Context-Aware Neural Video Compression on Solar Dynamics Observatory

Atefeh Khoshkhahtinat[†], Ali Zafari[†], Piyush M. Mehta[‡], Nasser M. Nasrabadi[†], Barbara J. Thompson[§], Michael S. F. Kirk[§], Daniel da Silva[§]
[†]Dept. of Computer Science & Electrical Engineering, West Virginia University, WV USA
[‡]Dept. of Mechanical & Aerospace Engineering, West Virginia University, WV USA
[§]NASA Goddard Space Flight Center, MD USA
{ak00043,az00004}@mix.wvu.edu,{piyush.mehta,nasser.nasrabadi}@mail.wvu.edu
{barbara.j.thompson,michael.s.kirk,daniel.e.dasilva}@nasa.gov

***Abstract*—NASA's Solar Dynamics Observatory (SDO) mission collects large data volumes of the Sun's daily activity. Data compression is crucial for space missions to reduce data storage and video bandwidth requirements by eliminating redundancies in the data. In this paper, we present a novel neural Transformer-based video compression approach specifically designed for the SDO images. Our primary objective is to efficiently exploit the temporal and spatial redundancies inherent in solar images to obtain a high compression ratio. Our proposed architecture benefits from a novel Transformer block called *Fused Local-aware Window (FLaWin)*, which incorporates window-based self-attention modules and an efficient *fused local-aware feed-forward (FLaFF)* network. This architectural design allows us to simultaneously capture short-range and long-range information while facilitating the extraction of rich and diverse contextual representations. Moreover, this design choice results in reduced computational complexity. Experimental results demonstrate the significant contribution of the FLaWin Transformer block to the compression performance, outperforming conventional hand-engineered video codecs such as H.264 and H.265 in terms of rate-distortion trade-off.**

***Index Terms*—Solar Dynamics Observatory, Neural Video Compression, Swin Transformer, FLaWin**

I. Introduction

NASA's Solar Dynamics Observatory (SDO) mission gathers 1.4 terabytes of data that can be used to understand the effect of the Sun on the Earth each day [1]. Due to the problem of onboard data storage and bandwidth limitations, data compression is inevitable in space missions. Both hand-crafted [2] and neural-based [3], [4] codecs have been proposed to tackle the challenge of data compression on this space mission.

Recently, neural image/video compression methods have achieved remarkable performance compared with their traditional counterparts [5]. All the compression methods attempt to exploit the redundancies in images and videos. There are three types of redundancies in image signals: spatial redundancy, visual redundancy, and statistical redundancy. In addition to the above-mentioned redundancies in image signals, video signals inherit the advantage of temporal redundancy, which allows video compression to obtain a higher compression ratio compared with the still image compression [6].

In image/video compression, a transformation function is utilized to map the data to an uncorrelated latent space. The more decorrelated and energy-compacted latent representation is obtained by transforming, the more effective coding can be achieved. Unlike traditional codecs which use linear transformations, neural data compression is based on nonlinear transformations. Neural networks are capable of approximating arbitrary functions and can operate as a nonlinear transformation [7]. This property of neural networks provides the opportunity to transform the data with nonlinear dependency into a more decorrelated representation.

Any improvement of the transformation function of a neural data compression algorithm can lead to coding supremacy. The transforming part of most neural data compression methods is based on convolutional neural networks, which have failed to take into account long-range dependencies. To address this shortage, we propose to replace the convolutional network with a Transformer-based architecture. Our proposed Transformer framework leverages the self-attention module to capture global relationships. In addition, we equip our Transformer block with a Fused Local-aware Feed Forward (FLaFF) layer to strengthen the extraction of rich and diverse local textures, which is crucial for compression tasks. These enhancements can promote transformation's ability to project the data into a more decorrelated space.

Contributions of this paper. This paper presents a novel learned video compression approach specifically designed for compressing SDO images. The proposed algorithm aims to effectively exploit both spatial and temporal redundancies inherent in the dataset which enables the achievement of a high compression ratio. To enhance the capabilities of the non-linear transform and generate more decorrelated and energy-compacted latent code, we propose a Transformer-based transformation. Our proposed Transformer block leverages window-based self-attention modules and locally enhanced blocks, enabling the capture of both short-range and long-range relationships which are crucial for compression

This research is based upon work supported by the National Aeronautics and Space Administration (NASA), via award number 80NSSC21M0322 under the title of *Adaptive and Scalable Data Compression for Deep Space Data Transfer Applications using Deep Learning*.

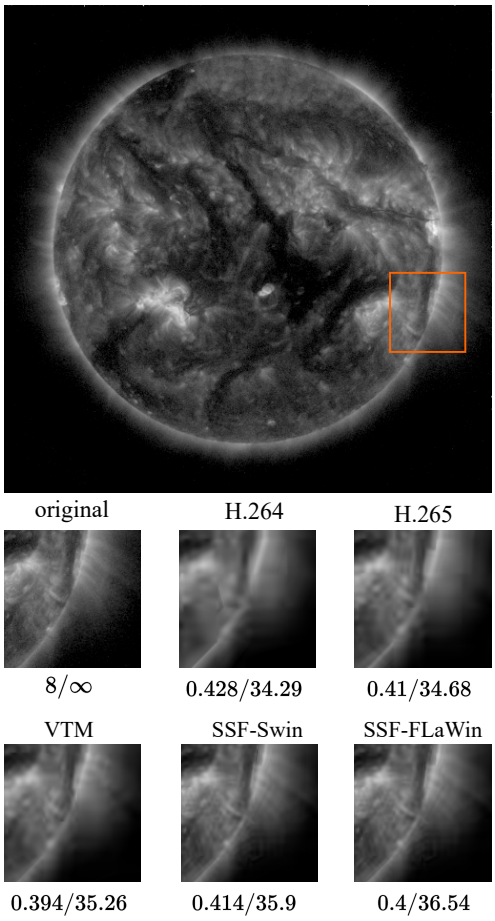

Fig. 1. Visual comparison of the proposed neural video compression approaches (SSF-Swin and SSF-FLaWin) with other traditional codecs in terms of bit-rate/distortion [bpp↓/PSNR↑]. SSF-FLaWin demonstrates lower distortion in terms of PSNR compared to the other codecs, indicating its superior ability to preserve image quality. *Best viewed on screen.*

tasks. This approach also helps in reducing computational complexity.

