Doctor of Philosophy Thesis

Université de Limoges

University of Limoges

ED 653 – Science and Engineering of Systems - XLIM

XLIM Research Institute

Dissertation submitted for the degree of

Doctor of Philosophy

Science and Engineering for Information

Presented and Publicly defended by

Wael SAIDENI

On October 13th, 2022

Optimization of Video Surveillance Networks in Smart Cities Using Video Compressive Sensing and Deep Learning Techniques

Thesis supervised by **Jean-Pierre CANCES**, **David HELBERT** and **Fabien COURREGES**

Committee:

Reporters

Mr. **Raphael COUTURIER**, Professor, Bourgogne-Franche-Comté University, and Senior Researcher, FEMTO laboratory, Bourgogne-Franche-Comté

Ms. **Sylvie TREUILLET**, Associate professor (HDR), Polytech Orléans University, and Senior Researcher, PRISME laboratory, Orléans

Reviewers

Mr. **David HELBERT**, Associate professor (HDR), Poitiers University, Poitiers

Mr. **Jean Pierre CANCES**, Professor, ENSIL/ENSCI Limoges, Limoges

Mr. **Fabien COURREGES**, Associate professor, IUT of Brive, Brive

Mr. **Mathieu CRUSSIERE**, Professor, INSA Rennes University, and Senior Researcher, IETR Laboratory, Rennes

Guests

Mr. **Karim TAMINE**, Associate professor, Limoges University, Limoges

Mr. **Thomas FROMENTEZE**, Associate professor, Limoges University, Limoges

To my grandfather **Mohsen Ben Hammouda Saoudi**

“Without data you’re just another person with an opinion”
W. Edwards DEMING

Acknowledgements

This research work was carried out in XLIM research institute within SRI (Intelligent Systems and Networks) department with RUBIH team.

First of all, I would like to express my gratitude to my supervisors Mr. David HELBERT, Associate professor (HDR) at Poitiers University, Mr. Jean Pierre CANCES, Professor at ENSIL/ENSCI Limoges, and Mr. Fabien COURREGES, Associate professor at IUT of Brive, for having accompanied me throughout these years. I thank them for the time they devoted to me to supervise the progress of this thesis. Their advice and judicious criticisms obviously contributed to the progress of this work.

I would also like to express my gratitude to Mr. Raphael COURTURIER, Professor at Bourgogne-Franche-Comté University and Senior Researcher at FEMTO laboratory of Bourgogne-Franche-Comté, and Ms. Sylvie TREUILLET, Associate professor (HDR) at Polytech Orléans University and Senior Researcher at PRISME laboratory in Orléans, for their interest to my work and being the scientific reviewers of this thesis.

My thanks also go to Mr. Mathieu CRUSSIERE, Professor at INSA Rennes University and Senior Researcher at IETR Laboratory in Rennes for the honor they have done me by accepting to examinate my work.

I am very grateful to Mr. Karim TAMINE, Associate professor at Limoges University and Mr. Thomas FROMENTEZE, Associate professor at Limoges University, for accepting the invitation and attending my thesis defense.

It is extremely pleasant to thank Mrs. Patricia LEROY for having taken care of all the administrative procedures to the good progress of my thesis. My gratitude also goes to the whole RUBIH team for their good mood and the friendly environment in which I spent all these years.

A big thank to all my friends, in particular Sana, Amira, Omar, Khaled, Bilel, Boutheyna and Abdou who never stopped supporting and encouraging me during the most difficult moments.

No words can express my gratitude to all my family (my grandmother Jamila, my mother Lamia, my brother Louay, my aunties Soumaya and Boutheina and my uncles Mohamed and Hafedh) for their support to complete this long journey.

And a special thanks to my love Sarra for always dreaming with me.

Copyrights

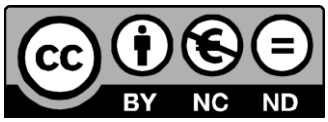


Table of content

Chapter II. Comparison Study of Deep Learning based approaches in Video Compressive Sensing

Chapter III. Video Compressive Sensing based on a novel video prediction framework

Chapter IV. Video Compressive Sensing based on Vision Transformers

List of figures

Introduction

Chapter I. Introduction to Compressive Sensing and Deep Learning

Chapter II. Comparison Study of Deep Learning based approaches in Video Compressive Sensing

Chapter III. Video Compressive Sensing based on a novel video prediction framework

Chapter IV. Video Compressive Sensing based on Vision Transformers

List of tables

Chapter II. Comparison Study of Deep Learning based approaches in Video Compressive Sensing

Chapter III. Video Compressive Sensing based on a novel video prediction framework

Chapter IV. Video Compressive Sensing based on Vision Transformers

List of acronyms

ADMM-Net	Alternating Direction Method of Multiplayers
AE	Auto-Encoders
AMP	Approximation Message Passing
ANN	Artificial Neural Networks
BCS	Bayesian compressive sensing
BIRNAT	BIdirectional Recurrent Neurol networks with Adversarial Training
BM3D	Block-Matching and 3D filtering
BP	Basis Pursuit
BPDN	Basis Pursuit DeNoising
CACTI	Coded Aperture Compressive Temporal Imaging
CCD	Charge Coupled Device
cGAN	conditional Generative Adversarial Networks
CNN	Convolutional Neural Network
ConvGRU	Convolutional Gated Recurrent Units
ConvLSTM	Convolutional Long Short-Term Memory
CoSaMP	Compressive Sampling Matching Pursuit
CR	Compression Ratio
CS	Compressive Sensing
CS-MRI	CS Magnetic Resonance Imaging
CSVideoNet	Compressive Sensing Video Network
DAVIS2017	Densely Annotated VIdeo Segmentation 2017
DCAN	Deep Convolutional Autoencoder Network
DCT	Discrete Cosine Transform
DeepUnfoldVCS	Deep Unfold Video Compressive Sensing
DE-RNN	DEnoising Recurrent Neural Networks
DeSCI	Denoising Snapshot Compressive Imaging
DL	Dictionary Learning
DMD	Digital Micromirror Device
DR2-Net	Deep Residual reconstruction Network
DVF	Deep Voxel Flow

DWT	Discrete Wavelet Tranform
E2E-CNN	End-to-End Convolutional Neural Network
ECG	EleCtrocardioGram
ELP-Unfolding	Ensemble Learning Priors
FBS	Forward-Backward Splitting
FFDNet	Fast and Flexible Denoising convolutional neural Network
FFN	Feed Forward Network
FISTA	Fast Iterative thresholding Algorithm
FPA-CS	Focal Plane array-based Compressive Imaging
FSTN	Flexible Spatio-Temporal Network
GAN	Generative Adversarial Network
GAP	Generalized Alternating Projection
GAP-FastDVDNet	Generalized Alternating Projection-Fast Deep Video Denoising Network
GAP-TV	Generalized Alternating Projection Total Variation
GMM	Gaussian Mixture Model
GPU	Graphical Processing Unit
GRUs	Gated Recurrent Units
IoT	Internet of Things
ISTA-Net	Iterative Shrinkage Thresholding Algorithm-based Network
JPEG	Joint Photographic Experts Group
K-SVD	K-Singular Value Decomposition
LASSO	Least Absolute Shrinkage and Selection Operator
LiSens	Line-SENSor-based compressive camera
LPIPS	Learned Perceptual Image Patch Similarity
LSTM	Long Short-Term Memory
MC-BCS-SPL	Motion Componsated Block Compressed Sensing with Smoothed Projected Landweber
MCS	Mobile Crowd Sensing
MEDYBAT	Modélisation Energétique DYnamique d'un BATiment
MetaSCI	Meta Modulated CNN for SCI reconstruction
MetaSCI-Net	Meta SCI Network
MIMO	Multiple Input and Multiple Output
MLP	Multi-Layer Perceptron

MMV	Multiple Measurement Vector
MNIST	Modified National Institute of Standards and Technology
MP	Matching Pursuit
MPEG	Moving Pictures Experts Group
MRI	Magnetic Resonance Imaging
MSE	Mean Square Error
MWCNN	Multilevel Wavelet Convolutional Neural Network
NLP	Natural Language Processing
NN	Neural Networks
OMP	Orthogonal Matching Pursuits
PE	Positional Encoding
PnP	Plug-and-Play
Pnp-FastDVDNet	Plug-and-Play-FastDVDNet
PnP-FFDNet	Plug-and-play-FFDNet
PredNet	Predictive Neural Network
PredRNN	Predictive Recurrent Neural Network
PSNR	Peak-Signal-to-Noise Ratio
RAMCES	Réseau Avancé de Mesure de Consommation Energétique et Supervision
ReconNet	Reconstruction Network
ReLU	Rectified Linear Unit
ResBlock	Residual Block
RevSCI	Reversible Snapshot Compressive Imaging
RevSCI-Net	RevSCI Network
RIP	Restricted Isometry Property
RL	Reinforcement Learning
RMSE	RootMean-Square Error
RNN	Recurrent Neural Network
Robust-ST-ConvLSTM	Robust SpatioTemporal Convolutional Long Short-Term Memory
ROMP	Regularized Orthogonal Matching Pursuits
SCADA	Supervisory Control And Data Acquisition
SCI	Snapshot Compressive Imaging
SD	Standard spatial multiplexing cameras Deviation

SDA	Stacked Denoising Autoencoder
SDA-CS	Stacked Denoising Autoencoder-Compressive Sensing
SGD	Stochastic Gradient Descent
SMC	Spatial Multiplexing Cameras
SMV	Single Measurement Vector
SP	Subspace Pursuit
SPC	Single Pixel Cameras
SPI	Single-Pixel Imaging
SSIM	Structural SIMilarity index
ST-ConvMHA	Spatio-Temporal Convolutional Multi-Head Attention
STM	SpatioTemporal Memory
StOMP	Stagewise Orthogonal Matching Pursuits
SVCS	Spatial Video Compressive Sensing
TISTA	Trainable ISTA Network
TMC	Temporal Multiplexing Cameras
TV	Total Variation
TVCS	Temporal Video Compressive Sensing
TwIST	Two-Step Iterative Shrinkage/Thresholding Algorithm
VCS	Video Compressive Sensing
VCS-RRS	VCS Reconstruction via Reweighted Residual Sparsity
ViT-SCI	Video Transformer for Snapshot Compressive Imaging
WSN	Wireless Sensor Network
Znet	Z-Order Recurrent Networks

Introduction

In three years, the global data information is expected to reach 181 zettabytes– the equivalent of 181 trillion gigabytes, which is 11 times more than the projected storage capacities as illustrated in Figure 1 and Figure 2. Specifically, smart sensors and phones are the main data producers of the global big data. To meet the huge demand for data storage, around 100 new colossal data centers are built every two years. However, it is not enough, and alarming studies expect that the number of digital bits would reach an impossible value, exceeding the number of all atoms on Earth in 150 years.

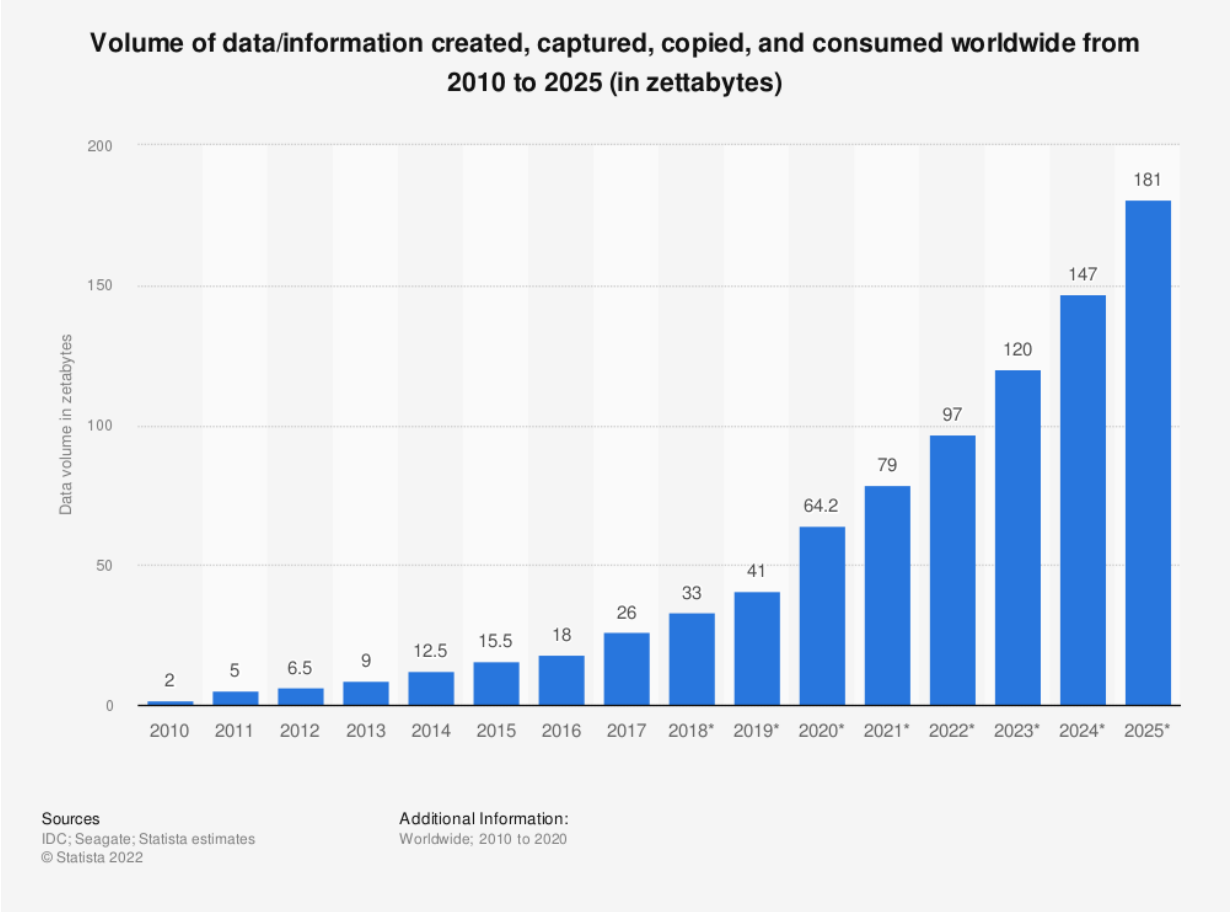


Figure 1: Total amount of data created, captured, copied, and consumed in the world from 2010 to 2025 [1].

Therefore, several challenges arise to deal with the serious problem of Big Data worldwide. On the one hand, high power efficiency at sensors level is required when processing and transmitting very large amount of data. On the other hand, it has been demonstrated from recent advances in Deep Learning and Machine Learning fields that Big Data can be a powerful weapon in many applications: large training datasets combined with robust models and great computational resources open the gate to many breakthroughs like smart object detection, intelligent decision-making systems, and smart Internet of Things (IoT) platforms.

In this context, implementing an IoT platform in Brive-la-Gaillarde has been an obvious research direction to experiment several approaches. These experiments aim to optimize the process of acquiring, transmitting, and reconstructing data in wireless sensor networks which enables to optimize the energy consumption of wireless devices and increases their autonomy.

Indeed, the idea of our IoT platform is based on the smart sensors that build up the acquisition layer. Data gathering is executed using several sophisticated IoT sensors that are deployed in different locations to collect many types of data over an extended period of time depending on the application. Collected data from different devices are usually huge and carry some redundant information. So, the idea is to firstly transmit data to a pre-processing unit with sufficient computational performances to extract meaningful features. The pre-processing unit is considered as a gateway to handle important information to the server rather than transmitting the entire information, which can remarkably reduce the system bandwidth. Therefore, the so called “Compressive Sensing” technique introduces a promising model to be explored in many IoT use cases.

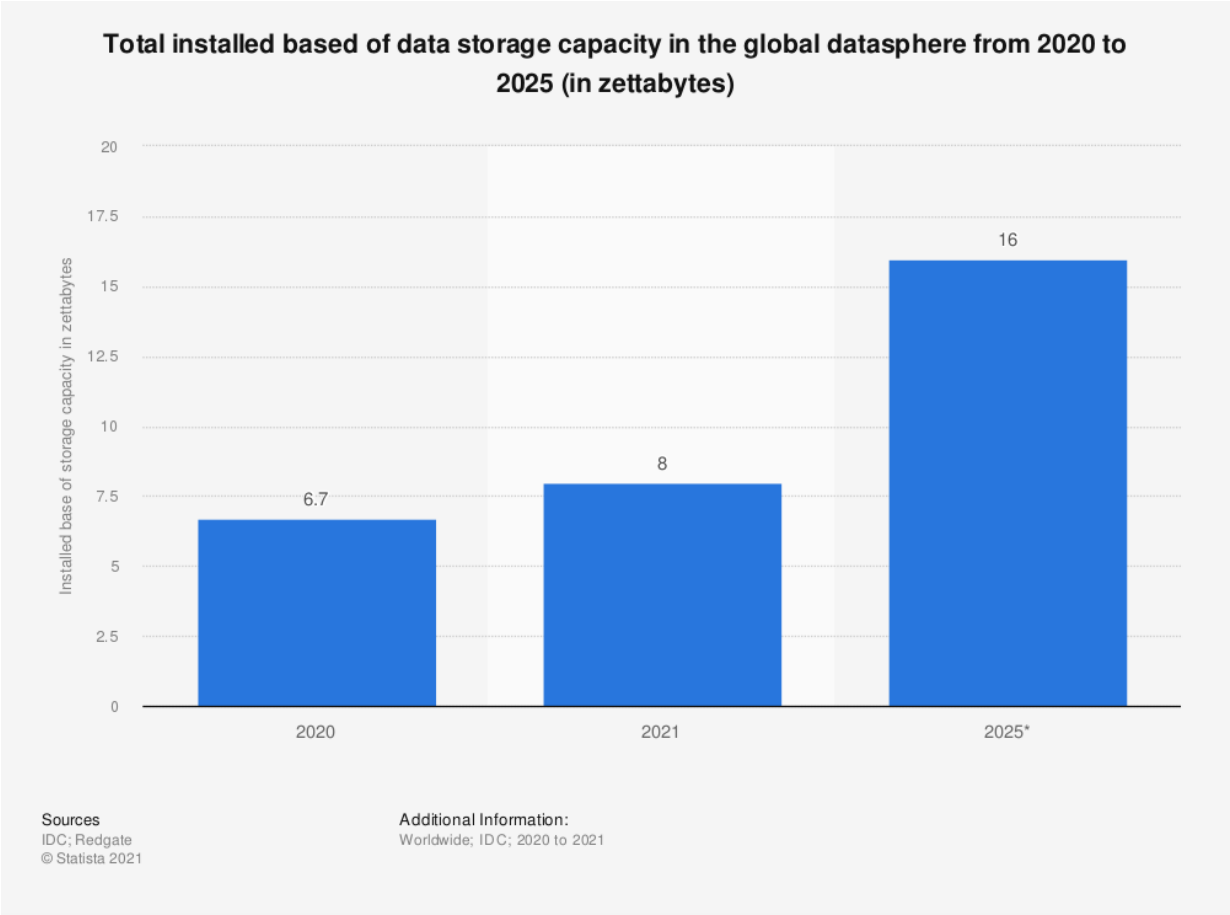


Figure 2: Total data storage capacity of all databases installed in the global datasphere from 2020 to 2025 [2].

The use of compressive sensing in several applications has allowed to capture impressive results, especially in various applications such as image and video processing and it has become a promising direction of scientific research. It provides an extensive application value in optimizing video surveillance networks. Conventional video compression techniques, such as Joint Photographic Experts Group (JPEG), Moving Pictures Experts Group (MPEG) standards and H.264 [3]-[4], are well developed and commonly used. Although these compression techniques are efficiently using data redundancies to compress video data, they are computationally asymmetric since they are composed of complex encoders and very simple decoders. Indeed, the encoding step is 5 to 10 times more complex than the decoding step [5]. However, in order to effectively exploit compression techniques in WSN, the computational cost have to move from the encoding sensor to the decoding server in an optimal manner. In our research project, we are interested in a video surveillance context where hundreds of camera

systems may be deployed. Thus, it is important to reduce the computational cost of the sensing components by reducing the complexity of the encoding process. On a large scale, using current video surveillance to encode video data in real-world system can dramatically increase its computational cost. This leads also to an effective higher power consumption at the sensor level. That's why, there is an urgent need for new video transmission techniques in smart cities.

Therefore, to address this issue: one main signal acquisition and compression technique has been developed: The Compressive Sensing (CS). It is used to compress and transmit sparse signals with a sampling rate much lower that the famous Shannon-Nyquist sampling theorem and enable an effective reconstruction of the original signal with very good quality performances.

In this project, we aim to provide solutions to energy constraints in WSN, in particular for video surveillance purposes. The idea is to design and implement a sophisticated framework to collect, transmit and store data from wireless video sensors placed in a wireless sensor network platform dedicated to smart buildings that is already deployed in the campus of Brive-la-Gaillarde, presented in Figure 3. The campus smart grid is composed of several blocks: SCADA (or Supervisory Control And Data Acquisition) system that is in charge of retrieving the measurements from the sensor network MEDYBAT (Modélisation Energétique DYnamique d'un BATiment) and RAMCES (Réseau Avancé de Mesure de Consommation Energétique et supervision). Indeed, SCADA, used as interface between users and the processes involved in the system to control its main functionalities, allows to:

- Generate graphics and reports using historical data,
- Detect alarm and automatically record events,
- Control the process.

SCADA is based on ScadaBR which is an open-source tool used in IoT related tasks to store and process measurements. ScadaBR enables to collect data from MEDYBAT and RAMCES. While RAMCES is a network which consists of measurement stations providing electrical measurements, boiler room parameters and gas consumption, MEDYBAT is the building's sensor network allowing to measure parameters like temperature, humidity, and luminosity. The idea behind this thesis is to add wireless surveillance systems to MEDYBAT in order to improve the campus traffic management and people mobility, making the university of Limoges safer and more efficient for every visitor. However, we intend to exploit Video Compressive Sensing (VCS) to enhance the energy consumption of different surveillance nodes and optimize the acquisition, transmission and recovery processes using advanced Deep Learning techniques. Indeed, Deep Learning is getting a lot of attention in Big Data related problems and our purpose in this thesis is to devise how to mix both CS and Deep Learning techniques to gain the benefits of both these approaches and achieve unprecedented performance in video compression within a WSN.

The major contributions of this research work are summarized as follows:

- A complete comparison study of recent Deep Learning-based research works in a video compressive sensing context is provided in Chapter 2. These works have been classified into different categories. This comparison aims to overview the current approaches video compressive sensing and demonstrate their powerful impact in computer vision applications when using well designed compressive sensing algorithms.
- A novel video prediction algorithm called "Robust Spatio Temporal Convolutional Long Short-Term Memory" (Robust-ST-ConvLSTM) is introduced. It is a memory flow algorithm based on higher order ConvLSTM. This memory flow algorithm is holding the

spatiotemporal information to optimize and control the prediction abilities of the ConvLSTM cell. This algorithm was developed for a specific compressive sensing context. However, some limitations, discussed in Chapter 3, have prevented us to extend the work to video compressive sensing applications.

- A complete framework of Video Compressive Sensing (VCS), from capturing a sequence of video frames in one single compressed measurement to reconstructing the original frames, is studied in Chapter 4. In this work, we present the first end-to-end sampling and recovery network built upon Transformers which are recently explored in vision related tasks to capture long-range spatio-temporal relations. Our proposed Video Transformer for Snapshot Compressive Imaging recovery (ViT-SCI) is based on Spatio-temporal Convolutional Multi-Head Attention (ST-ConvMHA) which is an extended version of the fully-connected attention adapted for vision problems. Our comprehensive qualitative and quantitative experiments on several datasets demonstrate that ViT-SCI outperforms previous state of the art methods with much faster reconstruction capacities, which pave the way for applying our algorithm in real-time applications. Indeed, ViT-SCI achieves high quality reconstruction on 64 × 64 video frames at the unprecedented rate of 1 frame per *ms*. In addition, an important ablation study on the Transformer network is provided to inspire future research works aiming to test the abilities of Transformers in vision tasks.

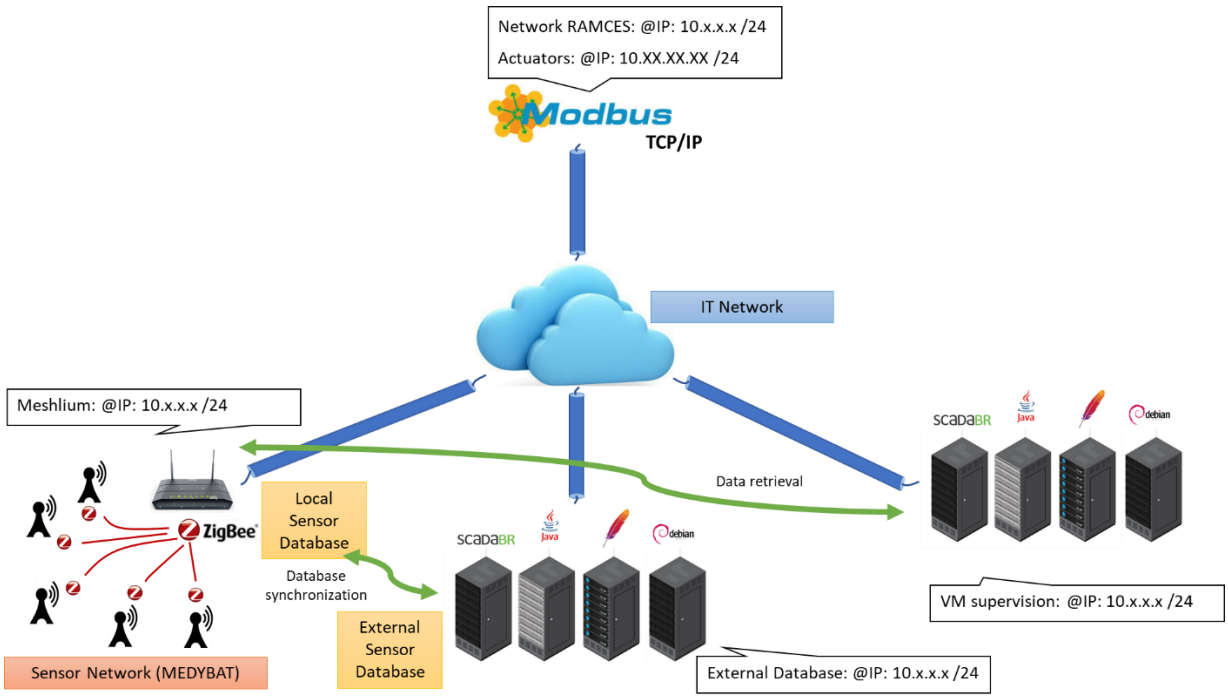


Figure 3: Architecture of the wireless sensor network platform dedicated to smart buildings deployed in the campus of Brive-la-Gaillarde, France.

References

[1] IDC, und Statista. “Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025 (in zettabytes).” Chart. June 7, 2021. Statista. Accessed April 13, 2022. https://www.statista.com/statistics/871513/worldwide-data-created/

[2] IDC, und Statista. “Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2025 (in zettabytes).” Chart. June 7, 2021. Statista. Accessed April 13, 2022. https://www.statista.com/statistics/871513/worldwide-data-created/

[3] Ev, Thomas Wiegand, and Gary J. Sullivan. “The H. 264/MPEG4 advanced video coding standard and its applications.” *IEEE communications magazine* 44, no. 8 (2006): 134-143.

[4] Sikora, Thomas. ”MPEG digital video-coding standards.” *IEEE signal processing magazine* 14, no. 5 (1997): 82-100.

[5] Wiegand, Thomas, Gary J. Sullivan, Gisle Bjontegaard, and Ajay Luthra. “Overview of the H. 264/AVC video coding standard.” *IEEE Transactions on circuits and systems for video technology* 13, no. 7 (2003): 560-576.

Table of content

Chapter I: Introduction to Compressive Sensing and Deep Learning

Chapter I. Introduction to Compressive Sensing and Deep Learning

I.1. Introduction

Conventional sensors are based on the sampling theorem of Shannon–Nyquist which is based on the following principle: the minimum sampling frequency of a signal that does not distort its underlying information, should be the double of its highest frequency component. However, this theorem which imposes an unnecessary high sampling rate is becoming outdated for applications that require a large amount of data. Thus, the Compressive Sensing paradigm seeks to decrease the rate of the Shannon–Nyquist principle and meets the expectations of the massive data-intensive applications. To keep it simple, for our application case, a CS camera takes several measurements coded from the scene much smaller than the number of reconstructed pixels. In fact, CS is an approach that facilitates the efficient acquisition of the sparse signals where detection and compression are performed at the same time. In this research work, we aim to optimize video compressive sensing frameworks by exploiting Deep Learning-based architectures. Therefore, some concepts must be introduced before presenting the main contributions of the thesis project. Indeed, in this chapter, Section I.2 provides a general introduction to the mathematical background behind compressive sensing and its main optimized-based approaches. In section I.3, we present recent Image Compressive Sensing methods that can be extended to be applied in a Video Compressive Sensing context. In Section I.4, we introduce the main applications of Compressive Sensing. Finally, Section I.5 presents the key elements of Deep Learning that will be exploited in this thesis.

I.2. Key elements of Compressive Sensing

In this section, mathematical background of Compressive Sensing will be detailed. In addition, some well-known optimized based reconstruction algorithms will be highlighted.

I.2.1. Definition

Compressive Sensing (CS) is a revolutionary mathematical theory in combining compression with sampling. In traditional methods, the compression step is executed after sampling the whole signal. However, CS introduces a framework, illustrated in Figure I.1, for spare signals to be efficiently recovered from a limited number of linear and non-adaptive measurements [I.1]-[I.2]-[I.3]. Indeed, CS has considerably surpassed the Shannon-Nyquist sampling theorem in terms of the required number of measurements for a reliable reconstruction [I.4]. In addition, this number depends on the design measurements and the signal's sparsity. However, the recovery process is non-linear and needs a specific undetermined system of equations to be decoded. CS supposes the sparsity or the low dimensionality of a model during recovery steps to limit the input signal to a small segment of the vector space, which enables the reconstruction of original signals from a small number of measurements [I.5].

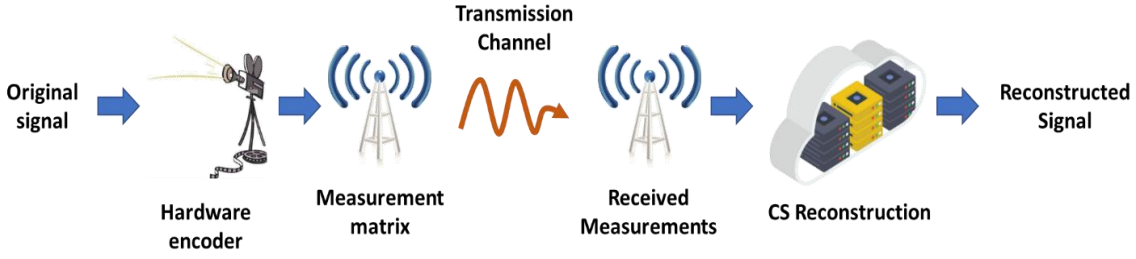


Figure I.1: Block diagram of the basic Compressive sensing framework.

I.2.2. Mathematical Introduction

To understand the mathematics behind the CS technique we recall here some basic principles: Instead of acquiring $\boldsymbol{N}$ samples of a signal $\boldsymbol{x} \in \mathbb{R}^{N\times 1}$, $\boldsymbol{M}$ random measurements are acquired with $\boldsymbol{M} \ll \boldsymbol{N}$ (CS theory states that the number of measurements sufficient to reconstruct the signal $\boldsymbol{x}$ is $\boldsymbol{M} = O(K\,log(N/K))$ such that (I.1):

$$\boldsymbol{y} = \boldsymbol{\Phi x}, \tag{I.1}$$

where $\boldsymbol{y} \in \mathbb{R}^{M\times 1}$is the known compressed measurement vector and $\boldsymbol{\Phi} \in \mathbb{R}^{M\times N}$ is the sensing matrix that will be discussed in section I.2.2. To recover the signal $\boldsymbol{x}$ given $\boldsymbol{y}$ and $\boldsymbol{\Phi}$, $\boldsymbol{x}$ must be sparse in a given base $\boldsymbol{\Psi}$ (I.2):

$$\boldsymbol{x} = \boldsymbol{\Psi s}, \tag{I.2}$$

where $\boldsymbol{s}$ is a K-sparse signal which means that $\mathbf{s}$ has at most K non-zero elements. From (I.1) and (I.2), we have (I.3):

$$\boldsymbol{y} = \mathbf{As}, \tag{I.3}$$

where $\boldsymbol{A} = \boldsymbol{\Phi\Psi}$. Figure I.2 illustrates the compressed sensing framework. However, the reconstruction of $\boldsymbol{x}$ or $\boldsymbol{s}$ from $\boldsymbol{y}$ is not possible. Therefore, an approximate solution can be obtained by solving the following ℓ_1 minimization problem which is a good approximation to the original ℓ_0 minimization problem (NP-hard problem) [I.6]-[I.7] (I.4):

$$\hat{\boldsymbol{s}} = argmin\,\|\boldsymbol{s}\|_1\; s.t.\, y = \boldsymbol{\Phi\Psi s}. \tag{I.4}$$

To reconstruct $\mathbf{s}$ from $\boldsymbol{y}$, CS algorithms can use different reconstruction approaches that will be discussed in Section I.2.4. Then $\boldsymbol{x}$ can be reconstructed from $\hat{\boldsymbol{x}} = \boldsymbol{\Psi}\hat{\boldsymbol{s}}$.

Since there is only one measurement vector, the above problem is generally referred to as a Single Measurement Vector (SMV) problem in the compressive sensing. However, when the input becomes a 3D signal (video) instead of 1D signal, the SMV problem becomes a Multiple Measurement Vector (MMV) problem. The sparse vector $\boldsymbol{s}$ becomes in this case a set of vectors $\boldsymbol{s_i}$ which must be recovered jointly from a set of measurement vectors $\mathbf{y_i}$ [I.8].

The set of the known measurement vectors $\mathbf{y_i}$ can correspond to different frames of the video signal. In fact, the video could be cut into series of images and then each image obtained could be associated to a measurement vector $\mathbf{y_i}$ and then it is possible to apply MMV model on the video. Therefore, the common approach used to deal with sequence data is Recurrent neural networks (RNN). However, RNN work well when we are dealing with short-term dependencies. In other words, these neural networks remember things for short periods of time and if a lot of information has been entered, it suffers from important losses. This problem could be solved by applying a modified version of the RNN: LSTM (Long Short-Term Memory) [I.9]. The advantage of LSTM is that it avoids the problem of long-term dependency i.e., it allows to remember information for a long period of time.

As a result, and in agreement with CS properties, CS has a great potential to be applied to images and videos because of their huge spatial and temporal redundancies which allow to have sparse representations to enable their reconstruction.

Nevertheless, RNNs are not the only Deep Learning approach experimented in video compressive sensing recovery phase. Indeed, many methods will be discussed in the following sections.

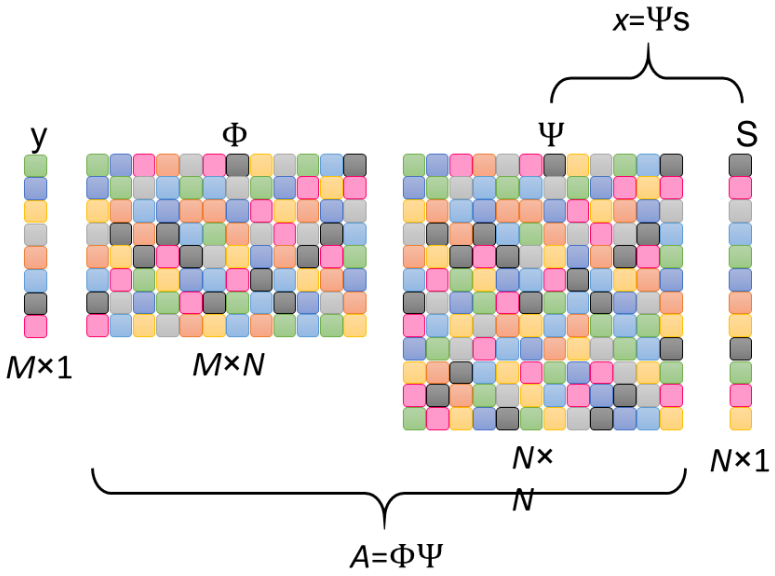


Figure I.2 : Compressive sensing framework.

