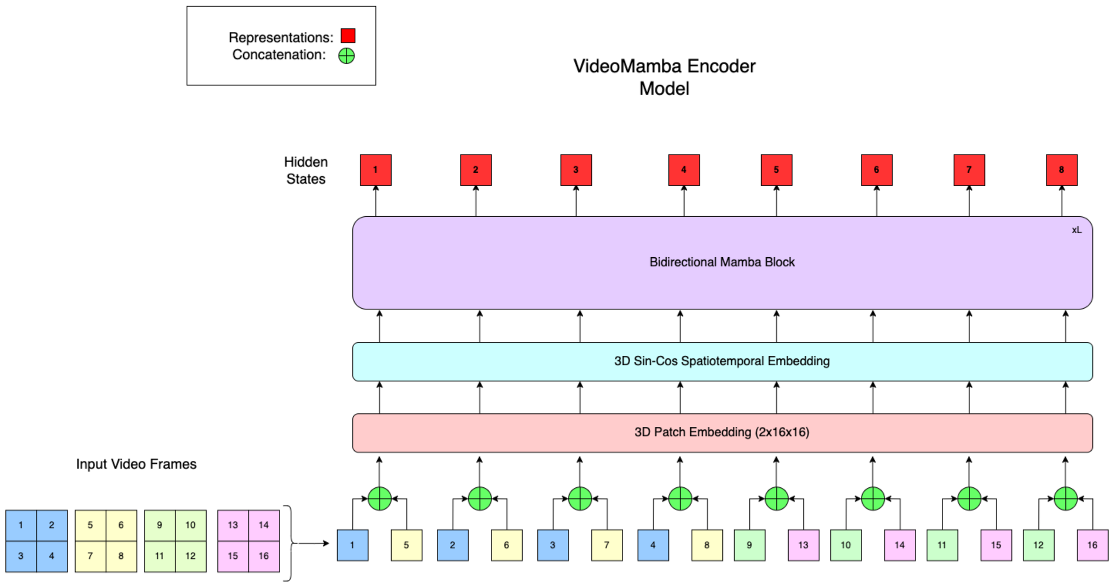
Representations:
Concatenation:
VideoMamba Encoder Model
Hidden States
Bidirectional Mamba Block
xL
3D Sin-Cos Spatiotemporal Embedding
3D Patch Embedding (2x16x16)
Input Video Frames

Figure 1: The figure illustrates the overall architecture of the VideoMamba encoder model. The process begins with the input video frames, which are divided into non-overlapping spatiotemporal patches using a 3D convolutional layer with a kernel size of $2 \times 16 \times 16$. These patches are then embedded using 3D sin-cos spatiotemporal embeddings to retain positional information across space and time. The embedded patches are processed through multiple layers of Bidirectional Mamba Blocks, which handle both forward and backward paths for robust spatiotemporal feature extraction. The output of these blocks forms the hidden states, which can be used for downstream tasks. This architecture leverages the linear complexity and efficiency of the Selective State Space Model (SSM) to handle long video sequences effectively.

et al. found in [2] the paper introducing the original Image JEPA, the multi-block masking strategy significantly outperformed other masking strategies such as random masking and row masking when training a vision transformer. Meanwhile, the VideoMamba paper showed only a 2 % performance difference between masking strategies, we opted for the V-JEPA 3D Multi-Block Masking to leverage its proven effectiveness. This strategy encourages the model to learn both fine-grained local features and broader global patterns, leading to a comprehensive understanding of the video content.

4.1.3 3D Convolutional Patch Embedding

The first step in the VideoMamba model involves converting input video frames into a sequence of non-overlapping spatiotemporal patches. This is achieved using a 3D convolutional layer with a kernel size of $2 \times 16 \times 16$, slightly different from the original $1 \times 16 \times 16$ used in the VideoMamba paper to align with the V-JEPA masking strategy. The $2 \times 16 \times 16$ kernel captures both spatial and temporal information by processing blocks of two consecutive frames at a time, creating patches that retain essential details across both dimensions. These patches are then transformed into embedded tokens, serving as the input for subsequent stages of the model, and a visual can be seen in figure 3. This approach ensures that the model efficiently handles video data, leveraging spatiotemporal coherence for more accurate feature extraction.