The remainder of the paper is organized as follows. Section II reviews the neural-based compression methods and the importance of compression in the SDO mission. Section III describes our proposed method. The experiments and ablation studies are discussed in section IV with a conclusion in section V.

II. Related Work

A. Neural Image Compression

Learned image compression methods often employ the transform coding scheme, which comprises four core steps [8]. The first step utilizes an analysis transform to convert the input image into a compact and decorrelated latent representation. This transformation is crucial in reducing the data's redundancy. Once the latent representation is obtained, the second step involves quantization, where the continuous-valued latent variables are discretized to obtain discrete values. In the third step, entropy coding is employed, where an entropy model is utilized to generate a stream of ones and zeros. Finally, in the fourth step, a synthesis transform is applied to the quantized latent representations to reconstruct the original image [9].

Neural image compression networks commonly utilize the autoencoder architecture [9], which allows for the implementation of an approximately invertible nonlinear transformation. Alongside the transformation network, the entropy model is utilized for entropy coding, responsible for estimating the rate of the latent representation, and both are learned in an end-to-end fashion. However, learning the network parameters poses a challenge due to the non-differentiable nature of quantization, resulting in gradients that can be either zero or infinity. To address this issue, several solutions have been proposed to approximate quantization using differentiable operations [10], [11], [12]. A prevalent method to tackle the challenge of non-differentiable quantization is to replace it with additive uniform noise [10]. This substitution effectively transforms the autoencoder into a variational autoencoder (VAE) [13] with a uniform encoder.

In early work, Ballé *et al.* [14] introduced the compressive autoencoder as a powerful image compression framework that achieved comparable performance to the JPEG2000 standard [15]. The compressive autoencoder employed the generalized divisive normalization (GDN) function to enable effective nonlinear transformations and use a fully-factorized entropy model to accurately estimate the bit rate associated with the latent representation. To further improve the entropy model, Ballé et al. [16] designed the hyperprior model, which conditions the distribution of the latent representation on hyperprior. This conditional distribution is approximated using a Gaussian scale mixture (GSM), with the scale parameters acquired from the decoding hyperprior. Building upon this research, Minnen *et al.* [17] extended the entropy model from a Gaussian scale mixture to a Gaussian mixture model (GMM) by incorporating an autoregressive ingredient.

B. Neural Video Compression

Most neural video compression algorithms consist of two components: predictive coding and transform coding [5]. Predictive coding is employed in inter-frame coding to exploit temporal redundancy. Inter-frame coding tries to predict the current frame from one or more previously reconstructed frames. In addition to inter-frame coding, intra-frame coding exists in the video compression pipeline, which leverages spatial redundancy to compress the frame. In intra-frame coding, the process is analogous to image compression methods. As a result, frames are classified into three groups in video codecs: 1. I-frame (Intra-coded): compressed independently using image codecs; 2. P-frame (predicted): predicted from the past frames. 3. B-frames (bi-directional): predicted from the past and future frames [18].

Recently, neural video compression methods have outperformed traditional video compression methods. Taking inspiration from the traditional hybrid video compression schemes, Lu *et al.* [19] introduced the Deep Video Compression (DVC) network as the first end-to-end deep video compression