I.2.3. Sensing Matrix

One of the most interesting research directions in compressive sensing is the construction of the sensing matrices. Indeed, the sensing matrix must satisfy some constraints. Firstly, it should be coherent with the sparsifying matrix **Ψ** to capture the salient 3 information of the initial signal with the minimum number of projections. Secondly, it may satisfy the restricted isometry property (RIP) to preserve the original signal main information in the compression process. However, it has been proved in [I.10] that RIP property is not always required to hold neither the sparsity level in a CS context, nor the random model of a signal. In addition, for real-time applications and low power requirements, we should design low complexity and hardware friendly sensing matrices. In most works, especially for those who are focusing on the reconstruction stage, the problem of the sampling matrix is not discussed since it is chosen as a random matrix such as Gaussian or Bernoulli matrix which meets the restricted isometry property (RIP) of CS. Although random matrices are easy to implement and can ensure better reconstruction results, they have many disadvantages. In fact, they require large storage resources and the recovery process may be difficult when dealing with large signal dimensions [I.11]. It can also be chosen as circulant sensing matrix [I.12]. However, other researchers use some features of the original input to design these matrices which is known as data-driven sampling matrix design. Other works are oriented to binary and bipolar sampling matrices that can be easily implemented on hardware devices and they do not require large computation resources.

I.2.4. Reconstruction Algorithms

The reconstruction process is the key to efficiently incorporate compressive sensing in real-world applications. Therefore, designing and implementing new optimization algorithms is the major concern of CS researchers. These algorithms can be categorized into several categories. In this section, we will cover the main two types of the recovery algorithms in CS: convex optimization algorithms and greedy algorithms.

I.2.4.1. Convex Optimization

To reconstruct the original signal $\boldsymbol{x}$, the trivial approach is to solve the l_0 minimization problem (I.5):

$$\hat{\boldsymbol{x}} = \underset{x}{\operatorname{argmin}} \|\boldsymbol{x}\|_0 \; s.t. \; \boldsymbol{y} = \boldsymbol{\Phi x}. \tag{I.5}$$

Since, l_0 minimization is an NP-hard problem for large-scale matrices, in our case $\boldsymbol{\Phi}$ is computationally complex, ℓ_1 minimization process is proposed to overcome the limitations of ℓ_0. In this case, the minimization problem, known as basis pursuit (BP) [I.13], becomes (I.6):

$$\hat{\boldsymbol{x}} = \underset{x}{\operatorname{argmin}} \|\boldsymbol{x}\|_1 \; s.t. \; \boldsymbol{y} = \boldsymbol{\Phi x}. \tag{I.6}$$

Another approach called basis pursuit denoising (BPDN) [I.14] is adapted when dealing with noisy systems. In addition, Least Absolute Shrinkage and Selection Operator (LASSO) [I.15] can be used when we have no prior knowledge about the noise level. The minimization process of some variational problems can also practically be solved using fast iterative thresholding algorithm (FISTA) [I.16], forward-backward splitting (FBS) [I.17] or approximation message passing (AMP) [I.18].

I.2.4.2. Greedy Algorithms

Greedy algorithms are commonly used in CS applications because of their low complexity and their fast reconstruction. Currently, the most exploited greedy algorithms are classified into sequential and parallel greedy pursuit techniques. Sequential methods count gradient pursuit [I.19], matching pursuit (MP) [I.20]-[I.21], orthogonal matching pursuits (OMP) [I.22], regularized OMP (ROMP) and stagewise OMP (StOMP) [I.23]-[I.24]-[I.25]. Although OMP allows a faster signal reconstruction than convex relaxation approaches, it deteriorates the recovery quality for signals with low sparsity. Therefore, improved versions of OMP have been proposed to avoid these drawbacks such as compressive sampling matching pursuit (CoSaMP) [I.26], subspace pursuit (SP) [I.27], Regularized OMP [I.24], Stagewise OMP [I.23], and orthogonal multiple matching pursuit [I.28]. Those techniques are considered as parallel greedy pursuit methods. Obviously, the performance of the reconstitution algorithms depends on the applications and there is no obvious metric to determine the best reconstruction algorithm. However, for some algorithms, we can compare their complexity and the minimum measurements required for the CS recovery.

I.3. Image Compressive Sensing

Recently, deep learning is used in various computer vision tasks, and it shows high performance results in several applications such as CS reconstruction algorithms. Since many computer vision algorithms applied on 2D signals (e.g., [I.29] in which ISTA-Net is applied in a video CS context) are extended to be applied on 3D signals (e.g., videos), we introduce in this section recent image CS algorithms. Among the reconstruction methods, various block-by-block methods are already proposed such as stacked denoising autoencoder (SDA) [I.30], non-iterative reconstruction using CNN (ReconNet) [I.31] and DR2-Net [I.32] which are deep learning-based end to end reconstruction networks. However, the outputs of these algorithms suffer generally from blocky artifacts. Therefore, the use of a BM3D algorithm, as a post processed procedure, is compulsory to eliminate the blocky artifacts in reconstructions. Among the well mentioned algorithms in image reconstruction, we have the iterative shrinkage

thresholding algorithm-based network (ISTA-Net) [I.33] that integrates the traditional ISTA into a neural network to achieve superior reconstructed quality, its enhanced version ISTA-NET+, trainable ISTA for sparse signal recovery (TISTA) [I.34] and ADMM-Net [I.35] which is proposed by adapting ADMM method for CS magnetic resonance imaging (CS-MRI) using neural networks. Experimental results in various research works prove that deep learning networks can successfully solve the two main issues of compressive sensing: the design of proper sampling matrices and the reconstruction process. The performances are significantly increased, and lower computation complexity is obtained than traditional methods. Shi et al. [I.36] and T.N. Canh et al. [I.37] proposed CNN based methods for 2D image reconstruction that split the reconstruction process into two stages. Firstly, the initial reconstruction which aims to recover the images from the patches. Secondly, a better-quality reconstruction is obtained from the enhancement of the initial reconstruction. In [I.36], deep networks are used in the reconstruction phase by imitating the traditional CS image recovery and the training of the sampling matrix through a CNN network. These two theoretically separated networks are considered as an encoder-decoder approach to generate the CS measurements and to reconstruct the 2D images. Deep compressive sensing was extended to multi-scale schemes [I.37]-[I.38]-[I.39] utilizing image decomposition. In [I.38], a multiphase reconstruction process is proposed. The first phase is dedicated to a multi-scale sampling and an initial reconstruction that are jointly trained. Then, the quality of the initial image is enhanced with convolution layers and ReLU activation function. The third phase, used in the experimental comparison because of its better performances, is enhanced with Multilevel Wavelet Convolution (MWCNN).

I.4. Applications of Compressive Sensing

Compressive sensing is becoming a promising field of research and many applications have benefited from its powerful models. This section presents the main applications of CS, illustrated in Figure I.3:

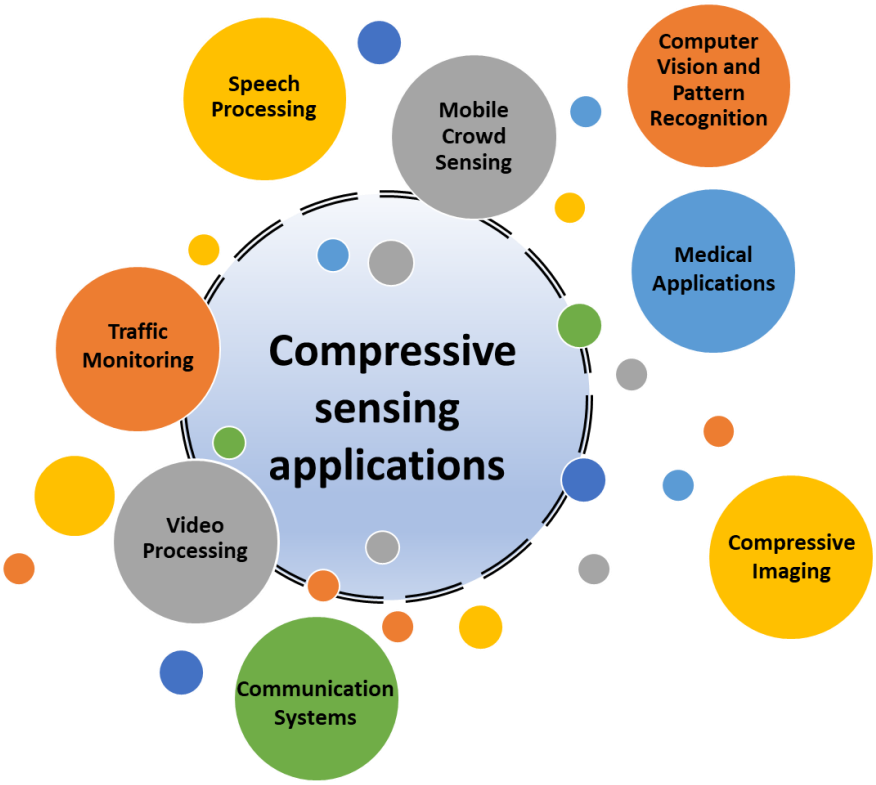


Figure I.3: Main applications of Compressive Sensing.

I.4.1. Compressive Imaging

Compressive Imaging systems can be classified into two main categories: Single pixel cameras and radar imaging systems. In fact, many imaging architectures have been introduced in the literature. One of the most popular systems in compressive imaging is single pixel camera proposed in [I.40]. In this system and during one exposure time, the video scene is gathered by an objective lens and then coded by temporal variant mask (example DMD), Then the output is detected by a charge coupled device (CCD) and then integrated into one single measurement frame. These measurements can be simply transmitted or stored at the sensor level. Furthermore, at the receiver level, the original video clip can be recovered using CS reconstruction algorithms.

I.4.2. Medical Applications

Imaging is one of the most important facets in medical science. It enables to identify and diagnose several health problems. Also, it is used by doctors to treat diseases and monitor the response of some therapies. Thus, several radiological imagining systems are invented to deal with the complexity and difficulty of human body. Among these complex systems, we have one of the most efficient tumor diagnosis methods: The Magnetic Resonance Imaging (MRI) [I.41]. CS technique is commonly used in MRI since it decreases the sampling rate without dropping useful information. Indeed, reducing the number of detected measurements in MRI is beneficial for patients because the number of measurements is proportional to the scrutiny duration, defined by the time allowed to excite human body hydrogen atoms. Therefore, CS approaches enable to have good quality images and decrease the exposure duration to magnetic fields. The key feature of MRI images that enables the use of CS approaches is their potential to have a sparse representation in the spatial or a transform domain. The original MRI signals will be reconstructed from the sparse data using nonlinear recovery frameworks. Apart from that, CS can be applied to some medical signals such as electrocardiogram (ECG) [I.42] and electrochemical signals [I.43] by using their sparsity feature.

I.4.3. Communication systems

Wireless sensor network (WSN) technology has been identified as one of the key components in designing future internet of things platforms [I.44]. It has been gaining a lot of attention since smart sensors have become an important part in our daily lives. However, in real life, these devices are resource-constrained: the storage resources, the energy capacity and the computing performances are all limited. That is why the processing of huge data especially video data is becoming very challenging. In order to shift the computation burdens from the sensor level to the decoder in WSN, compressive sensing is used as an effective way to reduce the complexity of the encoder, which means that by optimizing the way the acquire and transmit data over wireless channels, we optimize the computational resources of the devices and enhance their performances. In fact, the compressive sensing technique significantly enhances the coding efficiency of the wireless devices (considered as encoders) by reducing the sampling rate (in comparison with the well-known Shannon–Nyquist) and synchronizing the data sampling process. Another problem can be detected from a macro perspective in WSN platforms: the sporadic (infrequent) transmission rate. Indeed, not all wireless sensors send their data simultaneously to the central server, which means that the WSN architecture sparsity should be exploited to reach high data reliability with a limited number of sensors. In addition, IoT platforms can easily integrate compressive sensing into their several applications because many real-world datasets can be well approximated by sparse signals using an appropriate transform (e.g.,

DCT, DWT… to represent images, videos. . .). So, in many applications related to WSN, energy consumption is a principal concern because sensors have to send regularly their sensing data to the coordinator node. Data transmission being considered as a principal factor of energy consumption, many research efforts are focusing on reducing the amount of data acquired at the sensor level. In order to reduce the amount of transmission data, we have to compress them inside the network. As a result, compressive sensing (CS) algorithms have led to new ways of designing energy efficient WSN with low-cost data acquisition [I.45]. In addition to WSN, CS has been commonly exploited in other communication systems such as Antenna arrays [I.46] and Cognitive Radio networks [I.47].

I.4.4. Computer Vision and Pattern Recognition

Sparse signal representations approaches have significantly impacted computer vision fields [I.48]. It is an important mechanism for collecting, representing and compressing high dimensional data. This potential is predominantly due to the fact that most types of signals or data such as images and videos have obviously sparse representations in some basis (i.e. Fourier, Wavelet). Furthermore, efficient algorithms based on convex optimization, greedy pursuit or Deep Learning techniques are commonly used to compute such representations with good performances and high fidelity. One famous application of CS in computer vision is pattern representation and recognition such as face recognition. Indeed, the fact of considering face expression changes as sparse in an entire image has allowed to exploit the powerful tools of CS [I.49]. Also, approaches like ℓ_1-minimization provide great computational tools to extract significant features and structures in order to control the semantics of the data. In [I.50], a gesture recognition problem is solved using an ℓ_1-minimization approach and the theory of random projection.

I.4.5. Speech Processing

Although CS has been widely exploited in digital image and video processing for decades, it is used today to process speech and audio signals. Recently, speech data is generated at exponentially growing rates which increases the pressure on voice communication systems. However, the limited capacity of transmission bandwidths and storage resources requires the implementation of more performant compression methods for speech signals. Thus, using CS, as an emerging compression technique in signal processing for acquiring speech data at much lower rate than conventional approaches, was in most cases an efficient and effective solution. Among CS based speech processing approaches, few are: Speech processing and enhancement based on Bayesian compressive sensing (BCS) [I.51] and speech coding by exploiting the sparsity in phonological characteristics [I.52]. In addition, CS is commonly used in audio security and speech predictions [I.53].

I.4.6. Video Processing

Video signals have both intraframe and interframe correlations. It is an important feature proving the significant information redundancy in video data that can be practically sparse in some domains. Accordingly, video signals can be reconstructed from relatively few measurements in agreement with the CS theory. In fact, CS has shown tremendous potential for video processing applications. It has made real-time video acquisition and reconstruction possible by exploiting single pixel cameras [I.54]. The real-time acquisition system is used in various applications such as remote sensing and autonomous vehicles. Among video processing models, few are: Adaptive video sampling exploiting block-based video compressive sensing reconstruction [I.55]

and distributed VCS where the sampling phase of different frames in executed independently while the recovery step is done jointly [I.56].

I.4.7. Mobile Crowd Sensing

Mobile crowd sensing (MCS) is a sensing and computing technique exploiting the data generated by smartphones to visualize and monitor environmental and urban conditions. Traditional MCS techniques use a huge number of smartphones to collect environmental data. However, these techniques suffer from the extremely huge power consumption which causes high financial costs. Also, many users need to transmit their data simultaneously which causes bandwidth occupancy issues. Thus, CS based approaches are proposed to solve this problem. CS based approach enable to reduce the number of users collecting environmental data (from N users to M users, $M \ll N$) and predict the data generated by all N smartphones from the already received information. Several challenges in MCS are addressed in [I.57] especially clustering models aiming to select the optimal nodes with the best coverage abilities and the recovery techniques aiming to forecast the estimated data non-sensed by the rest of nodes. In addition, several research works have been done to enhance the performances of CS-based methods for MCS in terms of decreasing the number of sensing nodes needed for data collection [I.58] and ensuring the privacy of the transmission process [I.59].

I.4.8. Traffic Monitoring

In smart cities, traffic monitoring is an important step in designing modern infrastructures. In traditional monitoring systems, mobile smart phones and moving vehicles are the main source of periodic reports about the state of roads (traffic status, rush hours, speeds of different vehicles, …). However, these methods are power consuming since they require a huge number of users to cover the whole area of interest. Nevertheless, in real-world applications, the number of users is always limited which forces companies to invest more resources on collecting meaningful data and increases the projects costs. Thus, in [I.60], it has been proven that CS-based approaches can successfully be used for traffic surveillance purposes where large datasets can be approximated with low rank matrices. So, CS is exploited to predict the entire dataset and impute missing values from the sparse dataset already collected by limited resources. In [I.60], the CS-based algorithm used showed great performances by predicting 80% of the entire dataset. This research has inspired industries to reduce the number of sensing devices while maintaining the same good quality for the whole traffic dataset.

I.5. Background Knowledge

In the last few years, introducing Deep Learning in computer vision applications to learn representations of data with various levels of abstraction, has considerably enhanced the state of the art and made an incredible advance on solving problems such as pattern recognition, visual object detection, frames prediction and many other visions processing related tasks. Designing efficient training models enables to achieve better performances that human level precision on many use cases. The historical evolution of Deep Learning in presented in Figure I.4, where the major milestones of neural networks research is provided. Obviously, the huge impact of Deep Learning and the exceptional progress in different domains is the result of several research works that have marked the history of science.

In our research project, especially in terms of reconstructing video signals that have already been acquired in a VCS paradigm, Deep Learning is considered as a promising direction to exploit VCS in real-time applications. Indeed, since 2006, CS is mainly used for research

purposes and deploying this signal processing framework is still challenging. The major challenges faced by CS are the quality of the reconstructed signals and the reconstruction time. However, the drawbacks of the optimization-based reconstruction methods are the long recovery time and the relatively bad reconstruction quality. Therefore, Deep Learning is exploited to enhance the performances of the recovery approaches. In this section, some basic notions, that will be frequently used in this thesis, will be introduced.

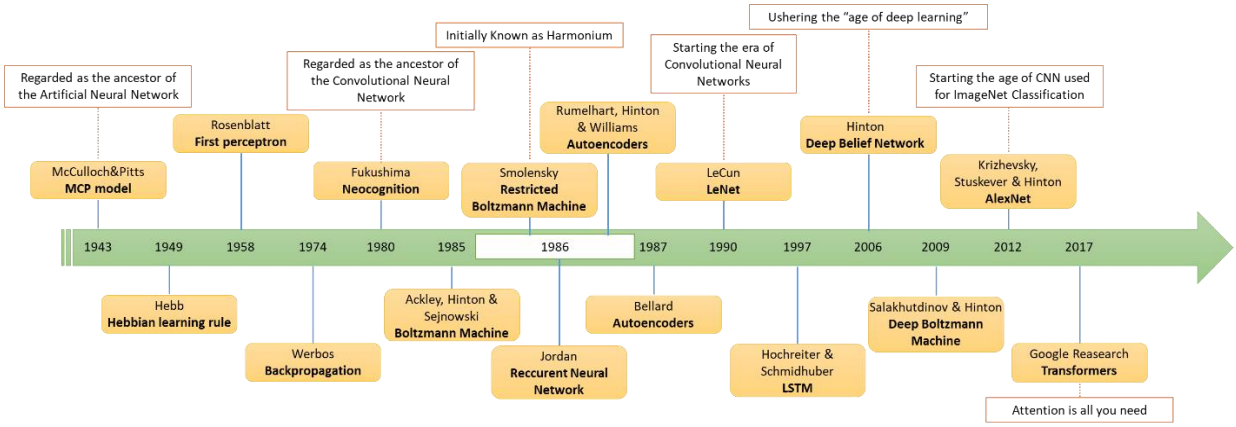


Figure I.4: The history of Neural Networks, Machine Learning and Deep Learning.

I.5.1. Neural Networks: basics

Neural Networks (NN) are computational learning systems of hardware and software functionalities aiming to extract meaningful features from input data and solve common artificial intelligence problems. It is the main backbone of deep learning. It enables to learn an approximation of a complex function. It consists of interconnected neurons, updated during the training phase. The updating process is based on error functions. Then, the linear combination of weights and bias parameters of the neurons is processed through the activation functions to generate suitable outputs. This paradigm is the main learning mechanism of deep learning-based models.

In order to understand the different Deep Learning architectures, some basic notions will be defined in this part.

I.5.1.1. Neuron

Inspired by biological neurons, artificial neurons are the basic units of neural networks. Technically, they receive input data from either row data sets or from artificial neurons of the previous layer, process it and produce outputs to the next hidden layer or to the final generated return.

The output of a neuron can be expressed as follows (I.7):

$$Output = \sum (Weight \times Input) + Bias. \tag{I.7}$$

Then, the performance of neural networks depends essentially on calculating the optimal values for weights and biases by repeatedly updating them.

I.5.1.2. Weights

Weights are learnable parameters of neural networks to transform input data and to impact the output. Before starting the training process, weights are initialized randomly. Then, they are continuously updated during the learning phase. Higher weights are assigned to more important features, thus a weight of zero represents an inconsequential feature. Accordingly, weights are in charge of supervising the stability of the connection between two artificial neurons.

I.5.1.3. Bias

In addition to weights, bias are also learnable parameters of neural networks [I.61]. It is added to modify the range of the input multiplied by the weight value. Bias is the second part of the linear transformation of input data. One of the most important utilities of biases is that they guarantee that outputs of artificial neurons are not null values even when inputs are zeros.

I.5.1.4. Activation Function

Activation functions [I.62], also known as transfer functions, are non-linear transformations applied to the linear combination of input data. It is used to regulate the output of neural network. As shown in Figure I.5, the output value of an activation function can be defined as follows (I.8):

$$\boldsymbol{y_k} = f\left(\sum_{\boldsymbol{k=1}}^{\boldsymbol{n}} \boldsymbol{XW_k} + \boldsymbol{b_k}\right). \tag{I.8}$$

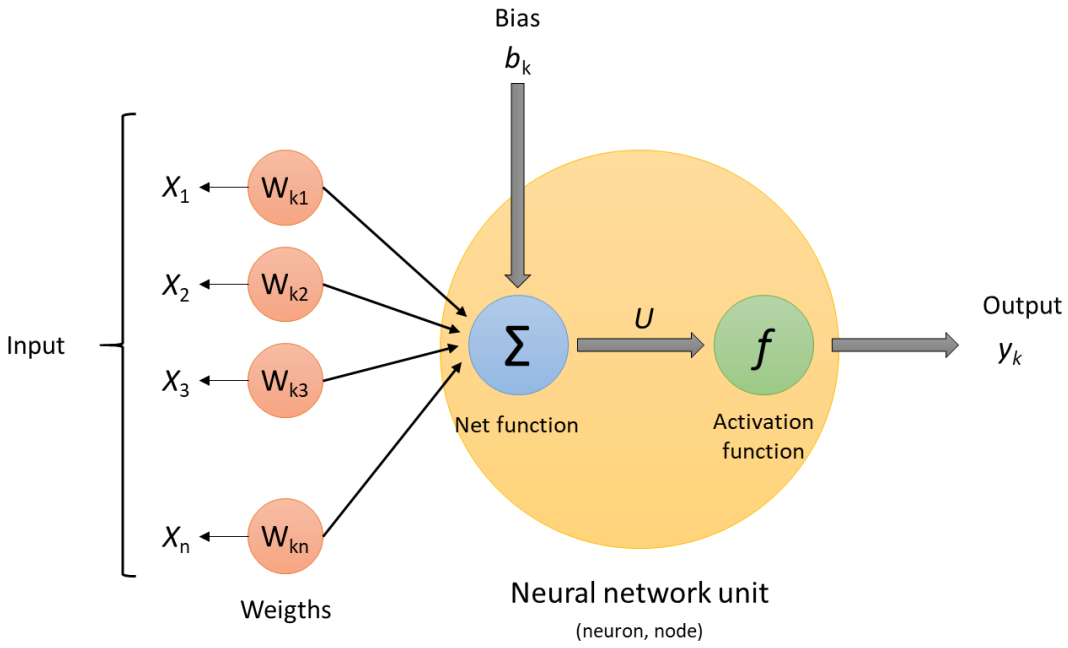


Figure I.5: Activation function.

Four main activation functions will be cited in this thesis: Sigmoid, Tanh, ReLU and Softmax.

Sigmoid or logistic activation function - It is defined by the following equation (I.9):

$$Sigmoid\ (\boldsymbol{x}) = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e^{-x}}}. \tag{I.9}$$

Since the sigmoid activation function outputs a range of values between 0 and 1, it is used for machine learning models to forecast a probability value.

Hyperbolic tangent activation function (Tanh) - It is better than sigmoid because it has a range of values between -1 and 1. Negative values are mapped completely negative and zero values are mapped near to zero in the tanh representation. It frequently used in recurrent neural network-based model for NLP and speech recognition use cases. The Tanh equation is defined as follows (I.10):

$$Tanh(\boldsymbol{x}) = \frac{\boldsymbol{e^{x}} - \boldsymbol{e^{-x}}}{\boldsymbol{e^{x}} + \boldsymbol{e^{-x}}}. \quad \text{(I.10)}$$

Rectified Linear Units (ReLU) - ReLU activation function is commonly used for hidden layers. It is defined as(I.11):

$$ReLU(\boldsymbol{x}) = \max(\boldsymbol{x}, \boldsymbol{0}). \quad \text{(I.11)}$$

Then, the output of ReLU is $\boldsymbol{x}$ when $\boldsymbol{x}$ is strictly positive, and 0 otherwise. ReLU is mainly exploited because of its constant derivative value for positive inputs. The constant derivative allows to fasten the training process. It is the most popular and most advanced function among the other activation functions because it attenuates the impacts of the Vanishing Gradient problem which accelerate the training phase.

Softmax - It is frequently used for multi-class classification models. Technically, it transforms a vector of n values into a vector n values, between 0 and 1, that sum to 1. Indeed, each value represents the probability of belonging to each class. Then, the classifier classifies the input based on the softmax result.

I.5.1.5. Input/ Output/ Hidden Layer

Input layers are the first neural networks components to receive input data. Hidden layers are the processing layers where the entire learning mechanism is executed. Output layers are the final layers in neural network, and they transform data generated from hidden layers to the final appropriate output.

I.5.1.6. Multi-Layer Perceptron

Multi-Layer Perceptron (MLP) [I.63] is a fully connected feedforward network in which every neuron in one layer is connected to all the neurons in the next layer. An MLP network has an input layer, one or many hidden layers and an output layer.

I.5.1.7. Cost Function

It is considered as the average loss over the whole training data. It is a metric to measure the difference between the predicted values and the actual ones. It penalizes the network when making prediction errors and enables to update the learnable parameters. Indeed, the main goal of the learning process is to minimize the value of this function. Thus, the optimal output is associated with the lowest cost value. Many cost functions are used in deep learning-based architectures such as mean square error (MSE) [I.64], Root Mean-Square Error (RMSE) [I.65], etc.

I.5.1.8. Forward Propagation

It is defined by the movement of information in a single direction forward from the input layer to the output layer. It is obvious that backward movement is not executed.

I.5.1.9. Backpropagation

The back propagation is the movement of the weight updating process when is done from the output layer to the first hidden layer. The weight updating process uses the gradient of the cost function.

I.5.1.10. Gradient Descent

It is a first order iterative optimization method to find the minimum of a convex function of multiple variables (the cost function). It has been widely used to train Artificial Neural Networks (ANN) [I.66]. One of the most used gradient descent algorithms is Stochastic Gradient Descent (SGD). It takes up some random instances of the training dataset at each iteration and then calculates the gradient. The process of finding the minima of the cost function with SGD is slower than typical Gradient descent algorithms because only one sample of the dataset is taken randomly into consideration and reaching the minima is possible in a significantly longer training time (noisy paths).

I.5.1.11. Learning Rate

The Learning Rate is defined as the rate, at each iteration, the model descends towards the value of the minima in the loss function. A trade-off should be made when selecting this hyperparameter. Indeed, the model may start diverging instead of converging and fails to determine the minimum if the learning rate is too large and it may take much more time to converge when the learning rate is very low. Also, it may get stuck in a fixed local minima. To deal with the above challenges, three main techniques can be used to reduce the value of the learning rate hyperparameter while training the model: Firstly, a constant $\boldsymbol{\lambda}$ can be applied to reduce the learning rate with a defined step. Secondly, it can be regulated during the training phase using the following equation (I.12):

$$\boldsymbol{lr_t = lr_0\, \alpha^{\frac{t}{\epsilon}},} \tag{I.12}$$

where $\boldsymbol{lr_t}$ and $\boldsymbol{lr_0}$ are the t^{th} and the initial learning rates, respectively. $\boldsymbol{\alpha}$ is predetermined decay factor. Finally, the third reduction technique is called the exponential decay and it is defined as follows (I.13):

$$\boldsymbol{lr_t = lr_0\, e^{-kt},} \tag{I.13}$$

where $\boldsymbol{k}$ is an hyperparameter.

I.5.1.12. Batches

In general, when training a deep learning model, the dataset is divided into several parts called batches. It is an important step before training because in most cases it is impossible to train the entire dataset in one go. Training the model on batches makes it more generalizable.

I.5.1.13. Epochs

One epoch is considered as a single iteration when the entire dataset (i.e., all the batches) is processed forward and backward through the deep learning model. The number of epochs used to train the model is predetermined by the user before starting the training. A trade-off must be made when fixing this hyperparameter to have high accuracy without over-fit the network.

I.5.1.14. Dropout

Dropout is a regularization approach commonly used to avoid over-fitting in deep learning models [I.67]. It consists in removing some neurons, randomly chosen, from one or many hidden layers during the training process. It enables to train different DL models (different neurons combinations) on the training dataset. Also, it decreases the complexity of the network in order to be able to generalize well on new test datasets.

I.5.1.15. Batch Normalization

It is a normalization technique processed between the hidden layers of very deep neural networks and aims to standardize the input of the next layer [I.68]. It enables to have the suitable distribution that can fit into the next hidden layer. In fact, batch normalization is exploited to solve the internal covariate shift problem. This internal problem comes from the changes of data distribution from one hidden layer to another during the training process. Thus, data should be explicitly normalized before sent to the next layer. In general, it allows to regularize the network and reduces the use of dropout and other regularization approaches.

I.5.2. Convolutional Neural Networks

One of the most established computer vision algorithms among various deep learning models is Convolutional Neural Network (CNN) thanks to its extraordinary results in many vision related tasks such as object recognition [I.69] using the deep layer structure and back-propagation to adaptively learn spatial features. Despite its heavy computational cost, CNNs are able to extract useful information from compressed visual signals such as objects and movements which are considerably exploited in the reconstruction process. Also, they have notably enhanced context learning, object segmentation and classification, and super-resolution.

It is used by applying convolution operations to the data before using fully connected networks. Indeed, Figure I.6 represents the main architecture of CNN used for image classification purposes. Each layer of the CNN mechanism converts the input volume into an output of feature maps. These feature maps are used by the fully-connected layers to classify the main input.

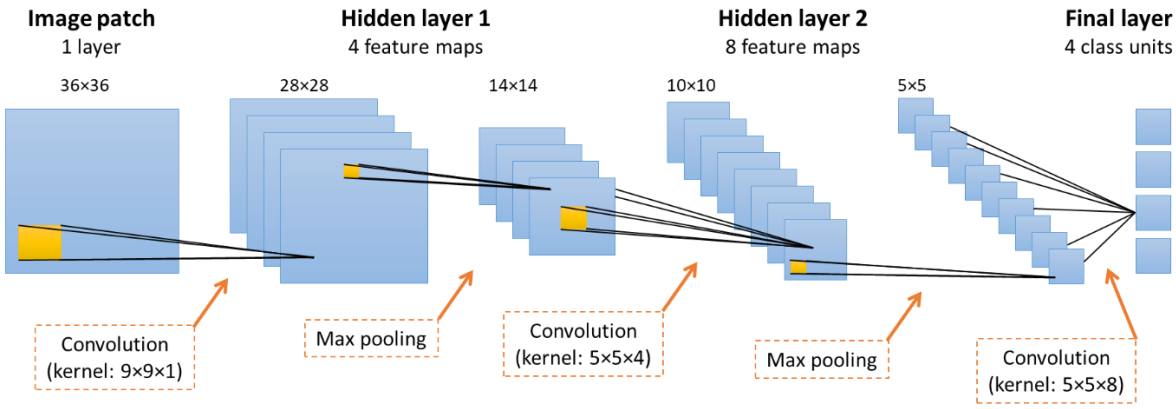


Figure I.6: CNN: main architecture.

I.5.2.1. Filters

In CNN, filters are considered as weight matrices commonly used to extract spatial patterns from images such edges and lines by identifying the variations in intensity values of the input frame. In general, the spatial dimensions of a filter are smaller than the size of the input image. The filters are updated by executing full convolutional operations on feature maps between the convolutional layer and its previous layer.

I.5.2.2. Pooling

Pooling is an operation performed to decrease the number of parameters in the network (subsampling or down sampling) and avoid over-fitting. It does not impact the depth dimension. It is used to generalize features extracted by filters and enable the model to identify features independent of their position in the frame. The most used type of pooling is a layer of 2 × 2 filters using the max function [I.70], as illustrated in Figure I.7, because it enables faster convergence and enhance the generalization abilities of the network. In addition, other pooling function can be exploited such as average pooling and min pooling.

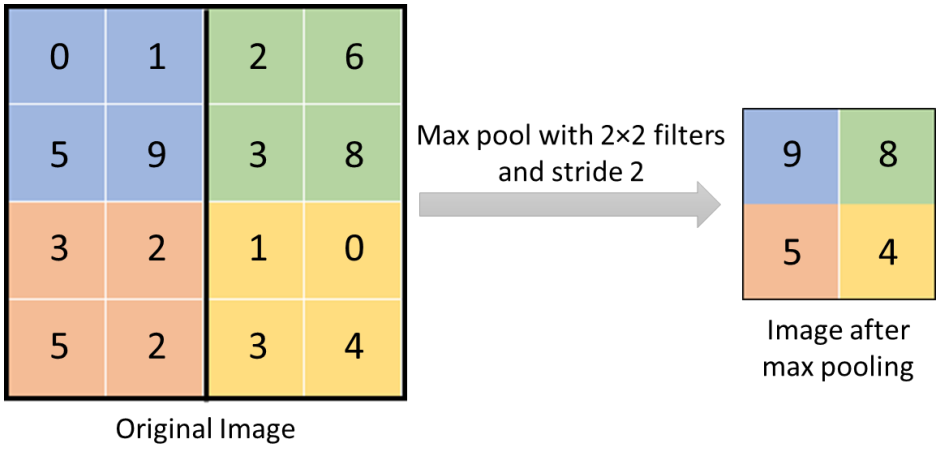


Figure I.7: MAX pooling.

I.5.2.3. Padding

In an image, the number of pixels appended when features are being extracted by the filter of a CNN model is called padding. Technically, it is the fact of adding layers of zeros the frame to efficiently extract features from pixels on corners and edges because those pixels are much less exploited than those in the middle. Accordingly, padding enables to prevent the shrinkage of an input image in consequence of the convolution operations.

I.5.3. Recurrent Neural Networks

Recurrent Neural Networks (RNN) [I.71] is another class of Deep Learning method that is commonly used to process time-series signals and other sequential data. It is considered as an extension to feed-forward networks to process long sequences. The main characteristic of recurrent networks is their internal memory to memorize information from previous layers and to impact the current input and future outputs. Among the commonly exploited recurrent architectures, long short-term memory (LSTM) and gated recurrent units (GRUs) are the most popular ones. RNN architectures (Figure I.8) have continuously been improved to model and process high dimensional data used in several applications such object tracking, video prediction and video synthesis.

