



Lightweight hybrid spatiotemporal deep learning models for video frame prediction: A comparative study across dataset complexities

Alberto Nogales^{a,b,*}, Jose M. Sánchez Velázquez^a, Mingbo Cai^{c,d},
Andrew Coney^a, Álvaro J. García-Tejedor^a

^a CEIEC Research Institute, Universidad Francisco de Vitoria, Ctra. M-515 Pozuelo-Majadahonda km. 1.800, 28223 Pozuelo de Alarcón, Madrid, Spain

^b University of Alcalá (UAH) Computer Science Department, Alcalá de Henares, Spain

^c Department of Psychology, University of Miami, USA

^d International Research Center for Neurointelligence, The University of Tokyo, Japan

ARTICLE INFO

Keywords:

Computer vision
Video frame prediction
Artificial intelligence
Deep learning
Feature extraction
Temporal sequence analysis

ABSTRACT

In recent years, advances in Artificial Intelligence have significantly impacted computer vision, enabling solutions to complex tasks such as video frame prediction. This task has critical applications in domains like autonomous cars, weather forecasting, and video compression. While Deep Learning has proven effective in spatiotemporal modeling, current architectures often face trade-offs between prediction accuracy and computational efficiency. To address this challenge, this study evaluates lightweight hybrid deep learning models that operate at the feature-map level, using autoencoders for spatial encoding and temporal sequence modeling through Convolutional Long Short-Term Memory (ConvLSTM), 3D Convolutional Neural Networks (3D CNNs), and related temporal architectures. The proposed solutions are tested in terms of performance, latency and energy consumption on three datasets of varying complexity, encompassing synthetic and real-world scenes, as well as grayscale and color imagery. Results show that architectures based on 3D CNNs and ConvLSTMs outperform other approaches on low-complexity real-world datasets. Moreover, the proposed models significantly reduce inference time and energy consumption, key advantages for practical deployment in scenarios such as video streaming or embedded vision systems.

1. Introduction

Computer science has experienced significant development in recent years, largely due to the impact of new techniques in Artificial Intelligence (AI). This branch of computer science studies and interprets the mechanisms behind human intelligent behavior, aiming to replicate such behaviors in machines, though not necessarily using the same underlying processes [16]. Although the foundations of AI were established in the 1950s, breakthroughs have only occurred in recent years, driven by the massive increase in available data and improvements in computational power.

These advancements have significantly benefited various application areas, such as signal processing, Natural Language Processing (NLP), and computer vision. The latter is defined by [7] as a discipline that seeks to equip computers with the ability to interpret, analyze, and understand visual information from their surroundings, including

images and videos. It is within computer vision that we find one of the great milestones of modern AI: the development of AlexNet in 2012, which won the ImageNet competition with a groundbreaking performance [20]. Since then, computer vision has become one of the most rapidly evolving areas of AI, with applications such as image classification, object recognition, and video frame prediction.

Video frame prediction remains a challenging task for modern AI techniques. Videos are sequences of frames over time, requiring not only image feature extraction but also sequential pattern analysis. The task has broad practical applications spanning surveillance, compression, and autonomous cars. These challenges are compounded by the diversity of video content, including differences in frame nature (synthetic vs. real) and color spaces (grayscale vs. color videos), which also affect Deep Learning models' performance. Deep Learning, as defined by [23], involves multi-layered hierarchical artificial neural networks capable of learning data representations with progressively higher levels of

* Corresponding author at: CEIEC Research Institute, Universidad Francisco de Vitoria, Ctra. M-515 Pozuelo-Majadahonda km. 1.800, 28223 Pozuelo de Alarcón, Madrid, Spain.

E-mail address: alberto.nogales@ceiec.es (A. Nogales).

<https://doi.org/10.1016/j.imavis.2026.106116>

Received 27 October 2025; Received in revised form 30 June 2026; Accepted 3 July 2026

Available online 3 July 2026

0262-8856/© 2026 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

abstraction from raw input.

The primary motivation of this study is to evaluate the effectiveness of lightweight hybrid Deep Learning architectures that combine two fundamental components for videos: convolutional models for spatial feature extraction and sequence modeling techniques for temporal pattern analysis. These architectures are intentionally designed to be computationally efficient, as video frame prediction tasks often demand real-time or low-resource deployment. To assess the generalizability of the proposed models, experiments were conducted on three datasets with varying characteristics: a synthetic grayscale dataset, a grayscale real-world dataset, and a color dataset consisting of natural video scenes. This diverse dataset selection enables a preliminary yet comprehensive exploration of model performance across different video types and complexities.

The main challenging novelty of our approach lies in designing and evaluating different hybrid (Autoencoder and temporal processing models) architectures that operate at the feature map level, rather than directly on full video frames. By using this approach, the networks are significantly more lightweight, as it processes the main representative features of each frame instead of raw pixel data. All proposed pipelines follow a unified three-step structure: first, an Autoencoder extracts frame features from the input video frames; second, various Deep Learning models for sequential pattern analysis, such as Recurrent Neural Networks (RNNs), are used to predict the subsequent feature map; and third, the decoder component of the previously trained Autoencoder reconstructs the video frame from the predicted features. Unlike previous studies that typically focus on a single temporal architecture or evaluate only predictive accuracy, the present work provides a unified benchmark of lightweight feature-map-level video prediction models. Models' performance is evaluated not only through frame prediction quality but also through latency and energy consumption metrics.

The contribution of this work does not reside in proposing new neural network components, but in designing and evaluating a lightweight hybrid framework for video frame prediction, which operates at the feature level by combining Convolutional Autoencoders with deep sequence models. This includes six three-stage pipelines that integrate feature extraction, temporal modeling, and frame reconstruction which are evaluated under identical experimental conditions. The study also analyzes the influence of three datasets of varying complexity, covering synthetic, grayscale, and color real-world videos. Finally, it jointly considers predictive quality, inference latency, and energy consumption, providing practical insights for real-time and resource-constrained deployments.

Results show that apart from achieving competitive predictive accuracy, the proposed approaches demonstrate faster inference and lower energy consumption compared to baseline methods, making them highly suitable for real-time and resource-constrained applications.

The structure of this paper is as follows. Section 2 reviews prior research on video frame prediction using AI techniques. Section 3 describes the data sources and methodology employed in this study. Section 4 presents the results and their analysis, followed by a discussion of their implications in Section 5. Finally, Section 6 concludes the paper with a summary of key findings and directions for future research.

2. Related works

This paper aims to evaluate hybrid Deep Learning models for video frame prediction. In particular, the proposed approaches benefit from other Deep Learning models and their capabilities to extract features from images and make an analysis of sequential patterns. Related to the present work, we find different previous papers that use Artificial Intelligence techniques for video frame prediction.

Video frame prediction has been addressed for a long time, but in the case of classical Machine Learning models, this was such a complex task that it was used for intra- and inter-frame prediction. In [22], a Machine

Learning-based approach using Support Vector Machines (SVM) is developed to improve macro-block mode decision in Moving Picture Experts Group 2 (MPEG-2) video compression, specifically for inter-frame prediction. Another interesting work is [29] that introduces GM-RF, a fast intra-frame prediction mode decision algorithm for the AOMedia Video 1 (AV1) encoder that uses Random Forest (RF) models to group and select prediction modes, significantly reducing computational complexity. [13] present a low-complexity macroblock decision algorithm for transcoding MPEG-2 P-frames to H.264, focusing on inter-frame prediction using Decision Trees based on the correlation of residual data between formats. Also, [11] propose a High Efficiency Video Coding (HEVC) strategy for intra-frame prediction using RF to accelerate depth decision and reduce complexity. As can be seen, classical Machine Learning models are not powerful enough to deal with frame video prediction, and all previous works are aimed at intra- or inter-frame prediction with compressed data.

With the development of Deep Learning models in recent years, the video frame prediction task has experienced great development. [3] evaluate two Deep Learning architectures, Convolutional Long Short-Term Memory (ConvLSTM) and Generative Adversarial Networks (GANs), for human motion frame prediction, training them on the UCF101 dataset. They find that the ConvLSTM model, which explicitly captures temporal and spatial patterns, outperforms GANs in predicting future frames. It should also be highlighted [10], which implements a ConvLSTM model to perform next-frame prediction using realistic human activity videos from the UCF101 dataset, obtaining great results. Also, [24] propose a Deep Learning framework for video compression and anomaly detection based on latent space frame prediction using ConvLSTM. Their method predicts future frames in the latent domain, achieving efficient compression and accurate anomaly detection by analyzing discrepancies between predicted and actual frame representations. In [9], a prediction method for video coding using a U-Net architecture is presented, performing both uni- and bi-directional inter-frame prediction without requiring motion vectors. Finally, [32] introduce PreCNet, a next-frame video prediction model implemented with hierarchical ConvLSTM modules. The model achieves state-of-the-art performance on benchmarks like the KITTI and Caltech Pedestrian datasets. As we can see, Deep Learning models could achieve video frame prediction accurately, but in our case, we wonder if we could benefit from the strengths of different Deep Learning models to develop a hybrid one.