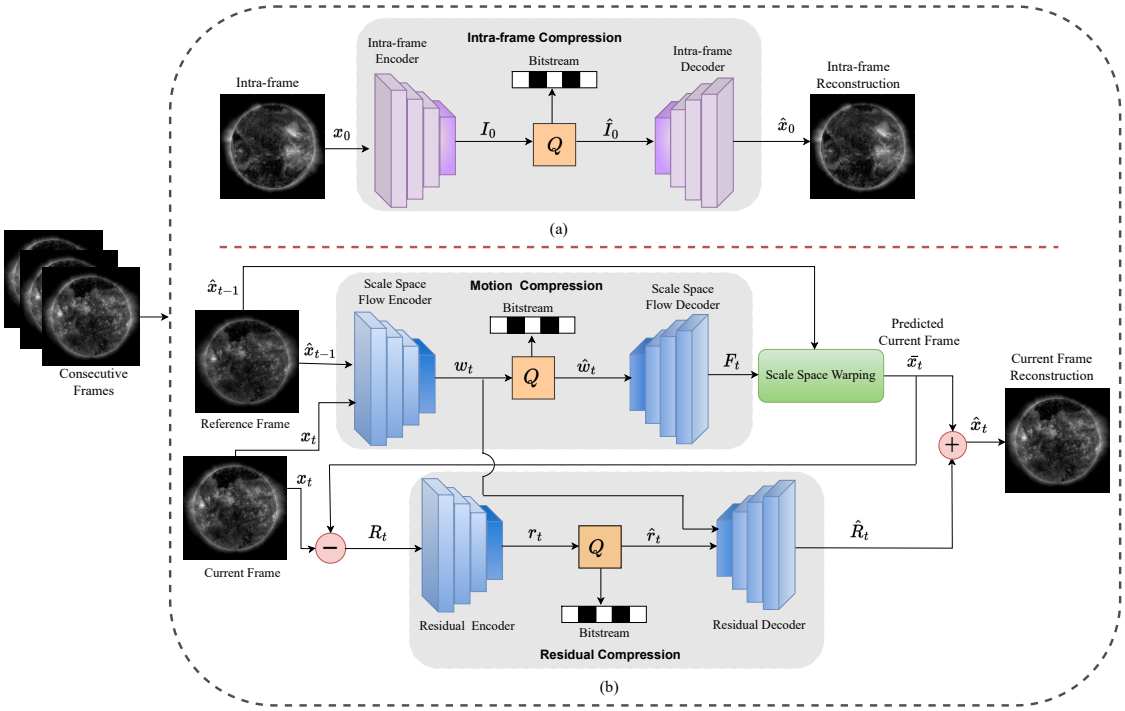

Fig. 2. An overview of neural video compression network. (a) The architecture of the I-frame compression model. (b) The architecture of the P-frame compression model, which consisting of motion compression and residual compression networks. The motion information and scale field are jointly estimated and encoded into a quantized latent representation $\hat{w}_t$. In the I-frame model, the previous reconstruction frame $\hat{x}_{t-1}$ is warped using the decoded motion and scale fields F_t, resulting in the prediction $\bar{x}_t$. The residual R_t is then computed as the difference between the original current frame x_t and the warped prediction $\bar{x}_t$. The residual is further encoded into a quantized latent representation $\hat{r}_t$, which is subsequently decoded to obtain $\hat{R}_t$. The final reconstructed current frame $\hat{x}_t$ is obtained by adding $\hat{R}_t$ to the warped prediction $\bar{x}_t$, resulting in $\hat{x}_t = \bar{x}_t + \hat{R}_t$. Entropy coding for each compression network is excluded for the sake of simplicity.

framework. This pioneering framework used a pre-trained Flownet [20] for optical flow estimation and employed bilinear warping techniques for motion compensation. For motion and residual compression, two autoencoder-based networks were used. Agustsson *et al.* [21] proposed the Scale-Space Flow (SSF) framework, which aims to mitigate the difficulties associated with fast motion in optical flow estimation. Their approach involves the incorporation of a scale channel as an uncertainty parameter, allowing the application of Gaussian blur to regions prone to disocclusions and rapid motion. Hu *et al.* [22] presented the Feature-space Video Compression (FVC) network as an advanced version of DVC. Their approach focuses on performing essential tasks, including motion estimation, motion compression, motion compensation, and residual compression, in the feature domain instead of the pixel space. In the framework presented in [23], a cross-scale prediction module is incorporated to facilitate efficient motion compensation. Inspired by the observation that videos consist of a series of images with temporal redundancy, researchers [24], [25] have extended image compression networks by adopting a 3D autoencoder-based framework to handle video data. The primary objective of this approach is to exploit spatial-temporal redundancies in videos by utilizing spatiotemporal transformations. To further enhance the performance of these networks, Habibian *et al.* introduced a temporally conditional entropy model to leverage temporal correlations within the latent space.

In contrast to DVC, SSF, and FVC networks, which rely on a single previous frame as a reference frame, Lin *et al.* [26] propose a method that utilizes several previous frames to improve the accuracy of predicting the current frame. Mentzer *et al.* [27] propose a neural video compression model based on Generative Adversarial Networks (GANs) [28]. Their objective is to enhance the perceptual quality of the reconstructed frames by leveraging the power of GANs. More recently, Mentzer *et al.* [29]present a novel network that avoids explicit motion estimation. Instead, they leverage a temporal transformer for entropy modeling.

III. Methods

A. Overview

Our baseline model is the Scale-Space Flow (SSF) network [21], which is one of the popular low-latency video compression models. As shown in Fig. 2, it is comprised of I-frame compression and P-frame compression models. The P-frame compression model consists of two parts: motion compression network and residual compression network. These three main networks, i.e., I-frame compression, motion compression, and residual compression are based on the autoencoder network architecture [10].

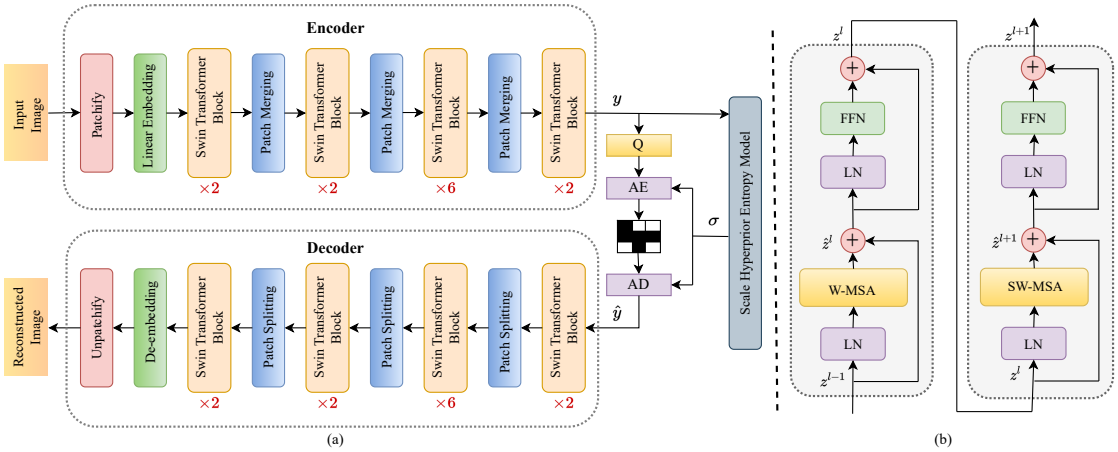

Fig. 3. Swin Transformer based architecture is designed for compressing I-frame, scale-space flow, and residual. (b) Two successive Swin Transformer blocks. Q shows scalar quantization. AE and AD refer to the arithmetic encoder and decoder, respectively.