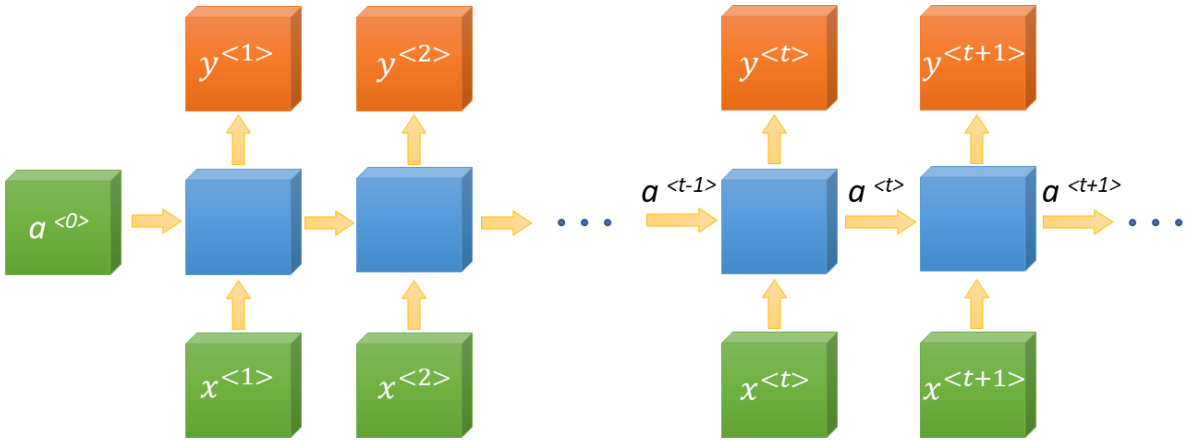


Figure I.8: RNN: main architecture.

I.5.3.1. Recurrent Neuron

Recurrent Neurons are different from standard artificial neurons because their outputs are sent back to them to re-process data. They enable to store data within the network and generate more generalized outputs.

I.5.3.2. Vanishing Gradient Problem

The Vanishing Gradient Problem [I.72] is caused by a very small value of the gradient of the activation function in the network. The weights are multiplied by these very small gradients, they become also very small which slows down the training phase and impacts the long-range dependency of the recurrent model. To solve this problem many techniques can be used such as using ReLU as an activation function.

I.5.3.3. Exploding Gradient Problem

In contrast to vanishing gradient problem, the exploding gradient problem happens when the gradient of the activation function is very large [I.73]. To solve this problem during the back propagation, gradients are clipped to not exceed some threshold.

I.5.4. Generative Adversarial Networks

Generative Adversarial Network (Figure I.9), which is a deep learning-based generative approach, was designed and developed by Ian Goodfellow in 2014 [I.74]. It is an alternative technique to the famous maximum likelihood estimation approach. In a GAN-based algorithm, two neural networks, the generator and the discriminator, are implicitly competing against each other to generate more accurate estimations. To make it simple, the generator network starts with a randomly generated data and the discriminator is designed to judge the accuracy of the output of the generator. This learning process stops when the generated data become not far from the actual real samples and can be expressed by (I.14):

$$min_G min_D V(D,G) = \mathbb{E}_{x\sim P_{data}(x)}\big[\log\big(D(x)\big)\big] + \mathbb{E}_{x\sim P_{data}(z)}[\log(1-D(G(x)))], \quad \text{(I.14)}$$

in this equation, GAN is considered as a minmax game with the value function $\boldsymbol{V}$. The generator $\boldsymbol{G}$ aims to minimize $\boldsymbol{V}$ and the discriminator $\boldsymbol{D}$ aims to maximize it.

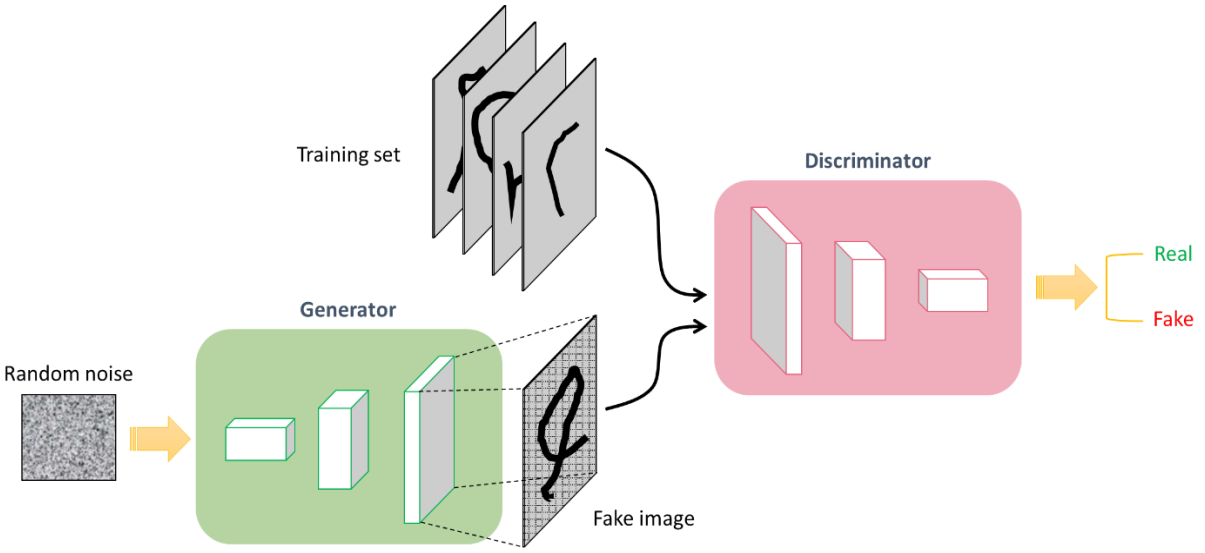


Figure I.9: GAN: main architecture.

I.5.4.1. Generator

It is a neural network aiming to generate fake data points (e.g. images) that seem to be realistic, and feed the discriminator network as a real data.

In fact, the generator has a random vector drawn from a Gaussian distribution as an input to feed the generative mechanism. It aims to make the discriminator network classify its generated output into real or fake data. The backpropagation mechanism is exploited to regulate each weight in the proper direction by measuring weights impact on the output.

I.5.4.2. Discriminator

It is a neural network that enables to differentiate between real and fake data. It is a prominent network in the learning strategy of generators.

In the training process, the discriminator is able to classify the output of the generator using the discriminator loss function. Same as generators, the learning process uses the backpropagation method to update learnable weights.

I.5.5. Auto-Encoders (AE)

Recently, many compression approaches are based on the dimensionality reduction aiming to transform signals from a high dimensional space into a low dimensional level. This transformation, exploiting the sparsity of some signals can be realized by training auto-encoders (AE) (Figure I.10) [I.75], which is another multilayer neural network commonly used for unsupervised feature learning to reduce data dimensions. AE are playing an important role in computer vision and video processing problems. They are an unsupervised learning technique in which the bottleneck enables a compressed knowledge representation of the original data.

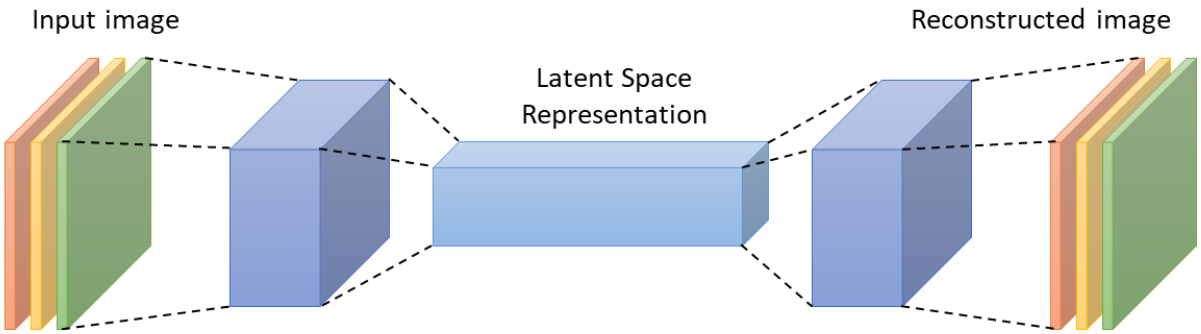


Figure I.10: AE: main architecture.

I.5.6. Transformers

Finally, there is a trend of replacing convolutional and recurrent neural networks with a recent topology to improve the network's abilities to exploit spatiotemporal correlations: Transformers. Indeed, a Transformer is a recent deep learning approach based on the mechanism of self-attention, deferentially measuring the impact of each part of the input training data. It is considered as a sequence-to-sequence model but does not include recurrent models. In fact, Transformers is an encoder-decoder architecture, as illustrated in Figure I.11. The encoder processes the input data/sequence and compresses it into a context representation called vector. Then, the decoder generated the output from the context vectors. Since its first appearance in 2017 with the well-known paper "attention is all you need" [I.76], Transformers have been widely exploited in Natural Language Processing (NLP). However, extended versions of Transformers have recently improved some computer vision applications. With enough input data, linear layers and matrix multiplications Transformers are revolutionizing the learning process of Deep Learning attention-based approaches.

In this thesis, we worked on exploiting these powerful architectures to build a robust video compressive sensing framework.

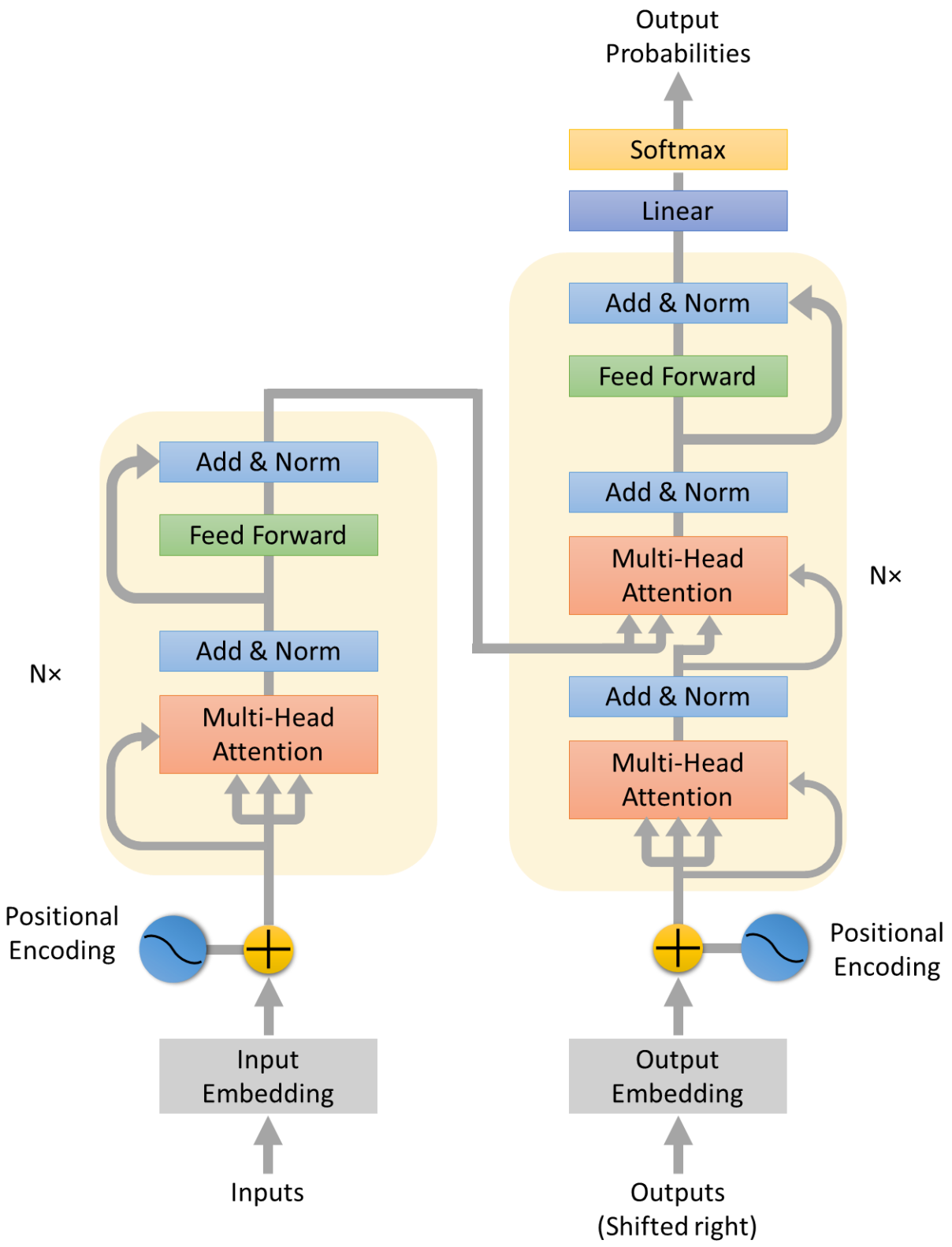


Figure I.11: Transformers: main architecture.

I.5.6.1. Embedding

Embedding is the operation of transforming high dimensional vector into low dimensional space. It is the vector representation of each token in the input sequence. It is an important step for Transformer-based architectures to feed the encoder block with the suitable data representation.

I.5.6.2. Attention Mechanism

Attention is one of the most powerful concepts in Deep Learning. It is an integral component of Transformers, which expressly represent the interactions between all units of an input image or video sequence for structured forecasting assignment. Attention layers update each unit of an input sequence by combining global information from the entire input data. Their main role is to extract correlations between different tokens of input data by evaluating the context and the structure of the signal (a video sequence in our thesis).

Mathematically, for an input sequence $\boldsymbol{X} \in \mathbb{R}^{n \times d}$, where n is the number of the sequence components and $\boldsymbol{d}$ is the embedding dimension modeling each unit, the goal of the attention mechanism is to detect the interactions between all n components. This contextual information is calculated by defining 3 learnable matrices to convert Queries ($\boldsymbol{W^Q} \in \mathbb{R}^{d \times d_q}$), Keys ($\boldsymbol{W^K} \in \mathbb{R}^{d \times d_k}$) and Values ($\boldsymbol{W^V} \in \mathbb{R}^{d \times d_v}$). Then, the input $\boldsymbol{X}$ is projected to calculate Queries, Keys and Values: $\boldsymbol{Q = XW^Q}$, $\boldsymbol{K = XW^K}$ and $\boldsymbol{V = XW^V}$. Finally, the output of the attention layer is defined by (I.15):

$$\boldsymbol{A = softmax\left(\frac{QK^T}{\sqrt{d_q}}\right)V}, \tag{I.15}$$

where $\mathbf{A} \in \mathbb{R}^{n \times d_v}$ is the attention map.

In several research works, multi-head attention is commonly used to capture various complex relationships between different elements in the input sequence. It is composed of several self-attention blocks (for example, 8 self-attention blocks in the original Transformer). Each attention unit has its own learnable matrices and its own attention map. In general, the number of attention layers depend on the number of objects illustrated in the input sequence.

I.6. Conclusion

In this chapter, we have provided an introductory review of Compressive Sensing and its applications over the past few years. Then, we introduced the main techniques used Image Compressive Sensing frameworks and can be extended in Video Compressive Sensing purposes. We also presented the main concepts of Deep Learning that will be discussed in this thesis and the major milestones in the history that have influenced the current development of deep learning. Indeed, we have explained in detail the different Deep Learning architectures exploited in recent Video Compressive Sensing frameworks such as CNN, RNN and the Transformers. Also, learnable parameters and hyperparameters were reviewed in detail.

At the end of this chapter, we believe that these concepts will provide a background knowledge to readers. The next chapter will focus on the recent advances in Deep Learning-based Video Compressive Sensing frameworks in order to theoretically prove the impact the neural networks in the field. Furthermore, a comparative study will be presented to qualitatively and quantitatively evaluate the main VCS approaches.

References

[I.1] Donoho, David L. “Compressed sensing.” *IEEE Transactions on information theory 52*, no. 4 (2006): 1289-1 CIBEL, https://cibel.com

[I.2] Candès, Emmanuel J., and Terence Tao. “Near-optimal signal recovery from random projections: Universal encoding strategies.” *IEEE transactions on information theory 52*, no. 12 (2006): 5406-5425 CERMAG LTD, www.cermag.co.uk/magnet_properties.html

[I.3] Candès, Emmanuel, and Justin Romberg. “Sparsity and incoherence in compressive sampling.” *Inverse problems 23*, no. 3 (2007): 969.

[I.4] Tropp, Joel A., Jason N. Laska, Marco F. Duarte, Justin K. Romberg, and Richard G. Baraniuk. “Beyond Nyquist: Efficient sampling of sparse bandlimited signals.” *IEEE transactions on information theory 56*, no. 1 (2009): 520-544.

[I.5] Candès, Emmanuel J., Justin Romberg, and Terence Tao. “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information.” *IEEE Transactions on information theory 52*, no. 2 (2006): 489-509.

[I.6] Donoho, D.L. Compressed sensing. *IEEE Trans. Inf. Theory* 2006, 52, 1289– 1306. doi:10.1109/TIT.2006.871582.

[I.7] Candès, E.J.; Romberg, J.K.; Tao, T. Stable signal recovery from incomplete and inaccurate measurements. *Commun. Pure Appl. Math*. 2006, 59, 1207–1223.

[I.8] langi, H.; Ward, R.; Deng, L. Distributed Compressive Sensing: A Deep Learning Approach. *IEEE Trans. Signal Process*. 2016, 64, 4504–4518.

[I.9] Hochreiter, S.; Schmidhuber, J. Long Short-term *Memory. Neural Comput.* 1997, 9, 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[I.10] Candès, E.J.; Plan, Y. A Probabilistic and RIPless Theory of Compressed Sensing. *IEEE Trans. Inf. Theory* 2011, 57, 7235–7254. doi:10.1109/TIT.2011.2161794.

[I.11] Nguyen, T.L.; Shin, Y. Deterministic sensing matrices in compressive sensing: *A survey. Sci. World J.* 2013, 2013, 192795.

[I.12] Rousseau, S.; Helbert, D. Compressive Color Pattern Detection Using Partial Orthogonal Circulant Sensing Matrix. *IEEE Trans. Image Process.* 2020, 29, 670–678. doi:10.1109/TIP.2019.2927334.

[I.13] Chen, S.S.; Donoho, D.L.; Saunders, M.A. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput*. 1998, 20, 33–61.

[I.14] Candès, E.J. The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Math.* 2008, 346, 589–592.

[I.15] Candès, E.; Tao, T. The Dantzig selector: Statistical estimation when p is much larger than n. *Ann. Stat.* 2007, 35, 2313–2351.

[I.16] Beck, A.; Teboulle, M. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. *IEEE Trans. Image Process.* 2009, 18, 2419–2434. 10.

[I.17] Combettes, P.L.; Pesquet, J.-C. Proximal splitting methods in signal processing. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*; Springer: New York, NY, USA, 2011; pp. 185–212.

[I.18] Donoho, D.L.; Maleki, A.; Montanari, A. Message-passing algorithms for compressed sensing. *Proc. Natl. Acad. Sci*. USA 2009, 106, 18914–18919.

[I.19] Figueiredo, M.A.; Nowak, R.D.; Wright, S.J. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. Signal Process.* 2007, 1, 586–597.

[I.20] Mallat, S.G.; Zhang, Z. Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Process.* 1993, 41, 3397–3415.

[I.21] Krstulovic, S.; Gribonval, R. MPTK: Matching pursuit made tractable. In Proceedings of the 2006 *IEEE International Conference on Acoustics*, Speech and Signal Processing (ICASSP), Toulouse, France, 14–19 May 2006; IEEE: Piscataway, NJ, USA, 2006; Volume 3, p. III.

[I.22] Tropp, J.A.; Gilbert, A.C. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inf. Theory* 2007, 53, 4655–4666.

[I.23] Donoho, D.L.; Tsaig, Y.; Drori, I.; Starck, J.-L. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *IEEE Trans. Inf. Theory* 2012, 58, 1094–1121.

[I.24] Needell, D.; Vershynin, R. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Found. Comp. Math.* 2009, 9, 317–334.

[I.25] Tropp, J. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inf. Theory* 2004, 50, 2231–2242.

[I.26] Needell, D.; Tropp, J.A. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmon. Anal.* 2009, 26, 301–321.

[I.27] Dai, W.; Milenkovic, O. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Trans. Inf. Theory* 2009, 55, 2230–2249.

[I.28] Liu, E.; Temlyakov, V.N. The orthogonal super greedy algorithm and applications in compressed sensing. *IEEE Trans. Inf. Theory* 2012, 58, 2040–2047.

[I.29] Xuan, Y.; Yang, C. 2Ser-Vgsr-Net: A Two-Stage Enhancement Reconstruction Based On Video Group Sparse Representation Network For Compressed Video Sensing. In Proceedings of the 2020 *IEEE International Conference on Multimedia and Expo (ICME)*, London, UK, 6–10 July 2020; pp. 1–6, doi:10.1109/ICME46284.2020.9102849.

[I.30] Mousavi, A.; Patel, A.B.; Baraniuk, R.G. A deep learning approach to structured signal recovery. In *Proceedings of the 2015 53rd Annual Allerton Conference on Communication*, Control, and Computing (Allerton), Monticello, IL, USA, 29 September–2 October 2015; pp. 1336–1343, doi:10.1109/ALLERTON.2015.7447163.

[I.31] Kulkarni, K.; Lohit, S.; Turaga, P.; Kerviche, R.; Ashok, A. ReconNet: Non-iterative reconstruction of images from compressively sensed measurements. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016; pp. 449–458.

[I.32] Yao, H.T.; Dai, F.; Zhang, S.L.; Zhang, Y.D.; Tian, Q.; Xu, C.S.; DR2 -Net: Deep residual reconstruction network for image compressive sensing. *Neurocomputing* 2019, 359, 483–493.

[I.33] Zhang, J.; Ghanem, B. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In Proceedings of *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1828–1837.

[I.34] Ito, D.; Takabe, S.; Wadayama, T. Trainable ISTA for Sparse Signal Recovery. *IEEE Trans. Signal Process.* 2019, 67, 3113–3125. doi:10.1109/TSP.2019.2912879.

[I.35] Su, H.; Bao, Q.; Chen, Z. ADMM–Net: A Deep Learning Approach for Parameter Estimation of Chirp Signals Under Sub-Nyquist Sampling. *IEEE Access* 2020, 8, 75714–75727. doi:10.1109/ACCESS.2020.2989507.

[I.36] Shi, W.; Jiang, F.; Liu, S.; Zhao, D. Image Compressed Sensing Using Convolutional Neural Network. *IEEE Trans. Image Process.* 2020, 29, 375–388. doi:10.1109/TIP.2019.2928136.

[I.37] Canh, T.N.; Jeon, B. Multi-Scale Deep Compressive Sensing Network. In Proceedings of the 2018 *IEEE Visual Communications and Image Processing (VCIP)*, Taichung, Taiwan, 9–12 December 2018; pp. 1–4, doi:10.1109/VCIP.2018.8698674.

[I.38] Canh, T.N.; Jeon, B. Difference of Convolution for Deep Compressive Sensing. In Proceedings of the 2019 *IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan, 22–25 September 2019; pp. 2105–2109, doi:10.1109/ICIP.2019.8803165.

[I.39] Shi, W.; Jiang, F.; Liu, S.; Zhao, D. Scalable Convolutional Neural Network for Image Compressed Sensing. In Proceedings of the 2019 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 15–20 June 2019; pp. 12282–12291. doi:10.1109/CVPR.2019.01257.

[I.40] Duarte, Marco F., Mark A. Davenport, Dharmpal Takhar, Jason N. Laska, Ting Sun, Kevin F. Kelly, and Richard G. Baraniuk. “Single-pixel imaging via compressive sampling.” *IEEE signal processing magazine 25*, no. 2 (2008): 83-91.

[I.41] Gamper, Urs, Peter Boesiger, and Sebastian Kozerke. “Compressed sensing in dynamic MRI.” Magnetic Resonance in Medicine: *An Official Journal of the International Society for Magnetic Resonance in Medicine 59*, no. 2 (2008): 365- 373.

[I.42] Poh, Kok-Kiong, and Pina Marziliano. “Compressive sampling of EEG signals with finite rate of innovation.” *EURASIP journal on advances in signal processing* 2010 (2010): 1-12.

[I.43] Liu, Xilin, Hongjie Zhu, Milin Zhang, Andrew G. Richardson, Timothy H. Lucas, and Jan Van der Spiegel. “Design of a low-noise, high power efficiency neural recording front-end with an integrated real-time compressed sensing unit.” In 2015 *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 2996-2999. IEEE, 2015.

[I.44] Akyildiz, Ian F., Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. “Wireless sensor networks: a survey.” *Computer networks 38*, no. 4 (2002): 393- 422.

[I.45] Amarlingam, M., Pradeep Kumar Mishra, Pachamuthu Rajalakshmi, Mukesh Kumar Giluka, and Bheemarjuna Reddy Tamma. “Energy efficient wireless sensor networks

utilizing adaptive dictionary in compressed sensing." In 2018 IEEE *4th World Forum on Internet of Things (WF-IoT)*, pp. 383-388. IEEE, 2018.

[I.46] Liu, Yimin, Hang Ruan, Lei Wang, and Arye Nehorai. "The random frequency diverse array: A new antenna structure for uncoupled direction-range indication in active sensing." *IEEE Journal of Selected Topics in Signal Processing 11*, no. 2 (2016): 295-308.

[I.47] Sharma, Shree Krishna, Eva Lagunas, Symeon Chatzinotas, and Björn Ottersten. "Application of compressive sensing in cognitive radio communications: A survey." *IEEE communications surveys tutorials 18*, no. 3 (2016): 1838-1860.

[I.48] Wright, John, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S. Huang, and Shuicheng Yan. "Sparse representation for computer vision and pattern recognition." *Proceedings of the IEEE 98,* no. 6 (2010): 1031-1044.

[I.49] Nagesh, Pradeep, and Baoxin Li. "A compressive sensing approach for expression-invariant face recognition." In 2009 *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1518-1525. IEEE, 2009.

[I.50] Akl, Ahmad, Chen Feng, and Shahrokh Valaee. "A novel accelerometer-based gesture recognition system." *IEEE transactions on Signal Processing 59*, no. 12 (2011): 6197-6205.

[I.51] You, Hanxu, Zhixian Ma, Wei Li, and Jie Zhu. "A speech enhancement method based on multi-task Bayesian compressive sensing." *IEICE TRANSACTIONS on Information and Systems 100*, no. 3 (2017): 556-563.

[I.52] Asaei, Afsaneh, Milos Cernak, and Hervé Bourlard. "On compressibility of neural network phonological features for low bit rate speech coding." In *Proceeding of Interspeech, no. CONF*. 2015.

[I.53] George, Sudhish N., Nishanth Augustine, and Deepthi P. Pattathil. "Audio security through compressive sampling and cellular automata." *Multimedia Tools and Applications 74*, no. 23 (2015): 10393-10417.

[I.54] Edgar, Matthew P., Ming-Jie Sun, Graham M. Gibson, Gabriel C. Spalding, David B. Phillips, and Miles J. Padgett. "Real-time 3D video utilizing a compressed sensing time-of-flight single-pixel camera." In *Optical Trapping and Optical Micromanipulation XIII*, vol. 9922, pp. 171-178. SPIE, 2016.

[I.55] Mun, Sungkwang, and James E. Fowler. "Residual reconstruction for block-based compressed sensing of video." In 2011 *Data Compression Conference*, pp. 183-192. IEEE, 2011.

[I.56] Veeraraghavan, Ashok, Dikpal Reddy, and Ramesh Raskar. "Coded strobing photography: Compressive sensing of high speed periodic videos*." IEEE Transactions on Pattern Analysis and Machine Intelligence 33*, no. 4 (2010): 671-686.

[I.57] Wang, Leye, Daqing Zhang, Yasha Wang, Chao Chen, Xiao Han, and Abdallah M'hamed. "Sparse mobile crowdsensing: challenges and opportunities." *IEEE Communications Magazine 54*, no. 7 (2016): 161-167.

[I.58] Xu, Liwen, Xiaohong Hao, Nicholas D. Lane, Xin Liu, and Thomas Moscibroda. "More with less: Lowering user burden in mobile crowdsourcing through compressive sensing."

In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 659-670. 2015.

[I.59] Wang, Leye, Daqing Zhang, Dingqi Yang, Brian Y. Lim, and Xiaojuan Ma. “Differential location privacy for sparse mobile crowdsensing.” In 2016 *IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1257-1262. IEEE, 2016.

[I.60] Zhu, Yanmin, Zhi Li, Hongzi Zhu, Minglu Li, and Qian Zhang. “A compressive sensing approach to urban traffic estimation with probe vehicles.” *IEEE Transactions on Mobile Computing 12*, no. 11 (2012): 2289-23.

[I.61] Wang, Shengjie, Tianyi Zhou, and Jeff Bilmes. “Bias also matters: Bias attribution for deep neural network explanation.” *In International Conference on Machine Learning*, pp. 6659-6667. PMLR, 2019.

[I.62] Karlik, Bekir, and A. Vehbi Olgac. “Performance analysis of various activation functions in generalized MLP architectures of neural networks.” *International Journal of Artificial Intelligence and Expert Systems 1*, no. 4 (2011): 111-122.

[I.63] Pal, Sankar K., and Sushmita Mitra. “Multilayer perceptron, fuzzy sets, classifiaction.” (1992).

[I.64] Allen, David M. “Mean square error of prediction as a criterion for selecting variables.” *Technometrics 13*, no. 3 (1971): 469-475.

[I.65] Willmott, Cort J., and Kenji Matsuura. “Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance.” *Climate research 30*, no. 1 (2005): 79-82.

[I.66] Ruder, Sebastian. “An overview of gradient descent optimization algorithms.” arXiv preprint arXiv:1609.04747 (2016).

[I.67] Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting.” *The journal of machine learning research 15*, no. 1 (2014): 1929-1958.

[I.68] Ioffe, Sergey, and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift.” *In International conference on machine learning*, pp. 448-456. PMLR, 2015.

[I.69] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. “Imagenet classification with deep convolutional neural networks.” *Advances in neural information processing systems 25* (2012).

[I.70] Nagi, Jawad, Frederick Ducatelle, Gianni A. Di Caro, Dan Cire¸san, Ueli Meier, Alessandro Giusti, Farrukh Nagi, J¨urgen Schmidhuber, and Luca Maria Gambardella. “Max-pooling convolutional neural networks for vision-based hand gesture recognition.” In 2011 *IEEE international conference on signal and image processing applications (ICSIPA)*, pp. 342-347. IEEE, 2011.

[I.71] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[I.72] Hochreiter, Sepp. “The vanishing gradient problem during learning recurrent neural nets and problem solutions.” *International Journal of Uncertainty*, Fuzziness and Knowledge-Based Systems 6, no. 02 (1998): 107-116.

[I.73] Philipp, George, Dawn Song, and Jaime G. Carbonell. “The exploding gradient problem demystified-definition, prevalence, impact, origin, tradeoffs, and solutions.” arXiv preprint arXiv:1712.05577 (2017).

[I.74] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David WardeFarley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. “Generative adversarial nets.” *Advances in neural information processing systems 27* (2014).

[I.75] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[I.76] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention is all you need.” *Advances in neural information processing systems 30* (2017).

Table of contents

Chapter II: Comparison study of Deep Learning based approaches in Video Compressive Sensing

Chapter II. Comparison Study of Deep Learning based approaches in Video Compressive Sensing

II.1. Introduction

Compressive sensing is a technique exploited today in several applications as explained in the previous chapter. In fact, CS is a theory which can efficiently acquire and reconstruct sparse signals [II.1]. CS theory suggests that the sampling rate necessary to acquire and reconstruct the signal can be significantly lower than the minimal rate required by the Nyquist-Shannon sampling theorem. This lower sampling rate can reduce the processing and energy requirement at the sensor nodes which can lead to revolutionary results for embedded video sensors. In fact, the video signal in general is sparse so it contains a significant amount of redundancy in both spatial and temporal domains and therefore video compression is one of the most important fields where CS can be applied. The advent of CS has led to the emergence of new image devices such as Single Pixel Cameras [II.2]. CS techniques are commonly used to deal with high transmission throughput and large storage spaces. Indeed, an impressive progress has been made in Video Compressive Sensing (VCS) with the appearance of single pixel cameras where the video is represented in the Fourier domain [II.3] or the Wavelet domain [II.4]. Then, video CS cameras tried to integrate temporal compression into the systems with the arrival of the optical flow-based algorithms for video reconstruction [II.5]. In addition, Total Variation (TV) [II.6] and Dictionary Learning [II.7] were among the popular approaches used for VCS. TV methods suppose the sparsity of the gradient of each video frame and try to minimize the l_1 norm of the gradient frames. However, dictionary-based approaches consider the video patches as a sparse linear extension in the dictionary elements.

Another challenge of VCS, especially for the video reconstruction process is the complexity of the mathematical formulations handled by the reconstruction system. For the sake of simplicity, video recovery techniques can be classified into two main categories: Optimization based algorithms, categorized also into convex and greedy algorithms, and Deep Learning methods. Sections II.3 introduces the main approaches used to reconstruct the main video scenes from the compressed measurements. On the one hand, we clearly notice that iterative based approaches have high complexity (from few seconds to few minutes to recover an image). However, these techniques are not applicable for real-time applications. On the other hand, Neural Networks (NN) are applied in our topic of interest: the optimization of the transmission and reconstruction of video signals in wireless sensor networks.

Neural networks have shown excellent performances in terms of quality of image reconstruction and reconstruction processing time (in the order of milliseconds). This makes the NN approach a good candidate for real-time applications of video-monitoring in a smart city context. Thus, this paper aims at better characterizing and comparing existing state of the art NN reconstruction-based methods. The remaining of the chapter is organized as follows: In Section II.2 we present different image compressive sensing architectures, whilst Section II.3 discusses different video compressive sensing sampling and reconstruction architectures while classifying them based on their sampling strategy. In Section II.4 we classify recent deep learning-based video compressive sensing algorithms according to their modulation strategy. In Section II.5, we provide recent research results with an experimental study on several VCS approaches to compare their performances in terms of the quality of their output and the testing time. Section II.6 eventually concludes the chapter by identifying open research challenges and pointing out future research directions.

II.2. Image Compressive Sensing

Recently, deep learning is used in various computer vision tasks and it shows high performance results in several applications such as CS reconstruction algorithms. Since many computer vision algorithms applied on 2D signals (e.g., [II.8] in which ISTA-Net is applied in a video CS context) are extended to be applied on 3D signals (e.g., videos), we introduce in this section recent image CS algorithms.

Among the reconstruction methods, various block-by-block methods are already proposed such as stacked denoising autoencoder (SDA) [II.9], non-iterative reconstruction using CNN (ReconNet) [II.10] and DR2-Net [II.11] which are deep learning based end to end reconstruction networks. However, the outputs of these algorithms suffer generally from blocky artifacts. Therefore, the use of a BM3D algorithm, as a post processed procedure, is compulsory to eliminate the blocky artifacts in reconstructions. Among the well mentioned algorithms in image reconstruction, we have the iterative shrinkage thresholding algorithm based network (ISTA-Net) [II.12] that integrates the traditional ISTA into a neural network to achieve superior reconstructed quality, its enhanced version ISTA-NET+, trainable ISTA for sparse signal recovery (TISTA) [II.13] and ADMM-Net [II.14] which is proposed by adapting ADMM method for CS magnetic resonance imaging (CS-MRI) using neural networks. Experimental results in various research works prove that deep learning networks can successfully solve the two main issues of compressive sensing: the design of proper sampling matrices and the reconstruction process. The performances are significantly increased and lower computation complexity is obtained than traditional methods. Shi et al. [II.15] and T.N. Canh et al. [II.16] proposed CNN based methods for 2D image reconstruction that split the reconstruction process into two stages. Firstly, the initial reconstruction which aims to recover the images from the patches. Secondly, a better-quality reconstruction is obtained from the enhancement of the initial reconstruction. In [II.15], deep networks are used in the reconstruction phase by imitating the traditional CS image recovery and the training of the sampling matrix through a CNN network. These two theoretically separated networks are considered as an encoder-decoder approach to generate the CS measurements and to reconstruct the 2D images (Figure II.1).

Deep compressive sensing was extended to multi-scale schemes [II.16]-[II.17]-[II.18] utilizing image decomposition. In [II.17], a multiphase reconstruction process is proposed. The first phase is dedicated to a multi-scale sampling and an initial reconstruction that are jointly trained. Then, the quality of the initial image is enhanced with convolution layers and ReLU activation function. The third phase, used in the experimental comparison because of its better performances, is enhanced with Multilevel Wavelet Convolution (MWCNN).