In the case of hybrid Deep Learning models. Hybrid feature extraction and fusion strategies have also proven effective beyond video prediction, for instance, in remote sensing change detection [37] and salient object detection in optical remote sensing images [36], further supporting the generalizability of hybrid feature-level architectures across computer vision tasks. For video frame prediction, we can highlight the following. [25] presents a novel next-frame video prediction method that combines MobileNetV2 for frame-level feature extraction with a Transformer model for temporal sequence modeling of embeddings. Also [26], which presents proposed HF²-VAD, a hybrid video anomaly detection framework that uses Autoencoders for optical flow reconstruction and a Conditional Variational Autoencoder (CVAE) for flow-guided future frame prediction. In [26], a hybrid video frame prediction model that combines Transformers and LSTM with 3D to efficiently model long- and short-term spatiotemporal dependencies is presented. We also found [2], which develops a novel hybrid video frame prediction model that fuses ConvLSTM and 3D-CNN architectures to capture temporal and spatial dynamics, respectively. As demonstrated in previous studies, many approaches combine Deep Learning models to extract spatial features from images and capture the temporal dynamics of frame sequences.

In this work, we adopt a similar strategy; however, to the best of our knowledge, this is the first study to benchmark multiple lightweight temporal architectures within a unified feature-map prediction framework, jointly evaluating reconstruction quality, computational

efficiency, and energy consumption across heterogeneous datasets. While prior methods either operate directly in pixel space, incurring substantial computational overhead, or perform latent-space prediction for specific applications such as video compression or representation learning, they rarely provide systematic comparisons among recurrent, convolutional, and hybrid temporal models under identical experimental conditions. Consequently, there is limited understanding of how these lightweight paradigms trade off predictive performance against latency and energy efficiency in resource-constrained deployment scenarios.

3. Materials and methods

In this section, we describe the datasets used in the experimentation, and then we formalize all the models that are part of the approaches.

3.1. Video frame datasets

As part of the paper, we are evaluating different Deep Learning model approaches. As the nature of the data could influence it, we are using three different datasets: one which contains synthetic images in black and white, another also in grayscale but with moving people, and another one which is the closest to real applications of people performing different tasks, which is colored. Following, we are providing more information about the different datasets.

The first dataset, Moving MNIST,¹ consists of 10,000 sequences of 20 frames each of size 64×64 pixels, portraying two moving digits in black and white on a dark background and serving as a synthetic testbed for studying object motion and interaction in a single plane. The second dataset, ICPR'04 Recognition of Human Actions,² comprises 599 grayscale videos with 25 subjects performing six actions: walking, jogging, running, boxing, waving, and clapping. These videos exhibit variations in scale, clothing, and setting, are recorded at 25 frames per second with a resolution of 160×120 pixels and have an average duration of four seconds. Finally, the UCF101³ dataset contains 13,320 realistic videos sourced from YouTube, covering 101 action categories with considerable diversity in camera movement, appearance, pose, and background. Each video, at a resolution of 320×240 pixels, belongs to one of 25 groups, making the dataset a highly representative benchmark for real-world human action recognition.

3.2. A set of hybrid models for video frame prediction

As said before, in this work, we are evaluating different hybrid Deep Learning approaches tested in the three datasets mentioned above. The goal is to leverage the strengths of Autoencoders in extracting the essential features of images, along with the capabilities of models such as Recurrent Neural Networks (RNNs) and ConvLSTMs to analyze sequential patterns in video. Since all proposed approaches follow the same overall workflow, we describe it as follows. First, an Autoencoder is trained on all video frames to extract feature maps from its bottleneck layer, which are then used to construct a new training dataset. Second, various Deep Learning models capable of learning from time series data are trained on this feature map dataset to predict the next frame's feature representation. Finally, the decoder part of the original Autoencoder is used to reconstruct the video frame from the predicted feature map. This workflow is applied consistently across all three datasets. A schematic overview of the proposed method is presented in Fig. 1.

As can be seen in the previous Figure, the workflow implies different Deep Learning models that include Autoencoders and sequential architectures, which are defined as follows.

Autoencoder. These are unsupervised learning models; the training data is not labeled, originally introduced in [4]. A key characteristic of Autoencoders is that the input and output layers normally have the same dimensionality, and the model is composed of two main components. The first is the encoder, which compresses the input data into a lower-dimensional representation that captures its essential features. The second is the decoder, which reconstructs the original input by progressively upsampling this compact representation back to its original size. In our case, we train an Autoencoder for each of the three datasets using pairs of the same frame. Once each Autoencoder is trained, we can obtain a dataset of feature maps to train the next models.

In the following, we formalize several spatiotemporal sequence models that have been used to predict the next feature map corresponding to a video frame.

Recurrent Neural Network (RNN). First introduced by [12], RNNs process variable-length input sequences by maintaining an internal hidden state, making them well-suited for temporal modeling tasks such as video frame prediction.

Long Short-Term Memory (LSTM). Developed by [15], LSTMs extend RNNs with gating mechanisms, input, output, and forget gates that selectively retain or discard information, enabling effective learning over long temporal sequences.

Gated Recurrent Unit (GRU). This model was first described in [8], which simplifies the LSTM architecture by using only two gates, update and reset, reducing computational complexity while maintaining competitive sequential modeling performance.

3D Convolutional Neural Network (3D-CNN). Introduced by [17], 3D-CNNs apply volumetric convolutions across stacked video frames, capturing spatial and temporal features simultaneously.

Convolutional LSTM (ConvLSTM). [31], ConvLSTM replaces the fully connected operations in LSTM with convolutional structures, enabling simultaneous modeling of spatial and temporal dependencies.

Convolutional RNN (CRNN). This model was introduced by [27], CRNNs combine convolutional layers for spatial feature extraction with recurrent layers for temporal modeling, processing sequential frames through a sliding window to capture spatiotemporal patterns [18].

The benchmark evaluates six representative temporal architectures spanning recurrent, convolutional, and hybrid paradigms, providing a comprehensive comparison under identical experimental conditions.

3.3. Performance metrics

The first metric, Mean Absolute Error (MAE), evaluates the average magnitude of errors between the reconstructed frame and the ground truth, [19]. It is computed by taking the absolute value of the difference at each pixel to avoid cancelation of positive and negative errors and then averaging these absolute differences across all pixels. In Eq. 1, the metric is formalized.

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i| \quad (1)$$

Like MAE, we also find the Mean Squared Error (MSE), which measures the Euclidean distance between the ground truth image and the image reconstructed by the Autoencoder, as described by [30]. A higher MSE value indicates a larger discrepancy between the reconstructed and actual image, suggesting poorer reconstructed performance. The metric is formally defined in Eq. 1.

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (2)$$

For Eqs. 1 and 2 above, y_i represents the value at a specific position in the ground truth image and $\hat{y}_i$ represents the corresponding value in the reconstructed image.