The most important contribution of the SSF network is the generalization of optical flow to scale-space flow by adding a scale field to the motion field as a third channel. The scale field contributes to the model to blur the regions where disocclusion and fast motion exist, leading to a better inter-frame prediction. The inter-frame prediction is obtained by performing a trilinear warping on progressively blurred versions of the reference frame [21]. In the following sections, we will describe their novel components in P-frame compression.

*1) **Motion Compression**:* The proposed motion compression network utilizes an autoencoder-based architecture, where the input consists of the current frame $\boldsymbol{x_t}$ and the previous reconstruction frame $\hat{\boldsymbol{x}}_{\boldsymbol{t-1}}$. The encoder of the architecture is designed to jointly compute and encode the motion information that presents between the two consecutive frames. On the decoder side, the quantized latent representation of motion is decoded into three vectors: $F = (F_x, F_y, F_z)$. The first two vectors, F_x and F_y, represent horizontal and vertical motion vectors, respectively, and have dimensions of $\mathbb{R}^{2\times H\times W}$. The third vector, F_z, corresponds to a one-channel scale field with dimensions of $\mathbb{R}^{H\times W}$.

*2) **Motion Compensation**:* The motion compensation module plays a crucial role in predicting the current frame $\bar{\boldsymbol{x}}_{\boldsymbol{t}}$ using the reference frame $\hat{\boldsymbol{x}}_{\boldsymbol{t-1}}$ and the motion and scale fields. To achieve this, a scale-space warping operation is employed, where the reference frame undergoes progressive convolution with a Gaussian kernel. This convolution generates a series of blurred versions of the reference frame:

$$\boldsymbol{X} = [\hat{x}_{t-1}, \hat{x}_{t-1} * G(s_1), ..., \hat{x}_{t-1} * G(s_M)], \tag{1}$$

where $G(s_i)$ represents the Gaussian kernel with a scale parameter of s_i, the motion-compensated pixel value for each pixel located at the coordinate $[x, y]$ is obtained by applying trilinear interpolation in the scale-space volume. The process can be described as follows:

$$\begin{aligned} \bar{\boldsymbol{x}}_{\boldsymbol{t}} &= \text{Scale-Space-Warp}(\hat{x}_{t-1}, F) \\ \bar{\boldsymbol{x}}_{\boldsymbol{t}}[x, y] &= \boldsymbol{X}[x + F_x[x, y], y + F_y[x, y], F_z[x, y]]. \end{aligned} \tag{2}$$

B. *Transformer-based Architecture*

The Transformer [30] is originally proposed in the field of natural language processing (NLP) and had a profound impact on this domain. The remarkable accomplishments of the Transformer in NLP have motivated researchers to embrace the Transformer architecture in computer vision tasks. These tasks encompass a broad spectrum of applications, such as object detection [31], image classification [32], semantic segmentation [33], and numerous other applications [34]. ViT [35] is the first vision Transformer which utilizes a pure Transformer-based architecture for image classification and yields impressive results compared with traditional CNN networks [36], [37], [38], [39]. It splits an image into non-overlapping patches and captures long-range dependencies by using multi-head self-attention module [35]. ViT has a high computational complexity due to the globally computed self-attention. The Swin Transformer [40] is proposed to reduce the computational complexity from quadratic to linear with respect to the patch numbers. The computational complexity is reduced because the Swin Transformer calculates self-attention locally within non-overlapping windows. The Swin Transformer network is also able to produce hierarchical representation which is very necessary for dense prediction tasks [41]. We have used the Swin Transformer to build the encoders and decoders of the Scale-Space Flow network. Fig. 3(a) shows the Swin Transformer-based architecture which is used to compress the I-frame, residual and scale-space flow. We have also extended the Swin Transformer block to the Fused Local-aware Window (FLaWin) Transformer block to enhance preserving local information.

C. *Swin Transformer-based Encoder/Decoder*

*1) **Encoder**:* The Swin Transformer, a pivotal component utilized as an encoder in the architecture [40], comprises four essential blocks: Patchify, Linear Embedding, Swin Transformer block, and Patch merging. To initiate the encoding process, the input image $x \in R^{C_{in}\times H\times W}$ undergoes Patchify,

which divides it into non-overlapping patches. Secondly, these patches are flattened and mapped into an embedding space with dimension C by a linear embedding block. The output of these two blocks is then fed into the multiple Swin Transformer blocks and patch merging layers. The Swin Transformer block, a fundamental building block, is instrumental in maintaining the number of patches while efficiently extracting semantic features. This is achieved by performing local self-attention computations within each non-overlapping window, allowing the model to capture fine-grained meaningful information effectively. The patch merging layer generates hierarchical feature maps by halving the resolution of the feature map and doubling the channel number of the feature map, ensuring effective feature extraction and spatial information aggregation.

*2) **Decoder**:* The Swin Transformer decoder is the inverse symmetric of the encoder. We replace the patchify block with a unpatchify block, the patch merging layer with the patch splitting layer, and the linear embedding layer with a deembedding layer.