II.3. Video Compressive Sensing

Obviously, the main function of video compressive sensing systems is to capture video data with low-dimensional detectors and then use the optimized based algorithms, as explained above in chapter 1, to solve the ill-posed reconstruction problem. These two systems: the hardware encoder and the software recovery system enable to optimize encoders resources, especially in the transmission process. However, their long running time prevents them from being exploited in real-time applications. So, thanks to recent advances in deep learning, we expand the variety of algorithms used in the reconstruction phase. Deep learning-based approaches enable a fast end-to-end recovery of video scenes with better quality performances despite the long training time. Indeed, the basic framework of video compressive sensing is composed of two main systems: the hardware encoder and the software decoder, and a channel to transmit

video data over it. This is the main digital video delivery system employed by communication systems that rely on compressive sensing to acquire, transmit and reconstruct data. In fact, the encoder uses special cameras (low-speed cameras such as single pixel cameras) to capture and process high speed videos. Then, it generates fewer compressive measurements that could be easily transmitted or stored. Finally, a reconstruction algorithm will be applied in order to reconstruct the received video at the receiver device (e.g., server). Figure II.1 illustrates the basic video compressive sensing framework.

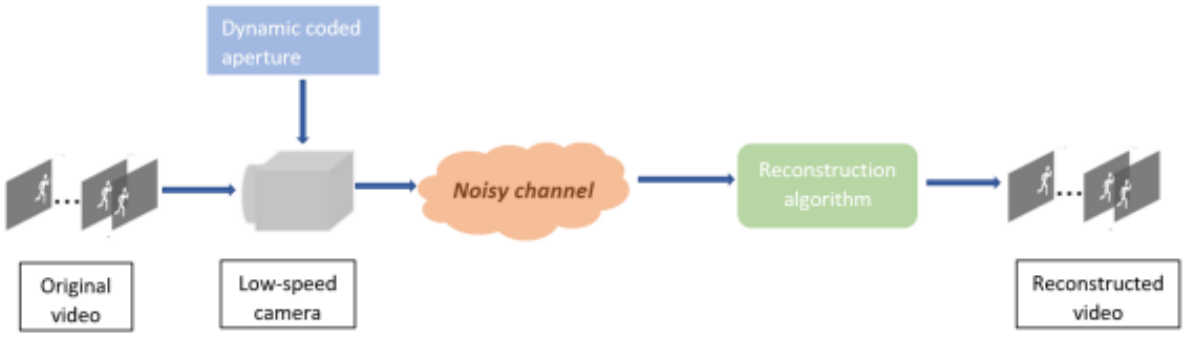


Figure II.1: Basic model of video compressive sensing.

Video CS algorithms have used various models and architectures to sample and reconstruct the signals. According to the way the video signals are sampled, we review these works in the following three categories: Temporal VCS, Spatial VCS and Spatiotemporal VCS.

II.3.1. Temporal VCS

The sampling phase of the Temporal VCS (TVCS) relies on the 2D measurements obtained from the sampling across the temporal axis which means that the compression is done in the temporal domain.

The non-neural networks approach exploits the sparsity of the video scenes and the variety of the existing algorithms for optimization problems. In [II.19], J. Yang et al. propose a Gaussian mixture model (GMM) based algorithm to reconstruct spatiotemporal video patches from temporally compressed measurements. This robust algorithm is less-dependent on the offline training dataset which enable to be extended to real-time applications. X.Yuan et al. [II.20] solved the compressive sensing problem by exploiting the Generalized Alternating Projection (GAP) to solve the Total Variation (TV) minimization mathematical problem.

Another approach to deal with TVCS, Deep learning has become one of the CS community promising trends. In [II.21], the authors present a deep fully connected network and non-iterative algorithm to recover the frames already sampled using a 3D Bernoulli sensing matrix to measure consecutive frames simultaneously. This article represents the first deep learning architecture for temporal compressive sensing reconstruction. The work of this article concerns temporal CS where the multiplexing is done through the temporal dimensions and its architecture is based on Multi-layer Perceptron (MLP) as shown in Figure II.2 Indeed, the MLP architecture is used to learn the non-linear function which maps a measured frame patch $\boldsymbol{y_i}$ via multiple layers to a video block $\boldsymbol{x_i}$.

Each hidden layer is defined by (II.1):

$$h_k(\boldsymbol{y}) = \sigma(\boldsymbol{b_k} + \boldsymbol{W_k}\boldsymbol{y}), \tag{II.1}$$

where h_k is the k-hidden layer, $\boldsymbol{b_k}$ is the bias vector and $\boldsymbol{W_k}$ is the weight matrix. The non-linear activation function used in this model is the rectified linear unit (ReLU) defined as $\sigma(\boldsymbol{y}) = \max(0, \boldsymbol{y})$. In this model, the 1st fully connected layer must provide a 3D signal from the 2D compressed measurements. The other layers are considered as 3D layers. The size of the video blocks used is 8 × 8 × 16 and increasing the block size would considerably increase the network complexity. This algorithm is tested by changing either the number of MLP layers (4 or 7) or the size of the learning database. The metrics used are the PSNR and SSIM [II.22]. In fact, increasing the number of layers for small datasets (not for large datasets) improves the metrics because several parameters are trained. However, increasing the number of layers will inevitably lead to an increase of the complexity of the network. Compressive sensing allows signals to be detected with far fewer measurements than those of Shannon–Nyquist. It entails lower costs for IOT projects and a reduction in the acquisition time. In this context, many papers have proposed architectures such as Single Pixel Cameras (SPC) providing a framework which seems to be effective for images in terms of acquisition using a reduced number of coded measurements with low-cost sensors. In [II.23], the authors were able to extend the CS imaging model beyond the images to work with the video. In the article quoted above, which talks about single-pixel cameras, it is a demonstration of the Deep Learning application with a convolutional auto-encoder network to retrieve a 128 × 128 real-time video pixels at 30 frames/s from a sampling of single-pixel cameras with a compression ratio of 2%. Thus, the proposed architecture is a Deep Convolutional Autoencoder Network (DCAN) architecture which represents a powerful and efficient computation pipeline to solve inverse problems with good quality and in real time. In this research work, deep neural networks have been exploited to produce an algorithm to reconstruct a video signal in real time from a single-pixel camera consisting of a Digital Micromirror Device (DMD) as a spatial modulator.

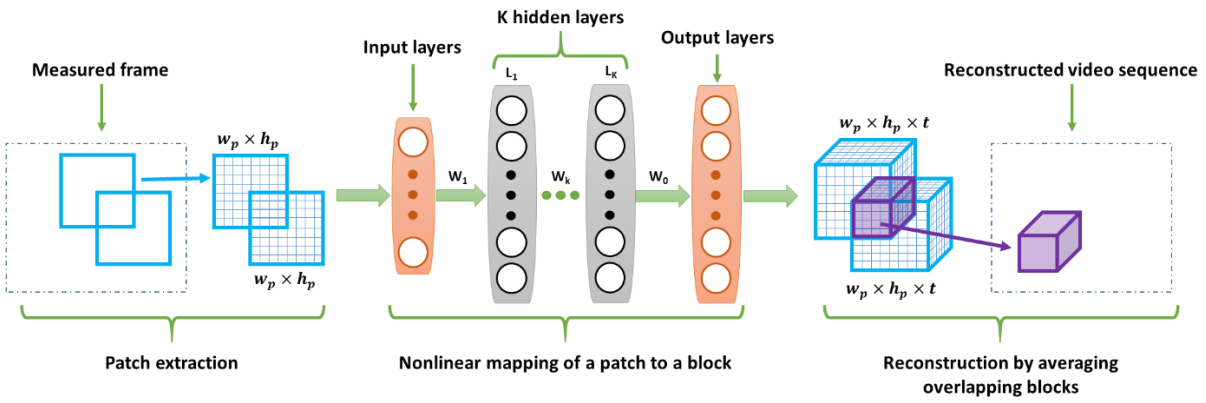


Figure II.2: Video Compressive Sensing Architecture based on an MLP Network.

It is obvious from the DCAN architecture, represented in Figure II.3, that it is a calculation model which includes coding and decoding layers. The main goal of these layers is to reconstruct an image or an input scene. The input of this network is measured by M (128×128) binary filters and reconstructed using fully connected layers and 3 convolutional blocks. After the fully connected layers, each convolution operation is followed by ReLU activation and batch normalization. The optimization of the filter weights is done using the gradient descent stochastic algorithm while respecting the minimization of the standard cost function in measuring the Euclidean distance between the observed and desired output. In order to test the performance of this algorithm, three metrics were used: peak-signal-to-noise ratio (PSNR), structural similarity index (SSIM) and standard deviation (SD). Thus, since authors can change the input resolution

size and compression ratio, the best results in terms of PSNR and SSIM were obtained with a resolution size of 128 × 128 and a compression ratio of 98%.

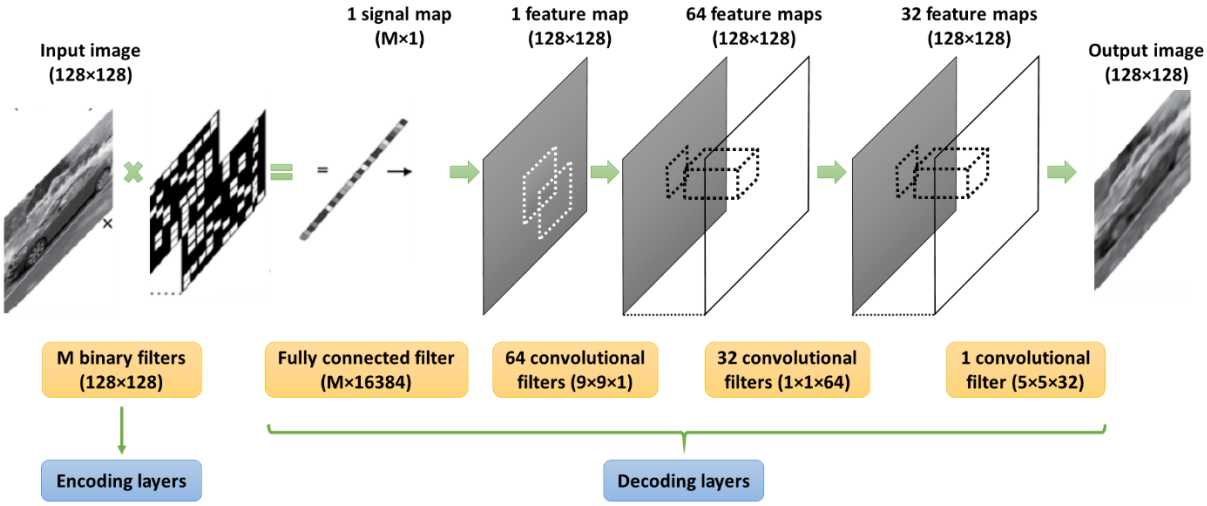


Figure II.3: DCAN architecture.

Thanks to the evolution in the field of deep learning, another compressive sensing system has been proposed in [II.24]. This system allows an instantaneous reconstruction by estimating the output from the input measurements. This approach requires a design based on a network model of neurons, a computing capability linked to the machine used to run the model designed and a large database of learning and validation data.

However, models based on neural networks are less flexible than iterative models because they are based on the learning process and subsequently work only on systems with parameters already determined during the learning phase such as image size and compression rate. The model proposed in [II.24] is a Snapshot Compressive Imaging (SCI) system which refers to compressive sensing systems where multiple frames are mapped into a single measurement frame. It is based on a DMD, an end-to-end CNN algorithm (E2E-CNN) and a plug-and-play (PnP) environment to solve the reverse problem related to the video compressive sensing.

This model is inspired from video CS and is shown in Figure II.4. The video is considered to be a dynamic scene that is represented as a sequence of images with different chrono-dating $[(t_1,\ldots,t_B)]$. The coded frames are then integrated over time on a camera forming a measurement compressed to a single image. In accordance with the measurement and coding models, the iterative algorithms or pre-formed neural networks are used to reconstruct the video.

The principle of SCI video is based on binary spatial coding. Unlike to traditional image processing approaches where signals are acquired directly, in computational imaging, the captured measurement may not be visually explainable but includes the original images. After reconstruction of the video with the model described in this article, the authors compare these performances with those of the best known algorithms in the field of SCI video such as TwIST [II.25], GAP-TV [II.20], GMM [II.19] and DeSCI.

Indeed, the advancement in the field of Deep Learning applied to images have inspired researchers to expand their work on the CS video. Among them, we have Deep fully connected neural network for video CS, Deep tensor ADMM-Net for video SCI problem or E2E-CNN [II.24].

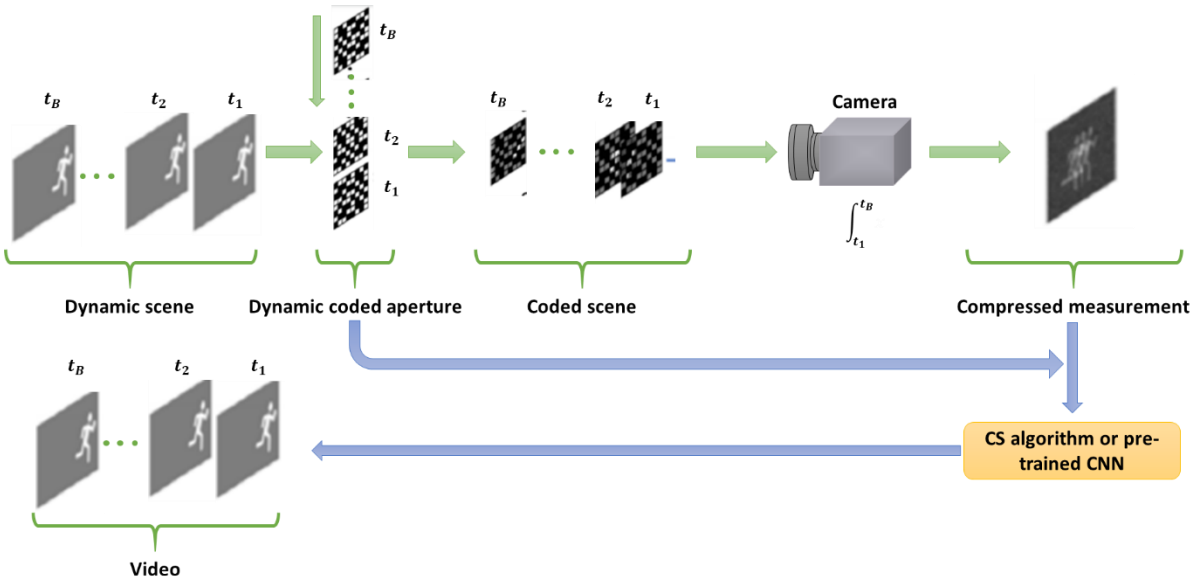


Figure II.4: Video SCI.

The learning of this model is done by applying a residual learning for the encoder-decoder in order to speed up the video CS. It is important to know that this deployment is based on an optical system using a high-speed DMD spatial modulator, because the idea behind this model was to apply a spatial modulation to the image sequences at high speed.

To understand this model, we will detail the mathematical approach behind this video CS model:

Let f represent the dynamic scene that has $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{t}$ as the spatial and temporal variables of the video. Let also $\boldsymbol{x}'$, $\boldsymbol{y}'$ and $\boldsymbol{t}'$ be the coordinates of spatial and temporal measurements. Then the measurement formed on the detector plane is given by the function g

(II.2):

$$g(\boldsymbol{x}',\boldsymbol{y}',\boldsymbol{t}') = \int_1^{N_x}\int_1^{N_y}\int_1^{N_t}\left[f(\boldsymbol{x},\boldsymbol{y},\boldsymbol{t})T(\boldsymbol{x},\boldsymbol{y},\boldsymbol{t}) \times p\left(\frac{\boldsymbol{x}-\boldsymbol{x}'}{\Delta},\frac{\boldsymbol{y}-\boldsymbol{y}'}{\Delta}\right)p_t\left(\frac{\boldsymbol{t}-\boldsymbol{t}'}{\Delta \boldsymbol{t}}\right)\right]dx,dy,dt, \quad \text{(II.2)}$$

where T is the time modulation introduced by the DMD, Δ the pixel pitch, Δt the camera integration time, N_x and N_y the spatial dimensions space, N_t the temporal dimension, p and pt the functions of spatial and temporal pixel sampling.

The sampling of the pixel is discrete and follows the following equation (II.3):

$$\boldsymbol{y} = \sum_{k=1}^{B} \boldsymbol{X_k} \circ \boldsymbol{C_k} + \boldsymbol{G}, \quad \text{(II.3)}$$

where B is the number of pixels, $\boldsymbol{X}$ is the high speed frames, $\boldsymbol{C}$ is the coding patterns, $\boldsymbol{G}$ represents the noise and $\circ$ is the Hadamard product.

Let $(\boldsymbol{i},\boldsymbol{j})$ the position of the pixel and thus the above equation becomes (II.4):

$$\boldsymbol{y_{i,j}} = \sum_{k=1}^{B} \boldsymbol{c_{i,j,k} x_{i,j,k}} + \boldsymbol{g_{i,j}}. \tag{II.4}$$

We define: $\boldsymbol{x} = \left[\boldsymbol{x_1^T}, \dots, \boldsymbol{x_B^T}\right]^{\boldsymbol{T}}$ where $\boldsymbol{x_k} = Vec\,(\boldsymbol{X_k})$. We have $\boldsymbol{D_k} = diag\,(Vec\,(\boldsymbol{C_k}))$ for $k = 1, \dots, B$.

It is obvious that our problem is a compressive sensing problem (II.5):

$$\boldsymbol{y} = \boldsymbol{\varphi x} + \boldsymbol{g}, \tag{II.5}$$

where $\boldsymbol{\varphi} \in \mathbb{R}^{n \times nB}$ is the detection matrix (which is only dense when $n = n_x n_y$), the signal $\boldsymbol{x} \in \mathbb{R}^{nB}$, $\boldsymbol{g} \in \mathbb{R}^n$ and n the noise vector. The matrix $\boldsymbol{\varphi} = [\boldsymbol{D_1}, \dots, \boldsymbol{D_k}]$ consists of diagonal matrices.

It is now clear that the goal of this problem is to reconstruct the signal $\boldsymbol{x}$ from the measurements $\boldsymbol{y}$. As a result, the E2E-CNN model has been proposed. However, this model needs a large database and huge execution time. In addition, if we change the matrix $\boldsymbol{\varphi}$, the neural network must execute another learning process which needs another temporal data. To cope with this, PnP framework is needed to use pre-trained data in an optimization framework in order to establish an equilibrium between the flexibility of the algorithm and its running time.

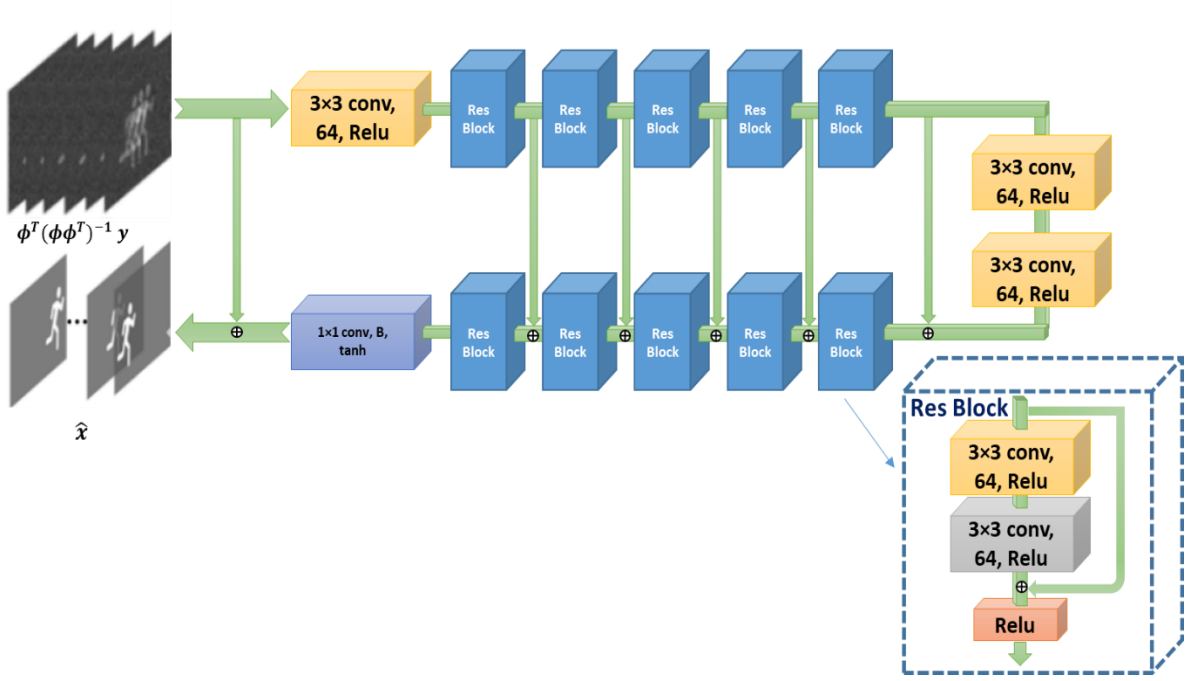


Figure II.5: E2E-CNN architecture.

E2E-CNN architecture, represented in Figure II.5, is based on convolutional encoder-decoder architecture. It consists of 5 residual blocks for the encoder and 5 other blocks for the decoder and the two structures are connected by 2 convolutional layers. Each convolution is followed by ReLU activation function and a batch normalization. In addition, the output of a residual block of the decoder is added to the input of the residual block of the mapped decoder. In this architecture, the authors did not use pooling layers nor the oversampling in order not to lose the details of the images.

The loss function of this model is (II.6):

$$\boldsymbol{L_{CNN}} = \boldsymbol{\alpha}\|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|_2^2 + \boldsymbol{\beta}[\mathbf{1} - MS.SSIM(\boldsymbol{x}, \widehat{\boldsymbol{x}})], \tag{II.6}$$

where $MS.SSIM$ is multiscale structural similarity index between the output of the network. The actual values of $\boldsymbol{x}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are predetermined.

It has been said before that E2E-CNN suffers from a problem of flexibility (for different tasks and different compression ratios) which means that when we change the measurement matrix $\boldsymbol{\varphi}$, we are forced to retrain our model which requires other databases and more execution time. This problem will be corrected by the PnP algorithm that allows to reconstruct $\boldsymbol{x}$ from $\boldsymbol{y}$ and $\boldsymbol{\varphi}$ (II.7):

$$\widehat{\boldsymbol{x}} = \underset{x}{\operatorname{argmin}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\varphi x}\|_2^2 + \tau R(\boldsymbol{x}), \tag{II.7}$$

where τ is an equilibrium parameter between the ℓ_2 norm and the deep denoising prior $R(\boldsymbol{x})$ used to solve the minimization problem without re-training the model which enables the flexibility of the algorithm.

To solve equation (II.7), the ADMM technique could be applied [II.24]. In addition, a denoising problem could be faced and then FFDNet algorithm is needed to solve it. The only drawback with the FFDNet is the undesirable artifacts produced with high compression ratios. This is due to the fact that learning with the FFDNet is made with a Gaussian noise for video compressive sensing: for each iteration, the noise is different. To conclude this approach, [II.24] proposes an implementation of a video compressive sensing algorithm that uses a DMD as a dynamic modulator and an E2E-CNN and PnP algorithms with FFDNet for the video reconstruction.

The most recent research in temporal VCS is presented in [II.26]. It uses 3D CNN from temporal compressive imaging and the residual network concept to exploit temporal and spatial correlation among successive object frames. The idea of measurement calibration algorithm in this approach has improved its final performances on both simulation experiments and optical ones. Another recent work is proposed by Zheng et al. [II.27]. It consists of an encoder-decoder flexible and concise architecture to reconstruct video frames in a CS framework. The reconstruction process is based on deep unfolding structure that uses 2 stages. This reconstruction algorithm outperforms recent deep learning-based algorithms as illustrated in Section II.5 in terms of quality performances.

II.3.2. Spatial VCS

The compression approach in spatial video compressive sensing (SVCS) is based only on the spatial domain which means that the sampling step is processed on the scene video frame by frame. In the reconstruction phase, the frames are recovered independently. Then, the reconstruction algorithm integrates an estimation process to predict the motions of the preliminary recovered frames.

One of the most known conventional (non-neural networks) SVCS methods used is [II.28]. C. Zhao et al. propose an initial recovery of each frame independently using the spatial correlation. Then, they optimize the output using the inter-frame correlation.

As in TVCS, Deep leaning is used to solve SVCS problems. In [II.29], K. Xu et al. propose a robust algorithm to sample the different frames in the spatial domain. Then, they use CNN and RNN to reconstruct the original video and enhance the recovery quality, respectively. The video compressive sensing model was proposed to overcome the limitations of CS cameras.

CSVideoNet was inspired from CNN [II.30], that is a type of deep networks in which filters and pooling operations are applied alternatingly on the input images to extract their main features, and RNN architectures in order to improve the trade-off between compression ratio and spatial-temporal resolution of reconstructed videos. High-speed cameras can capture videos with frame rates that arrive up to 100 frames/s. This model allows to improve the compression ratio and enhance the quality of the video.

Currently, two types of CS cameras are in use: the spatial multiplexing cameras (SMC) and the temporal multiplexing cameras (TMC). Since SMC cameras take fewer measurements than the number of pixels, they suffer from low spatial resolution. However, TMC cameras have low frame rate sensors in spite of their high spatial resolution. Thus, in [II.29], a new model has been proposed in order to overcome the problem of spatial resolution using SMC cameras. This model, represented in Figure II.6, consists of 3 parts: a static encoder, a CNN network dedicated for the extraction of spatial features for each frame of the compressed data and an LSTM network for motion estimation and video reconstruction.

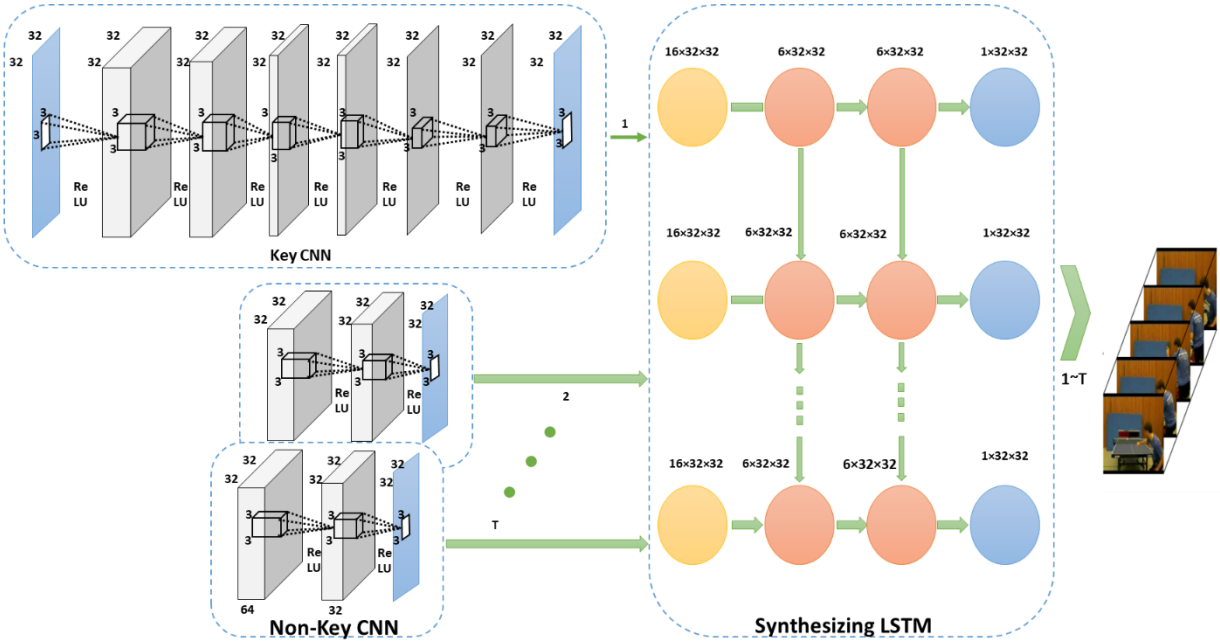


Figure II.6: CSVideoNet architecture.

In the proposed architecture, the design of the encoder is inspired from the CNN's architecture because the main goal does not only consist in extracting visual features but also in preserving the details of the dynamic scenes. For this reason, the authors eliminated the pooling layer which causes an information loss. In fact, the pooling layer allows to progressively decrease the spatial dimensions to reduce the number of parameters and as a result the complexity of the network. In addition, all feature maps have the same dimensions as the reconstructed videos. The first fully connected layer enables to convert the m-dimensional video data into 2D feature maps. The size of the video block in this model is 32 × 32. All convolutional layers are followed by the ReLU activation function except for the last layer. The CNN layers are divided into 2 types: 8 CNN Key layers and 3 non-key CNN layers.

The CNN key layers are compressed with a low compression ratio and non-key CNN layers with a high compression ratio. The weight of the non-key CNN layers is shared to reduce storage requirements. The key frame that represents the input of the CNN key layer is the key image of the video sequence and contains more information than the non-key frames of the non-key CNN

layers. In the implementation of the CSVideoNet solution, for every 10 frames of the video, the 1st one is defined as the key frame.

The LSTM decoder is designed to improve the spatial-temporal resolution. In fact, LSTM is used to extract the movement features that are essential to improve the temporal resolution of the CNN output. In addition, it allows to reduce the size of the model and therefore to obtain a faster speed of reconstruction. For this network, increasing the size of the CNN has been tested, but it does not provide any improvement for the reconstruction because the CNN network is unable to capture temporal features. So, the LSTM network is important to improve the PSNR, which shows that the temporal resolution is processed at this level. This proves the importance of LSTM for video reconstruction. Thus, CSVideoNet is a non-iterative algorithm for real-time applications. The main goal of CSVideoNet is to improve the reconstruction quality and the compression ratio.

In addition to the SVCS models already mentioned, two famous studies, based on stacked denoising autoencoders [II.9] or CNN [II.10] have been proposed for spatial CS to extremely fast reconstruct the frames from the compressively sensed measurements.

In conclusion, it is important to say that the SVCS is originally based on single pixel cameras (SPC) to execute spatial multiplexing and enable video reconstruction by accelerating the acquisition process. However, there have been many extensions to the SPC. One of the famous extensions aims to parallelize the SPC architecture by applying many sensors to separately sample spatial areas of the moving scene [II.31]-[II.32]. These prototypes are better than traditional SPC not only in terms of the manufacturing cost but also in terms of the measurement rate and the quality of the captured frames.

II.3.3. Spatio-Temporal VCS

Video compressive sensing approaches are mostly based on either temporal or spatial domain. These approaches consider one single domain to compress data which is not optimal. However, spatio-temporal data can convey more features that can be used to optimize the sensing and the recovery processes. In fact, the spatio-temporal approach consists in sampling both the temporal and spatial information simultaneously. In this case, the sensing matrix becomes a sensing cube that encode the video in its 3rd dimension. In [II.33], T. Xiong et al. implemented a hardware-friendly algorithm for video compressive sensing where the sensing cube, that is composed of either 1 or 0, is used to encode the video signal into a single coded image. Then, the recovery phase is processed using dictionary and simple sparse recovery. However, the computational cost of the recovery process used in [II.33] remains one the major limitations of this spatiotempral VCS algorithm. In [II.34], the same research team improved their previous work, by adding a CNN layer to extract key features from the frames to enhance the recovery process and improve the sensing quality. D. Lam et al. [II.35] propose a video sampling process divided into 2 steps. Firstly, the 3D image volume is decomposed by a 3D Wavelet transform. Then, a second measurement is obtained by a Noiselet transform. Using this sampling paradigm, the CS reconstruction, with Total Variation, performs successfully.

Motivated by the success of convolutional neural network (CNN) in image processing, 3D CNN are commonly used for decades to extract useful features from video signals. In [II.36], the authors apply a 3D CNN network to extract spatial and temporal features for action recognition. This architecture is used later in [II.37] to design a 3D video compressive sensing algorithm. One other similar approach is proposed in [II.38] which proposes a 3D Convolutional network

that is more suitable to extract spatiotemporal features compared to 2D ConvNets by exploring the effect of different depths and filter sizes.

In the later work of Weil et al. [II.39], an improved version of ISTA-Net+ is proposed which learns an adaptive sampling matrix by simultaneously optimizing the sampling and reconstruction procedures. A two-phase joint deep reconstruction is adopted to selectively exploit spatial-temporal information, consisting of a temporal alignment with a learnable occlusion mask and a multiple frames fusion with spatial temporal feature weighting (see Figure II.7). The separated frames (key and non-key) reconstructions are based on the attention mechanism that applies an adaptive shrinkage-thresholding for discriminative transform coefficients suppression. A specific measure loss is also proposed to ease the network optimization by reducing the inverse mapping space. Accordingly, the reconstruction network is able to adaptively exploit spatial-temporal correlations to recover the full video from few 3D samples of the original video tensor.

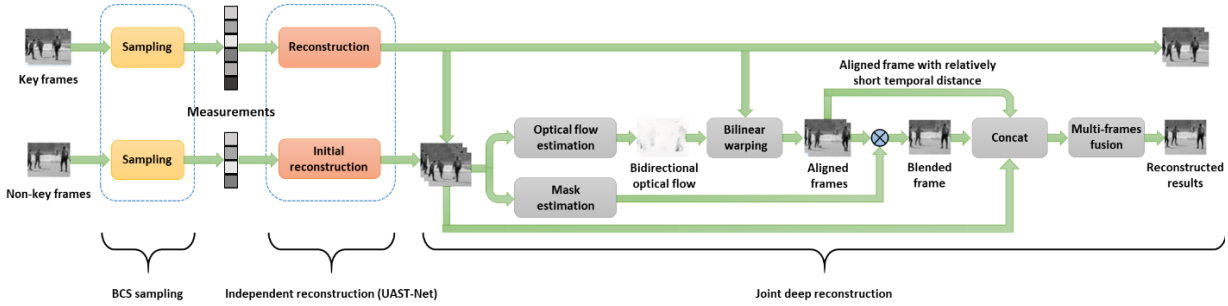


Figure II.7: Overall architecture of STEM-Net.

II.4. Video Single-Pixel Imaging and Video Snapshot Compressive Imaging

According to the modulation, video compressive sensing approaches can be categorized into two main groups: Single-Pixel Imaging systems and Video Snapshot Compressive Imaging (SCI).

II.4.1. Single Pixel Imaging

Single-Pixel Imaging (SPI) is a novel paradigm that enables a device, equipped only with a single point detector called single pixel camera (SPC), to produce high-quality images. The general implementation of the SPI can be schematized as in Figure II.8.

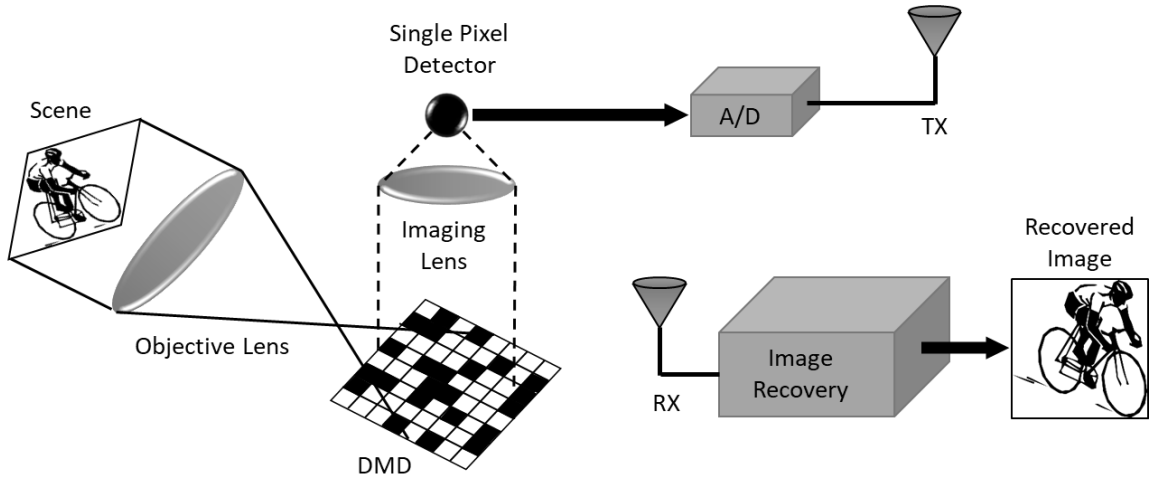


Figure II.8: Single Pixel Camera diagram.

Technically, the single-pixel camera essentially detects the inner product of the video and a set of patterns [II.2]. Then, need to solve an inverse problem to reconstruct the original scene from the raw measurement.