4.1.4 Position Embeddings

Since VideoMamba is a sequence model, it can only process patches in a sequence and so has only 1 dimension of positional understanding built into the model. To improve the model's spatial and temporal understanding, we implement positional embeddings to give the model information about where patches are in an image or video.

In our VideoMamba model, we employ 3D sin-cos positional embeddings for both spatial and temporal dimensions, instead of learnable positional embeddings. This decision aligns with the

V-JEPA architecture, where positional embeddings are applied to unmasked tokens in the context encoder and to the masked tokens in the predictor model as can be seen in figure 1. Using 3D sin-cos embeddings ensures consistency and stability, as these embeddings do not need to be learned from two different submodels simultaneously. The sin-cos embeddings capture positional information inherently, avoiding potential conflicts and instability that might arise from having to train learnable embeddings across different model components. This approach effectively encodes spatial and temporal positions, facilitating the model's ability to understand and process video sequences.

4.1.5 Bidirectional Mamba Blocks

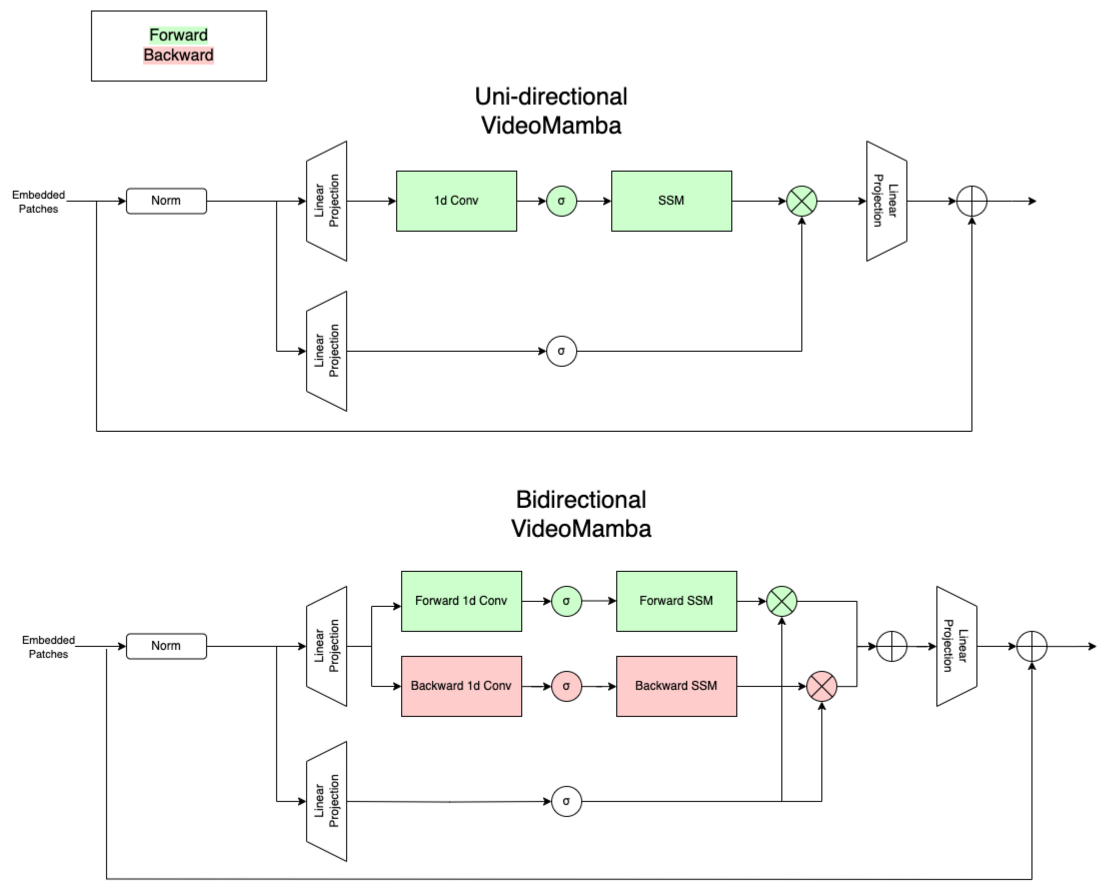

Figure 2: (a) Unidirectional VideoMamba Block: The unidirectional block processes embedded patches through a 1D convolutional layer followed by a Selective State Space Model (SSM). The output is then linearly projected and combined with the original input after normalization. (b) Bidirectional VideoMamba Block: The bidirectional block enhances the processing by incorporating both forward and backward paths. Embedded patches are processed by forward and backward 1D convolutional layers and corresponding SSMs. The outputs from the forward and backward paths are then combined and linearly projected, which helps capture both past and future context for more robust video representation.