*3) **Swin Transformer Block**:* The Swin Transformer block is the main part of the Swin Transformer architecture. Unlike the traditional Transformer block which is composed of multi-head self-attention (MSA), the Swin Transformer block is built upon a window-based multi-head self-attention (W-MSA) which conducts self-attention within local windows. The Window-based multi-head self-attention decreases the computational complexity; however, it fails to take into account the information interaction across different windows. To remedy this issue, the shifted-window-based multi-head self-attention (SW-MSA) is employed after the W-MSA module. As shown in Fig. 3(b), the Swin Transformer block consists of a layer normalization (LN), window-based multi-head self-attention (W-MSA) or shifted-window-based multi-head self-attention (SW-MSA), residual connection and a Feed-Forward Network (FFN), including a 2-layer MLP with GELU function as the nonlinearity. The process of two consecutive Transformer blocks can be defined as follows:

$$\begin{aligned} \hat{\boldsymbol{z}}^{\boldsymbol{l}} &= W\text{-}MSA(LN(\boldsymbol{z}^{\boldsymbol{l-1}})) + \boldsymbol{z}^{\boldsymbol{l-1}}, \\ \boldsymbol{z}^{\boldsymbol{l}} &= MLP(LN(\hat{\boldsymbol{z}}^{\boldsymbol{l}})) + \hat{\boldsymbol{z}}^{\boldsymbol{l}}, \\ \hat{\boldsymbol{z}}^{\boldsymbol{l+1}} &= SW\text{-}MSA(LN(\boldsymbol{z}^{\boldsymbol{l}})) + \boldsymbol{z}^{\boldsymbol{l}}, \\ \boldsymbol{z}^{\boldsymbol{l+1}} &= MLP(LN(\hat{\boldsymbol{z}}^{\boldsymbol{l+1}})) + \hat{\boldsymbol{z}}^{\boldsymbol{l+1}}, \end{aligned} \tag{3}$$

where $\hat{\boldsymbol{z}}^{\boldsymbol{l}}$ and $\hat{\boldsymbol{z}}^{\boldsymbol{l+1}}$ show the outputs of W-MSA and SW-MSA of the *l*, and *l+1* blocks, respectively. The self-attention mechanism employed in W-MAS and SW-MSA can be formulated as follows:

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = softmax(\frac{\boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}}}{\sqrt{d}} + \boldsymbol{B})\boldsymbol{V}, \tag{4}$$

Where Q, K, and $V \in R^{M^2 \times d}$ show the query, key and value matrices respectively. The dimension of the key is denoted by d, and M^2 represents the number of patches in a window. The learnable relative position encoding is captured by the matrix B, which is derived from the bias matrix $B' \in \mathbb{R}^{(2M-1)\times(2M-1)}$ using learnable parameters. When there are K attention heads, the attention mechanism is applied K times in parallel, and the outputs of all heads are concatenated together. Finally, the concatenated outputs are linearly projected to obtain the final result.

D. Fused Local-aware Window Transformer Block

The feed-forward network (FFN) plays a crucial role in the Transformer block, known for its feature enhancement capabilities. In our proposed Transformer block, named Fused Local-aware Window (FLaWin), we replace the conventional FFN of the Swin Transformer block, comprising MLP layers, with our introduced fused local-aware feed forward (FLaFF) network. This incorporation of FLaFF in the Transformer block enables the capture of both local and long-range information while facilitating the extraction of diverse and multi-scale representations. Notably, the inclusion of local information is required for image compression tasks, where preserving fine-grained details is indispensable.

*1) **Fused Local-aware Feed Forward (FLaFF)**:* Our proposed FLaFF is composed of an Inception module which helps to extract local information and multi-scale representations. In the FLaFF architecture, as depicted in Fig. 4(b), first each token is passed through a linear projection layer, consisting of 1×1 convolution layers, to increase its dimension. Second, the tokens are reshaped to a 2D token map, which is well-suited for the Inception block. Third, the Inception block is employed to extract diverse and local information from the 2-D token maps in parallel. Fourth, the 2D token maps are flattened and passed to another linear layer to project and lower the dimension of the input channels.

As illustrated in Fig. 4(c), the Inception block operates by dividing the 2-D input along the channel dimension and directing these split components into three separate branches. Each branch involves a depth-wise convolution with a kernel size of 3×3. Utilizing depth-wise convolution in the Inception block offers two valuable benefits: it reduces computational complexity and enhances the modeling capabilities for channel attention. The convolution operations of these three branches are performed in parallel, and their outputs are concatenated along the channel dimension to form the final output of the Inception block. This architecture allows the FLaFF to effectively capture local details and diverse representations, significantly contributing to the overall performance.

E. Training Strategy

*1) **Loss Function**:* The rate-distortion loss is used to train our network. If the length of the sequence is T, the total loss can be written as [21]:

$$D+\lambda R = \sum_{t=0}^{T-1} d(x_t, \hat{x_t}) + \lambda[R(I_0) + \sum_{t=1}^{T-1} R(w_t) + R(r_t)], \tag{5}$$

where D represents the distortion measure, such as the Mean Squared Error (MSE) between the original and reconstructed frames. R denotes the bitrate to encode the quantized latent representation. I_0, w_t, r_t represents I-frame, scale-space flow