Mathematically, let $(\boldsymbol{X_t})_{t\in\mathbb{N}} \in \mathbb{R}^{N\times 1}$, where $\boldsymbol{X_t}$ is the t-th frame of the detected video. The SPC enables the access to the measurement vector $(\boldsymbol{y_t})_{t\in\mathbb{N}} \in \mathbb{R}^{M\times 1}$, then the acquisition step can be modeled by (II.8):

$$\boldsymbol{y} = \boldsymbol{\varphi X_t \Delta_t}, \forall t, \tag{II.8}$$

where $\boldsymbol{\varphi} \in \mathbb{R}^{M\times N}$ is a dense matrix that encode the list of patterns (one row represents one pattern of the modulator) and $\Delta_{\boldsymbol{t}}$ defines the integration time for each pattern. At each time step, $\boldsymbol{\varphi} \in \mathbb{R}^{M\times N}$ is a matrix containing a set of M patterns. Generally, it is an orthogonal basis (e.g., Fourier, Wavelet, Hadamard). Indeed, using these structural matrices enables to accelerate the computational process because random matrices require huge storage resources which affect the computational mechanism (Figure II.9).

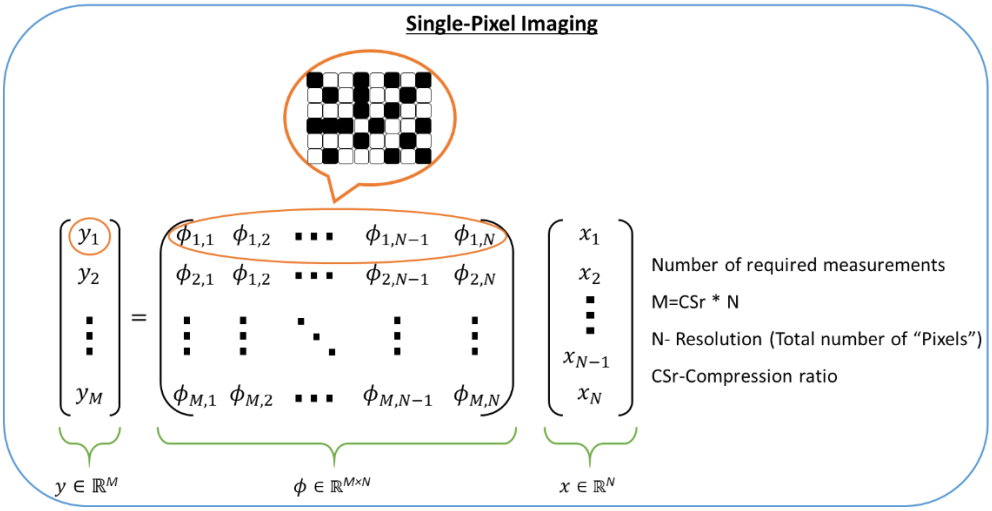


Figure II.9: Model of Single Pixel Imaging.

The most challenging part about single pixel imaging is the reconstruction paradigm. Therefore, many approaches were proposed in the last decade. These reconstruction approaches can be categorized into two groups: traditional approaches and deep learning-based model.

In traditional strategies we find l_2-regularized approaches [II.40] and l_1-regularized approaches [II.2]-[II.41] called also Total-variation approaches. Each approach has its advantages and drawbacks. For l_2-regularized approaches: they are faster but they lead to decreased frame quality. However, l_1-regularized approaches are much slower but they lead to better image quality.

Recently, deep neural networks have been used successfully in signal pixel imaging reconstruction problems. In [II.42], A. l. Mur et al. have exploited the spatio-temporal features of video and proposed a Convolutional Gated Recurrent Units (ConvGRU) based algorithm to reconstruct video frames already captured by a single pixel camera. N. Ducros et al. [II.43] defined a generic convolutional network to recover the original video. In addition, in [II.23], an auto-encoder network is proposed for SPI reconstruction purposes. However, this approach does not exploit the temporal features of video scenes since it enables to reconstruct the video frames independently.

II.4.2. Video Snapshot Compressive Imaging

Compressing high-speed videos is already possible due to the huge research work done in video snapshot compressive imaging (SCI). The video SCI system is composed of two main networks: the hardware encoder and the software reconstruction (decoder) network [II.44]. The hardware decoder represents the optical imaging framework and the software decoder denotes the reconstruction algorithm. The hardware decoder aims to compress the 3D video signal into a 2D measurement and the compression is done across the third dimension (the temporal dimension in this case). This compression aims to avoid huge memory storage and transmission bandwidth. The optical system is called the coded aperture compressive temporal imaging (CACTI) [II.45] system. In this system ad during one exposure time, the video scene is gathered by an objective lens and then coded by a temporal-variant mask (shifting physical mask [II.45]-[II.46] or different patterns on a Digital Micromirror Device (DMD) [II.5]-[II.47]. Then, the output is detected by a Charge Coupled Device (CCD) and then integrated into one single measurement frame.

From a mathematical perspective, a video SCI system captures a dynamic scene of B frames $\boldsymbol{X} \in \mathbb{R}^{h \times w \times B}$ (h and w are the height and the weight of the frame, respectively) is modulated by B masks $\boldsymbol{C} \in \mathbb{R}^{h \times w \times B}$ before being integrated into one single measurement frame $\boldsymbol{Y} \in \mathbb{R}^{h \times w}$ by a camera sensor in one exposure time (B frame). This operation is expressed as follows (II.9):

$$\boldsymbol{y} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{B}} \boldsymbol{X_k} \circ \boldsymbol{C_k} + \boldsymbol{G}, \tag{II.9}$$

where $\circ$ and $\boldsymbol{G} \in \mathbb{R}^{h \times w}$ denote the Hadamard product and noise, respectively. Then, we define $\boldsymbol{y} = Vec(\boldsymbol{Y}) \in \mathbb{R}^{hw}$ and $\boldsymbol{g} = Vec(\boldsymbol{G}) \in \mathbb{R}^{hw}$. Correspondingly, we define $\boldsymbol{x} \in \mathbb{R}^{hw}$ as (II.10):

$$\boldsymbol{x} = Vec(\boldsymbol{X}) = [Vec(\boldsymbol{X_1})^T, \dots, Vec(\boldsymbol{X_B})^T]^T. \tag{II.10}$$

The measurement y can then be expressed as (II.11):

$$\boldsymbol{y} = [\boldsymbol{D_1}, \dots, \boldsymbol{D_B}]\boldsymbol{x} + \boldsymbol{g}, \tag{II.11}$$

where $\boldsymbol{D_B} = diag\left(Vec\left(\boldsymbol{C_B}\right)\right) \in \mathbb{R}^{hw \times hw}$, for $b = 1 \dots B$. We have in this case a matrix $[\boldsymbol{D_1}, \dots, \boldsymbol{D_B}]$ that is highly structured and sparse. Depending on the theoretical study in [II.48], the original video can be reconstructed from the single measurement frame y (Figure II.10).

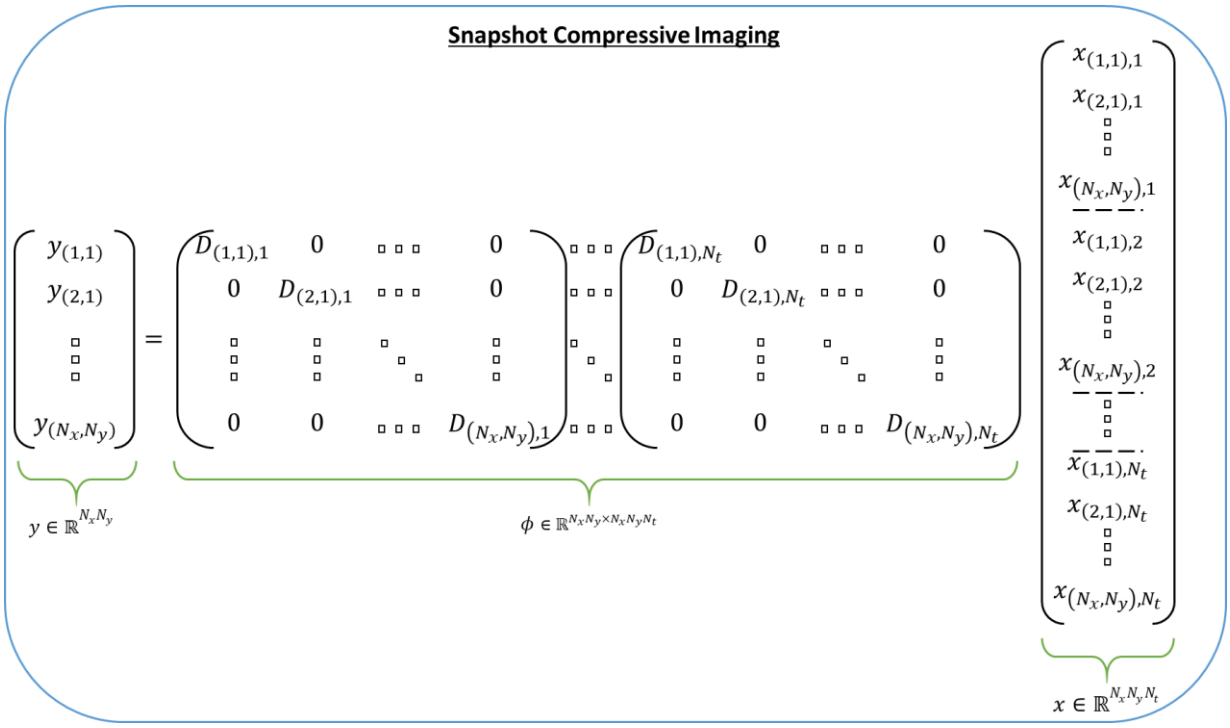


Figure II.10: Model of Snapshot Compressive Imaging.

The second important part of video SCI is the reconstruction process which aim to recover the original video from the 2D measurement frames and the masks. This process is crucial to have a practical and efficient video SCI system. In the literature, the reconstruction algorithms could be categorized into two categories: optimization-based methods and Deep Learning based algorithms. The optimization based algorithms, such as GAP-TV [II.20], GMM [II.19], DeSCI [II.49], and PnP-FFDNet [II.50], require huge computational resources and large reconstruction time. For instance, DeSCI, that has led recently the state-of-the-art optimization-based approaches, takes hours to generate a 256×256×8 video from one single measurement frame). However, GAP-TV is a fast algorithm but it cannot provide a good reconstruction. In general, to use an algorithm in a real-world application, we need a PSNR 30 which is not the case for GAP-TV [II.50].

In Deep Learning based methods [II.21]-[II.24]-[II.29]-[II.51]-[II.10]-[II.52]-[II.53]-[II.54], these problems have been ameliorated.

Indeed, Z. Cheng et al. [II.51] proposed a bidirectional neural network-based method to reconstruct the video frames from the measurement matrix and the masks by exploiting the correlation of sequential frames. The idea behind this approach, illustrated in Figure II.11, is based on two main sub-networks: A deep convolutional neural network (CNN) with ResBlock [II.55] and a self-attention module [II.56] in order to reconstruct the first frame (reference frame), and a bidirectional neural network to reconstruct the rest of the frames. To improve the quality of the reconstruction, an adversarial training is defined with the Mean Square Error (MSE) loss. However, the main drawback of BIRNAT is its impractical computational time in the training phase (weeks to train a model of size 256 × 256 × 8 [II.57]) and its huge GPU memory consumption that make it unsuitable for large-scale SCI applications especially with the high-resolution videos used in real life.

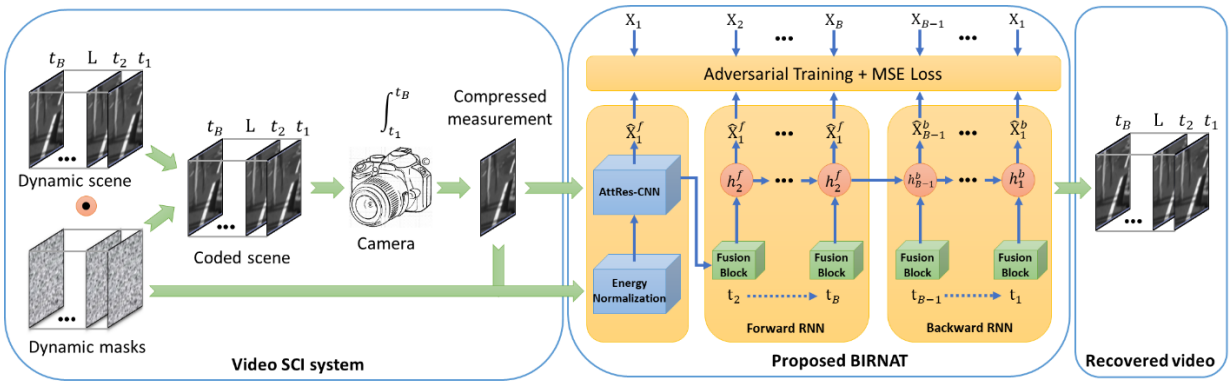


Figure II.11: BIRNAT architecture.

The GPU memory storage problem in the training phase is ameliorated in RevSCI-Net [II.57] by introducing a reversible CNN network to free the memory from the middle activation generated by each layer of the network. This technique enables to reduce the memory cost from $O(N)$ to $O(1)$ (where N is the number of layers). RevSCI-Net rely on an end-to-end CNN model exploring the temporal and spatial correlations of the original video.

In addition to the speed issue, some deep learning-based reconstruction algorithms, such as BIRNAT, suffer from flexibility and adaptability problems which affect their performances. Therefore, Z. Wang et al. [II.58] introduced a Meta Modulated Convolutional Network (MetaSCI) as a new scalable and adaptive reconstruction model. MetaSCI is a fully CNN approach that exploits the fast adaption encoding paradigm in order to efficiently reconstruct the video frames in terms of memory consumption.

Recently, an ensemble learning based algorithm is proposed in [II.59], originally exploited in inverse problems, to enhance the scalability of video SCI reconstruction approaches. Zongliang et al. [II.60] still work on combining iterative algorithms and deep neural networks. An online Plug-and-play algorithm is proposed to adaptively update the model's parameters using the PnP iteration, which enhance the network's noise resistance. The second part of the paper focus on color SCI videos. The authors present an ADMM optimization and deep neural network to improve the output quality. Finally, a deep equilibrium-based model is proposed in [II.61] that combines data-driven regularization and stable convergence to deal with the problems of memory requirement and unstable reconstruction in some exiting approaches.

Obviously, both categories have their advantages and drawbacks, which make this research direction challenging and very promising for the future if we aim to come up with a memory friendly model that consume less computational cost for our daily life applications.

II.5. Comparative Study

In this comparative study, we aim to compare the performances of some well-known optimization-based VCS algorithms and Deep Learning-based approaches.

II.5.1. Optimization-Based VCS Algorithms

Table II.1 presents the complexity of optimization-based sparse recovery algorithms as well as the minimum measurement requirement. It shows also some challenging issues considered as crucial when designing CS reconstruction algorithms: Sparsity information, Noise resistance and hardware feasibility:

- The sparsity information: it may not be provided for the reconstruction process.

- Noise resistance: It is important to design a recovery algorithm where the measurements are not affected by measurement noise.
- Hardware feasibility: low-complexity algorithms can usually be implemented on hardware devices for real-world applications.

Table II.1: Complexity, minimum measurement requirement and crucial properties of CS recovery algorithms.

Algorithms	Min. number of measurements	Complexity	No requirement of sparsity information	Noise resistance	Hardware implementation
Basis Pursuit	$klog(N)$	$O(N^3)$	✓		✓
OMP	$klog(N)$	$O(kMN)$	✓		✓
StOMP	$Nlog(N)$	$O(Nlog(N))$		✓	✓
ROMP	$klog(N)^2$	$O(kMN)$	✓		✓
CoSaMP	$klog(N)$	$O(MN)$		✓	✓
Subspace Pursuits	$klog\left(\frac{N}{k}\right)$	$O(MNlog(k))$		✓	✓

II.5.2. Deep Learning-Based VCS Algorithms

A quantitative and a qualitative comparison of Deep Learning-based approaches will be presented in this section.

II.5.2.1. Quantitative Comparison

Training Details

It is important to mention that video compressive sensing algorithms (acquisition and reconstruction) does not have a particular training dataset and can be applied on any scene. Indeed, all experiments are trained on Densely Annotated VIdeo Segmentation (DAVIS2017) [II.62] dataset. DAVIS2017 is an object segmentation dataset that contains 90 different videos with a resolution of 480 × 894. To efficiently train the state-of-the-art algorithms, 6516 videos of size 8 × 256 × 256 are generated from DAVIS2017 to learn different parameters on the same compression ratio $\frac{1}{8}$. Then, all algorithms are tested on 6 simulation datasets: Aerial, Drop, Kobe, Runner, Traffic, Vehicle to evaluate their performances. All experiments are tested on the RTX 2080 GPU and Intel® Core™ i7-9700K CPU (3.6 GHz, 32GB memory).

Comparison Metrics

The following three metrics are employed to compare different approaches:

- Peak Signal to Noise Ratio (PSNR) [II.22]: Quality metric.

- Structural Similarity Index (SSIM) [II.22]: Quality metric.
- Reconstruction Time: this metric is used to prove whether the algorithm can be applied in real-time applications at the testing step.

Benchmark Results

We present a quantitative comparison to compare the quality performances of the following VCS algorithms: GAP-TV [II.20], DeSCI [II.49], PnP-FFDNet [II.50], Pnp-FastDVDNet [II.63], GAP-FastDVDNet (online) [II.60], DE-RNN [II.61], DE-GAPFFDnet [II.61], E2E-CNN [II.24], BIRNAT [II.51], MetaSCI [II.58], RevSCI [II.57], DeepUnfold-VCS [II.27], ELP-Unfolding [II.59].

Table II.2: Quantitative comparison of different approaches for video compressive sensing system. The average results of PSNR in dB, SSIM and reconstruction time (seconds) per measurement. GAP-TV and DeSCI are tested on CPU while other approaches are on GPU.

Algorithms	Year	Aerial	Drop	Kobe	Runner	Traffic	Vehicle	Average	Time
GAP-TV [II.20]	2016	25.03 0.828	33.81 0.963	26.45 0.845	28.48 0.899	20.90 0.715	24.82 0.838	26.58 0.848	4.2
DeSCI [II.49]	2019	25.33 0.860	43.22 0.993	33.25 0.952	38.76 0.969	28.72 0.925	27.04 0.909	32.72 0.935	6180
PnP-FFDNet [II.50]	2020	24.02 0.814	40.87 0.988	30.47 0.926	32.88 0.938	24.08 0.833	24.32 0.836	29.44 0.889	3.0
PnP-FastDVDNet [II.63]	2021	27.89 0.897	41.82 0.989	32.73 0.946	36.29 0.962	27.95 0.932	27.32 0.925	32.35 0.942	18
GAP-FastDVDNet (online) [II.60]	2022	28.24 0.897	41.95 0.989	32.95 0.951	36.41 0.962	28.16 0.934	27.64 0.928	32.56 0.944	35
DE-RNN [II.61]	2022	24.83 0.855	30.16 0.909	21.46 0.697	27.85 0.818	19.47 0.715	23.65 0.832	24.54 0.804	4.68
DE-GAP-FFDnet [II.61]	2022	26.02 0.892	39.89 0.992	29.32 0.952	33.06 0.971	24.71 0.907	25.85 0.905	29.81 0.936	1.90
E2E-CNN [II.24]	2020	27.18 0.969	36.56 0.949	27.79 0.807	34.12 0.947	24.62 0.840	26.43 0.882	29.45 0.882	0.0312
BIRNAT [II.51]	2020	28.99 0.927	42.28 0.992	32.71 0.950	38.70 0.976	29.33 0.942	27.84 0.927	33.31 0.951	0.16

MetaSCI [II.58]	2021	28.31 0.904	40.61 0.985	30.12 0.907	37.02 0.967	26.95 0.888	27.33 0.906	31.72 0.926	0.025
RevSCI [II.57]	2021	29.35 0.924	42.93 0.992	33.72 0.957	39.40 0.977	30.02 0.949	28.12 0.937	33.92 0.956	0.19
DeepUnfold-VCS [II.27]	2022	30.86 0.965	44.43 0.997	35.24 0.984	41.47 0.994	31.45 0.977	30.32 0.976	35.63 0.982	1.43
ELP-Unfolding [II.59]	2022	30.68 0.943	44.99 0.995	34.41 0.966	41.16 0.986	31.58 0.962	29.65 0.960	35.41 0.969	0.24

Table II.2 summarizes the comparison of several VCS algorithms on PSNR, SSIM and the reconstruction time. From this table, different performance results are plotted in Figure II.12 and Figure II.15 for visualization purposes. From Figure II.12 and Figure II.13, we notice that iterative algorithms (GAP-TV, DeSCI, PnP-FFDnet and PnP-FastDVDnet) provide inferior quality performance results (both in terms of PSNR and SSIM) with low recovery speed (from one second to even hours) which threaten their hardware implementation for real-time applications. However, the other deep learning-based algorithms outperforms these iterative approaches in terms of quality performances with faster reconstruction time. These performances can prove the potential usability of deep learning-based approaches in real-time applications. From Figure II.14 and Figure II.15, we notice that DeSCI, the iterative algorithm, provides little improvement over some deep learning-based algorithms on the Kobe, Runner and Drop (e.g., PSNR: +2.22%, +1.65% and +0.15% over BIRNAT, +6.42%, +10, 39% and +4.7% over MetaSCI on Drop, Kobe and Runner, respectively). Indeed, these datasets are characterized by high-speed motions of some objects. However, we infrequently find these features in DAVIS2017 dataset, which explain these results. As a result, datasets of high-speed motions are recommended while training these deep learning-based algorithms to enhance their quality performances. In addition, we note that the recent ensemble learning-based algorithm (ELP-Unfolding) is proposed to enhance the performance of the previous algorithms by strategically generate and combine multiple models which confirm the fact to consider this technique as a promising research topic in video reconstruction. In addition, we notice from Figure II.14 and Figure II.15, that DeepUnfold-VCS outperforms the rest of the proposed algorithms in terms of quality performances (PSNR and SSIM) on almost all experiments. In fact, the authors propose an algorithm that combines iterative strategy and deep learning. In addition, they used a deep unfolding approach and exploit its interpretability to reconstruct the video scene. In the other hand, Meta-SCI is the fastest VCS reconstruction approach with good quality performances. Also, in contrast to DeepUnfold-VCS, it proposes a CNN-based network which is much faster than recurrent neural nets. It can be used in real-time applications that require prompt capture and reconstruction time. To conclude, recent deep learning-based approaches proposed for VCS purposes present good quality performances and research in this field becomes very competitive and very challenging to come up with the fastest algorithm.

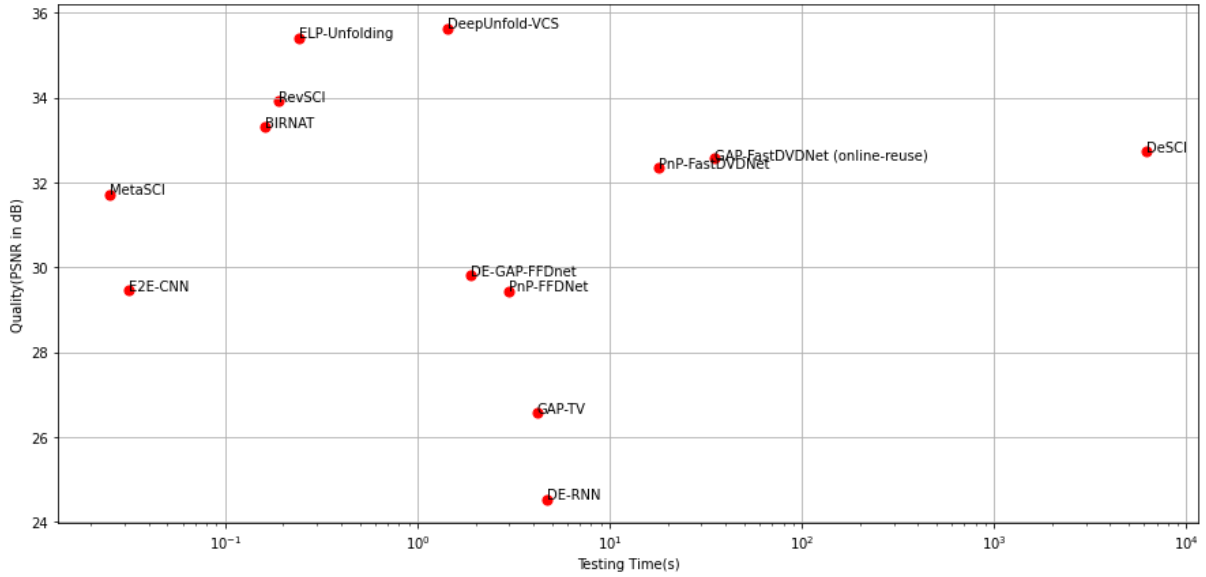


Figure II.12: Trade-off between quality (in PSNR) and testing time of several VCS reconstruction algorithms.

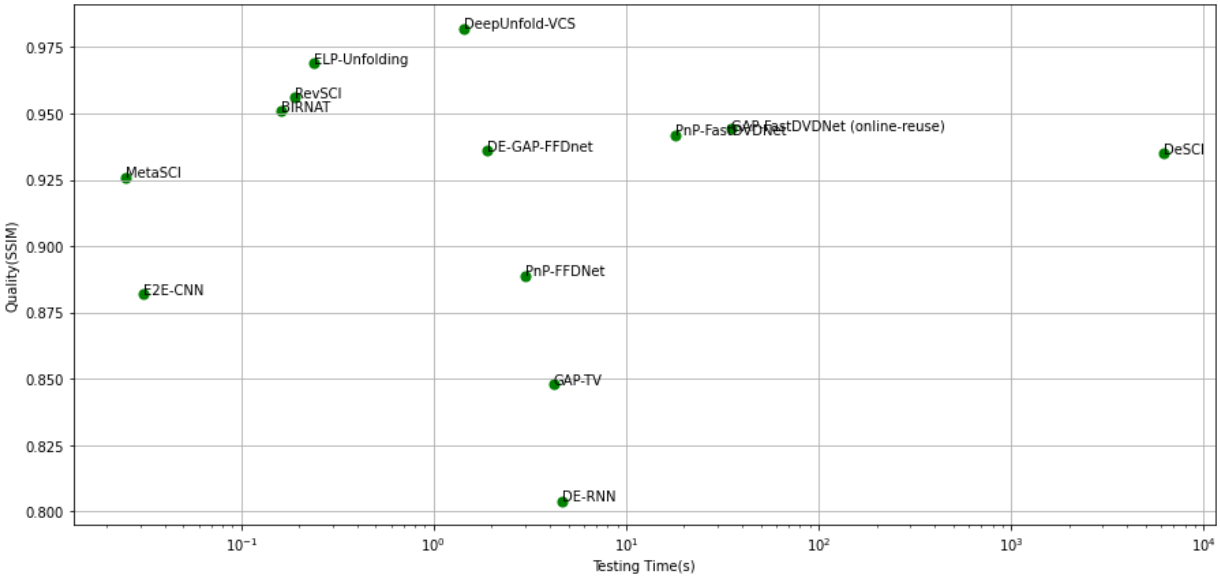


Figure II.13: Trade-off between quality (in SSIM) and testing time of several VCS reconstruction algorithms.

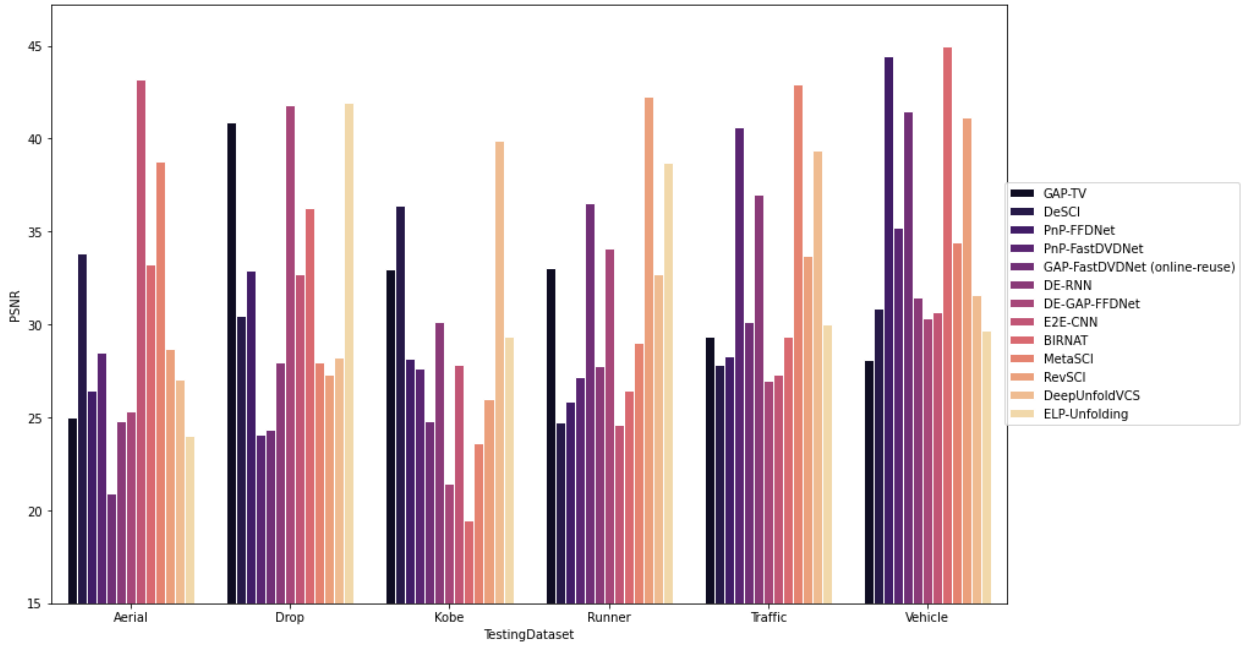


Figure II.14: Performance comparison based on PSNR obtained by several VCS reconstruction algorithms on 6 grayscale benchmark data.

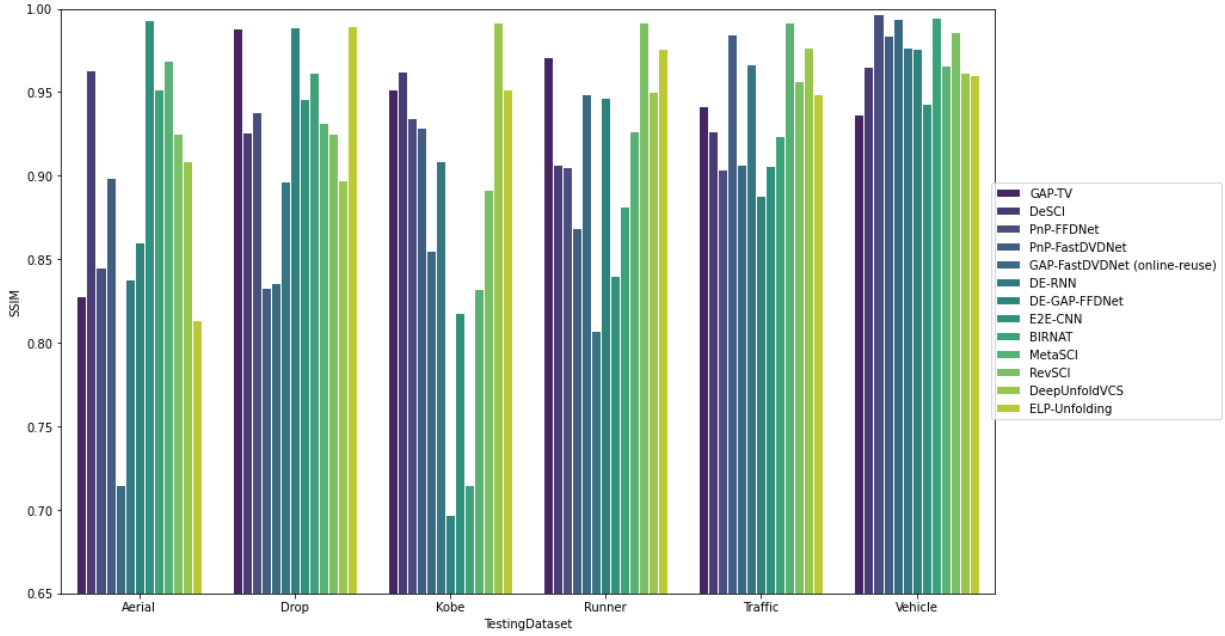


Figure II.15: Performance comparison based on SSIM obtained by several VCS reconstruction algorithms on 6 grayscale benchmark data.

II.5.2.2. Qualitative Comparison

Different VCS approaches, together with their specific advantages and limitations, are summarized in Table II.3 and Table II.4 to compare their qualitative performances that should be taken into consideration while implementing the network for a particular application.

Table II.3: Different algorithms for video compressive sensing (Part 1).

Classification type	Category	Traditional/DL	Algorithm's class	Examples	Advantages	Limitations
Sampling strategy	Temporal VCS	Traditional	GMM based	GMM [II.19]	Parallel processing can be used, good quality performances, flexibility	Too computationally slow, slow reconstruction process, use only the temporal domain to compress the video
			TV based	GAP-TV [II.20]		
		DL		Deep fully connected network for VCS [II.21], DCAN [II.23], E2E-CNN [II.24]		
	Spatial VCS	Traditional	Reweighted residual sparsity	VCS-RRS [II.28]	Good performances, flexibility	use only the spatial domain to compress the video, Low scalability
			Extended architectures of SPC	FPA-CS [II.31], LiSens [II.32]	High spatial resolution, flexibility	Expensive
		DL	RNN based	CSVideoNet [II.29], SDA-CS [II.9]		
			CNN based	ReconNet [II.10]		

	Spatio-temporal VCS	Traditional		ST-approach [II.33]	Sample the temporal and spatial dimension simultaneously	Huge computational cost
			TV based	3D-Wavelet and 3D-Noiselet approach [II.35]		
		DL	CNN based	[II.35]-[II.34]-[II.36]-[II.37]-[II.38]		

Table II.4: Different algorithms for video compressive sensing (Part 2).

Classification type	Category	Traditional/DL	Algorithm's class	Examples	Advantages	Limitations
Modulation strategy	Video Snapshot Compressive Imaging	Traditional	Sparse based	Low-Cost Compressive Sensing for Color Video and Depth	Good flexibility	Very slow algorithms
			TV based	TwIST [II.25], GAP-TV [II.20]		
			GMM	GMM (Off-line training) [II.19]		
			Dictionary Learning	3D K-SVD		
		DL	Deep Unfolding	ADMM-Net [II.53] BIRNAT [II.51], RevSCI-Net [II.57] MetaSCI-Net [II.58]	Good reconstruction quality, Fast algorithms, less GPU memory consumption (RevSCI-Net, MetaSCI-Net)	Less flexible, Not robust to real data noise,huge GPU memory consumption (BIRNAT, ADMM-Net)

		DL	Plug and Play	[II.24]-[II.50]	Good trade-off between accuracy, speed and flexibility	The training phase can be slow
			End-to-End	E2E-CNN [II.24]	Fast algorithms	Low flexibility
	Single pixel Cameras	Traditional	l_1-regularized approach		Good quality	Slow
			l_2-regularized approach		Fast	Less good quality
		DL	RNN based	[II.42]	Good reconstruction quality,	Huge computational time
			CNN based	[II.43]	Faster training	Huge memory consumption
			Auto-encoder based	[II.23]		

II.6. Conclusions

In this chapter, after reformulating the compressive sensing paradigm, we have closely reviewed the fundamentals of image and video compressive sensing. In addition, we analyzed the backbone deep learning based architectures for image and video CS in order to provide the CS community the essential background knowledge. Indeed, we classified different concepts of compressive sensing in general and image and video compressive sensing in particular into categories to facilitate their understanding. The methods have been analyzed in this review from different angles: network architecture, contribution, complexity and performance results. In conclusion, compressing sensing is a promising research direction in order to optimize data gathering and processing. Although there have been great achievements in this field, there is still room for improvement in image and video compressive sensing using neural networks. In the next chapters, two video approaches will be designed, implemented and deeply discussed.

References

[II.1] Donoho, D.L. Compressed sensing. *IEEE Trans. Inf. Theory* 2006, 52, 1289– 1306. doi:10.1109/TIT.2006.871582.