The Structural Similarity Index Measure (SSIM), introduced by [35], assesses the perceptual similarity between images, regardless of their

¹ https://www.cs.toronto.edu/~nitish/unsupervised_video/

² <https://www.csc.kth.se/cvap/actions/>

³ <https://www.crcv.ucf.edu/data/UCF101.php>

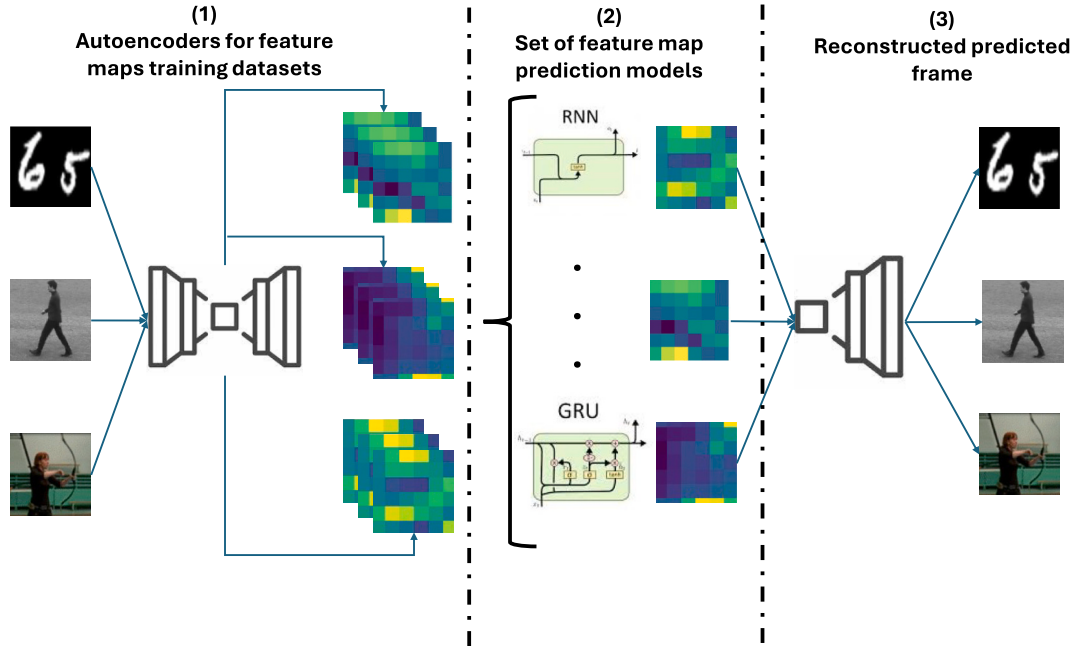


Fig. 1. Proposed workflow for video frame prediction.

absolute quality. It evaluates three key image components, luminance (l), contrast (c), and structure (s), each weighted by constants α , β , and γ , respectively. In Eq. 2, SSIM is formalized.

$$SSIM(x, y) = [l(x, y)]^\alpha \cdot [c(x, y)]^\beta \cdot [s(x, y)]^\gamma \quad (3)$$

Finally, we have the Kullback–Leibler divergence (D_{KL}) [21], which is an asymmetric measure that quantifies how one probability distribution, $P(x)$, diverges from a baseline distribution, $Q(x)$, over the same set of values of a discrete variable x . In this work, D_{KL} measures discrepancies between the global intensity distributions of two images. Consequently, it provides additional insight into whether the proposed feature-map representations preserve broader statistical characteristics such as contrast, texture composition, and intensity allocation, even when individual pixel values differ. We found a mathematical formalization in Eq. 4.

$$D_{KL}(N(\mu_i, \sigma_i) || N(0, 1)) = \frac{1}{2} \sum_i (1 + \ln(\sigma_i^2) - \mu_i^2 - \sigma_i^2) \quad (4)$$

In the last Equation, $N(\mu, \sigma)$ is a normal distribution having as mean the mean μ and as standard deviation the standard deviation σ of the output data obtained in training, and $N(0, 1)$ is the standard normal distribution.

Summarizing and in terms of interpretation, the formalized metrics answer the following questions. MSE and MAE answer “How close are the pixels?”, SSIM answers “How well is the scene structure preserved?”, and D_{KL} answers “Do the reconstructed frames maintain the same statistical characteristics as the originals?”. The four metrics were selected to characterize different aspects of reconstruction quality, but also, they should be interpreted as complimentary in some cases.

4. Experimental setup and results

This section presents how the resulting datasets used for the evaluation have been pre-processed and how the different approaches were trained with them. It also details the metric values obtained to build the evaluated frame prediction models. Finally, a comparison between the performance of the best approaches for each of the datasets is presented.

The experiments presented in this study were conducted on a high-performance computing server. This server is equipped with an AMD

Ryzen Threadripper 2950X processor, featuring 16 physical cores and 32 threads, based on a 64-bit x86_64 architecture and capable of frequency boosting up to approximately 3.5 GHz. The system includes four GPUs: two NVIDIA RTX A6000 units with approximately 49 GB of GDDR5X memory each (interconnected via NVLink), and two NVIDIA GeForce GTX 1080 Ti cards with 11 GB of GDDR5X memory each. The server is supported by 128 GB of RAM and a hybrid storage configuration comprising a 256 GB SSD for the operating system and applications, along with an additional 3.6 TB hard drive dedicated to data storage. All experiments were run on Ubuntu 22.04.1 LTS. The server architecture is optimized for performance, featuring advanced networking and I/O capabilities, AMD-V virtualization support, integrated hardware monitoring sensors, remote management options, and built-in mitigations for known security vulnerabilities.

4.1. Preprocessing the datasets for a fair comparison

The following preprocessing stage was conducted to ensure a fair comparison across the three datasets.

To ensure a fair comparison among the three datasets, a rigorous preprocessing stage was conducted, addressing differences in image type, number of videos, sequence lengths, and frame sizes.

First, the temporal continuity of each dataset was verified. In the case of MNIST, which was already provided as individual frames in a NumPy⁴ array, geometric and centroid analyses confirmed that consecutive frames contained consistent object shapes, and that distance increased naturally with frame separation. Selected sequences were then reassembled into videos to visually confirm continuous digit motion. For the ICPR’04 and UCF101 datasets, originally provided as compressed video files, each video was decomposed into individual frames using OpenCV,⁵ and a sample video from each set was reconstructed to validate that the frame extraction process had preserved the correct sequence order.

Next, a common reference framework was established. All sequences were standardized to 20 frames each, using truncation for those with more frames, since MNIST originally contained 20-frame sequences. A

⁴ <https://numpy.org/>

⁵ <https://opencv.org/>

total of 599 sequences from each dataset were then selected, matching the smallest dataset size, where MNIST was randomly reduced from its 10,000 available sequences, and UCF101 was stratified by action category to ensure balanced representation. Spatial resolution was uniformly set to 64×64 pixels, which included removing black borders in certain UCF101 frames and using Lanczos interpolation for consistent resizing. MNIST frames were binarized using Otsu's method, improving contrast for white digits on a black background, while ICPR'04 frames remained grayscale, and UCF101 frames remained in color. Consequently, all models trained on these datasets receive data of uniform size and length, allowing for a more direct and unbiased performance comparison.

4.2. Training a set of video frame prediction approaches

As described in the proposed workflow, the process can be divided into three main steps. First, three Autoencoders are individually trained to reconstruct the images from each of the three datasets. Second, various spatiotemporal models are trained on the corresponding feature maps extracted from each dataset, enabling them to predict the next feature map in the sequence of a video. Finally, the predicted feature map is passed through the corresponding Autoencoder to reconstruct the target video frame. This three-stage pipeline involves separate training processes and requires the evaluation of different metrics at each stage to assess the performance of the models. Following, we have compiled all the information related to this part of the experimentation.

This design is motivated by the principle of decoupling spatial representation learning from temporal dynamics modeling: by operating the temporal models on compact bottleneck rather than full-resolution frames, the input dimensionality is reduced, directly reducing the computational cost per temporal step while preserving the structural information necessary for accurate frame reconstruction.

An Autoencoder for image reconstruction. The first model that we are training for each dataset is an Autoencoder, which has two aims. First, as the present work is based on the idea of predicting feature maps, we need to construct these datasets, which are obtained from the three trained Autoencoders (one for each studied dataset). Second, this Autoencoder will be used to reconstruct the frame by using a predicted feature map obtained with the models from the second stage of our method.

With this in mind, we have trained three Convolutional Autoencoders, one for each evaluated dataset. In all the cases, we have applied different best practices to obtain accurate models. First, before training the models, the creation of the training, validation, and test subsets is needed.

In our case, however, there are specific conditions that must be considered to maintain entire sequences without splitting individual videos. The 80–20 ratio was followed, yielding 480 sequences for training and 119 for testing per dataset, corresponding to 9600 training frames and 2380 testing frames in each. This approach prevents temporal information leaks by keeping entire sequences within one partition and ensures a balanced, reproducible data division. Apart from that, all images were normalized in ranges from 0 to 1 so they can adapt better to the used models.

Once we have our training and test subsets, we are training our Convolutional Autoencoders. A grid search strategy was employed to optimize hyperparameters by exploring various combinations [6].

Table 1
Hyperparameters and values explored during the Autoencoder grid search.

Hyperparameter	Values
Conv. Layer Dimensions	[32, 64, 128], [64, 128, 256]
Loss function	L1, MSE, MSLE, RMSE
Optimizer	Adam RSMProp
Learning rate	0.001, 0.0005

Table 1 summarizes the hyperparameters used across the different machine learning models during this process, which comprises 100 epochs.