The core of the VideoMamba model consists of Bidirectional Mamba Blocks, implemented by incorporating a second state-space model in each block that processes the sequence of frames in reverse. This bidirectional approach, first introduced in Vision Mamba by Zhu et al. [10], addresses the inductive bias present in state-space models regarding the order of inputs in the sequence. By processing the video frames both forwards and backwards, the model reduces the likelihood of losing track of the beginning or end of the sequence. This dual-pass mechanism allows the Bidirectional Mamba Blocks to capture comprehensive contextual information, enhancing the model's ability to understand and represent video content effectively. The detailed approach can be seen in figure 2.

4.1.6 Spatiotemporal Scans

In the VideoMamba model, we utilize a typical raster scan method, processing the spatial dimensions first and the temporal dimension second. This approach, found by Li et al. [7] to be the most performant, aligns with the default practice for Vision Transformer (ViT) models. Although the scan order does not significantly impact ViT models, it proves advantageous for state-space models like VideoMamba by ensuring efficient and effective handling of spatiotemporal data.

4.1.7 Integration with V-JEPA

To integrate the VideoMamba model with the V-JEPA training structure, we removed the classification head from the VideoMamba model and incorporated it into the V-JEPA framework. This modification allows the VideoMamba model to function effectively within the V-JEPA architecture, leveraging its strengths in handling spatiotemporal data. In our implementation, we used the VideoMamba encoder for both the context encoder and the target encoder, as well as for the predictor submodel within the V-JEPA structure. This integration aims to combine the robust spatiotemporal feature extraction capabilities of the VideoMamba model with the efficient and stable training process facilitated by V-JEPA, enhancing overall performance in video understanding tasks.