[II.2] Duarte, M.F.; Davenport, M.A.; Takhar, D.; Laska, J.N.; Sun, T.; Kelly, K.F.; Baraniuk, R.G. Single-pixel imaging via compressive sampling. *IEEE Signal Process. Mag.* 2008, 25, 83–91, doi:10.1109/MSP.2007.914730.

[II.3] Veeraraghavan, A.; Reddy, D.; Raskar, R. Coded Strobing Photography: Compressive Sensing of High Speed Periodic Videos. *IEEE Trans. Pattern Anal. Mach. Intell.* 2011, 33, 671–686. doi:10.1109/TPAMI.2010.87.

[II.4] Wakin, M.; Laska, J.N.; Duarte, M.F.; Baron, D.; Sarvotham, S.; Takhar, D.; Kelly, K.F.; Baraniuk, R.G. Compressive imaging for video representation and coding. In *Proceedings of the Picture Coding Symposium*, Beijing, China, 24–26 April 2006; pp. 1–6.

[II.5] Reddy, D.; Veeraraghavan, A.; Chellappa, R. P2C2: Programmable pixel compressive camera for high speed imaging. In Proceedings of *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, CO, USA, 20–25 June 2011; pp. 329–336.

[II.6] Kittle, D.; Choi, K.; Wagadarikar, A.; Brady, D.J. Multiframe image estimation for coded aperture snapshot spectral imagers. *Appl. Opt.* 2010, 49, 6824–6833.

[II.7] Hitomi, Y.; Gu, J.; Gupta, M.; Mitsunaga, T.; Nayar, S.K. Video from a single coded exposure photograph using a learned over-complete dictionary. In Proceedings of the *IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, 6–13 November 2011; pp. 287–294.

[II.8] Xuan, Y.; Yang, C. 2Ser-Vgsr-Net: A Two-Stage Enhancement Reconstruction Based On Video Group Sparse Representation Network For Compressed Video Sensing. In Proceedings of the 2020 *IEEE International Conference on Multimedia and Expo (ICME)*, London, UK, 6–10 July 2020; pp. 1–6, doi:10.1109/ICME46284.2020.9102849.

[II.9] Mousavi, A.; Patel, A.B.; Baraniuk, R.G. A deep learning approach to structured signal recovery. In Proceedings of the 2015 *53rd Annual Allerton Conference on Communication*, Control, and Computing (Allerton), Monticello, IL, USA, 29 September–2 October 2015; pp. 1336–1343, doi:10.1109/ALLERTON.2015.7447163.

[II.10] Kulkarni, K.; Lohit, S.; Turaga, P.; Kerviche, R.; Ashok, A. ReconNet: Non-iterative reconstruction of images from compressively sensed measurements. In Proceedings of the *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 27–30 June 2016; pp. 449–458.

[II.11] Yao, H.T.; Dai, F.; Zhang, S.L.; Zhang, Y.D.; Tian, Q.; Xu, C.S.; DR2 -Net: Deep residual reconstruction network for image compressive sensing. *Neurocomputing* 2019, 359, 483–493.

[II.12] Zhang, J.; Ghanem, B. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1828–1837.

[II.13] Ito, D.; Takabe, S.; Wadayama, T. Trainable ISTA for Sparse Signal Recovery. *IEEE Trans. Signal Process.* 2019, 67, 3113–3125. doi:10.1109/TSP.2019.2912879.

[II.14] Su, H.; Bao, Q.; Chen, Z. ADMM–Net: A Deep Learning Approach for Parameter Estimation of Chirp Signals Under Sub-Nyquist Sampling. *IEEE Access* 2020, 8, 75714–75727. doi:10.1109/ACCESS.2020.2989507.

[II.15] Shi, W.; Jiang, F.; Liu, S.; Zhao, D. Image Compressed Sensing Using Convolutional Neural Network. *IEEE Trans. Image Process.* 2020, 29, 375–388. doi:10.1109/TIP.2019.2928136.

[II.16] Canh, T.N.; Jeon, B. Multi-Scale Deep Compressive Sensing Network. In *Proceedings of the 2018 IEEE Visual Communications and Image Processing (VCIP)*, Taichung, Taiwan, 9–12 December 2018; pp. 1–4, doi:10.1109/VCIP.2018.8698674.

[II.17] Canh, T.N.; Jeon, B. Difference of Convolution for Deep Compressive Sensing. In *Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan, 22–25 September 2019; pp. 2105–2109, doi:10.1109/ICIP.2019.8803165.

[II.18] Shi, W.; Jiang, F.; Liu, S.; Zhao, D. Scalable Convolutional Neural Network for Image Compressed Sensing. In *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 15–20 June 2019; pp. 12282–12291. doi:10.1109/CVPR.2019.01257.

[II.19] Yang, J.; Yuan, X.; Liao, X.; Llull, P.; Brady, D.J.; Sapiro, G.; Carin, L.; Video compressive sensing using Gaussian mixture models. *IEEE Trans. Image Process.* 2014, 23, 4863–4878.

[II.20] Yuan, X. Generalized alternating projection based total variation minimization for compressive sensing. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, 25–28 September 2016; pp. 2539–2543.

[II.21] Iliadis, M.; Spinoulas, L.; Katsaggelos, A.K. Deep fully-connected networks for video compressive sensing. *Digit. Signal Process.* 2018, 72, 9–18.

[II.22] Horé, A.; Ziou, D. Image Quality Metrics: PSNR vs. SSIM. In Proceedings of the 2010 *20th International Conference on Pattern Recognition*, Istanbul, Turkey, 23–26 August 2010; pp. 2366–2369, doi:10.1109/ICPR.2010.579.

[II.23] Higham, C.F.; Murray-Smith, R.; Padgett, M.J.; Edgar, M.P. Deep learning for realtime single-pixel video. *Sci. Rep.* 2018, 8, 2369.

[II.24] Qiao, M.; Meng, Z.; Ma, J.; Yuan, X. Deep learning for video compressive sensing. *APL Photonics 2020*, 5, 030801. doi:10.1063/1.5140721.

[II.25] Bioucas-Dias, J.M.; Figueiredo, M.A.T. A New TwIST: Two-Step Iterative Shrinkage/Thresholding Algorithms for Image Restoration. *IEEE Trans. Image Process.* 2007, 16, 2992–3004. doi:10.1109/TIP.2007.909319.

[II.26] Zhang, L.; Lam, E.Y.; Ke, J. Temporal compressive imaging reconstruction based on a 3D-CNN network. *Opt. Express* 2022, 30, 3577–3591.

[II.27] Zheng, S.; Yang, X.; Yuan, X. Two-Stage is Enough: A Concise Deep Unfolding Reconstruction Network for Flexible Video Compressive Sensing. *arXiv* 2022, arXiv:2201.05810.

[II.28] Zhao, C.; Ma, S.; Zhang, J.; Xiong, R.; Gao, W. Video compressive sensing reconstruction via reweighted residual sparsity. *IEEE Trans. Circuits Syst. Video Technol.* 2017, 27, 1182–1195.

[II.29] Xu, K.; Ren, F. CSVideoNet: A real-time end-to-end learning framework for high-frame-rate video compressive sensing. In *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, Lake Tahoe, NV, USA, 12–15 March 2018; pp. 1680–1688.

[II.30] Albawi, S.; Mohammed, T.A.; Al-Zawi, S. Understanding of a convolutional neural network. In *Proceedings of the 2017 International Conference on Engineering and Technology (ICET)*, Antalya, Turkey, 21–23 August 2017; pp. 1–6. doi:10.1109/ICEngTechnol.2017.8308186.

[II.31] Chen, H.; Salman, Asif, M.; Sankaranarayanan, A.C.; Veeraraghavan, A. FPACS: Focal plane array-based compressive imaging in short-wave infrared. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 7–12 June 2015; pp. 2358–2366, doi:10.1109/CVPR.2015.7298849.

[II.32] Wang, J.; Gupta, M.; Sankaranarayanan, A.C. LiSens—A Scalable Architecture for Video Compressive Sensing. In Proceedings of the *2015 IEEE International Conference on Computational Photography (ICCP)*, Houston, TX, USA, 24–26 April 2015; pp. 1–9, doi:10.1109/ICCPHOT.2015.7168369.

[II.33] Xiong, T.; Rattray, J.; Zhang, J.; Thakur, C.S.; Chin, S.; Tran, T.D.; Etienne-Cummings, R. Spatiotemporal compressed sensing for video compression. In *Proceedings of the 2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Boston, MA, USA, 6–9 August 2017.

[II.34] Wang, X.; Zhang, J.; Xiong, T.; Tran, T.D.; Chin, S.P.; Etienne-Cummings, R. Using deep learning to extract scenery information in real time spatiotemporal compressed sensing. In *Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, Florence, Italy, 27–30 May 2018; pp. 1–4.

[II.35] Lam, D.; Wunsch, D. Video compressive sensing with 3-D wavelet and 3-D noiselet. In Proceedings of the *19th IEEE International Conference on Image Processing (ICIP '12)*, Orlando, FL, USA, USA, 30 September–3 October 2012. doi:10.1109/ICIP.2012.6467004.

[II.36] Ji, S.; Xu, W.; Yang, M.; Yu, K. 3D convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 2013, 35, 221–231.

[II.37] Zhao, Z.; Xie, X.; Liu, W.; Pan, Q. A hybrid-3D convolutional network for video compressive sensing. *IEEE Access* 2020, 8, 20503–20513.

[II.38] Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning spatiotemporal features with 3D convolutional networks. In Proceedings of the *IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 7–13 December 2015; pp. 4489–4497.

[II.39] Wei, Z.; Yang, C.; Xuan, Y. Efficient Video Compressed Sensing Reconstruction via Exploiting Spatial-Temporal Correlation With Measurement Constraint. In *Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME)*, Shenzhen, China, 5–9 July 2021; pp. 1–6.

[II.40] Rousset, F.; Ducros, N.; Farina, A.; Valentini, G.; D'Andrea, C.; Peyrin, F. Adaptive Basis Scan by Wavelet Prediction for Single-pixel Imaging. *IEEE Trans. Comput. Imaging* 2016, 3, 36–46.

[II.41] Baraniuk, R.G.; Goldstein, T.; Sankaranarayanan, A.C.; Studer, C.; Veeraraghavan, A.; Wakin, M.B. Compressive video sensing: Algorithms, architectures, and applications. *IEEE Signal Process. Mag.* 2017, 34, 52–66.

[II.42] Mur, A.L.; Peyrin, F.; Ducros, N. Recurrent Neural Networks for Compressive Video Reconstruction. In *Proceedings of the IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, Iowa City, IA, USA, 3–7 April 2020; pp. 1651–1654.

[II.43] Ducros, N.; Lorente Mur, A.; Peyrin, F. A completion network for reconstruction from compressed acquisition. In Proceedings of the 2020 *IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, Iowa City, IA, USA, 3–7 April 2020; pp. 619–623.

[II.44] Yuan, X.; Brady, D.; Katsaggelos, A.K. Snapshot compressive imaging: Theory, algorithms and applications. *IEEE Signal Process. Mag.* 2020, 38, 65–88.

[II.45] Llull, P.; Liao, X.; Yuan, X.; Yang, J.; Kittle, D.; Carin, L.; Sapiro, G.; Brady, D.J. Coded aperture compressive temporal imaging. *Opt. Express* 2013, 21, 10526–10545. doi:10.1364/OE.21.010526.

[II.46] Koller, R.; Schmid, L.; Matsuda, N.; Niederberger, T.; Spinoulas, L.; Cossairt, O.; Schuster, G.; Katsaggelos, A.K. High spatio-temporal resolution video with compressed sensing. *Opt. Express* 2015, 23, 15992–16007.

[II.47] Sun, Y.; Yuan, X.; Pang, S. Compressive high-speed stereo imaging. *Opt Express* 2017, 25, 18182–18190.

[II.48] Jalali, S.; Yuan, X. Snapshot compressed sensing: Performance bounds and algorithms. *IEEE Trans. Inf. Theory* 2019, 65, 8005–8024.

[II.49] Liu, Y.; Yuan, X.; Suo, J.; Brady, D.J.; Dai, Q. Rank Minimization for Snapshot Compressive Imaging. *IEEE Trans. Pattern Anal. Mach. Intell.* 2019, 41, 2990– 3006. doi:10.1109/TPAMI.2018.2873587.

[II.50] Yuan, X.; Liu, Y.; Suo, J.; Dai, Q. Plug-and-Play Algorithms for Large-Scale Snapshot Compressive Imaging. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, 13–19 June 2020; pp. 1444–1454, doi:10.1109/CVPR42600.2020.00152.

[II.51] Cheng, Z.; Lu, R.; Wang, Z.; Zhang, H.; Chen, B.; Meng, Z.; Yuan, X. BIRNAT: Bidirectional recurrent neural networks with adversarial training for video snapshot compressive imaging. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Glasgow, UK, 23–28 August 2020.

[II.52] Yuan, X.; Pu, Y. Parallel lensless compressive imaging via deep convolutional neural networks. *Opt. Express* 2018, 26, 1962–1977.

[II.53] Ma, J.; Liu, X.; Shou, Z.; Yuan, X. Deep tensor admm-net for snapshot compressive imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision (ICCV)*, Seoul, Korea, 27 October–2 November 2019.

[II.54] Iliadis, M.; Spinoulas, L.; Katsaggelos, A.K. Deepbinarymask: Learning a binary mask for video compressive sensing. *Digit. Signal Process.* 2020, 96, 102591.

[II.55] He, K.; Zhang, X.; Ren, S.; J.; S. Deep residual learning for image recognition. In *Proceedings of the CVPR*, Las Vegas, NV, USA, 27–30 June 2016.

[II.56] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; pp. 5998–6008.

[II.57] Cheng, Z.; Chen, B.; Liu, G.; Zhang, H.; Lu, R.; Wang, Z.; Yuan, X. Memory-efficient network for large-scale video compressive sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 20–25 June 2021.

[II.58] Wang, Z.; Zhang, H.; Cheng, Z.; Chen, B.; Yuan, X. Metasci: Scalable and adaptive reconstruction for video compressive sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 20–25 June 2021.

[II.59] Yang, C.; Zhang, S.; Yuan, X. Ensemble learning priors unfolding for scalable Snapshot Compressive Sensing. *arXiv* 2022, arXiv:2201.10419.

[II.60] Wu, Z.; Yang, C.; Su, X.; Yuan, X. Adaptive Deep PnP Algorithm for Video Snapshot Compressive Imaging. *arXiv* 2022, arXiv:2201.05483.

[II.61] Zhao, Y.; Zheng, S.; Yuan, X. Deep Equilibrium Models for Video Snapshot Compressive Imaging. arXiv 2022, *arXiv*:2201.06931.

[II.62] Pont-Tuset, J.; Perazzi, F.; Caelles, S.; Arbelaez, P.; Sorkine-Hornung, A.; Gool, L.V. The 2017 DAVIS challenge on video object segmenta tion. *arXiv* 2017, arXiv:1704.00675.

[II.63] Yuan, X.; Liu, Y.; Suo, J.; Dur,; F.; Dai, Q. Plug-and-play algorithms for video snapshot compressive imaging. *arXiv* 2021, arXiv:2101.04822

Table of contents

Chapter III. Video Compressive Sensing based on a novel video prediction framework

Chapter III. Video Compressive Sensing based on a novel video prediction framework

III.1. Introduction

Recently, many video compression techniques based on prediction frameworks have been proposed to enhance their performances. These frameworks are proposed to deal with the several challenges faced by almost all traditional video codecs (e.g.H.264) including the large computational cost and the huge memory resources needed to store dense matrices. Therefore, this idea of prediction may be extended to join the compressive sensing theory in order to optimize the computational resources of the transmission devices.

In this chapter, we noticed that video prediction is a promising research direction and many innovations could be done. Indded, we proposed “Robust Spatiotemporal Convolutional Long Short-Term Memory” (Robust-ST-ConvLSTM) algorithm as a novel algorithm for video prediction. It presents a new internal mechanism that is able to regulate efficiently the flow of spatiotemporal information from video signals based on higher order Convolutional-LSTM. The remaining chapter is organized as follows: Section III.2 discusses related works in video prediction. In Section III.3, we describe the main idea behind our proposed algorithm and its key components. In Section III.4.1.1, we evaluate the capability of Robust-ST-ConvLSTM for multi-step video prediction on two spatiotemporal datasets, including a synthetic dataset of handwritten digits and a human motion dataset and report its performance by comparing it against the state-of-the-art algorithms. In Section III.5, we discuss the potential perspectives to integrate this work in a VCS context. Finally, Section III.6 provides conclusion and the future research directions.

III.2. Related Works

Video prediction or predicting what happens in the next frames is the key component of intelligent decision-making systems. It is also, an emerging field of computer vision and deep learning that is facing many challenges [III.1]-[III.2]-[III.3][III.4]-[III.5]-[III.6]. Actually, these predictive systems have many real-world applications such as video surveillance or human and buildings security which is one of the most frequently debated issues nowadays.

Video prediction networks are based on historical information gathered from continuous and unlabeled video frames. These networks aim to forecast future frames in a video after having some previous images. Formally, we suppose $\boldsymbol{X_t} \in \mathbb{R}^{w \times h \times c}$ is the t-th frame of a dynamic scene $\boldsymbol{X} = (\boldsymbol{X_{t-n}}, \ldots, \boldsymbol{X_t})$ with $\boldsymbol{n}$ frames, where $\boldsymbol{w}$, $\boldsymbol{h}$, and $\boldsymbol{c}$ denote width, height and number of channels, respectively. The main target from this project is to predict the next m frames $\boldsymbol{Y} = (\boldsymbol{Y_{t+1}}, \ldots, \boldsymbol{Y_{t+m}})$ from the input $\boldsymbol{Y}$.

III.2.1. Optical flow-based methods

Many research projects have proposed video prediction solutions based on optical flow or dense trajectory [III.7]-[III.8]-[III.9]-[III.10]. In fact, optical flow is applied to report motion information about objects of successive frames. Technically, these approaches take the given dynamic scene as input to forecast the optical flow of the future frame. The obtained result is then merged with the last input frame to generate the future predicted video frame. However, those approaches that necessitate supervised training, use training datasets that contain optical flow information which is not obviously provided in the commonly used video datasets.

III.2.2. Deep Learning based methods

While the optical flow-based models use the motion information to predict the frames, neural network approaches analyze the frames and extract their features in order to exploit the spatiotemporal representation to forecast the next frames. In this section, recent deep learning models for video prediction will be discussed after being classified into three categories: recurrent neural networks, convolutional networks and generative networks.

Although these neural networks-based methods are better than the traditional optical flow-based solutions in terms of performances, they are challenging and produce sometimes blurry results. Obviously, it is a promising research area.

III.2.2.1. Recurrent models

Recurrent networks are commonly used for video sequences related problems since they are considered as sequential data with spatio-temporal representation.

Recurrent neural networks (RNN) have demonstrated considerable success in video prediction research works that are detailed in [III.11]-[III.12]-[III.13]-[III.14]-[III.15]-[III.16]-[III.17]-[III.18]-[III.19]-[III.20]-[III.21]-[III.22]-[III.23]-[III.24]-[III.25]-[III.26]-[III.27]. In fact, along with the advancements in neural networks architectures, video prediction has been studied extensively in recent years. Zhang et al. proposed a ConvLSTM-based architecture where hidden states are updated along a z-order curve [III.16]. The model presents a novel training approach based on two Z-Order Recurrent Networks (Znet): Znet-Predictor and Znet-Probe. Since most video prediction algorithms based on ConvLSTM have duplicated features with same functionality in both cell state and hidden state of the LSTM unit, Znet came up with a novel route updating to enhance the hidden states. Technically, to trick the neural network, the model is set to choose inputs that minimize the loss function instead of updating weights and biases that minimize the cost.

W. Lotter et al. [III.19] presented a predictive neural network (PredNet) architecture. This network aims to forecast future video frames in dynamic scenes. Technically, every layer in the network makes local predictions and only sends the deviations from those predictions to the following layers. The PredNet model is a series of recurrent blocks that make local predictions that are subtracted from the input before being forwarded to the subsequent network layer.

C. Lu et al. [III.22] propose a Flexible Spatio-Temporal Network (FSTN). This model enables the generation of the frames lying between the observed frames in order to output slow-motion video sequences. Also, it proposes a novel loss function to optimize the training phase of the model. The architecture described above is based on two main models: extrapolation model and interpolation model. Both of them are considered as spatio-temporal autoencoders. However, the extrapolation model has a guided training phase by the ground truth frames feeding each layer by the supervised information needed, while the interpolation model does not need the ground truth images. Another difference of the two models lies in their definition. The interpolation is the estimation of a value between given data points, but the extrapolation is useful when looking for a value that is either higher or lower than the values in the dataset.

A recent RNN architecture was proposed by Wang et al. in [III.28]. The idea behind this research work remains behind the new spatiotemporal LSTM (ST-LSTM) unit that takes out and memorizes spatial appearances and temporal variations simultaneously since for video prediction we need to consider both the spatial and the temporal structures. In fact, the Predictive Recurrent Neural Network (PredRNN) is based on spatiotemporal memory flow which

allows the memory cells to move vertically across stacked RNN layers and horizontally through all RNN states. This approach is different from stacked LSTM. Actually, in stacked LSTM, memory states are updated independently from the visual features which means that the first layer of the present time step could ignore the information memorized by the last layer at the previous time step. However, in PredRNN, a memory cell is introduced to handle the information between different time steps. Another problem is solved in [III.29]. The new memory cell can handle long-term and short-term information at the same time which can limit the predictive performances of the model. So, a pair of memory cells is used and explicitly decoupled in order to satisfy the different variations. This model reduces the loss of visual information from the very first layer to the top of the recurrent network. Furthermore, another learning strategy was proposed called reverse scheduled sampling. This strategy enables to learn temporal dynamics from longer periods of the input video and reduces the training discrepancy between the encoding network and the prediction network.

III.2.2.2. Convolutional models

Different from recurrent neural networks, convolutional networks are feed-forward neural networks that are commonly used for computer vision challenges such as visual prediction.

Many models are based on convolutions for video prediction. One of these architectures is a multi-model combining temporal and spatial sub-networks which is proposed in [III.30] and called MixPred. The future frame prediction approach described is divided into two parts: a temporal model for modeling the time series of the input video and a spatial sub-network to model the spatial texture on the content. Then, the authors tested an information fusion method for feature map interaction between the two parts. This approach allows to copy the unchanging pixels from the last frame thanks to the temporal mask which means that the predicted frame has the same clearness as the original frame. Also, synchronously exchanging temporal and spatial information enables to fill the changed pixels in order to have a complete predicted image. This model uses only convolutional layers, but it could be theoretically enhanced by using other models like the generative networks. The model described above could be used not only in future frames prediction but also in several applications such as object tracking, action recognition and video compression.

In [III.31], the model trains a deep neural network to generate video frames by flowing pixel values from existing ones instead of initializing them from scratch. The model, called Deep Voxel Flow (DVF), usually takes 3 frames from the video scene without pre-processing: two frames are taken as input and the third frame is used as the generated target. This approach is based on the idea of borrowing voxels (3D-pixels) from the adjacent frames to generate more realistic results. The architecture is composed of a convolutional encoder-decoder to forecast the voxel flow and a volume sampling layer to generate the target image.

As in [III.30], the model can predict the in-between frames (interpolation) and the future frames (extrapolation) of the input dynamic scene. The voxel flow, used to sample the input frames with the volume sampling function to synthesize the target frame, has two main components: the spatial component and the temporal one. The spatial element is the optical flow for the predicted frame and the temporal part is used to form a color in that frame.

The framework described above aims to predict one frame, but it can naturally be extended to a multi-frame prediction framework with a fairly simple manipulation. In fact, the target becomes a 3D volume and not 2D image and the learning rate will be reduced to maintain stability in the training phase. In addition, the spatiotemporal coherence is maintained because of the

preservation of local correlations due to the convolutions across the temporal layers. The strength of this model is that it combines the advantages of the optical-flow-based approach and the newer neural network-based models. Also, it can be trained and tested on any real-world video with any resolution. However, it fails in scenes with repetitive patterns. Also, it generates some blurry scenes, like most of neural network-based implementations.

III.2.2.3. Generative models

Generative models are used to generate new samples from the same distribution as the input data. The target behind training generative models is to learn a probability distribution that is similar to the data's probability distribution. In video prediction, the models described above aim to output a single eventual outcome. However, generative approaches generate a wide spectrum of feasible predictions.

The most common network structure in the field of video prediction and image generation in general is Generative Adversarial Networks (GAN). These networks are composed of two sub-networks jointly trained, the discriminator and the generator, to create fake samples that look like real data. Technically, the generator fools the discriminator by generating new samples from a random noise (e.g. Gaussian noise). Then the discriminator features the probability distribution function that describes real data. Nevertheless, in video prediction, some conditions could be added to the general implementation of GAN in order to forecast the future frames.

In [III.32], a generative approach was proposed to predict frames based on cycle GAN. The main model is composed of one generator and two discriminators. In fact, the generator uses the retrospective cycle to predict both future and past frames and we train it with reversed input sequences. Moreover, one discriminator is dedicated to identify fake frames while the other is implemented to distinguish the sequences that contain fake frames which is crucial in forecasting temporally consistent frames. Technically, the loss function and the network architecture make this approach special when we compare it with the general formulation of GAN networks. Since this model enables to predict a limited number of frames before generating blurry images, a multi-frame prediction strategy is employed. The model starts by forecasting the next frame from an input video. Then, it constructs a new input video by concatenating the last frames of the input video and the predicted frame. Finally, the new input video will enable the prediction of the next frame. This strategy is repeated until we get the desired number of predicted frames.

In [III.33], the authors insisted on the fact that conditional Generative Adversarial Networks (cGAN) are suitable for video frames prediction because it can guarantee the spatio-temporal coherence between the predicted frames and the input video. Another approach is discussed in [III.34] and is based on the idea of dividing the video signal into two parts: content and motion. Content to specify the objects in the sequence and motion to describe their movements. The model is based on mapping a sequence of random vectors to a sequence of frames in order to generate the predicted videos. These random vectors are composed of two parts: one for the content and the other for the motion. Since this framework is based on GAN, discriminators are used to learn motion and content decomposition in an unsupervised way by introducing a new adversarial learning scheme.

III.3. Overview of the proposed Robust Spatiotemporal ConvLSTM algorithm

To understand the idea behind Robust Spatiotemporal ConvLSTM algorithm, it is obvious to present the main inspiring recurrent architectures, i.e., LSTM and ConvLSTM.

III.3.1. From LSTM to ConvLSTM

The idea behind the proposed algorithm is based on Convolutional LSTM (ConvLSTM) which is Long Short-Term Memory (LSTM) network applied on high dimensional data.

III.3.1.1. LSTM

Long short-Term Memory Network is considered as an advanced type of RNN that was designed and developed by Hochhreiter and Schmidhhuber (1997) [III.35] to solve the vanishing gradient problem of standard RNNs. Theoretically, RNNs are designed to learn long term dependencies. However, in practice, many issues appear such as vanishing gradient that prevents those neural networks to learn long term dependencies. Therefore, it has been proven that LSTM is a powerful tool to remember information for longer period of time. Indeed, the main idea behind LSTM consists of connecting the previous information to the future task.

The main structure of LSTM based neural networks is the same: it consists of a chain of LSTM modules. However, the structure of those modules depends on the application.

One of the most powerful components of LSTM is the cell state which is represented by the horizontal line on the top of Figure III.1. It is used to handle the main information through the whole network and from one LSTM block to another. This function is controlled by 3 different structures: the forget gate, the input gate, and the output gate.

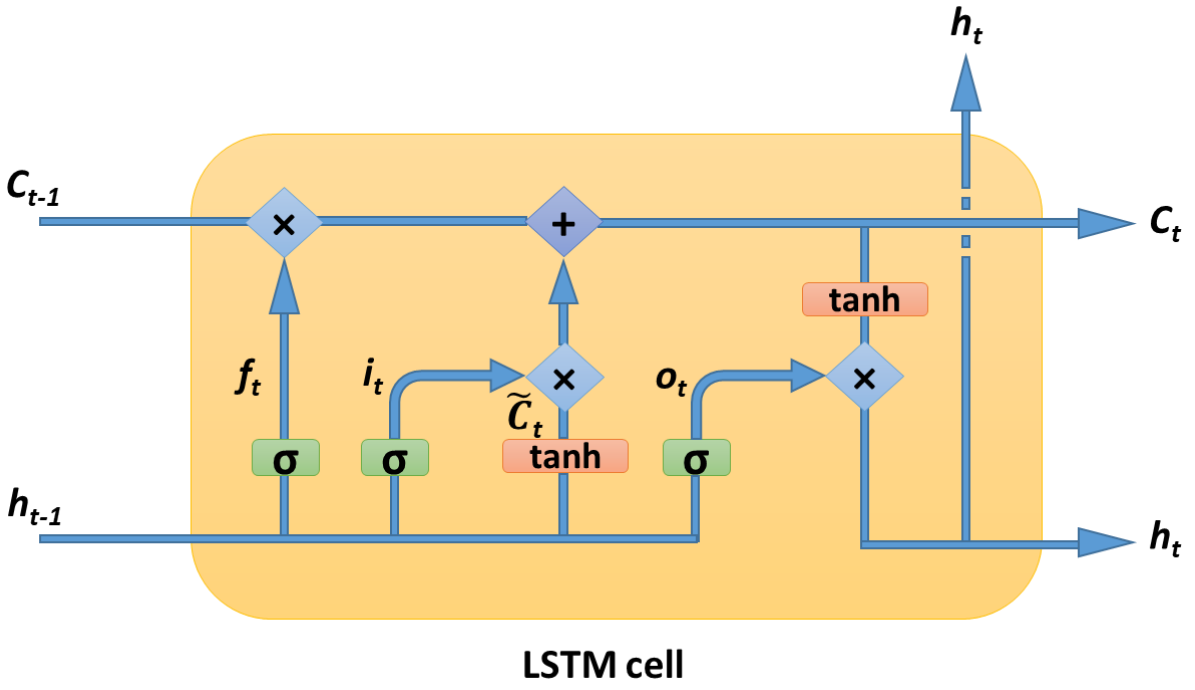


Figure III.1: The structure of a standard LSTM module.

When we look at the LSTM cell in Figure III.1, we notice that the information coming from the cell state $\boldsymbol{c_{t-1}}$ which passes through the forget gate that decide which information is going to be forgotten thanks to the sigmoid layer that outputs a number between 0 and 1. Then, the input gate uses the input $\boldsymbol{x_t}$ and the hidden state $\boldsymbol{h_{t-1}}$ to update the cell state. Then, a *tanh* layer outputs new candidate values $\hat{\boldsymbol{c}}_t$ that have the possibility to be added to the cell state. The cell state update is created from the combination of $\hat{\boldsymbol{c}}_t$ and $\boldsymbol{i_t}$.

Now, everything is ready to update the cell state. Firstly, the old cell state $\boldsymbol{c_{t-1}}$ is multiplied by $\boldsymbol{f_t}$ then $\boldsymbol{i_t} * \hat{\boldsymbol{c}}_{\boldsymbol{t}}$ is added. Finally, the output of the LSTM unit will be based on the cell state $\boldsymbol{c_t}$, the input $\boldsymbol{x_t}$ and the hidden state $\boldsymbol{h_{t-1}}$. Indeed, a sigmoid gate is applied to decide the parts of the cell state that will be involved in the output process. Then, the cell state $\boldsymbol{c_t}$ is put through tanh and then multiplied by the output of the sigmoid layer. The main target of this last step is to output the new hidden state $\boldsymbol{h_t}$. To sum up the mechanism of LSTM: This neural network unit has 3 inputs: the input $\boldsymbol{x_t}$, the cell state $\boldsymbol{c_{t-1}}$ and the hidden state $\boldsymbol{h_{t-1}}$ that will be passed through 3 different gates in order to output 2 structures: the cell state $\boldsymbol{c_t}$ and the new hidden state $\boldsymbol{h_t}$. The mechanism described above is explained by the following equations (III.1):

$$\begin{aligned}
\boldsymbol{i_t} &= \sigma(\boldsymbol{w_i} \times \boldsymbol{x_t} \times \boldsymbol{s_i} \times \boldsymbol{h_{t-1}}),\\
\boldsymbol{f_t} &= \sigma\big(\boldsymbol{w_f} \times \boldsymbol{x_t} \times \boldsymbol{s_f} \times \boldsymbol{h_{t-1}}\big),\\
\boldsymbol{o_t} &= \sigma(\boldsymbol{w_o} \times \boldsymbol{x_t} \times \boldsymbol{s_o} \times \boldsymbol{h_{t-1}}),\\
\hat{\boldsymbol{c}}_{\boldsymbol{t}} &= tanh(\boldsymbol{w}_{\hat{\boldsymbol{c}}} \times \boldsymbol{x_t} \times \boldsymbol{s}_{\hat{\boldsymbol{c}}} \times \boldsymbol{h_{t-1}}),\\
\boldsymbol{c_t} &= \boldsymbol{f_t} \circ \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \circ \hat{\boldsymbol{c}}_{\boldsymbol{t}},\\
\boldsymbol{h_t} &= \boldsymbol{o_t} \circ \tanh(\hat{\boldsymbol{c}}_{\boldsymbol{t}}),
\end{aligned} \tag{III.1}$$

where σ is the sigmoid function, $\times$ is a pointwise multiplication, $+$ is a pointwise addition and $\circ$ denotes the Hadamard product.

III.3.1.2. ConvLSTM

Although LSTM is considered as a powerful network for dealing with temporal relationship, its main drawback is that it is unable to handle spatial information because we need to flatten high dimensional data to 1D vectors to be compatible to the input common structure. However, Spatiotemporal data are commonly used in many applications such as video surveillance. So, we were forced to look for a new structure where we take advantage of LSTM by integrating spatiotemporal data.

Convolutional LSTM (Figure III.2) is used to capture the spatial dimension for the prediction mode. The special feature of ConvLSTM is that the inputs $\boldsymbol{x_t}$, the cell states $\boldsymbol{c_t}$, the hidden states $\boldsymbol{h_t}$ and the 3 gates are 3D tensors. In addition, the convolution operation is used instead of simple matrix multiplication as shown in the following equations (III.2):

$$\begin{aligned}
\boldsymbol{I_t} &= \sigma(\boldsymbol{W_i} \odot \boldsymbol{X_t} + \boldsymbol{S_i} \odot \boldsymbol{H_{t-1}}),\\
\boldsymbol{F_t} &= \sigma(\boldsymbol{W_f} \odot \boldsymbol{X_t} + \boldsymbol{S_f} \odot \boldsymbol{H_{t-1}}),\\
\boldsymbol{O_t} &= \sigma(\boldsymbol{W_o} \odot \boldsymbol{X_t} + \boldsymbol{S_o} \odot \boldsymbol{H_{t-1}}),\\
\widehat{\boldsymbol{C}}_{\boldsymbol{t}} &= tanh(\boldsymbol{W}_{\hat{\boldsymbol{c}}} \odot \boldsymbol{X_t} + \boldsymbol{S}_{\hat{\boldsymbol{c}}} \odot \boldsymbol{H_{t-1}}),\\
\boldsymbol{C_t} &= \boldsymbol{F_t} \circ \boldsymbol{C_{t-1}} + \boldsymbol{I_t} \circ \widehat{\boldsymbol{C}}_{\boldsymbol{t}},\\
\boldsymbol{H_t} &= \boldsymbol{O_t} \circ \tanh\big(\widehat{\boldsymbol{C}}_{\boldsymbol{t}}\big),
\end{aligned} \tag{III.2}$$

where $\odot$ denotes the convolution operation.