We have used MSE as the guidance metric to choose the model that performs the best for each dataset. MSE was selected as the grid search guidance metric due to its computational efficiency during the large number of evaluation cycles required. This metric has been complemented with the others used in this work: MAE, SSIM, and D_{KL} . Although all metrics have been computed for all experiments, we have only included those that are relevant for describing the best models. **Table 2** compiles the MSE for training, validation, and test, for each stage.

All the results presented in **Table 2** indicate that the models perform well, showing no signs of overfitting or underfitting. This observation aligns with the bias-variance trade-off, as described by [5]. Regarding the fact that the performance of the models cannot be compared with human performance, bias seems good, as the metrics are very low. In terms of variance, none of the models appear to exhibit overfitting as the differences between training, validation, and test are very low. We have used MSE as a guidance metric to measure the performance of the model at first, but we have complemented this with **Table 3**, which confirms the good performance of the three Autoencoders.

The consistently high SSIM values across all three datasets confirm that each Autoencoder reconstructs frames with high perceptual fidelity. The progressive decrease in SSIM from MNIST to UCF101 reflects the increasing visual complexity of each dataset rather than a failure of the model, and the low D_{KL} values indicate that the latent distributions remain well-structured across all cases.

Once we confirmed that each Autoencoder accurately reconstructs frames in each of the datasets. The encoder consists of three convolutional layers; the decoder mirrors this structure using transposed convolutions. We compile the features of the architectures in **Table 4**.

The selected architectures reflect a natural adaptation to dataset complexity: the MNIST and UCF101 datasets benefit from the larger filter configuration, producing a bottleneck with greater representational capacity, while the ICPR'04 dataset is well served by a lighter configuration. The spatial resolution of the bottleneck is a direct consequence of applying three convolutional layers to the input frames, reducing spatial dimensions by a factor of 2 at each layer. The channel depth (128 or 256) was selected via the grid search strategy depicted in **Table 1**.

Apart from the parameters obtained through the grid search, the following hyperparameters were applied to each model. Convolutional layers were used with a kernel size of 3, a stride of 2 (halving and expanding the image dimensions), and padding set to 1 to prevent information loss at the edges. Each layer includes normalization and uses Leaky ReLU as the activation function. The weights were initialized using He Initialization, adapted for Leaky ReLU. Finally, the Sigmoid activation function is employed in the output layer to constrain the predictions between 0 and 1.

A set of spatiotemporal models for predicting feature maps. The six architectures were selected to systematically span the temporal modeling design space: pure recurrent models representing increasing levels of gating complexity (RNN, LSTM, GRU); a spatiotemporal recurrent model that jointly captures spatial and temporal dependencies (ConvLSTM); a purely convolutional-temporal model that processes temporal context through volumetric convolutions (3D-CNN); and a hybrid recurrent-convolutional model (CRNN) combining both paradigms. Purely convolutional sequence models, such as Transformers and

Table 2
MSE values for Autoencoders.

Dataset	Training	Validation	Test
Moving MNIST	2.78×10^{-5}	3.43×10^{-5}	4.20×10^{-5}
ICPR'04	0.0010	0.0010	0.0015
UCF101	0.0017	0.0018	0.0018

Table 3
Complete evaluation of Autoencoders.

Dataset	MAE	SSIM	D _{KL}
Moving MNIST	0.0014	0.9994	0.0063
ICPR'04	0.0281	0.9089	0.0014
UCF101	0.0313	0.8928	0.0139

others, were not included in the present study as they fall outside the lightweight recurrent-convolutional design space targeted here and represent a natural direction for future comparison. These limitations of modern architectures are reported in ([28,34]. This architectures' selection enables a structured comparison across fundamentally different temporal inductive biases within a unified evaluation framework. Once we have obtained the Autoencoder for each dataset, we use the latent space to train the different spatiotemporal Deep Learning approaches. In total, we are training six models for the three obtained datasets, which makes a total of 18 solutions. Following the same 80–20 split described above, a grid search was applied for each model. In Table 5, we compile all the tuned hyperparameters and their values. All models share the same hyperparameters and values, except for the number of hidden layers; 3D-CNN and CRNN do not utilize this parameter. The number of input frames (3, 5, and 10) was chosen to evaluate the effect of short, medium, and longer temporal contexts on prediction quality. In this case, K-Fold validation has been applied to ensure the generalizability of the results and prevent overfitting. K-Fold validation was applied to ensure the generalizability of the results and prevent overfitting. Specifically, 5-fold cross-validation was applied with a batch size of 128, a maximum of 100 training epochs, and early stopping with a patience of 10 epochs, monitoring validation loss.

Once we have applied the grid search for the 6 models for each dataset, the MSE is used to choose the best approach in each of the 3 use cases. Again, MSE was selected as the grid search guidance metric due to its computational efficiency during the large number of evaluation cycles required; SSIM, which involves more complex perceptual computation, was reserved for final model evaluation to provide a complementary perceptual quality assessment. Table 6 provides the values of this metric for each dataset in training, validation, and test.

The best models for each dataset have been bolded, so we can see that the ConvLSTM is the best for two datasets and the 3D-CNN for one. In the three use cases, the best model accomplishes the bias-variance trade-off, where the MNIST and ICPR'04 have the best metrics.

To analyze the sensitivity of model performance to each hyperparameter choice, we examined the average test MSE across all six architectures and three datasets for each hyperparameter value, derived from the grid search top 5 configurations per model and dataset combination. Table 7 summarizes these results.

In the Table above, we can highlight the following things. The learning rate (lr) has the highest impact on the simpler Moving MNIST dataset (lr = 0.0001 achieves mean MSE = 0.077 ± 0.082). In the case of real-world dataset its effect is less consistent on ICPR'04, lr = 0.001 achieves the lowest mean (0.127 ± 0.044). In some cases, lr shows a high standard deviation (higher value is 0.488), indicating that its performance is highly sensitive to model and dataset interactions. In the UCF101 best model the metric is more robust, but the MSE is high. The optimizer is the most consistently influential hyperparameter across all datasets: Adam outperforms RMSprop across all three datasets, and this

Table 4
Architecture of Autoencoders.

Dataset	Encoder–Decoder Filter Dimensions	Bottleneck Size	Loss function	Optimizer	Learning rate
Moving MNIST	[64, 128, 256]	$8 \times 8 \times 256$	L1	Adam	0.001
ICPR'04	[32, 64, 128]	$8 \times 8 \times 128$	RMSE	Adam	0.005
UCF101	[64, 128, 256]	$8 \times 8 \times 256$	MSLE	RMSProp	0.001

gap widens substantially on the more complex UCF101 (0.237 ± 0.174 vs. 0.443 ± 0.323). The loss function also exhibits notable sensitivity: L2 consistently achieves the lowest errors in two datasets, while MSLE performs the worst, particularly on UCF101 (0.584 ± 0.412), suggesting it is poorly suited for feature-map-level prediction tasks.

Table 5
Grid search hyperparameters for spatiotemporal models.

Hyperparameter	Values
Number of hidden layers	1, 2, 3
Size of hidden layers	128, 256
Loss function	L1, MSE, MSLE, RMSE
Optimizer	Adam, RMSProp
Learning rate	0.01, 0.001, 0.0001
Number of input frames	3, 5, 10

Table 6
MSE for the different spatiotemporal models regarding the dataset.

Dataset	Model	Training	Validation	Test
Moving MNIST	RNN	0.0650 ± 0.0003	0.0651 ± 0.0003	0.0642
	LSTM	0.0510 ± 0.0004	0.0517 ± 0.0002	0.0508
	GRU	0.0562 ± 0.0002	0.0556 ± 0.0003	0.0562
	3D-CNN	0.0380 ± 0.0013	0.0402 ± 0.0004	0.0394
	ConvLSTM	0.0407 ± 0.0003	0.0408 ± 0.0002	0.0402
	CRNN	0.0454 ± 0.0005	0.0462 ± 0.0006	0.0451
ICPR'04	RNN	0.1738 ± 0.0086	0.1739 ± 0.0088	0.1646
	LSTM	0.1255 ± 0.0012	0.1266 ± 0.0028	0.1314
	GRU	0.1547 ± 0.0026	0.1553 ± 0.0047	0.1498
	3D-CNN	0.0541 ± 0.0074	0.0576 ± 0.0059	0.0583
	ConvLSTM	0.0471 ± 0.0002	0.0479 ± 0.0010	0.0476
	CRNN	0.0460 ± 0.0011	0.0538 ± 0.0011	0.1031
UCF101	RNN	0.4411 ± 0.0064	0.4413 ± 0.0091	0.4356
	LSTM	0.4417 ± 0.0003	0.4419 ± 0.0012	0.4441
	GRU	0.4426 ± 0.0003	0.4427 ± 0.0012	0.4448
	3D-CNN	0.1081 ± 0.0061	0.1171 ± 0.0050	0.1639
	ConvLSTM	0.1121 ± 0.0005	0.1133 ± 0.0023	0.1168
	CRNN	0.3303 ± 0.0154	0.3317 ± 0.0153	0.3388

Table 7
Hyperparameter sensitivity analysis with MSE.