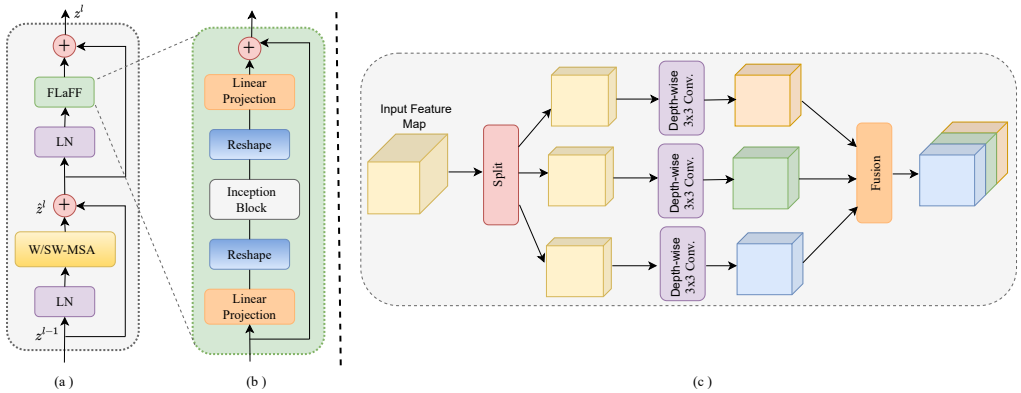

Fig. 4. (a) The architecture of FLaWin Transformer Block. (b) Fused local-aware feed-forward (FLaFF) network. (c) Structure of the proposed Inception block, which is used in FLaFF.

and residual latent, respectively. λ is the Lagrangian coefficient that controls the trade-off between rate and distortion.

*2) **Quantization**:* To do entropy coding, quantization process need to be replaced with a soft differentiable function to make the end-to-end training feasible. In this paper, we add uniform noise to latent representations [10] to approximate the hard quantization during training. In the test phase, hard quantization i.e., a rounding operation is employed.

*3) **Entropy Model**:* The entropy measurement should be used to estimate the bitrate for encoding the quantized latent representation. Therefore, it is required to estimate the probability distribution of quantized latent representation to compute the corresponding entropy. To do so, the hyper-prior network [16] is utilized to estimate the probability distribution. The hyper-prior network proposes a hyper-prior latent representation z, as side information to capture the latent representation's spatial dependencies. It results in computing the probability distribution of the latent representation precisely. The probability distribution of the quantized hyper-latent is estimated with a non-parametric fully factorized density model [16]. The probability of quantized latent $\hat{y}$ conditioned on quantized hyper-prior $\hat{z}$ is modeled by a zero-mean Gaussian distribution:

$$P_{\hat{y}|\hat{z}}(\hat{y}|\hat{z}) \sim \mathcal{N}(0,\, \sigma^2), \tag{6}$$

the scale parameter σ is determined by the decoded quantized hyper-prior $\hat{z}$.

IV. Experiments

A. Dataset

This research project relies on the extensive data collected during the Solar Dynamics Observatory (SDO) mission. The SDO mission is equipped with three instruments that operate continuously to capture essential information from the Sun [42], [43], [44], [45]. The Helioseismic and Magnetic Imager (HMI) is specifically designed to study oscillations and the magnetic field present on the solar surface, known as the photosphere [46]. It provides valuable insights into the dynamic behavior and magnetic properties of the Sun. The Atmospheric Imaging Assembly (AIA) captures full-sun images of the solar corona, covering a wide area of approximately 1.3 solar diameters. With a spatial resolution of around 1 arcsec, the AIA captures images at multiple wavelengths every 12 seconds [47]. This instrument offers detailed observations of the solar corona, which plays a significant role in understanding solar phenomena. To gain a deeper understanding of the variations that influence Earth's climate and near-Earth space, the Extreme Ultraviolet Variability Experiment (EVE) investigates the solar Extreme Ultraviolet (EUV) irradiance with high spectral precision [48].

The original SDO dataset has undergone preprocessing to generate a machine learning-ready dataset known as SDOML [49], which is used in this study. The SDOML dataset comprises AIA images captured at different wavelengths, including 94, 131, 171, 193, 211, 304, 335, 1600, and 1700 Å with a sampling rate of 6 minutes. In this paper, AIA images at the wavelength of 94 are utilized for both the training and testing phases. To train video compression networks on the SDOML dataset, we put four consecutive images together to make temporal chunks of four frames. Following the traditional codecs, during the test phase, we stack 30 consecutive images and create video clips with a GOP size of 30.

B. Implementation Details

During the training process, our models are trained with a wide range of hyperparameters $\lambda \in \{0.00125, 0.0025, 0.005, 0.01, 0.02, 0.04, 0.08, 0.160, 0.320\}$ to cover various rate and distortion scenarios. The training is conducted for 100 epochs, with batches of size 16. Each batch comprises randomly cropped patches with dimensions of 256x256, extracted from the original 512x512 images. To optimize the model, we employ the Adam optimizer [50] with an initial learning rate of 10^{-4}, which gradually decreases to 1.2×10^{-6} throughout the training process.

C. Ablation Study

To evaluate the effect of the Swin Transformer and how adding FLaFF to the Swin Transformer block help the video

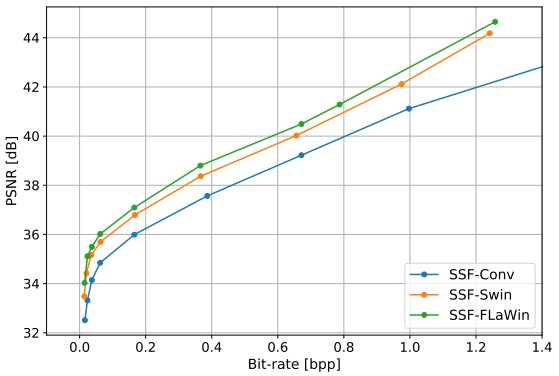

Fig. 5. The convolution-based model is compared with two different Transformer-based models, the Swin Transformer and FLaWin Transformer, in terms of the rate-distortion criteria.