4.2 V-JEPA Model

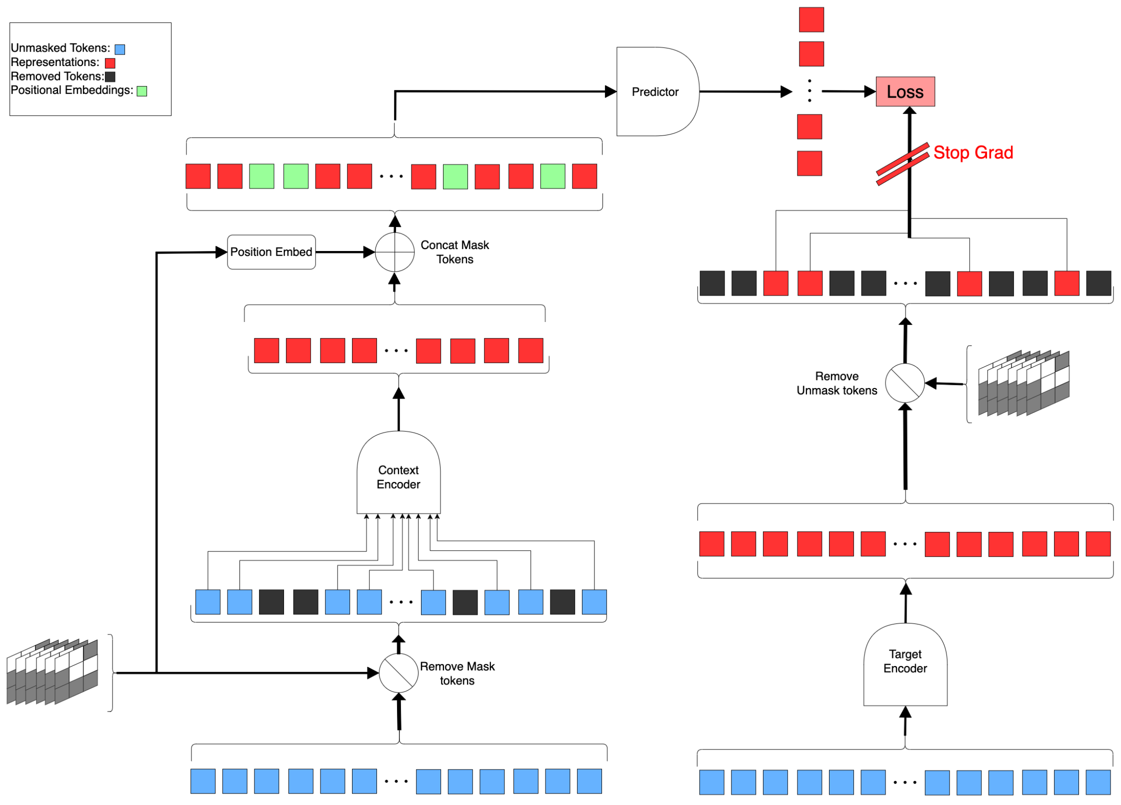

Figure 3: The figure illustrates the architecture of the V-JEPA (Video-based Joint-Embedding Predictive Architecture) model. The process begins with input video frames, which are divided into spatiotemporal tokens. These tokens are passed through the position embedding layer, where positional information is added. The context encoder processes the unmasked tokens, while the mask tokens are concatenated and fed into the predictor. The predictor attempts to predict the embeddings of the masked tokens. Simultaneously, the target encoder generates ’ground truth’ representations for the full (unmasked) sequence. The loss is computed between the predictor’s output and the target encoder’s output. The "Stop Grad" operation prevents gradients from flowing back to the target encoder, ensuring stability in the training process.

4.2.1 Introduction

The Video-based Joint-Embedding Predictive Architecture (V-JEPA) is a training strategy designed to enhance the efficiency and effectiveness of vision encoders for video analysis. V-JEPA extends the principles of the Image-based Joint-Embedding Predictive Architecture (I-JEPA) introduced by Assran et al. [3] to handle spatiotemporal data in videos. This approach focuses on training an encoder to learn robust latent space representations by minimizing the contrastive loss between a predictor and a target encoder, eliminating the need for a decoder to reconstruct pixels. The primary goal of the V-JEPA training strategy is to produce a target encoder that outputs high-quality representations of videos.

I-JEPA comprises three main components: an input encoder, a target encoder, and a predictor. The input encoder processes partially visible inputs, while the target encoder, which is a rolling average of the input encoder, provides stable 'ground truth' representations. This structure prevents trivial encoding solutions and ensures meaningful representation learning. V-JEPA adapts this framework for video by predicting masked regions within the representation space, making it applicable to any patch-based vision encoder.

In the context of V-JEPA, large ViT models have shown significant performance improvements, achieving results comparable to VideoMAE and VideoMAEv2 on datasets like Kinetics-400 (K400) and Something-Something v2 (SSv2) with reduced pre-training iterations. This efficiency is beneficial for training scenarios with limited computational resources. In this work, we integrate the VideoMamba model into the V-JEPA framework, utilizing its robust spatiotemporal feature extraction capabilities to further advance video understanding.

4.2.2 Context Encoder

In the V-JEPA framework, the context encoder is responsible for processing partially visible video frames, generating embeddings that capture the available spatiotemporal information. For our implementation, we utilize the VideoMamba encoder as the context encoder. The VideoMamba model, with its robust spatiotemporal feature extraction capabilities, is well-suited for this task.

The context encoder receives the video frames with certain regions masked out, as determined by the 3D Multi-Block Masking strategy described earlier. The VideoMamba encoder processes these masked video sequences using its 3D convolutional patch embedding, position embeddings, and Bidirectional Mamba Blocks. This approach ensures that the embeddings generated by the context encoder are rich in both spatial and temporal features, despite the partial visibility of the input frames. The approach can be seen in figure 3.