Hyperparameter	Value	MNIST	ICPR'04	UCF101
Learning rate	0.01	0.354 ± 0.232	0.129 ± 0.047	0.331 ± 0.425
	0.001	0.176 ± 0.209	0.127 ± 0.044	0.385 ± 0.129
	0.0001	0.077 ± 0.082	0.321 ± 0.488	0.338 ± 0.313
Optimizer	Adam	0.138 ± 0.168	0.111 ± 0.046	0.237 ± 0.174
	RMSprop	0.154 ± 0.193	0.257 ± 0.384	0.443 ± 0.323
Loss function	L1	0.156 ± 0.214	0.340 ± 0.604	0.246 ± 0.180
	L2	0.087 ± 0.100	0.106 ± 0.047	0.296 ± 0.164
	MSE	0.139 ± 0.159	0.128 ± 0.045	0.260 ± 0.178
	MSLE	0.268 ± 0.258	0.263 ± 0.344	0.584 ± 0.412

To complement the previous results, we have compiled another metric at the test stage. In this case, regarding Table 6, only the best model for each dataset is shown. The information can be seen in Table 8.

The MAE values confirm the degradation pattern already observed in Table 6: performance deteriorates monotonically from Moving MNIST to ICPR'04 to UCF101 (0.2364), consistent with increasing visual complexity.

In the following, we describe the best models for each dataset. The best model for MNIST is the 3D-CNN with hidden layers of size 256. It uses RMSE as a loss function, Adam as an optimizer, and a learning rate of 0.01. The number of input map features is 5. In the case of ICPR'04, the best model predicting feature maps is a ConvLSTM with 2 hidden layers of size 256. It uses MSE as a loss function, Adam as an optimizer, and works with a learning rate of 0.0001. The input comprises 3 instances using a kernel of 3. Regarding UCF101, the best model is also a ConvLSTM with the same hyperparameters.

Testing the reconstructed frames from the predictions. The results obtained from the test sets were decoded using the corresponding Autoencoders to reconstruct future frames. Reconstruction quality was subsequently evaluated through SSIM, providing a measure of perceptual fidelity between predicted and ground-truth images. Table 9 summarizes the results across all datasets and architectures.

As we can see, once again, the best-performing model for each dataset remains the same. Notably, the ConvLSTM model achieves the highest performance on the ICPR'04 dataset, outperforming all other solutions.

The combined analysis of MAE and SSIM highlights how different aspects of reconstruction quality evolve across datasets of increasing complexity. For Moving MNIST, the low MAE and near-perfect SSIM indicate that the models accurately preserve both local pixel information and the overall structural organization of highly regular synthetic sequences. In ICPR'04, MAE increases, and SSIM decreases relative to Moving MNIST, reflecting the additional challenges introduced by real human activities and natural image variability. Nevertheless, the relatively high SSIM values suggest that the structural characteristics of human silhouettes and the predominantly static backgrounds remain well preserved despite the increase in pixel-level errors. This behavior illustrates the complementary nature of the two metrics: MAE captures local reconstruction discrepancies, whereas SSIM measures the preservation of perceptually relevant spatial structures. By contrast, UCF101 exhibits a simultaneous degradation in both measures. The higher MAE and lower SSIM indicate that complex textures, color information, camera motion, and dynamic backgrounds substantially hinder both numerical reconstruction accuracy and structural fidelity, revealing the limitations imposed by the compact latent representation in highly diverse video scenes.

A further analysis of the degradation is accomplished in Table 10. It compiles the percentage degradation in MSE and SSIM for each model across datasets, relative to Moving MNIST. In other words, it measures how the performance improves or worsens depending on the complexity of the dataset, which increases from Moving MNIST to ICPR'04 and then to UCF101.

Analyzing the Table above, ConvLSTM consistently shows the lowest degradation across both metrics (+18.4% and +190.5% MSE increase for ICPR'04 and UCF101, respectively; only -7.1% SSIM drop on UCF101), confirming its suitability for deployment across varying levels of video complexity.

Table 8
Complete evaluation for the best spatiotemporal model.

Dataset	MAE Test
Moving MNIST	0.0859
ICPR'04	0.1255
UCF101	0.2364

Table 9

Mean SSIM values of the reconstructed feature maps of the test set images.

Dataset	Model	SSIM
Moving MNIST	RNN	0.5804
	LSTM	0.6289
	GRU	0.6668
	3D-CNN	0.7628
	ConvLSTM	0.7518
	CRNN	0.7079
ICPR'04	RNN	0.6133
	LSTM	0.6419
	GRU	0.6871
	3D-CNN	0.8013
	ConvLSTM	0.8081
	CRNN	0.7132
UCF101	RNN	0.1618
	LSTM	0.1931
	GRU	0.1935
	3D-CNN	0.5483
	ConvLSTM	0.6981
	CRNN	0.2640

Table 10

Percentage degradation in MSE and SSIM.

Model	Δ ICPR'04 (MSE)	Δ UCF101 (MSE)	Δ ICPR'04 (SSIM)	Δ UCF101 (SSIM)
RNN	+156.4%	+578.5%	+5.7%	-72.1%
LSTM	+158.7%	+774.2%	+2.1%	-69.3%
GRU	+166.5%	+691.5%	+3.0%	-71.0%
3D-CNN	+48.0%	+316.0%	+5.0%	-28.1%
ConvLSTM	+18.4%	+190.5%	+7.5%	-7.1%
CRNN	+128.6%	+651.2%	+0.7%	-62.7%

The observed degradation in reconstruction quality across datasets of increasing complexity can be attributed to the information loss inherent in the bottleneck compression stage. The Autoencoder reduces each 64×64 frame to an 8×8 feature map, imposing a strong information bottleneck that substantially reduces representational capacity. While the latent representation preserves the most salient visual structures, fine-grained spatial details such as textures, edges, and background information may not be fully retained, particularly in visually complex scenes, [14]. For Moving MNIST, this loss is minimal because the relevant content (binary digits on uniform black backgrounds) is low frequency by nature and well-preserved at 8×8 resolution. For ICPR'04, the relatively static backgrounds and limited color depth allow the bottleneck to retain the most structurally important features despite real-world complexity. In the case of UCF101, however, the combination of rich color textures, dynamic backgrounds, camera motion, and fine-grained human pose detail creates a high-frequency content profile not easy to represent. As a result, the temporal model operates on feature maps that are already worse representations of the original frames, and the decoder must reconstruct fine spatial detail from insufficient information, leading to the low SSIM values observed in Table 8.

The observed differences across architectures can largely be explained by their respective inductive biases. RNNs, LSTMs, and GRUs process flattened feature representations and therefore capture temporal dependencies predominantly at a global level, with limited preservation of explicit spatial relationships. By contrast, ConvLSTM maintains the spatial organization of extracted features through convolutional recurrent operations, allowing local spatial correlations and temporal dynamics to be modeled jointly. This behavior appears particularly beneficial for ICPR'04, whose relatively stable backgrounds and structured human motions favor locality-preserving representations. Similarly, 3D-CNNs employ fixed spatiotemporal kernels that effectively capture smooth and predictable motion patterns. However, highly heterogeneous datasets such as UCF101 challenge all evaluated architectures and expose the limitations imposed by the compact latent

representation. Overall, the results suggest that maintaining spatial structure during temporal modeling is an important factor for improving robustness as dataset complexity increases.

These findings suggest several directions that could improve reconstruction fidelity in feature-map-based video prediction frameworks. Adjusting the bottleneck spatial resolution as a configurable trade-off between compression efficiency and representational capacity could better preserve high-frequency spatial detail in complex scenes. Similarly, replacing pixel-wise loss functions with perceptual training objectives could more directly optimize for structural similarity, reducing the gap observed relative to the baseline. Finally, multi-scale architectures maintaining representations at different spatial resolutions could mitigate the information loss inherent in single-bottleneck designs.

But frame predictions do not always respond to the same difficulty. To study that, we have done a more in-depth analysis of how each model performs this task. First, we have divided our test set into four subsets regarding how similar the predicted frame is compared to its ground truth. The interval boundaries were derived empirically using quartile analysis of the test set SSIM score distribution for each dataset. This dataset-adaptive approach was preferred over fixed universal thresholds, which would be inappropriate given the substantially different SSIM ranges across the three datasets, ensuring that the quality categories remain meaningful and comparable within each dataset's own performance distribution. Results have been compiled in Table 11, which allows us to understand how our method performs depending on the nature of the videos.