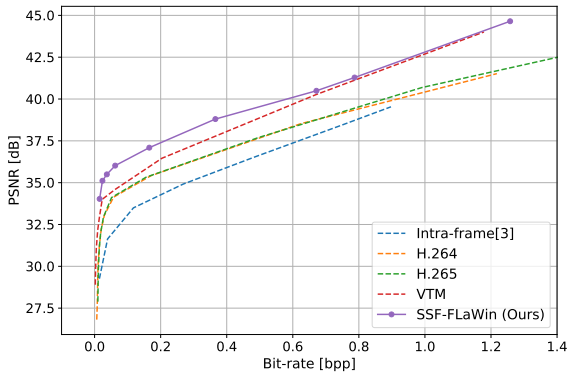

Fig. 6. Rate distortion curves on the test video clips. Distortion is measured by PSNR.

compression, we construct two versions of the Scale-Space Flow network. First, we replace the three convolutional autoencoders in I-frame and P-frame models with the Swin Transformer-based architecture. This model is called SSF-Swin. Then, we further enhance the Swin Transformer block by incorporating the proposed FLaWin Transformer block, which aims to improve the information extraction capability of the Transformer. This variant is termed SSF-FLaWin. As results are depicted in Fig. 5, the SSF-Swin network achieves better performance in terms of rate-distortion trade-off compared to the SSF-Conv model. The superior performance of the SSF-Swin can be attributed to its ability to exploit the long-range correlations within the data, enabling better exploitation of spatial and temporal redundancies. Moreover, the introduction of the FLaWin Transformer block in SSF-FLaWin further contributes to the compression performance. The architectural design of FLaWin allows for the simultaneous capture of local details and long-range correlations, leading to the extraction of diverse and informative representations.

D. Comparison with the Traditional Video Codecs

We conducted a comparison of the rate-distortion performance of our proposed network, SSF-FLaWin, with classical video compression standards and neural image compression on the SDOML dataset [3], [51], which serves as an equivalent to intra-frame compression. The distortion is measured by the Peak Signal-to-Noise Ratio (PSNR) metric. As depicted in Fig. 6, the rate-distortion performance of the neural video compression network, SSF-FLaWin, surpasses that of traditional video codecs such as H.264 and H.265, while achieving comparable performance with VTM [52]. These findings clearly demonstrate the effectiveness of the FLaWin Transformer block in enhancing video compression performance. Furthermore, our results strongly emphasize the considerable advantage of video compression over image compression when applied to the SDOML dataset. This highlights the effectiveness of exploiting the temporal redundancies inherent in video data, which leads to significantly improved compression efficiency.

V. Conclusion

we have presented a Transformer-based neural video compression approach for the NASA'SDO mission. Our experimental results have clearly demonstrated the effectiveness of applying video compression techniques to the dataset, resulting in improved compression ratios. This improvement can be attributed to the high temporal correlation observed between the images in the dataset. Additionally, we have conducted an in-depth investigation into the coding efficiency of the Swin Transformer and FLaWin Transformer-based networks. The findings indicate the potential of these architectures for achieving efficient video compression. Overall, our work highlights the benefits of utilizing advanced Transformer models for enhancing video compression in the context of the SDO mission.

References

[1] J. Schou, P. H. Scherrer, R. I. Bush, R. Wachter, S. Couvidat, M. C. Rabello-Soares, R. S. Bogart, J. Hoeksema, Y. Liu, T. Duvall *et al.*, "Design and ground calibration of the helioseismic and magnetic imager (HMI) instrument on the solar dynamics observatory (SDO)," *Solar Physics*, 2012.

[2] C. E. Fischer, D. Müller, and I. De Moortel, "JPEG2000 image compression on solar EUV images," *Solar Physics*, 2017.

[3] A. Zafari, A. Khoshkhahtinat, P. M. Mehta, N. M. Nasrabadi, B. J. Thompson, D. Da Silva, and M. S. Kirk, "Attention-based generative neural image compression on solar dynamics observatory," in *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2022, pp. 198–205.

[4] A. Zafari, A. Khoshkhahtinat, N. Nasrabadi, and P. Mehta, "Neural image compression on solar dynamics observatory," *The Third Triennial Earth-Sun Summit (TESS*, vol. 54, no. 7, 2022.

[5] Y. Yang, S. Mandt, and L. Theis, "An introduction to neural data compression," *CoRR*, 2022.

[6] S. Ma, X. Zhang, C. Jia, Z. Zhao, S. Wang, and S. Wang, "Image and video compression with neural networks: A review," *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[7] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function," *Neural networks*, 1993.

[8] V. K. Goyal, "Theoretical foundations of transform coding," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 9–21, 2001.

[9] J. Ballé, P. A. Chou, D. Minnen, S. Singh, N. Johnston, E. Agustsson, S. J. Hwang, and G. Toderici, "Nonlinear transform coding," *IEEE Journal of Selected Topics in Signal Processing*, vol. 15, no. 2, pp. 339–353, 2020.

[10] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *ICLR*, 2017.

[11] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," *Advances in neural information processing systems*, vol. 30, 2017.

[12] Y. Yang, R. Bamler, and S. Mandt, "Variational bayesian quantization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 10 670–10 680.

[13] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[14] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," *arXiv preprint arXiv:1611.01704*, 2016.

[15] D. S. Taubman and M. W. Marcellin, *JPEG2000 - image compression fundamentals, standards and practice*, ser. The Kluwer international series in engineering and computer science. Kluwer, 2002.

[16] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *ICLR*, 2018.

[17] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," *Advances in neural information processing systems*, vol. 31, 2018.