By leveraging the VideoMamba encoder's architecture, the context encoder can effectively handle the complexities of video data, ensuring that the partially visible inputs are transformed into meaningful representations. These representations are then used by the predictor to infer the missing information, facilitating robust and efficient learning within the V-JEPA framework.

4.2.3 Target Encoder

The target encoder in the V-JEPA framework provides stable 'ground truth' representations for the video frames. To achieve this stability, the target encoder is implemented as a rolling average of the context encoder. This design choice mitigates the risk of representation collapse, where the model might otherwise learn trivial or constant representations for different inputs.

In our implementation, the target encoder utilizes the same VideoMamba encoder architecture as the context encoder. However, unlike the context encoder, which processes partially visible video frames, the target encoder operates on the complete, unmasked video frames. By maintaining a rolling average of the context encoder's weights, the target encoder ensures that its output representations evolve smoothly over time, reflecting the learned features without abrupt changes.

This rolling average mechanism helps stabilize the learning process, providing consistent and reliable target representations for the predictor. By leveraging the robust spatiotemporal feature extraction capabilities of the VideoMamba encoder, the target encoder contributes to the effective training of the V-JEPA framework, ensuring that the predictor is guided by high-quality embeddings.

4.2.4 Predictor

In the V-JEPA framework, the predictor is tasked with inferring the representations of the masked regions from the embeddings generated by the context encoder. For our implementation, we utilize the VideoMamba encoder as the predictor. This choice ensures consistency in the architecture and training dynamics, leveraging the same spatiotemporal feature extraction capabilities as the context and target encoders.

Initially, we experimented with using a Vision Transformer (ViT) as the predictor. However, this configuration led to significant instability in the training process, and the model failed to converge. We attribute this instability to the differing training hyperparameters required by the ViT model compared to the VideoMamba model. Implementing a dual-optimizer setup, where each component uses its respective optimal training parameters, proved impractical due to the complexity of managing two different optimizers and loss functions within the same training pipeline.

By using the VideoMamba encoder for the predictor, we maintain a cohesive training regime with unified hyperparameters, resulting in a more stable and convergent training process. The predictor processes the embeddings of the masked tokens, leveraging the inductive biases and feature extraction strengths of the VideoMamba architecture to accurately infer the missing information. This integration ensures that the V-JEPA framework operates efficiently, capitalizing on the robust performance of the VideoMamba model across all its components.

4.2.5 Training Mechanism

The training mechanism in the V-JEPA framework involves minimizing the difference between the predicted representations generated by the predictor and the target representations provided by the target encoder. We use a loss function designed to achieve this goal, along with a regularization term to ensure stable and consistent learning.

The loss function used in our implementation is simply the average L1 distance between the output of the predictor and the target encoder. Let z_i be the predicted representations and h_i be the target representations for the i-th masked region. The loss function $\mathcal{L}$ is given by:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{L} \sum_{j=1}^{L} |z_{ij} - h_{ij}|$$

where:

- N is the number of masked regions.

- L is the dimensionality of the representations.

The regularization term $\mathcal{R}$ is defined as:

$$\mathcal{R} = \frac{1}{N} \sum_{i=1}^{N} \sqrt{\mathrm{Var}(z_i) + \epsilon}$$

where:

- $\mathrm{Var}(z_i)$ is the variance of the predicted representations for the i-th masked region.

- ϵ is a small constant added for numerical stability (set to 0.0001 in our implementation).

The final objective is to minimize the combined loss and regularization term:

$$\mathcal{L}_{\text{total}} = \mathcal{L} + \lambda \mathcal{R}$$

where λ is a regularization parameter.

This training mechanism ensures that the predictor learns to generate representations that closely match the target encoder's outputs while maintaining stable and well-distributed embeddings. By leveraging this loss function, the V-JEPA framework facilitates effective and efficient learning of robust video representations.

5 Experimental Results

5.1 Implementation Details

To conduct our experiments, we used the released code for V-JEPA to train our models, along with the Video Mamba implementation of Vim with bidirectional raster scanning across the video patches. Bidirectional scanning, as found by L. Zhu et al. and Y. Liu et al., improves performance in video understanding tasks. For hyperparameters, we used the recommended pre-training parameters from L. Zhu et al. for each model size. A smaller version of VideoMamba was used for our predictor model during V-JEPA training.