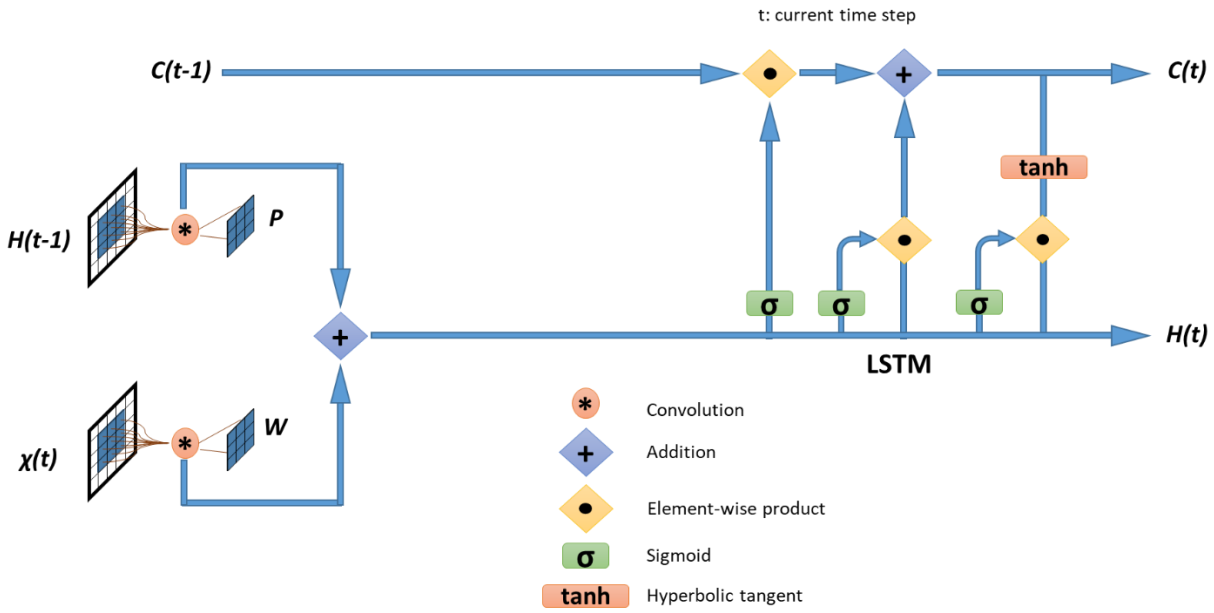


Figure III.2: The structure of convolutional LSTM.

III.3.2. Main contributions in the video prediction context

The great progress made by RNN architectures in a wide range of applications and research fields, has motivated us to explore some recent approaches to predict future video frames. The main advantage of these models is their potential to learn adequate features from high-dimensional data, such as videos, in an end-to-end manner without hand-designed features. However, despite the significant progress in deep learning architectures, video prediction is still considered as a big challenge especially in terms of output visual quality and long-term prediction. Therefore, our Robust Spatiotemporal Convolutional Long Short-Term Memory (Robust-ST-ConvLSTM) algorithm is proposed as a long-term prediction algorithm that outperforms the state-of-the-art approaches in terms of quality performances. Our algorithm is based on a modified version of ConvLSTM cell. Obviously, ConvLSTM is not very efficient in handling long sequences. Indeed, ConvLSTM based algorithms focus on stochastic features of the data rather than its spatial distortion. Also, a temporal information encoding in ConvLSTM unit [III.12] is based on 1st-order Markovian architecture. Thus, making long-range temporal correlations hard to extract. In addition, the vanishing gradient problem often occurs in training 1st-order RNN based predictive algorithms [III.36].

Bearing all these drawbacks in mind, we propose our Robust-ST-ConvLSTM algorithm for video prediction. With the following properties, we hope our algorithm will pave the way for the application of recurrent neural network on real-world datasets:

- Spatial and temporal data are taken into consideration jointly.
- The new spatiotemporal memory (STM) cell transfers low-level and semantic aspects of the dynamic scene which are the key of generating future frames.
- The Robust-ST-ConvLSTM new internal mechanism offers new cell state and hidden state transition functions to efficiently regulate the flow of spatiotemporal information from the input videos.
- The algorithm aims to rely on $\boldsymbol{N}$ previous hidden states, that provide temporal context for the motion in video scenes, to update one cell state at every timestep.

III.3.3. Robust Spatiotemporal ConvLSTM proposed algorithm

The proposed Robust Spatiotemporal ConvLSTM (Robust-ST-ConvLSTM) algorithm is a memory flow algorithm based on higher order ConvLSTM. To make it simple, the novel algorithm aims to decide the cell state $\boldsymbol{C_t}$ not only from the previous hidden state $\boldsymbol{H_{t-1}}$ but also from $\boldsymbol{N}$ previous hidden states ($\boldsymbol{H_{t-2}}, \dots, \boldsymbol{H_{t-N}}$) ($\boldsymbol{N}$ will be fixed by the user and it can only affect the computational time). The second part of the algorithm is to implement a memory flow to hold spatiotemporal information to optimize and control the prediction capacities of ConvLSTM. In fact, the memory flow will be a second cell state for spatiotemporal data. However, the cell state will not be removed and will handle temporal data.

Indeed, the novel algorithm uses a stack of ConvLSTM units to learn the spatial correlations and the temporal dynamics from the input video. These features will be used later to forecast the future frames. So, a novel transition function is introduced based on spatiotemporal memory flow and is able to leverage a deterministic number of previous hidden states. In the original implementation of ConvLSTM, the temporal memory states $\boldsymbol{C_t}$ are updated only from one-time step to another. However, in video prediction, the consecutive frames are having close data distributions in the spatial dimensions and many temporal correlations. Thus, we need to exploit these properties to make better predictions in terms of quality performances. Therefore, we believe that this higher order ConvLSTM based on memory flow will take advantage from the global motion changes of the consecutive frames and the information of the spatiotemporal memory to predict future frames. The memory state update process for the original stacked ConvLSTM model can be represented graphically with a horizontal diagram flow. We propose here to enhance this previous model by updating the memory state horizontally (the cell state) and vertically (the spatiotemporal memory state) as illustrated in Figure III.3. This approach ameliorates the way we handle the spatiotemporal information from the input to the output and enables to connect all the recurrent units of the entire network.

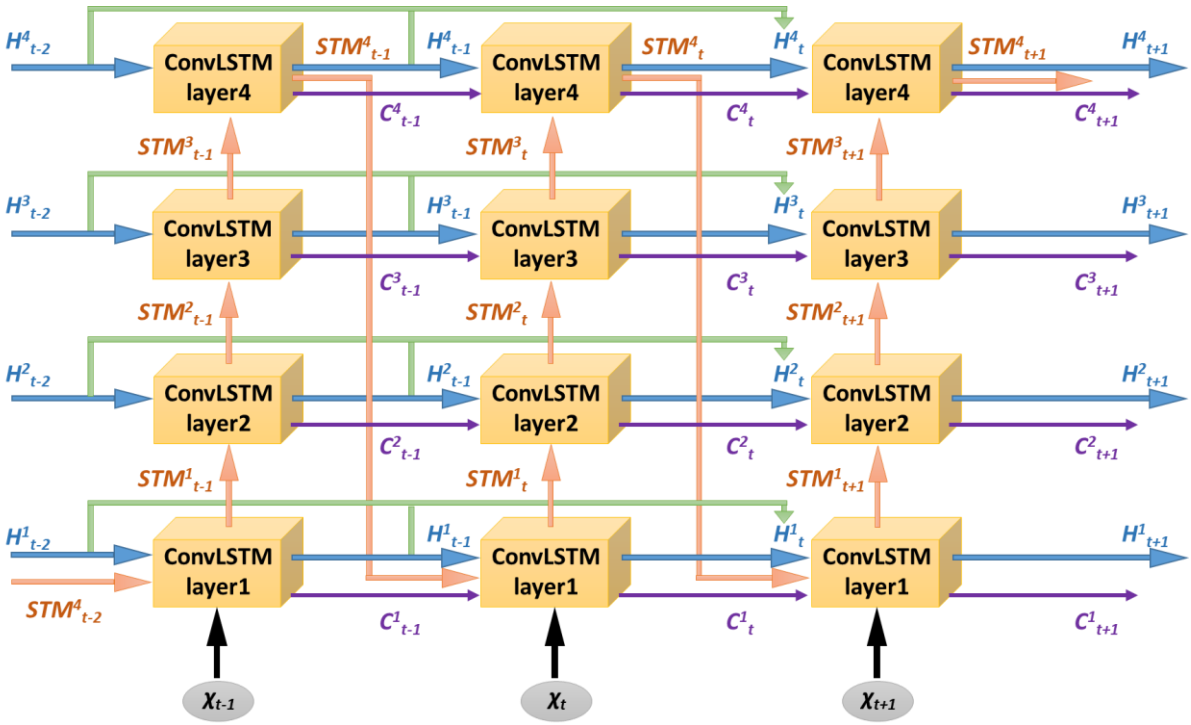


Figure III.3: The main structure of Robust Spatiotemporal LSTM.

The main equations of the new robust spatiotemporal unit represented in Figure III.4 are (III.3):

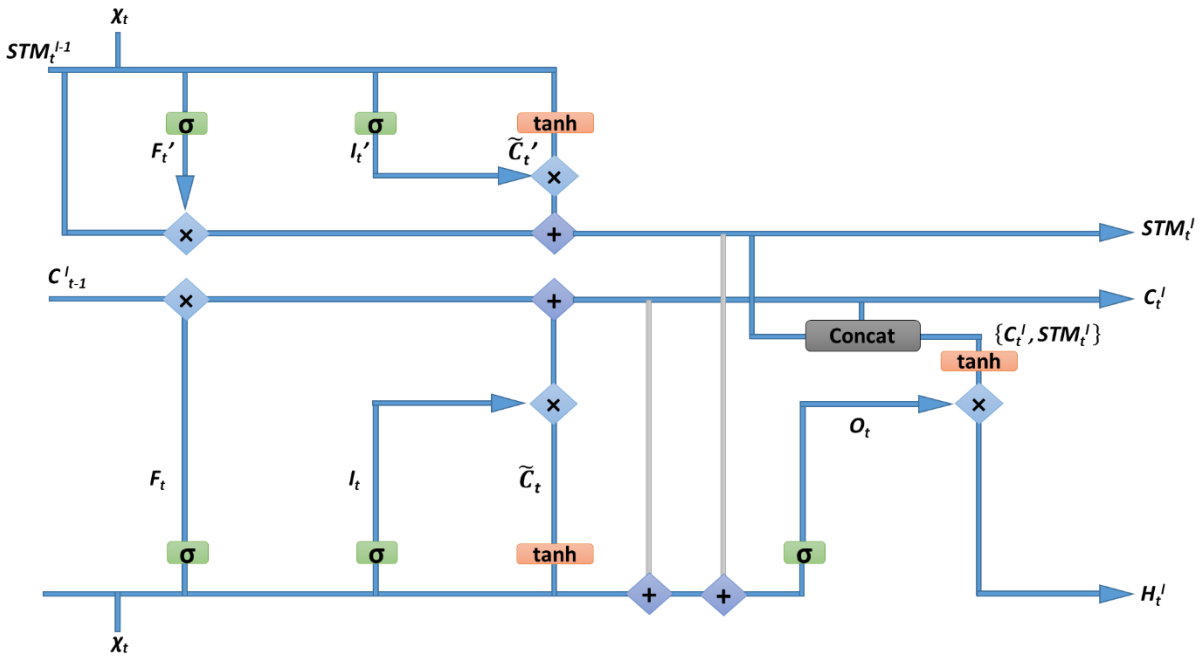


Figure III.4: Robust Spatiotemporal Unit.

$$
\begin{aligned}
\boldsymbol{I_t} &= \sigma\left(\boldsymbol{W_i} \odot \boldsymbol{X_t} + f\left(\boldsymbol{H_{t-1}^l}, \dots, \boldsymbol{H_{t-N}^l}\right)\right), \\
\boldsymbol{F_t} &= \sigma\left(\boldsymbol{W_f} \odot \boldsymbol{X_t} + f\left(\boldsymbol{H_{t-1}^l}, \dots, \boldsymbol{H_{t-N}^l}\right)\right), \\
\widehat{\boldsymbol{C}}_t &= \tanh\left(\boldsymbol{W_{\hat{c}}} \odot \boldsymbol{X_t} + f\left(\boldsymbol{H_{t-1}^l}, \dots, \boldsymbol{H_{t-N}^l}\right)\right), \\
\boldsymbol{C_t^l} &= \boldsymbol{F_t} \circ \boldsymbol{C_{t-1}^l} + \boldsymbol{I_t} \circ \widehat{\boldsymbol{C}}_t, \\
\boldsymbol{I_t'} &= \sigma\left(\boldsymbol{W_i'} \odot \boldsymbol{X_t} + \boldsymbol{M_i'} \odot \boldsymbol{STM_t^{l-1}}\right), \\
\boldsymbol{F_t'} &= \sigma\left(\boldsymbol{W_f'} \odot \boldsymbol{X_t} + \boldsymbol{M_f'} \odot \boldsymbol{STM_t^{l-1}}\right), \\
\widehat{\boldsymbol{C}}_t' &= \tanh\left(\boldsymbol{W_{\hat{C}}'} \odot \boldsymbol{X_t} + \boldsymbol{M_{\hat{C}}'} \odot \boldsymbol{STM_t^{l-1}}\right), \\
\boldsymbol{STM_t^l} &= \boldsymbol{F_t'} \circ \boldsymbol{STM_t^{l-1}} + \boldsymbol{I_t'} \circ \widehat{\boldsymbol{C}}_t', \\
\boldsymbol{O_t} &= \sigma\left(\boldsymbol{W_{ox}} \odot \boldsymbol{X_t} + f\left(\boldsymbol{H_{t-1}^l}, \dots, \boldsymbol{H_{t-N}^l}\right) + \boldsymbol{W_{oc}} \odot \boldsymbol{C_t^l} + \boldsymbol{W_{ostm}} \odot \boldsymbol{STM_t^l}\right), \\
\boldsymbol{C_t} &= \boldsymbol{F_t} \circ \boldsymbol{C_{t-1}} + \boldsymbol{I_t} \circ \widehat{\boldsymbol{C}}_t, \\
\boldsymbol{H_t^l} &= \boldsymbol{O_t} \odot \tanh\left(\boldsymbol{W_{1\times1}} \odot \left[\boldsymbol{C_t^l}, \boldsymbol{STM_t^l}\right]\right),
\end{aligned}
\tag{III.3}
$$

Where σ is the sigmoid activation function. Like the original ConvLSTM, $\boldsymbol{I_t}$ and $\boldsymbol{I_t'}$: the input gates, $\boldsymbol{F_t}$ and $\boldsymbol{F_t'}$: the forget gates, $\widehat{\boldsymbol{C}}_t$ and $\widehat{\boldsymbol{C}}_t'$: the potential candidates for the cell states, $\boldsymbol{O_t}$: the output gate. $\boldsymbol{X_t}$ denotes the input at the time step t. $\boldsymbol{H_t^l}$ denotes the hidden state of the l^{th} layer at the time step t. $\boldsymbol{C_t^l}$ is the memory state of the l^{th} layer at the time step t. $\boldsymbol{STM_t^l}$ denotes the spatiotemporal memory of the l^{th} layer at the time step t. f is the function that should be designed to combine $\boldsymbol{N}$ previous hidden states. The design of the f is quite difficult since it must satisfy the following conditions: the spatial structure of the hidden states must be preserved, the size of the filters that control the previous hidden states must increase with the time steps in

order to capture the context of these structures and finally, the algorithm's complexity must not explode.

To implement f we tested two main approaches. The first approach aims to return the mean value of all elements in the input tensor that handle the previous hidden states. In Robust-ST-ConvLSTM, the feedback signal is generated by combining multiple preceding hidden states. Therefore, the state of the $\boldsymbol{N}$-order Robust-ST-ConvLSTM is recursively updated with the following f function (1st approach)

(III.4):

$$f\left(\boldsymbol{H_{t-1}^{l}}, \dots, \boldsymbol{H_{t-N}^{l}}\right) = \frac{\mathbf{1}}{\boldsymbol{N}} \sum_{\boldsymbol{n=1}}^{\boldsymbol{N}} \boldsymbol{W_{hn}}\, \boldsymbol{H_{t-n}^{l}}. \tag{III.4}$$

Analogous to the filter structures used in signal processing, the second approach in designing the f function in inspired from recursive least squares filters [III.37]. It is now based on the weighted sum of the previous hidden states. Consequently, f is straightforward (2nd approach)

(III.5):

$$f\left(\boldsymbol{H_{t-1}^{l}}, \dots, \boldsymbol{H_{t-N}^{l}}\right) = \frac{\mathbf{1}}{\boldsymbol{N}} \sum_{\boldsymbol{n=1}}^{\boldsymbol{N}} \boldsymbol{\alpha_{n}^{n}} \boldsymbol{W_{hn}}\, \boldsymbol{H_{t-n}^{l}}, \tag{III.5}$$

where $\boldsymbol{\alpha}$ is the forgetting factor. The parameter $\boldsymbol{\alpha}$ $(0 < \boldsymbol{\alpha} < 1)$ gives more weight to recent hidden states.

The gates of the Robust Spatiotemporal unit are no longer dependent on the the hidden state and the temporal memory state from the previous time step of the same layer. However, they depend on the previous hidden states from previous time steps at the same layer and the spatiotemporal memory state. To be clear, the first layer in a stacked ConvLSTM model at time step t receives the spatiotemporal memory of the last layer in the stacked model of the previous time step as illustrated in Figure III.3 ($\boldsymbol{STM_{t}^{l}} = \boldsymbol{STM_{t-1}^{L}}$ with $\boldsymbol{L}$ is the number of stacked layers). So, we adopt the original structure of ConvLSTM, and we added a second gated structure for the spatiotemporal memory $\boldsymbol{STM_{t}^{l}}$. However, the final hidden state $\boldsymbol{H_{t}^{l}}$ depends on the combination of the temporal memory state $\boldsymbol{C_{t}^{l}}$ and the spatiotemporal memory state $\boldsymbol{STM_{t}^{l}}$.

The spatiotemporal memory parameter is dedicated to reduce the loss of spatiotemporal information in the video sequences from the first layer to the last layer of the network. Besides, the previous hidden states used as input for the ConvLSTM blocks are implemented to expand the visibility of the neural units about the context of the current events at different time steps.

It is clear that the proposed model increases the number of parameters when we compare it with the standard ConvLSTM but it will prevent as from unnecessarily expanding the ConvLSTM model to obtain the same performances.

III.4. Performance evaluation, comparison, and discussion

III.4.1. Datasets

As far as we are concerned, there are currently no datasets for video prediction because it is an emerging area of research. However, researchers basically use motion video datasets such as

KTH and MovingMNIST used to compare the performances of our proposed algorithm with the state-of-the-art approaches.

III.4.1.1. KTH

This dataset has 2391 video sequences of 6 human actions (Walking, Jogging, Running, Boxing, Hand waving, Hand clapping) performed by 25 people in 4 different scenarios. Static cameras were used to capture the video scenes with 25 fps as a frame rate. The sequences have a length of 4 seconds in average with a frame size of 160 × 120. The videos are stored in 600 video files for each combination of 25 subjects, 6 actions and 4 scenarios. To train different approaches, the original frames are resized to 128 × 128. Then, we followed the setup of [III.38], which uses persons 1 − 16 to train different algorithms and persons 17 − 25 for testing. Different models are trained to forecast 10 frames from 10 input frames. To evaluate the robustness of our algorithms compared to state-of-the-art approaches at test step, we widen the predictions abilities to 20 frames (timesteps). In addition, we trained different models to forecast 10 frames from only 5 observations to compare their quality performances on a limited number of input frames (Figure III.5).

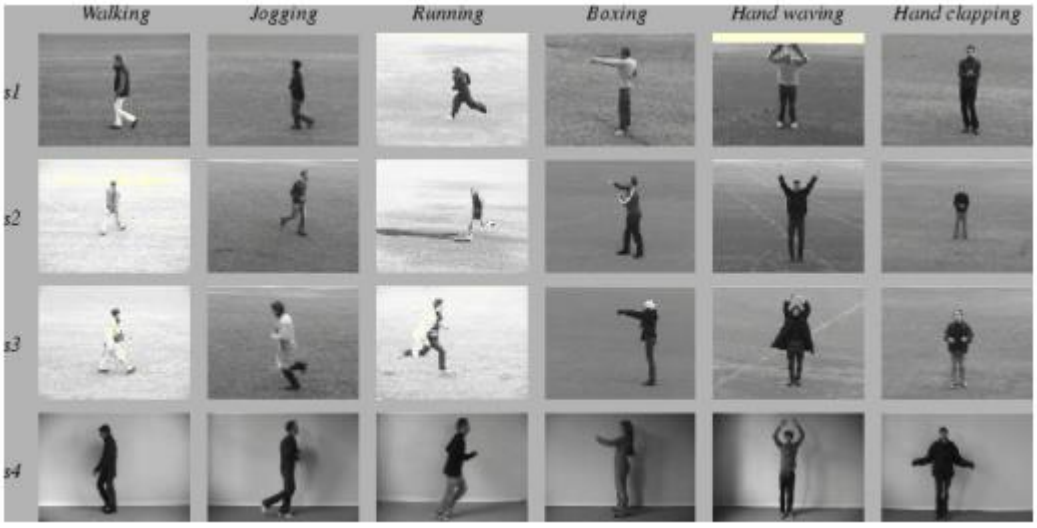


Figure III.5: KTH action dataset.

III.4.1.2. Moving MNIST

We followed the original setting of Moving MNIST dataset proposed for video representation purposes [III.18]. The idea of this dataset is to generate two moving digits, randomly placed in 64 ×64 grid, that move around with a constant velocity. To evaluate the performances of different algorithms, we generate 10000 sequences for the training process, 3000 for validation and 5000 for testing. The main performances are obtained by following the common setting in previous research works: generating 10 future frames after receiving the previous 5 and 10 observations.

III.4.2. Compared methods and performance metrics

III.4.2.1. Compared methods

To evaluate the performance of our proposed Robust-ST-ConvLSTM, we compare it with the performance of some advanced video prediction models:

- **ConvLSTM**: is commonly used for spatiotemporal predictive systems with a traditional roadway for the memory state. This algorithm is mentioned in almost every research work as the least efficient approach. However, it is the source of inspiration for video prediction algorithms based on recurrent neural networks.
- **PredRNN 2017**: based on the spatiotemporal LSTM (ST-LSTM) unit that take out and memorize spatial appearances and temporal variations simultaneously.
- **PredRNN 2021**: In this algorithm, a pair of memory cells is used and explicitly decoupled in order to enhance the performances of the previous algorithm and surpass its limitations. In addition, another learning strategy was proposed called reverse scheduled sampling.

The differences in results can be explained by the implementation details of the training process: we did not use the pretrained checkpoints to test the performances of the existing state of the art algorithms, but we actually re-trained ConvLSTM, PredRNNv-2017 and PredRNN-v2021 in the same conditions as our approach to achieve a fair and consistent comparison. Some hyperparameters are also adjusted to allow the comparison:

- The training process is stopped after 100.000 iterations (not 80.000 iterations in PredRNN).
- Mini-batch = 2 sequences (not 8 sequences in PredRNN because in this case the mini-batch of data does not fit onto our GPU memory).

III.4.2.2. Performance metrics

Because the results are video frames, we will use the most commonly used metrics to evaluate the quality of images between the ground truth and the prediction. Those metrics are Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM) [III.39] and Learned Perceptual Image Patch Similarity (LPIPS) [III.40].

- The PSNR measures, in decibels, the quality ratio between the original frame and the predicted one. The higher the PSNR, the better the quality of the predicted image.

  The PSNR is calculated by (III.6):

$$PSNR\left(\boldsymbol{Y},\widehat{\boldsymbol{Y}}\right)=10log_{10}\frac{max^{2}\widehat{\boldsymbol{Y}}}{\frac{\mathbf{1}}{\boldsymbol{N}}\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{N}}\left(\boldsymbol{Y}-\widehat{\boldsymbol{Y}}\right)^{\mathbf{2}}}, \tag{III.6}$$

  where $\boldsymbol{Y}$ is the ground truth, $\widehat{\boldsymbol{Y}}$ is the generated prediction, $\boldsymbol{N}$ is the number of pixels and $max\widehat{\boldsymbol{Y}}$ is the maximum value of the frame intensities.

- The SSIM measures the similarity between two images in terms of luminance, contrast, and structure. It is calculated as follows (III.7):

$$SSIM\left(\boldsymbol{Y},\widehat{\boldsymbol{Y}}\right)=\frac{(2\boldsymbol{\mu_{Y}}\boldsymbol{\mu_{\widehat{Y}}}+\boldsymbol{C_{1}})+(2\boldsymbol{\sigma_{Y\widehat{Y}}}+\boldsymbol{C_{2}})}{\left(\boldsymbol{\mu_{Y}^{2}}+\boldsymbol{\mu_{\widehat{Y}}^{2}}+\boldsymbol{C_{1}}\right)\left(\boldsymbol{\sigma_{Y}^{2}}+\boldsymbol{\sigma_{\widehat{Y}}^{2}}+\boldsymbol{C_{2}}\right)}, \tag{III.7}$$

  where $\boldsymbol{\mu_{Y}}$ and $\boldsymbol{\mu_{\widehat{Y}}}$ are the average of $\boldsymbol{Y}$ and $\widehat{\boldsymbol{Y}}$, respectively, $\boldsymbol{\sigma_{Y}}$ and $\boldsymbol{\sigma_{\widehat{Y}}}$ are the variance of $\boldsymbol{Y}$ and $\widehat{\boldsymbol{Y}}$, respectively, $\boldsymbol{\sigma_{Y\widehat{Y}}}$ is the covariance of $\boldsymbol{Y}$ and $\widehat{\boldsymbol{Y}}$. $\boldsymbol{C_{1}}$ and $\boldsymbol{C_{2}}$ are constants. The higher the $SSIM$, the greater similarity between two images.

- The LPIPS metric is used to measure the distance between video frames patches. It evaluates the perceptual distance between ground truth patches and predicted patches to judge how similar they are in a way that agrees with human judgement. This perceptual metric is defined by using deep features. In our implementation, it is pre-trained on AlexNet [III.41] architecture. Lower LPIPS scores indicate better prediction results.

III.4.3. Implementation details

The proposed algorithm is implemented with Python 3.6 and Pytorch 1.4.0 as a deep learning framework. Pytorch is used because it offers an effective way to manipulate tensors or multi-dimensional matrices needed to store and process multi-dimensional data.

We use Adam optimizer to train our model which is an optimization algorithm that combines the properties of AdaGrad and RMSProp algorithms to provide an optimization algorithm that is faster than the commonly used Stochastic Gradient Descent (SGD) algorithm especially with sparse data. A mini-batch of 2 sequences is chosen at each training iteration and it is reduced to the maximum to handle the out of memory problem of our GPU. We choose a learning rate of 0.0001 and the training process is stopped after 100000 iterations. The main architecture of our proposed model is composed of 4 ConvLSTM layers for each time step as illustrated in Figure III.3. The number of hidden states used to update the cell state is limited (in our case = 3), we can increase it to enhance the performance of our algorithm. However, an additional computational cost will slow down the training process. So, a trade-off between the number of hidden states and computational complexity should be done. The entire training process was on an NVIDIA GeForce RTX 2060GPU, Intel(R) Core(TM) i7-9700K CPU (3.60 GHz), a 32GB device memory, and Windows 10 operating system.

III.4.4. Experimental results

The performances of our approach will be evaluated on KTH and Moving MNIST datasets.

III.4.4.1. On KTH dataset

Table III.1 presents quantitative results of the proposed algorithm and state-of-the-art networks and the corresponding frame-wise comparisons are shown in Figure III.6, Figure III.7 and Figure III.8. We adopt PSNR and SSIM as evaluation metrics. We can obviously confirm that our proposed algorithm shows significant improvements in terms of short-term and long-term forecasting over the commonly used ConvLSTM approach. In fact, it increases the average PSNR and SSIM over the same number of predicted frames by 26% and 21.31%, respectively, by comparing it with the algorithm mentioned above. Also, it performs favorably against the PredRNN-v2017 and the PredRNN-v2021 algorithms of Wang et al. Our Robust-ST-ConvLSTM (with α = 0.9) performs better than PredRNN-v2021 by 1.72% and 2.77% in terms of PSNR and SSIM, respectively. These empirical results demonstrate the effectiveness and the efficiency of the Robust Spatiotemporal Convolutional Long Short-Term Memory algorithm in predicting future frames. In concordance with PSNR and SSIM results, we clearly notice that our Robust-ST-ConvLSTM algorithm outperforms the state-of-the-art approaches in terms of LPIPS which prove that our algorithm is effectively able to predict high-fidelity video frames. In accordance with these results, Figure III.9 that compares representative generated frames, proves that our algorithm outperforms the state-of-the-art approaches in terms of future movement and frames details. Robust ST-ConvLSTM predicts more accurate motion trajectories into the future because of the memory flow component that strengthen the long-term prediction ability of the

ConvLSTM cell and also because of updating the ConvLSTM cell using information from some previous time steps.

We can notice also that the second approach in designing f which is inspired from recursive least squares filters slightly outperforms the first approach in terms of PSNR and SSIM. This means that further research work could be done in order to determine the optimal value of $\boldsymbol{\alpha}$ that gives the best PSNR and SSIM performances. In this work, various values of α have been tested randomly ($0 < \boldsymbol{\alpha} < 1$) and the optimum one among them was the selected value 0.9.

The presented results and the computational cost depend on the number of memory units used for feedback. In our implementation, we used 3 hidden states which means that we have 3rd order Robust-ST-ConvLSTM. Furthermore, the number of hidden states can affect the performances of our model in terms of the quality of its output and also in terms of the computational process. From the previous observations about the value of α and the number of hidden states, we can confirm that a trade-off should be done between quality performances and computational costs, in future research work, to have the best performances without training a computationally very expensive algorithm.

Table III.1: Quantitative evaluation of different algorithms on KTH dataset. The metrics are averaged over the 10 and 20 predicted frames based on 5 and 10 observations, respectively. Higher PSNR and SSIM scores and lower LPIPS scores indicate better prediction results.

	10 → 20			5 → 10		
Model	PSNR (dB)	SSIM	LPIPS	PSNR (dB)	SSIM	LPIPS
ConvLSTM (Shi et al., 2015)	23.009	0.704	0.237	23.300	0.712	0.178
PredRNN (Wang et al., 2017)	27.624	0.839	0.208	28.752	0.845	0.153
PredRNN (Wang et al., 2021)	28.502	0.831	0.143	28.622	0.860	0.152
Robust-ST-ConvLSTM (1st approach)	**28.828**	**0.848**	**0.124**	**28.785**	**0.880**	**0.110**
Robust-ST-ConvLSTM (2nd approach)	**28.992**	**0.854**	**0.122**	**28.905**	**0.892**	**0.106**

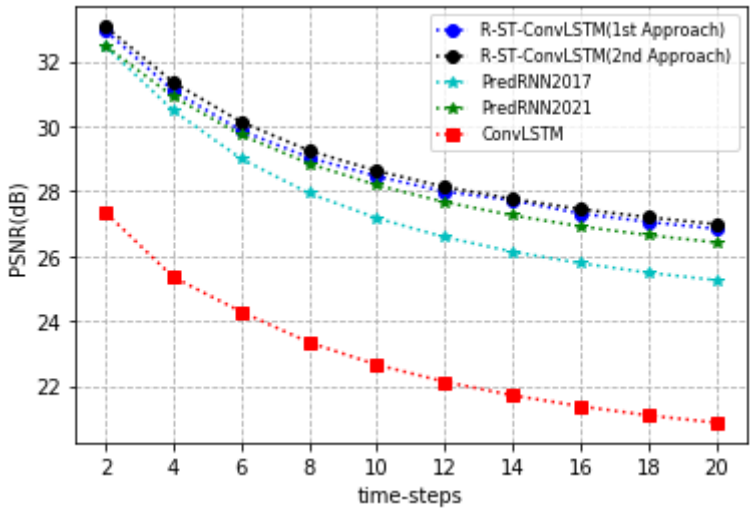


Figure III.6: Frame-wise PSNR comparisons of different models on KTH dataset after 100 000 iterations.

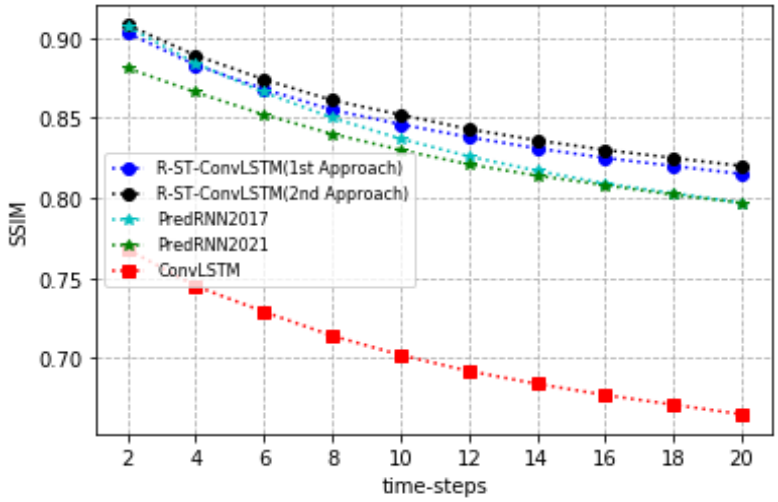


Figure III.7: Frame-wise SSIM comparisons of different models on KTH dataset after 100 000 iterations.

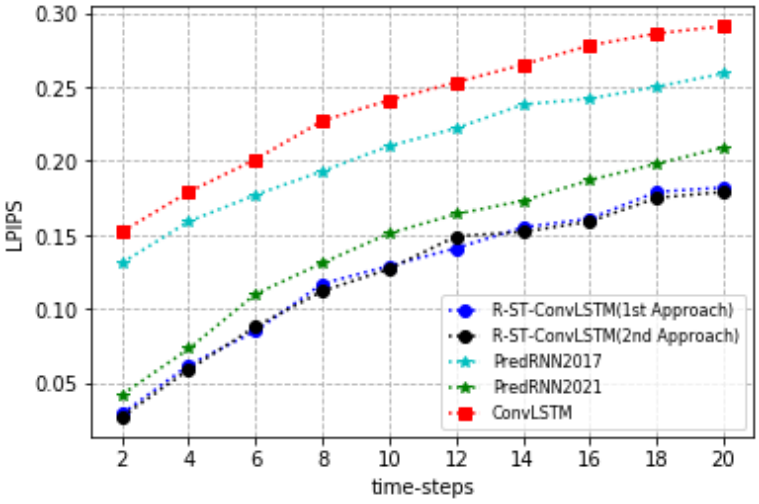


Figure III.8: Frame-wise LPIPS comparisons of different models on KTH dataset after 100 000 iterations.

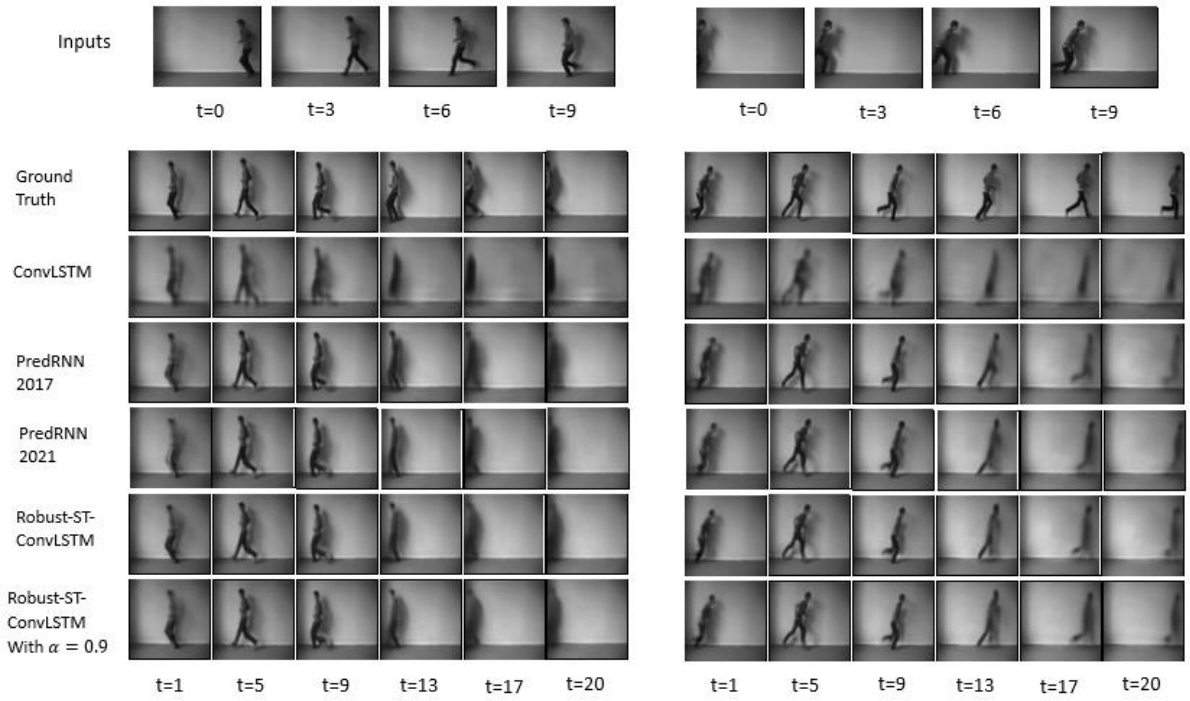


Figure III.9: Prediction examples on the KTH data set, where we predict 20 frames into the future based on the past 10 frames.