The best model on the MNIST dataset performs very well, as the best values are around 0.90. A total of 1015 out of 1785 frames, which is around 50% of the tested frames, fall within the best and second-best prediction intervals. Notably, only 49 frames are classified in the worst interval. The narrow range width of 0.07 suggests high consistency and low variability in the model's predictions. This strong performance can be attributed to the relatively simple visual patterns and limited variability present in the dataset, as most of the pixels are black.

On the ICPR'04 dataset, the model also demonstrates robust performance, with 1015 out of 2023 frames ($\sim 50\%$) falling into the two best categories. This is the same number of frames as in the previous dataset, but it should be highlighted that the number of frames predicted as the best is notably higher. The remaining 546 frames are in the categories predicting the worst. Compared to MNIST, the prediction quality is slightly more variable, as reflected by the wider range of 0.09. This indicates that while the model generalizes well to moderately complex scenarios, some frames are challenging to predict accurately.

Finally, regarding UCF101, it presents the most challenging scenario for models. Although 1916 out of 2023 frames ($\sim 94\%$) are within the top two prediction categories, which contain a much higher number of frames than the other approaches. Additionally, metrics in the worst intervals are very low, demonstrating that performance can be improved. The significantly broader range width of 0.16 reveals a greater degree of prediction variability, indicating that the model's performance is less stable in the presence of complex spatiotemporal patterns and background motion.

These results demonstrate that while the proposed model achieves high-quality predictions across all datasets, its performance degrades slightly as the visual and motion complexity of the input increases. The

shifting distribution of prediction quality, especially the decrease in the best predictions and increase in range width, highlights the limitations of the model when dealing with real-world, high-variability video sequences. Nonetheless, the relatively high number of the two top categories, even in challenging datasets, confirms the generalizability and robustness of the model.

Finally, to show real use cases of the performance of the models, we show some predictions based on the previous frames for each of the datasets. In this case, for each of the datasets, we compile 4 predictions in each Figure. These predictions correspond to the 4 upper limits of the intervals and the lower limit of the interval. For each instance, we show the previous frame, the ground truth, and the predicted one.

As shown in Fig. 2, the two best intervals closely match the ground truth in structural terms, while the two worst intervals exhibit increasing blurring and loss of fine detail, consistent with the bottleneck compression limitations discussed above. The degradation is most pronounced in UCF101, where complex backgrounds and color gradients are the hardest to reconstruct from the 8×8 feature representation.

Comparison against a baseline. Regarding the previous results, our approach confirms that the feature-map-based approach not only delivers good performance in the latent space but also retains its good results once the images are restored. To evaluate the effectiveness of our proposed approach, we establish a baseline consisting of spatiotemporal Deep Learning models trained directly on frames of the original image sequences, **without leveraging feature maps**. These baseline models represent the **standard versions** of the architectures later used in our hybrid framework, namely **RNN, GRU, LSTM, ConvLSTM, RCNN, and 3D-CNN**.

To ensure a fair comparison, both the proposed models and the baseline models were trained under identical conditions: the same training and test splits, the same 5-fold cross-validation procedure, the same batch size (128), the same maximum number of epochs (100), the same early stopping criterion (patience = 10), the same optimizer search space, the same loss function search space and the same learning rate search space. The only difference between the two approaches is the input representation: baseline models receive full-resolution frames (64×64), while the proposed models receive the compressed bottleneck (8×8) produced by the Autoencoder.

Tables 12 (MSE) and 13 (SSIM) show the results of comparing the performance of our approaches on the three datasets against the proposed best values of the baseline.

In Table 12 (MSE comparison), lower values indicate better performance. The proposed method outperforms the baseline on the Moving MNIST dataset (0.0209 vs. 0.0286), suggesting improved reconstruction accuracy. However, it performs worse on the ICPR'04 (0.0028 vs. 0.0013) and UCF101 (0.0064 vs. 0.0028) datasets, where the baseline achieves significantly lower error.

In Table 13 (SSIM comparison), higher values indicate better perceptual similarity. Again, the baseline achieves superior results across all datasets: Moving MNIST (0.8032 vs. 0.7628), ICPR'04 (0.9023 vs. 0.8081), and UCF101 (0.8531 vs. 0.6981). This implies that while the proposed approach may reduce pixel-wise error in some cases, it does not preserve structural and perceptual information as effectively as the baseline.

Overall, these results suggest that the proposed method has some potential, especially on the Moving MNIST dataset in terms of MSE, but it generally underperforms the baseline when it comes to perceptual fidelity and generalization to more complex datasets like UCF101 and ICPR'04. It is important to contextualize these differences relative to the intended deployment scenarios. The SSIM gap between the proposed approach and the baseline varies substantially across datasets: 0.0404 on Moving MNIST, 0.0942 on ICPR'04, and 0.1550 on UCF101. For Moving MNIST and ICPR'04, the achieved SSIM values (0.76 and 0.81, respectively) preserve most of the structural content of the original scenes while enabling substantial gains in computational efficiency, as reported later on. The acceptability of this trade-off ultimately depends

Table 11
Evaluation by performance intervals for the best spatiotemporal model.

Dataset	Q1	Q2	Q3	Q4	Range width
MNIST	[0.91, 0.83]: 130	[0.83, 0.75]: 885	[0.75, 0.67]: 721	[0.67, 0.59]: 49	0.07
ICPR'04	[0.96, 0.87]: 739	[0.87, 0.78]: 738	[0.78, 0.69]: 372	[0.69, 0.60]: 174	0.09
UCF101	[0.84, 0.68]: 1355	[0.68, 0.51]: 561	[0.51, 0.35]: 90	[0.35, 0.19]: 17	0.16

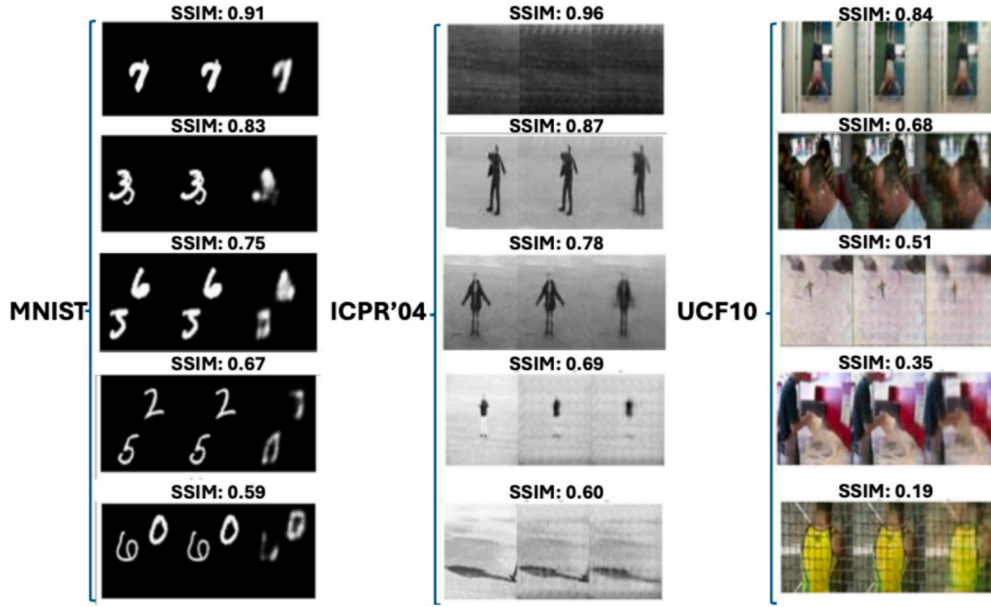


Fig. 2. Examples of the intervals' limits for each dataset.

Table 12

Comparison against a baseline computing MSE.

Dataset	Our approach	Baseline
Moving MNIST	0.0209	0.0286
ICPR'04	0.0028	0.0013
UCF101	0.0064	0.0028

Table 13

Comparison against a baseline computing SSIM.