For pre-training ViT-Tiny and ViT-Small using V-JEPA, we employed the recommended hyperparameters from the V-JEPA paper [3] and used a ViT predictor model during JEPA training.

To explore the application of V-JEPA to VideoMamba, we planned to use three different model sizes from the VideoMamba paper: VideoMamba-Tiny, VideoMamba-Medium, and VideoMamba-Base. We selected VideoMamba-Tiny for ablation studies due to its efficiency, VideoMamba-Medium because it was the largest model that did not overfit in the original VideoMamba paper, and VideoMamba-Base because it was unable to avoid overfitting using the original training strategy. We also planned to conduct ablation studies on positional encodings, comparing the sin-cos encodings used in V-JEPA with the learned encodings used in VideoMamba, using VideoMamba-Tiny. Additionally, we intended to test different patch masking strategies, including random tube masking, clip-row masking from VideoMamba, and the multi-block masking strategy from the V-JEPA paper.

We evaluated the encoder in the ablation studies by training an attentive probe on top of the frozen encoder for downstream tasks, including action recognition on Kinetics-400, motion classification on Something-Something v2, and object recognition on ImageNet-1K. This approach provides insights into the effectiveness of different masking and embedding strategies for motion detection, video temporal understanding, and image understanding, allowing direct comparison between ViT and VideoMamba when trained with V-JEPA.

5.2 Intermediate Results

To date, we have conducted experiments with VideoMamba-Tiny and ViT-Tiny pre-trained on Something-Something v2 (SSv2). We used the multi-block masking strategy from V-JEPA for both models and then trained an attentive probe on SSv2 video action classification following A. Bardes et al. Due to limited computational resources, we were unable to fully fine-tune the attentive probes, resulting in poor accuracy. The VideoMamba-Tiny model achieved 3.20% accuracy after four epochs, suggesting convergence issues.

We believe the poor performance is due to using the hidden states of VideoMamba as output tokens, which may not be well-suited for reconstructing masked patches. In contrast, the ViT-Tiny model achieved a slightly higher accuracy of 4.20%. The instability and convergence issues observed when using a ViT predictor model with VideoMamba further complicate the training process. This instability likely arises from the differing training hyperparameters required by the ViT model compared to the VideoMamba model, and the complexity of managing two different optimizers and loss functions.

To address these challenges, we plan to explore using an attention-based predictor model to better align with the VideoMamba architecture. This approach aims to improve the mapping of hidden states to output tokens, enhancing overall training stability and performance.

	SSv2 Action Classification acc	**fine-tune epochs**
VideoMamba-Tiny	3.20%	4
ViT-Tiny	4.20%	4

Table 1: Comparison of Action Classification Models

6 Conclusion

This study explored integrating the VideoMamba model with the V-JEPA training strategy to improve video representation learning efficiency. By combining the computational efficiency of the Mamba architecture with the robust training framework of V-JEPA, we aimed to create high-performance vision encoders with fewer training iterations and reduced overfitting.

Initial experiments with VideoMamba-Tiny and ViT-Tiny on the Something-Something v2 dataset revealed challenges in achieving stable convergence and high accuracy. These findings suggest the need for further refinement of the predictor model and optimization of training hyperparameters.

Future work will focus on exploring alternative predictor architectures and conducting ablation studies on positional encoding strategies and masking techniques. Despite the challenges, the integration of VideoMamba and V-JEPA shows promise for efficient video representation learning.

In summary, this study provides insights into the potential and challenges of applying the V-JEPA training strategy to the VideoMamba model, laying the groundwork for future advancements in video understanding.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer, 2021.

[2] 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, 2023.

[3] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. V-JEPA: Latent video prediction for visual representation learning, 2024.

[4] 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, 2021.

[5] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[6] Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

[7] Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. Videomamba: State space model for efficient video understanding, 2024.

[8] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model, 2024.

[9] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training, 2022.

[10] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model, 2024.