[18] O. Rippel, S. Nair, C. Lew, S. Branson, A. G. Anderson, and L. Bourdev, "Learned video compression," in *ICCV*, 2019.

[19] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, "DVC: An end-to-end deep video compression framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 006–11 015.

[20] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4161–4170.

[21] E. Agustsson, D. Minnen, N. Johnston, J. Balle, S. J. Hwang, and G. Toderici, "Scale-space flow for end-to-end optimized video compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8503–8512.

[22] Z. Hu, G. Lu, and D. Xu, "FVC: A new framework towards deep video compression in feature space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1502–1511.

[23] Z. Guo, R. Feng, Z. Zhang, X. Jin, and Z. Chen, "Learning cross-scale prediction for efficient neural video compression," *arXiv e-prints*, 2021.

[24] A. Habibian, T. v. Rozendaal, J. M. Tomczak, and T. S. Cohen, "Video compression with rate-distortion autoencoders," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7033–7042.

[25] J. Pessoa, H. Aidos, P. Tomás, and M. A. Figueiredo, "End-to-end learning of video compression using spatio-temporal autoencoders," in *2020 IEEE Workshop on Signal Processing Systems (SiPS)*. IEEE, 2020, pp. 1–6.

[26] J. Lin, D. Liu, H. Li, and F. Wu, "M-LVC: Multiple frames prediction for learned video compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3546–3554.

[27] F. Mentzer, E. Agustsson, J. Ballé, D. Minnen, N. Johnston, and G. Toderici, "Neural video compression using gans for detail synthesis and propagation," in *European Conference on Computer Vision*. Springer, 2022, pp. 562–578.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[29] F. Mentzer, G. Toderici, D. Minnen, S.-J. Hwang, S. Caelles, M. Lucic, and E. Agustsson, "VCT: A video compression transformer," *arXiv preprint arXiv:2206.07307*, 2022.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[31] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*. Springer, 2020, pp. 213–229.

[32] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International conference on machine learning*. PMLR, 2021, pp. 10 347–10 357.

[33] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, "End-to-end video instance segmentation with transformers," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 8741–8750.

[34] A. Zafari, A. Khoshkhahtinat, P. Mehta, M. S. E. Saadabadi, M. Akyash, and N. M. Nasrabadi, "Frequency disentangled features in neural image compression," in *2023 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2023, pp. 2815–2819.

[35] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[38] A. Sarlak, A. Razi, X. Chen, and R. Amin, "Diversity maximized scheduling in roadside units for traffic monitoring applications," in *2023 IEEE 48th Conference on Local Computer Networks (LCN)*. IEEE, 2023, pp. 1–4.

[39] B. Adami, S. Tehranipoor, N. M. Nasrabadi, and N. Karimian, "A universal anti-spoofing approach for contactless fingerprint biometric systems," in *2023 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2023, pp. 1–8.

[40] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021.

[41] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen, and J. Wang, "HRFormer:high-resolution transformer for dense prediction," *NeurIPS*, 2021.

[42] "A guide to the mission and purpose of nasa's solar dynamics observatory," 2010. [Online]. Available: https://sdo.gsfc.nasa.gov/assets/docs/SDO_Guide.pdf

[43] R. Nematirad and A. Pahwa, "Solar radiation forecasting using artificial neural networks considering feature selection," in *2022 IEEE Kansas Power and Energy Conference (KPEC)*. IEEE, 2022, pp. 1–4.

[44] P. Bhuvela and A. Nasiri, "Design methodology for a medium voltage single stage llc resonant solar pv inverter."

[45] H. Taghavi, A. El Shafei, and A. Nasiri, "Liquid cooling system for a high power, medium frequency, and medium voltage isolated power converter."

[46] J. Schou, P. H. Scherrer, R. I. Bush, R. Wachter, S. Couvidat, M. C. Rabello-Soares, R. Bogart, J. Hoeksema, Y. Liu, T. Duvall *et al.*, "Design and ground calibration of the Helioseismic and Magnetic Imager (HMI) instrument on the Solar Dynamics Observatory (SDO)," *Solar Physics*, vol. 275, pp. 229–259, 2012.

[47] J. R. Lemen, A. M. Title, D. J. Akin, P. F. Boerner, C. Chou, J. F. Drake, D. W. Duncan, C. G. Edwards, F. M. Friedlaender, G. F. Heyman *et al.*, "The atmospheric imaging assembly (AIA) on the solar dynamics observatory (SDO)," *Solar Physics*, vol. 275, pp. 17–40, 2012.

[48] T. N. Woods, F. Eparvier, R. Hock, A. Jones, D. Woodraska, D. Judge, L. Didkovsky, J. Lean, J. Mariska, H. Warren *et al.*, "Extreme Ultraviolet Variability Experiment (EVE) on the Solar Dynamics Observatory (SDO): Overview of science objectives, instrument design, data products, and model developments," *The solar dynamics observatory*, pp. 115–143, 2012.

[49] R. Galvez, D. F. Fouhey, M. Jin, A. Szenicer, A. Muñoz-Jaramillo, M. C. Cheung, P. J. Wright, M. G. Bobra, Y. Liu, J. Mason *et al.*, "A machine-learning data set prepared from the NASA solar dynamics observatory mission," *The Astrophysical Journal Supplement Series*, 2019.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[51] A. Zafari, A. Khoshkhahtinat, P. M. Mehta, N. Nasrabadi, B. J. Thompson, D. da Silva, and M. Kirk, "Attention-based generative neural image compression on solar dynamics observatory," in *103rd AMS Annual Meeting*. AMS, 2023.

[52] "Versatile Video Coding Reference Software," Available at https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM, 2022.