Dataset	Our approach	Baseline
Moving MNIST	0.7628	0.8032
ICPR'04	0.8081	0.9023
UCF101	0.6981	0.8531

on the intended application and its relative emphasis on perceptual fidelity versus runtime and energy constraints. In real-time streaming and embedded vision systems, inference latency and energy efficiency are often the primary operational constraints rather than maximizing perceptual fidelity, [33]. In this context, the reduction in inference time per iteration and in energy consumption that will be analyzed later in this paper represents a deliberate and practically meaningful trade-off. For UCF101, however, the 0.155 SSIM gap is more substantial and reflects a genuine limitation: the aggressive dimensionality reduction imposed by the bottleneck introduces information loss that disproportionately affects high-frequency spatial detail in complex color scenes. The proposed framework is therefore particularly well suited to lower-complexity or resource-constrained deployment scenarios, while improving reconstruction fidelity on highly dynamic videos remains an explicit direction for future work.

A complementary analysis of the models' performance is based on the parameters that vary regarding the approach, such as the architectural depth and trainable parameters. Table 14 provides this information for both our framework and baselines, confirming that any performance differences are attributable to the input representation rather than architectural or training advantages.

Table 14 indicates that parameter reduction is architecture-dependent rather than universal. Convolutional models, especially 3D-CNN, significantly benefit from feature-map prediction, whereas

Table 14

Architecture depth and trainable parameter counts for each model.

Model	Architecture Depth	Params Proposed (MNIST / ICPR'04 / UCF101)	Params Baseline (MNIST / ICPR'04 / UCF101)
RNN	1 recurrent layer 1 FC	4.2 M / 4.3 M / 8.5 M	2.2 M / 2.2 M / 6.4 M
LSTM	2 recurrent layers 1 FC	21.3 M / 10.8 M / 21 M	6.0 M / 6.0 M / 16.5 M
GRU	1 recurrent layer 1 FC	17.4 M / 8.6 M / 17 M	4.4 M / 4.4 M / 12.8 M
3D-CNN	3 Conv3D (encoder) 4 ConvTranspose2D 1 Conv2D (decoder)	11.7 M / 10.3 M / 11.7 M	24.0 M / 24.0 M / 24.0 M
ConvLSTM	2 ConvLSTM layers 1 Conv2D output	10 M / 8.6 M / 10 M	7.1 M / 7.1 M / 7.1 M
CRNN	2 Conv2D (encoder) 2 RNN layers 1 FC 2 Conv2D (decoder)	19.4 M / 19.1 M / 10.4 M	5.3 M / 5.0 M / 5.3 M

recurrent models may require additional parameters due to the processing of latent representations. Therefore, the principal advantage of the proposed framework lies in computational and energy efficiency rather than in a consistent decrease in model size.

But just having this performance comparison is not fair. As these approaches have significant practical applications, such as streaming or video compression, other aspects, such as runtime or execution time. In real-time applications such as video streaming, the runtime per inference iteration is critical for meeting latency constraints. Meanwhile, in offline (i.e., video compression) or batch processing scenarios, total execution time, encompassing preprocessing, model inference, and post-processing, becomes the key metric for practical deployment. In Table 15, we provide a baseline comparison in these terms.

Analyzing Table 15, on average, our approaches require between five and eight times less time per iteration than their baseline counterpart across all datasets. This saving is independent of sequence length, since the dimensionality reduction performed by the encoder compresses

Table 15

Inference analysis of the approaches.

Dataset	Approach	Run time per iteration of predictive models	Total execution time of the approach
Moving MNIST	3D-CNN	0.03 s	1267 s + 1079 s
	Baseline	0.21 s	6990 s
ICPR'04	ConvLSTM	0.06 s	706 s + 1209 s
	Baseline	0.48 s	18,892 s
UCF101	ConvLSTM	0.06 s	1019 s + 1038 s
	Baseline	0.32 s	14,455 s

Table 16

Energy-consumption analysis in Jules (J) for the different approaches.

Dataset	Approach	Energy cost of the approach
Moving MNIST	3D-CNN	120,160 J + 292,388 J
	Baseline	1,317,228 J
ICPR'04	ConvLSTM	62,628 J + 186,861 J
	Baseline	3,848,750 J
UCF101	ConvLSTM	94,024 J + 138,031 J
	Baseline	3,233,107 J

every frame into a fixed, compact representation.

Even when the one-off costs of feature extraction and final image reconstruction are added (terms shown after the plus sign), the cumulative wall-clock time decreases by factors ranging from $\approx 3 \times$ (Moving MNIST) to $\approx 10 \times$ (ICPR'04).

Another interesting analysis in DL is energy consumption due to its growing environmental impact and the escalating computational demands. As highlighted in [1], understanding and comparing energy efficiency across architectures is essential for promoting sustainable AI development through standardized evaluation frameworks. Energy consumption was estimated using the NVIDIA Management Library (pynvml), by sampling GPU power draw at the start and end of each test run and computing energy as the product of average power and total execution time, following the evaluation framework described in the previous paper. In Table 16, we provide such a comparison of our approach against the proposed baseline.

Analyzing results in Table 16. Even after accounting for the encoder and decoder phases, the feature-map strategy reduces the energy budget by 70 to 94%, an advantage that increases with input complexity: ICPR'04 and UCF101, which contain richer motion and higher spatial variability, benefit the most. This significant reduction translates directly into a smaller carbon footprint and longer battery life when the models are deployed on embedded or edge devices.

Dimensionality reduction, therefore, yields substantial gains in both training time and energy consumption, which are two critical factors for continuous-inference scenarios. While the first baseline comparison sections showed that our approaches yield slightly lower performance compared with the baseline, Tables 15 and 16 demonstrate that there is an improvement in computational sustainability, which is key in the practical application of these approaches.

An interesting observation is that as shown in Table 14 the proposed feature-map-based models do not systematically contain fewer trainable parameters than their frame-level counterparts. In several cases, including LSTM, GRU, and ConvLSTM, the proposed architectures are larger. Consequently, the reductions in runtime and energy consumption observed in Tables 15 and 16 can be attributed primarily to the lower-dimensional input representation rather than to reduced model capacity. This finding reinforces the effectiveness of feature-level temporal modeling as a lightweight strategy for video frame prediction.

5. Conclusions and future works

This work proposes novel lightweight hybrid approaches based on Deep Learning models for video frame prediction. The core idea is to

combine the feature extraction capabilities of Autoencoders with the sequential modeling power of others, such as RNNs. Based on this principle, we introduce a three-stage method: first, Autoencoders extract feature maps from the input frames; next, spatial-temporal models are trained on these latent representations to predict future frames; finally, the same Autoencoder from the first stage is used to reconstruct the future frames from the predictions.

To explore different solutions for time-series analysis, we implemented and evaluated distinct model configurations across three video datasets. Results show that the most effective models were those using ConvLSTM and 3D-CNN architectures, achieving SSIM scores of 0.72, 0.82, and 0.69 depending on the dataset. However, when compared with the baseline, our methods yielded slightly lower accuracy. In contrast, we observed that this modest trade-off in predictive performance resulted in significant gains in computational efficiency and energy consumption. This finding highlights the potential of our models for real-world applications where resource constraints are a concern, and lightweight architectures are necessary.

Despite these promising results, this work has several limitations that should be acknowledged. First, the aggressive dimensionality reduction imposed by the bottleneck inevitably discards high-frequency spatial information, which disproportionately affects reconstruction fidelity on visually complex datasets such as UCF101. Second, while three datasets of varying complexity were used, the UCF101 subset is limited to 599 sequences, which may not fully represent the diversity of real-world color video. Third, the evaluation is limited to single-step next-frame prediction; performance on multi-step horizons, where errors accumulate over time, remains untested. These limitations define clear directions for future investigation.

For future work, we plan to conduct separate analyses on a more diverse set of datasets that share the same underlying characteristics, including synthetic grayscale data, real grayscale images, and real color images. Apart from that, we plan to explore the integration of advanced architectures that do not directly operate on spatiotemporal features, such as Generative Adversarial Networks (GANs). Additionally, we aim to test our models in real-world scenarios such as autonomous driving systems and first-person (Point of View) video streams to demonstrate practical applicability. Incorporating multi-modal inputs, such as optical flow, semantic segmentation maps, or audio, could further enhance the accuracy and temporal coherence of predictions, particularly in complex or dynamic scenes. From an architectural perspective, the present work reveals several limitations. In particular, for real-world color datasets, the aggressive dimensionality reduction imposed by the Autoencoders highlights the need to better understand the trade-off between compression efficiency and predictive performance. Another important research direction is benchmarking the proposed framework against non-lightweight state-of-the-art models, including Transformer-based approaches such as TimeSformer and VideoGPT, to clarify the accuracy ceiling achievable when computational cost is not the primary constraint. From a temporal modeling perspective, extending the evaluation to multi-step prediction, particularly in scenarios where latency requirements are less stringent, would provide insight into error accumulation over longer horizons and the temporal consistency of the evaluated architectures. Finally, from a representation learning perspective, investigating variable bottleneck resolutions, perceptual training objectives, and multi-scale Autoencoder designs represents a promising direction for reducing the reconstruction fidelity gap observed in high-complexity datasets such as UCF101.

CRedit authorship contribution statement

Alberto Nogales: Writing – original draft, Supervision, Methodology, Investigation, Formal analysis, Conceptualization. **Jose M. Sánchez Velázquez:** Writing – review & editing, Supervision, Methodology, Investigation. **Mingbo Cai:** Writing – review & editing, Methodology, Investigation, Conceptualization. **Andrew Coney:**

Visualization, Validation, Software, Data curation. **Álvaro J. García-Tejedor**: Resources, Project administration, Funding acquisition.

Funding

Not applicable.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

Not applicable.

Data availability

Datasets are open. Code is available upon request.

References

- [1] S. Aquino-Brítez, P. García-Sánchez, A. Ortiz, D. Aquino-Brítez, Towards an energy consumption index for deep learning models: a comparative analysis of architectures, GPUs, and measurement tools, *Sensors* 25 (3) (2025) 846, <https://doi.org/10.3390/s25030846>.
- [2] C.V. Aravinda, T. Al-Shehari, N.A. Alsadhan, S. Shetty, G. Padmajadevi, K.R.U. K. Reddy, A novel hybrid architecture for video frame prediction: combining convolutional LSTM and 3D CNN, *J. Real-Time Image Proc.* 22 (1) (2025) 50, <https://doi.org/10.1007/s11554-025-01626-w>.
- [3] M. Backus, Y. Jiang, D. Murphy, *Video Frame Prediction with Deep Learning*, 2022.
- [4] D.H. Ballard, Modular learning in neural networks, in: *AAAI* 647, 1987, pp. 279–284.
- [5] M. Belkin, D. Hsu, S. Ma, S. Mandal, Reconciling modern machine-learning practice and the classical bias–variance trade-off, *Proc. Natl. Acad. Sci. USA* 116 (32) (2019) 15849–15854.
- [6] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, *J. Mach. Learn. Res.* 13 (2) (2012).
- [7] A. Bimpas, J. Violas, A. Leivadakis, I. Varlamis, Leveraging pervasive computing for ambient intelligence: a survey on recent advancements, applications and open challenges, *Comput. Netw.* 239 (2024) 110156, <https://doi.org/10.1016/j.comnet.2023.110156>.
- [8] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, in: *ArXiv Preprint ArXiv:1406.1078*, 2014.
- [9] H. Choi, I.V. Bajic, Deep frame prediction for video coding, *IEEE Trans. Circuits Syst. Video Technol.* (2020) 1, <https://doi.org/10.1109/TCSVT.2019.2924657>.
- [10] P. Desai, C. Sujatha, S. Chakraborty, S. Ansuman, S. Bhandari, S. Kardiguddi, Next frame prediction using ConvLSTM, *J. Phys. Conf. Ser.* 2161 (1) (2022) 012024, <https://doi.org/10.1088/1742-6596/2161/1/012024>.
- [11] B. Du, W.-C. Siu, X. Yang, Fast CU partition strategy for HEVC intra-frame coding using learning approach via random forests, in: 2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2015, pp. 1085–1090, <https://doi.org/10.1109/APSIPA.2015.7415439>.
- [12] J.L. Elman, Finding structure in time, *Cogn. Sci.* 14 (2) (1990) 179–211.
- [13] G. Fernandez-Escribano, H. Kalva, P. Cuenca, L. Orozco-Barbosa, A. Garrido, A fast MB mode decision algorithm for MPEG-2 to H.264 P-frame transcoding, *IEEE Trans. Circuits Syst. Video Technol.* 18 (2) (2008) 172–185, <https://doi.org/10.1109/TCSVT.2008.918115>.
- [14] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science* 313 (5786) (2006) 504–507.
- [15] S. Hochreiter, J. Schmidhuber, Long short-term memory, *Neural Comput.* 9 (8) (1997) 1735–1780.
- [16] M.-P. Hosseini, D. Pompili, K. Elisevich, H. Soltanian-Zadeh, Random ensemble learning for EEG classification, *Artif. Intell. Med.* 84 (2018) 146–158.
- [17] S. Ji, W. Xu, M. Yang, K. Yu, 3D convolutional neural networks for human action recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* 35 (1) (2013) 221–231, <https://doi.org/10.1109/TPAMI.2012.59>.
- [18] G. Keren, B. Schuller, Convolutional RNN: An Enhanced Model for Extracting Features from Sequential Data, 2017.
- [19] V. Kotu, B. Deshpande, Introduction, in: *Data Science*, Elsevier, 2019, pp. 1–18, <https://doi.org/10.1016/B978-0-12-814761-0.00001-0>.
- [20] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, *Adv. Neural Inf. Proces. Syst.* 25 (2012) 1097–1105.
- [21] S. Kullback, R.A. Leibler, On information and sufficiency, *Ann. Math. Stat.* 22 (1) (1951) 79–86, <https://doi.org/10.1214/aoms/1177729694>.
- [22] V. Kumar, K.G. Sharma, A.S. Jalal, Macro-Block Mode Decision in MPEG-2 Video Compression Using Machine Learning, 2013, pp. 149–158, https://doi.org/10.1007/978-81-322-1000-9_14.
- [23] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, *Nature* 521 (7553) (2015) 436–444.
- [24] B. Liu, Y. Chen, S. Liu, H.-S. Kim, Deep learning in latent space for video prediction and compression, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 701–710, <https://doi.org/10.1109/CVPR46437.2021.00076>.
- [25] Z. Liu, Y. Nie, C. Long, Q. Zhang, G. Li, A hybrid video anomaly detection framework via memory-augmented flow reconstruction and flow-guided frame prediction, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 13568–13577, <https://doi.org/10.1109/ICCV48922.2021.01333>.
- [26] M. Mathai, Y. Liu, N. Ling, A hybrid transformer-LSTM model with 3D separable convolution for video prediction, *IEEE Access* 12 (2024) 39589–39602, <https://doi.org/10.1109/ACCESS.2024.3375365>.
- [27] Ming Liang, Xiaolin Hu, Recurrent convolutional neural network for object recognition, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3367–3375, <https://doi.org/10.1109/CVPR.2015.7298958>.
- [28] B. Peng, S. Narayanan, C. Papadimitriou, On Limitations of the transformer architecture, in: *Conference on Language Modeling*, 2024.
- [29] P. Rosa, D. Palomino, M. Porto, L. Agostini, GM-RF: an AV1 intra-frame fast decision based on random forest, in: 2022 IEEE International Conference on Image Processing (ICIP), 2022, pp. 3556–3560, <https://doi.org/10.1109/ICIP46576.2022.9897488>.
- [30] C. Sammut, G.I. Webb, *Encyclopedia of Machine Learning*, Springer Science & Business Media, 2011.
- [31] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W. Wong, W. Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, in: *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, 2015, pp. 802–810.
- [32] Z. Straka, T. Svoboda, M. Hoffmann, PreCNet: next-frame video prediction based on predictive coding, *IEEE Trans. Neural Networks Learn. Syst.* 35 (8) (2024) 10353–10367, <https://doi.org/10.1109/TNNLS.2023.3240857>.
- [33] X. Tang, Q. Li, J. Peng, Y. Tan, Z. Yuan, Y. Song, G.-M. Muntean, Y. Jiang, Low-latency video streaming: applications, challenges, and trends, *IEEE Commun. Surv. Tutor.* 28 (2026) 6384–6414, <https://doi.org/10.1109/COMST.2026.3689266>.
- [34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Kaiser, Ł. ukasz, I. Polosukhin, Attention is all you need, *Adv. Neural Inf. Proces. Syst.* (2017) 5998–6008.
- [35] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, *IEEE Trans. Image Process.* 13 (4) (2004) 600–612.
- [36] Y. Zhang, T. Wang, L. Xue, W. Lian, R. Tao, ORSI salient object detection via progressive interaction and saliency-guided enhancement, *IEEE Geosci. Remote Sens. Lett.* 23 (2026) 1–5, <https://doi.org/10.1109/LGRS.2025.3645633>.
- [37] Y. Zhang, J. Zhen, S. Sun, T. Liu, L. Huo, T. Wang, SCAFFNet: a semantic compensated adaptive fusion network for remote sensing images change detection, *IEEE Geosci. Remote Sens. Lett.* 23 (2026) 1–5, <https://doi.org/10.1109/LGRS.2025.3650414>.