III.4.4.2. On Moving MNIST

Table III.2 presents the performance of the evaluated models on the Moving MNIST dataset by predicting the next 10 frames from the previous 10 input frames. We use the similarity index measure (SSIM) and the Peak signal-to-noise ratio (PSNR) for evaluation. As shown from Table III.2 and Figure III.10, Figure III.11 and Figure III.123, our architecture performs well against the state-of-the-art approaches in both metrics. Figure III.10 and Figure III.11 show the frame-wise PSNR and SSIM comparisons of different approaches on MNIST dataset. The results of these figures prove the ability of the Robust-ST-ConvLSTM in predicting future frames. Also, they prove that our approach outperforms the previous models on all the predicted frames. The memory flow algorithm based on 3rd order ConvLSTM with $\boldsymbol{\alpha}$ = 0.9 increases the average PSNR over the 10 predicted frames by 3.15% by comparing it with PredRNN (Wang et al., 2021). However, it outperforms the same approach by 0.22% in terms of SSIM. This means that the algorithms have similar performances on MNIST dataset. Moreover, our approach performs favorably against the traditional ConvLSTM approach in terms of PSNR and SSIM. It brings 14.59% PSNR improvement and 26.95% SSIM improvement over ConvLSTM based frames prediction approach. In concordance with the previous results, Figure III.12 shows that our approach brings also a remarkable improvement in terms of LPIPS. These results on Moving MNIST dataset prove that our algorithm generates more realistic predictions of future digits movements. These numerical results are confirmed by Figure III.13 that shows the quality of the 10 predicted frames generated by the different approaches. Robust-ST-ConvLSTM outputs clearer frames. However, the state-of-the-art algorithms produce blurry images. This means that Robust-ST-ConvLSTM is more precise and surer about the future variations which proves its robustness against the other long-term prediction algorithms mentioned above.

We can notice also that the recursive least squares filters-based approach in designing f has approximately similar results as the first approach and that for different values of $\boldsymbol{\alpha}$. Different

from KTH dataset, the value of the parameter α does not affect the quality performances of the outputs but it affects the computational cost of our algorithm since a number of multiplications are added to the calculation process. This means that, for MNIST dataset, only the first approach of designing f, which is based on returning the mean value of the previous hidden states, is taken into consideration.

Table III.2: Quantitative evaluation of different algorithms on Moving MNIST dataset. The metrics are averaged over the 10 predicted frames based on 5 and 10 observations. Higher PSNR and SSIM scores and lower LPIPS scores indicate better prediction results.

	10 → 10			5 → 10		
Model	PSNR (dB)	SSIM	LPIPS	PSNR (dB)	SSIM	LPIPS
ConvLSTM (Shi et al., 2015)	28.380	0.705	0.158	27.436	0.686	0.174
PredRNN (Wang et al., 2017)	30.569	0.869	0.108	29.786	0.806	0.125
PredRNN (Wang et al., 2021)	31.525	0.893	0.071	30.115	0.854	0.102
Robust-ST-ConvLSTM (1st approach)	**32.490**	**0.894**	**0.063**	**31.785**	**0.860**	**0.082**
Robust-ST-ConvLSTM (2nd approach)	**32.520**	**0.895**	**0.059**	**31.962**	**0.865**	**0.075**

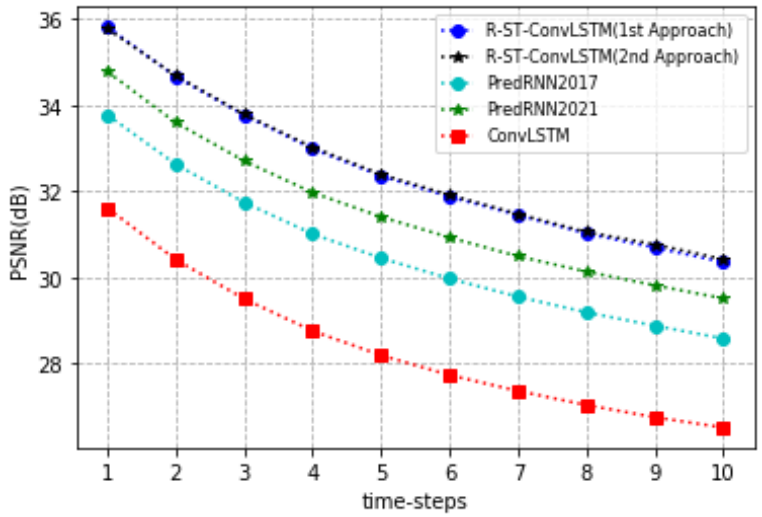


Figure III.10: Frame-wise PSNR comparisons of different models on Moving MNIST dataset after 100 000 iterations.

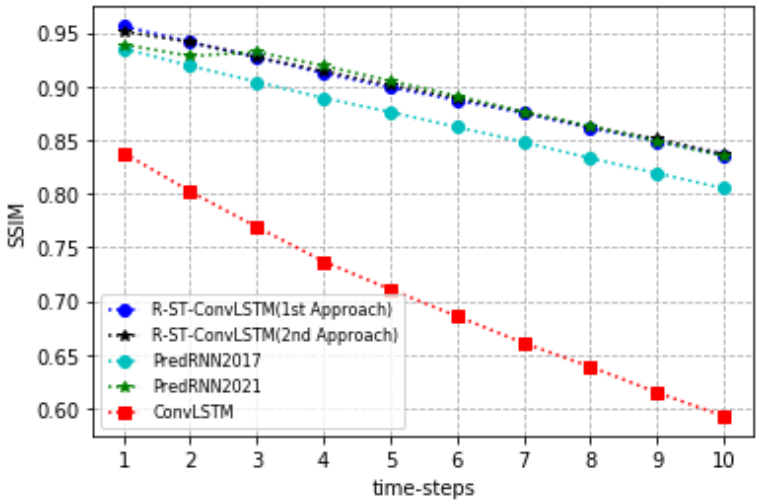


Figure III.11: Frame-wise SSIM comparisons of different models on Moving MNIST dataset after 100 000 iterations.

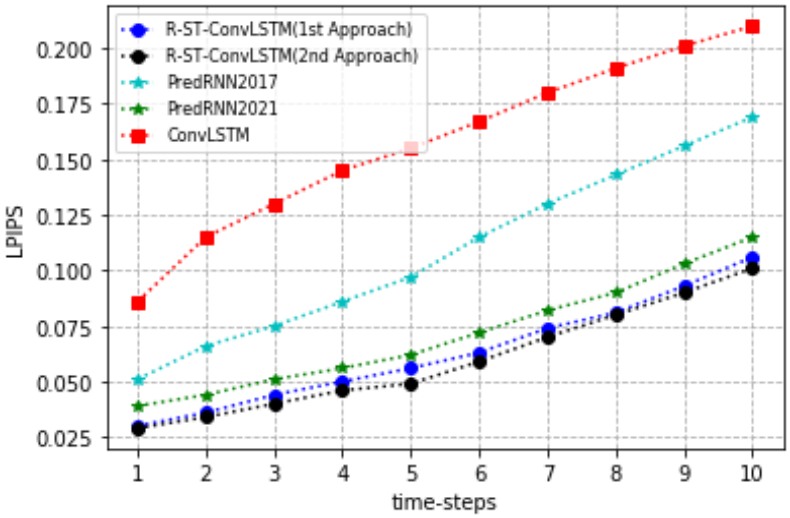


Figure III.12: Frame-wise LPIPS comparisons of different models on Moving MNIST dataset after 100 000 iterations.

Figure III.13: Prediction examples on the Moving MNIST dataset, where we predict 10 frames into the future based on the past 10 frames.

III.4.4.3. Experimental results on the number of predicted frames and the number of observations

When designing our prediction approach, one crucial choice is the length of the input video clip. However, the input video length directly impacts the complexity of the algorithm. Indeed, the number of the previous frames used to train the algorithm as well as the number of the number of frames to be predicted have to be fixed before the training. In this ablation study, we study the impact these variables on the quality performance of our algorithm. For evaluation, we choose $I = \{5, 10\}$ as sequence length under consideration on KTH and Moving MNIST datasets. However, we visualize the quality performances of our approach on the 10 predicted frames on Moving MNIST as shown in Figure III.10, Figure III.11 and Figure III.12, and on the 20 predicted frames on KTH as presented in Figure III.6, Figure III.7 and Figure III.8. In fact, Table III.1 and Table III.2 prove that increasing the number of input video frames can significantly improve the performances of the model. However, it increases its complexity which leads to longer training times. Figure III.10, Figure III.11 and Figure III.12 show that the performances of all models trained on Moving MNIST dataset decrease when predicting further frames which prove that designing very long-term predictions algorithms is still very challenging. Also, Figure III.6, Figure III.7 and Figure III.8 present the degradation of the quality performances of all models trained on KTH dataset over time-steps. However, on both datasets, our Robust-ST-ConvLSTM outperforms the state-of-the-art approaches on all quality metrics which prove the robustness and its quality precision against the other long-term prediction approaches.

III.4.4.4. Computational Complexity

Table III.3 shows that the computational complexity of our Robust-ST-ConvLSTM is more important than other state-of-the-art approaches. This result is due to the fact that we added new trainable parameters to the extended version of ConvLSTM cell to enhance its prediction abilities and handle spatiotemporal data. However, our RobustST-ConvLSTM slows down the training process which can be considered as a major limitation of our proposed algorithm.

Table III.3: Computational complexity comparison of the different approaches on Moving MNIST dataset as input.

Model	Number of training parameters ($\times 10^6$)
ConvLSTM (Shi et al., 2015)	16.60
PredRNN (Wang et al., 2017)	23.85
PredRNN (Wang et al., 2021)	23.86
Robust-ST-ConvLSTM	25.73

III.5. Discussion

It is obvious that video processing is facing several challenges including energy consumption in the transmission phase and limited storage resources. The resolution of these challenges is becoming urgent since video data represents today around 80% of the global internet traffic data [III.42] which require efficient systems to compress, acquire and transmit data. In this research work, we worked on a novel video prediction algorithm that enables to forecast video frames using relevant information from previous input frames. Indeed, the theory of conventional video codecs systems such as H256/HEVC [III.43] and motion-compensated reconstruction (MC-BCS-SP) [III.44] is significantly based on classical prediction to deal with complex coding issues. While achieving good compression performances, state-of-the-art prediction approaches used in this context are encountering various problems [III.45] especially when handling long sequences or in other words, long term prediction problems.

The idea behind this work was to design and implement a VCS framework based on an innovative prediction algorithm as presented in Figure III.14. This VCS framework was based on 3 main blocks:

- Key frame extraction module to select a group of the most informative frames of the video.
- Image CS module to acquire and transmit only few random measurements instead of the whole key frame.
- Next frame prediction to predict the non-key frames.

However, after implementing our novel video forecasting approach, we realized various problems in the designed VCS framework. Among these challenges, we had:

- The system should have the complete video sequence to extract the most relevant frames.
- Key frame extraction module should be run on sensing devices which increases their energy consumption. However, our main goal in this project is to optimize different resources usage including power consumption.

- Our algorithm is designed in a way to forecast a limited number of frames already defined by users. However, the number of non-key frames between two key frames is not constant and this information had to be transmitted with the random measurements of key frames. Also, in case we had a relatively high number of non-key frames, the quality performances would be affected and would affect the following frames.
- While our Robust-ST-ConvLSTM algorithm is based on some key features from previous frames to understand the context of the prediction, it is still considered as a limited approach because of the limited number of the previous hidden states exploited for the prediction process.
- Recently, an innovative architecture has revolutionized the computer vision field: The Transformers. In the last two years, many researchers and high-tech companies are working on exploring the advantages of this neural network in different fields. In fact, it is designed to process sequential data like RNN based models. Nevertheless, Transformers are faster than RNN based approaches because they do not process data in a sequential order, and they completely avoid recursion by processing a video as a whole to learn different relationships between all the patches using the well-known attention mechanism [III.46].

Bearing in mind the above considerations, we decided to follow the trend and explore the edges of transformers to build an end-to-end VCS framework. This work will be discussed in the next chapter.

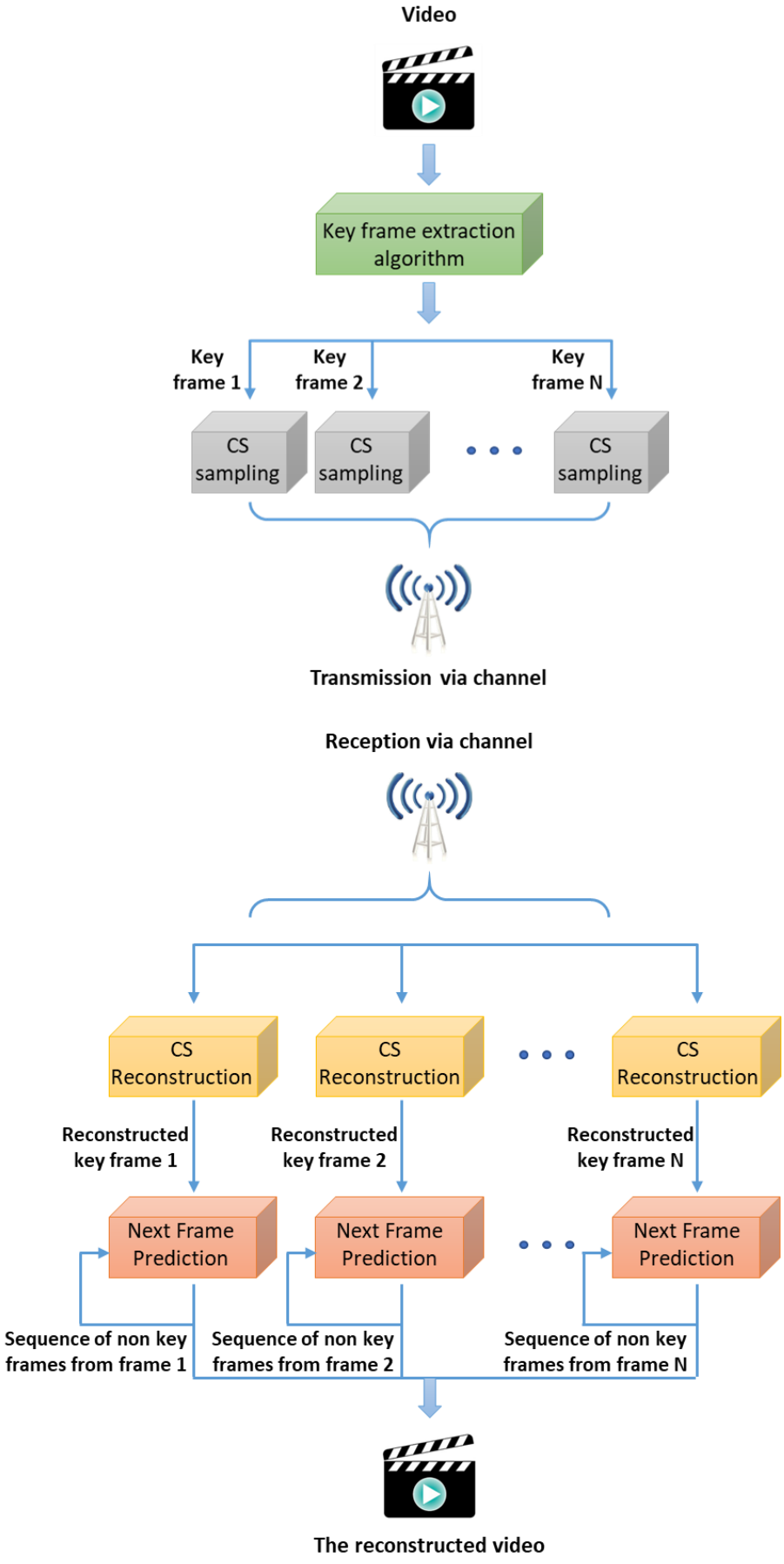


Figure III.14: VCS approach based on video prediction.

III.6. Conclusion

Video prediction is considered as a powerful tool to understand and model dynamic scenes. Therefore, in this work, we propose a new recurrent neural network (RobustST-ConvLSTM) for video prediction. It is based on new robust spatiotemporal unit inspired from the well-known ConvLSTM structure. This spatiotemporal unit rely on two different approaches in order to strengthen its prediction abilities: a memory flow to handle the spatiotemporal information and a higher order ConvLSTM approach that enable the cell states to decide their values from previous hidden states. Our approach outperforms the state-of-the-art research works on different datasets, including KTH dataset for human motion and Moving MNIST.

In conclusion, video prediction is a promising research direction and can be used in different applications such as video surveillance, video compression and intelligent decision-making systems. In our work, we noticed some limitations of our designed framework which prevent us from exploiting it in a VCS context as discussed in Section III.5. Therefore, we will propose, in the next chapter, the first end-to-end VCS framework built upon Transformers where we were able to explore this novel neural network in a video data compression context.

References

[III.1] Hoai, Minh, and Fernando De la Torre. “Max-margin early event detectors.” *International Journal of Computer Vision 107.2* (2014): 191-202.

[III.2] Kitani, Kris M., et al. “Activity forecasting.” *European conference on computer vision. Springer*, Berlin, Heidelberg, 2012.

[III.3] Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba. “Anticipating visual representations from unlabeled video.” *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[III.4] Zeng, Kuo-Hao, et al. “Visual forecasting by imitating dynamics in natural sequences.” Proceedings of the *IEEE International Conference on Computer Vision*. 2017.

[III.5] Bhattacharyya, Apratim, Mario Fritz, and Bernt Schiele. “Long-term on-board prediction of people in traffic scenes under uncertainty.” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[III.6] Hu, Anthony, et al. “Probabilistic future prediction for video scene understanding.” *European Conference on Computer Vision. Springer*, Cham, 2020.

[III.7] Walker, Jacob, Abhinav Gupta, and Martial Hebert. “Dense optical flow prediction from a static image.” *Proceedings of the IEEE International Conference on Computer Vision*. 2015.

[III.8] Walker, Jacob, et al. “An uncertain future: Forecasting from static images using variational autoencoders.” *European Conference on Computer Vision. Springer*, Cham, 2016.

[III.9] Wang, Ting-Chun, et al. “Video-to-video synthesis.” *arXiv* preprint arXiv:1808.06601 (2018).

[III.10] Sedaghat, Nima, Mohammadreza Zolfaghari, and Thomas Brox. “Hybrid learning of optical flow and next frame prediction to boost optical flow in the wild.” *arXiv* preprint arXiv:1612.03777 (2016).

[III.11] Terwilliger, Adam, Garrick Brazil, and Xiaoming Liu. “Recurrent flow-guided semantic forecasting.” 2019 *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019.

[III.12] Shi, Xingjian, et al. “Convolutional LSTM network: A machine learning approach for precipitation nowcasting.” *Advances in neural information processing systems 28* (2015).

[III.13] Lotter, William, Gabriel Kreiman, and David Cox. “Unsupervised learning of visual structure using predictive generative networks.” *arXiv* preprint arXiv:1511.06380 (2015).

[III.14] Villegas, Ruben, Dumitru Erhan, and Honglak Lee. “Hierarchical long-term video prediction without supervision.” *International Conference on Machine Learning. PMLR*, 2018.

[III.15] Villegas, Ruben, et al. “Learning to generate long-term future via hierarchical prediction.” *International conference on machine learning. PMLR*, 2017.

[III.16] Zhang, Jianjin, et al. “Z-order recurrent neural networks for video prediction.” *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2019.

[III.17] Ranzato, MarcAurelio, et al. “Video (language) modeling: a baseline for generative models of natural videos.” *arXiv* preprint arXiv:1412.6604 (2014).

[III.18] Srivastava, Nitish, Elman Mansimov, and Ruslan Salakhudinov. “Unsupervised learning of video representations using lstms.” *International conference on machine learning. PMLR*, 2015.

[III.19] Lotter, William, Gabriel Kreiman, and David Cox. “Deep predictive coding networks for video prediction and unsupervised learning.” *arXiv* preprint arXiv:1605.08104 (2016).

[III.20] Byeon, Wonmin, et al. “Contextvp: Fully context-aware video prediction.” Proceedings of the *European Conference on Computer Vision (ECCV)*. 2018.

[III.21] Patraucean, Viorica, Ankur Handa, and Roberto Cipolla. “Spatio-temporal video autoencoder with differentiable memory.” *arXiv* preprint arXiv:1511.06309 (2015).

[III.22] Lu, Chaochao, Michael Hirsch, and Bernhard Scholkopf. “Flexible spatio-temporal networks for video prediction.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[III.23] Denton, Emily L. “Unsupervised learning of disentangled representations from video.” *Advances in neural information processing systems 30* (2017).

[III.24] Oh, Junhyuk, et al. “Action-conditional video prediction using deep networks in atari games.” *Advances in neural information processing systems 28* (2015).

[III.25] Denton, Emily, and Rob Fergus. “Stochastic video generation with a learned prior.” *International conference on machine learning. PMLR*, 2018.

[III.26] Rochan, Mrigank. “Future semantic segmentation with convolutional lstm.” *arXiv* preprint arXiv:1807.07946 (2018).

[III.27] Vora, Suhani, et al. “Future segmentation using 3d structure.” *arXiv* preprint arXiv:1811.11358 (2018).

[III.28] Wang, Yunbo, et al. “Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms.” *Advances in neural information processing systems 30* (2017).

[III.29] Wang, Yunbo, et al. “PredRNN: A recurrent neural network for spatiotemporal predictive learning.” *arXiv* preprint arXiv:2103.09504 (2021).

[III.30] Yan, Jie, et al. “Mixpred: video prediction beyond optical flow.” *IEEE Access 7* (2019): 185654-185665.

[III.31] Liu, Ziwei, et al. “Video frame synthesis using deep voxel flow.” *Proceedings of the IEEE International Conference on Computer Vision*. 2017.

[III.32] Kwon, Yong-Hoon, and Min-Gyu Park. “Predicting future frames using retrospective cycle gan.” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

[III.33] Oprea, Sergiu, et al. “A review on deep learning techniques for video prediction.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[III.34] Tulyakov, Sergey, et al. “Mocogan: Decomposing motion and content for video generation.” *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.

[III.35] Hochreiter, Sepp, and Jurgen Schmidhuber. “Long short-term memory.” *Neural computation 9.8* (1997): 1735-1780.

[III.36] Soltani, Rohollah, and Hui Jiang. “Higher order recurrent neural networks.” *arXiv* preprint arXiv:1605.00064 (2016).

[III.37] Cances, Jean-Pierre, and Vahid Meghdadi. “Joint channel estimation and data demodulation algorithms for fast time varying band-limited frequency selective Rayleigh fading channels: a comparison study.” *Annales des télécommunications.* Vol. 55. No. 5. Springer-Verlag, 2000.

[III.38] Schuldt, Christian, Ivan Laptev, and Barbara Caputo. “Recognizing human actions: a local SVM approach.” *Proceedings of the 17th International Conference on Pattern Recognition*, 2004. ICPR 2004. Vol. 3. IEEE, 2004.

[III.39] Hore, Alain, and Djemel Ziou. “Image quality metrics: PSNR vs. SSIM.” 2010 *20th international conference on pattern recognition. IEEE*, 2010.

[III.40] Zhang, Richard, et al. “The unreasonable effectiveness of deep features as a perceptual metric.” *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018

[III.41] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. “Imagenet classification with deep convolutional neural networks.” *Advances in neural information processing systems 25* (2012).

[III.42] Barnett, Thomas, et al. “Cisco visual networking index (vni) complete forecast update, 2017–2022.” *Americas/EMEAR Cisco Knowledge Network (CKN) Presentation* (2018).

[III.43] Sullivan, Gary J., et al. “Overview of the high efficiency video coding (HEVC) standard.” *IEEE Transactions on circuits and systems for video technology 22.12* (2012): 1649-1668.

[III.44] Mun, Sungkwang, and James E. Fowler. “Residual reconstruction for block-based compressed sensing of video.” *2011 Data Compression Conference*. IEEE, 2011.

[III.45] Liu, Chao, et al. “Learned Video Compression with Residual Prediction and Loop Filter.” *arXiv* preprint arXiv:2108.08551 (2021).

[III.46] Vaswani, Ashish, et al. “Attention is all you need.” *Advances in neural information processing systems 30* (2017).

Table of contents

Chapter IV. Video Compressive Sensing based on Vision Transformers

Chapter IV. Video Compressive Sensing based on Vision Transformers

IV.1. Introduction

Designing and implementing an efficient end-to-end algorithm in a Video Compressive Sensing context has always been a challenging task for researchers and Deep Learning engineers. Recently, most of Deep Learning-based VCS frameworks are based on convolutional and recurrent architectures. However, since 2017, a groundbreaking architecture has slowly taken its place in the Computer Vision field. Therefore, we worked on the first Video Compressive Sensing algorithm built upon Transformers. The remaining chapter is organized as follows: Section IV.2 discusses related works in video compressive sensing and Transformer based architectures. Section IV.3 discusses the advantages of using Transformers over the well-known convolutional and recurrent architectures commonly used in a Video Compressive Sensing context. In Section IV.4, we present the main architecture behind ViT-SCI. In Section IV.5, we evaluate the performance of our proposed algorithm in a video SCI context with an extensive ablation study on different hyperparameters. Finally, Section IV.6 provides our conclusion and the main perspectives of this research work.

IV.2. Background and Related Works

The present section introduces the main research works in Video Snapshot Compressive Imaging, which is the main paradigm exploited in our research direction. Also, it presents the recent exploitation of Transformers, originally used in Natural Language Processing (NLP), in Computer Vision.

IV.2.1. Video Snapshot Compressive Imaging

Compressing high-speed videos is already possible due to the huge research work done in video snapshot compressive imaging (SCI). The video SCI system is composed of two main networks: the hardware encoder and the software reconstruction (decoder) network [IV.1]. The hardware encoder represents the optical imaging framework and the software decoder denotes the reconstruction algorithm. The hardware encoder aims to compress the 3D video signal into a 2D measurement matrix and the compression is done across the temporal dimension. This compression aims to avoid huge memory storage and transmission bandwidth. The optical system is called the coded aperture compressive temporal imaging (CACTI) [IV.2] system (Figure IV.1). In this system, and during one exposure time, the video scene is gathered by an objective lens and then coded by a temporal-variant mask (shifting physical mask [IV.2]-[IV.3], or different patterns on a Digital Micromirror Device (DMD) [IV.4]-[IV.5]]. Then, the output is detected by a Charge Coupled Device (CCD) and then integrated into one single measurement frame.

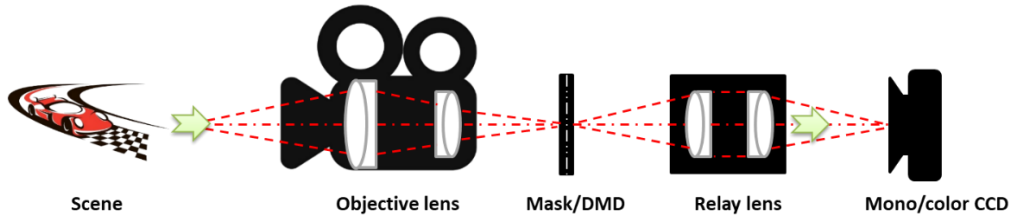


Figure IV.1: Schematic of the CACTI system.

From a mathematical perspective, a video SCI system captures a dynamic scene of $\boldsymbol{B}$ frames $\boldsymbol{X} \in \mathbb{R}^{h \times w \times B}$ ($\boldsymbol{h}$ and $\boldsymbol{w}$ are the height and the weight of the frame, respectively) which is

modulated by a number of masks ($\boldsymbol{B}$) noted $\boldsymbol{C_k} \in \mathbb{R}^{h \times w}$, $\boldsymbol{k} = 1 \ldots \boldsymbol{B}$, before being integrated into one single measurement frame $\boldsymbol{Y} \in \mathbb{R}^{h \times w}$ by a camera sensor in one exposure time ($\boldsymbol{B}$ frame). This operation is expressed as follows (IV.1):

$$\boldsymbol{Y} = \sum_{\boldsymbol{k=1}}^{\boldsymbol{B}} \boldsymbol{X_k} \circ \boldsymbol{C_k} + \boldsymbol{G}, \tag{IV.1}$$

Where $\boldsymbol{X_k}$ and denotes the k^{th} frame, $\circ$ and $\boldsymbol{G} \in \mathbb{R}^{h \times w}$ denote the Hadamard product and noise, respectively. Then, we define $\boldsymbol{y} = Vec(\boldsymbol{Y}) \in \mathbb{R}^{hw}$ and $\boldsymbol{g} = Vec(\boldsymbol{G}) \in \mathbb{R}^{hw}$ where Vec represents the vectorization operator. Correspondingly, we define $\boldsymbol{x} \in \mathbb{R}^{hwB}$ as (IV.2):

$$\boldsymbol{x} = [Vec(\boldsymbol{X_1})^{\boldsymbol{T}}, \ldots, Vec(\boldsymbol{X_B})^{\boldsymbol{T}}]^{\boldsymbol{T}}. \tag{IV.2}$$

The measurement $\boldsymbol{y}$ can then be expressed as (IV.3):

$$\boldsymbol{y} = [\boldsymbol{D_1}, \ldots, \boldsymbol{D_B}]\boldsymbol{x} + \boldsymbol{g}, \tag{IV.3}$$

where $\boldsymbol{D_b} = diag(Vec(\boldsymbol{C_b})) \in \mathbb{R}^{hw \times hw}$, for $\boldsymbol{b} = 1 \ldots \boldsymbol{B}$ denotes a diagonal matrix. We have in this case a matrix $[\boldsymbol{D_1}, \ldots, \boldsymbol{D_B}]$ that is highly structured and sparse. Depending on the theoretical study in [IV.6], the original video can be reconstructed from a single compressed measurement frame y and the coding patterns $\{\boldsymbol{C_k}\}_{k=1}^{B}$ [IV.6] with a sampling rate of $\frac{\boldsymbol{1}}{\boldsymbol{B}}$.

The second important part of video SCI is the reconstruction process which aims to recover the original video from the 2D measurement frame and the masks. This process is crucial to have a practical and efficient video SCI system. In the literature, the reconstruction algorithms could be classified into two categories: optimization-based methods and Deep Learning based algorithms. The optimization-based algorithms, such as GAP-TV [IV.7], GMM [IV.8], DeSCI [IV.9], and PnP-FFDNet [IV.10], require huge computational resources and large reconstruction time. For instance, DeSCI takes hours to generate a 256 × 256 × 8 video from one single measurement frame). In Deep Learning based methods [IV.11]-[IV.12]-[IV.13]-[IV.14]-[IV.15]-[IV.16]-[IV.17]-[IV.18]-[IV.19], this computational problem has been ameliorated. However, some architectures need a large memory and a huge amount of time for the training phase. BIRNAT [IV.14], for example, can take weeks to train a model of size 256 × 256 × 8 [IV.20]. Obviously, both categories have their advantages and drawbacks, which make this research direction challenging and very promising for the future if we aim to come up with a memory friendly model that consumes less computational cost for our daily life applications.

IV.2.2. From NLP to computer vision

Since there are various high-level analogies between video processing and NLP, we decided to take advantage of this architecture for our video reconstruction purpose. In fact, video and sentences have sequential features. In addition, if a word can be understood from the context in a sentence, patches could be reconstructed based on the contextual features gathered from the rest of the video or to be precise from the tokens having similar features based on the computations of the attention layer.

IV.2.3. Transformers in computer vision

Transformers are originally proposed in 2017 [IV.21] as a simple and scalable architecture in language translation and successfully dominate natural language processing (NLP) tasks [IV.22]-[IV.23]. Indeed, transformers are based on self-attention mechanism which is a highly efficient technique to learn the correlations between input features and update the embeddings in parallel. Thus, in contrast to recurrent architectures, transformers-based models allow modelling long dependencies between input data components and handle parallel processing. Indeed, they are characterized by their scalability to very high-complexity models. Recently, transformers started to improve computer vision tasks. They have been used in various computer vision applications such as classification [IV.24]-[IV.25], video segmentation [IV.26], object detection [IV.27] and video inpainting [IV.28].

IV.2.4. Challenges in computer vision applications

Although transformers are becoming a research trend in the last two years due to their excellent performances, they are facing some crucial challenges in the computer vision field. Some hindrances include their requirement for large amounts of training data engendering high computational costs (in terms of computational time and memory resources needed for processing) [IV.29].

IV.3. Transformers in a Video Compressive Sensing Context: main contributions

In this part, we aim to highlight the relevant advantages of Transformer-based architectures in comparison with the commonly used CNN and RNN models. Also, we detail our main contributions in this research concept.

IV.3.1. Why are Transformers steadily replacing CNN/RNN architectures?

With the huge demand for data acquisition and processing, Video Compressive Sensing or precisely Video Snapshot Compressive Imaging (SCI) becomes a promising research direction. It is the task to indirectly capture high dimensional data and encode it into one single 2D compressed measurement to optimize the memory storage of the system and its transmission bandwidth. Then, an efficient reconstruction algorithm is needed to reconstruct the original video from the compressed measurement. For the last decades, practical video recovery approaches are mainly based on convolutional and recurrent neural networks [IV.11]-[IV.12]-[IV.13]-[IV.14]-[IV.15]-[IV.16]-[IV.17]-[IV.18]-[IV.19]. While these models achieve practical performances, the recovery process in video compressive sensing remains very challenging in terms of flexibility, scalability and speed of the training and the testing phases [IV.1]-[IV.30].

On the one hand, recurrent neural networks are designed to process data sequentially which makes the implementation of parallel computing very difficult and slows down the training phase. Also, processing long sequences through recurrent networks leads to a loss of information and causes the vanishing gradient problem [IV.31]. To deal with the vanishing gradient problem, one of the most impactful papers in Deep Learning [IV.21] has proposed the attention mechanism which manages and quantifies the interdependence between input elements. This attention mechanism has contributed towards the designing and the implementation of transformer models. In fact, these models enable the efficient utilization of GPUs by parallelly processing input sequences and then speed up the training phase considerably. In addition, it is challenging to use transfer learning on recurrent models. However, it is practical to use pretrained transformers to reduce the training cost.

On the other hand, convolutional neural networks (CNN) are simple to parallelize. Also, for various applications, CNN based models are fast to train but for short input sequences. For long sequences, convolutional models are unable to learn different dependencies among all the possible combinations of the input elements. That's why, it is practical to process long sequences as a whole using transformers. Transformers are thus better than recurrent neural networks and convolutional neural networks for the following reasons:

- Computational complexity per layer: Self-attention layers $O(n^2.d)$ are faster than recurrent layers $(n.d^2)$ and convolutional layers $O(k.n.d^2)$ when the dimensionality d is bigger than the input sequence length n (which is the case in NLP models) [IV.21].
- The computation can be parallelized: Recurrent networks need $O(n)$ sequential operations. However, self-attention layers can be computed in a parallel manner.
- The path length between long-range dependencies: it is more important with recurrent and convolutional layers than with self-attention layers [IV.21].

IV.3.2. Main contributions in a VCS context

Bearing the above problems in mind, in this work, we intend to enhance the reconstruction performances by proposing an end-to-end transformers-based model for SCI video reconstruction trying to solve the trilemma of flexibility, scalability and speed.

However, applying efficient transformers for various computer vision applications such as SCI reconstruction is still facing several challenges. In fact, famous vision transformers (e.g. ViT [IV.24]) divide input 2D images into several patches which may threaten the local spatial information [IV.32] because some low-level visual features (e.g. edges, shapes) are divided into different patches. After the patch embedding step, global fully connected self-attention is applied to extract the global interactions between different tokens which ignores local details. Then, for video recovery problems, temporal data may be the key for better performances since missing information in one frame can be reconstructed from adjacent frames.

As a result, the idea is to come up with a new transformer-based architecture for video snapshot compressive imaging (SCI) with an attention layer that exploit local and spatio-temporal data information.

In a nutshell, our contributions are summarized as follows:

- To the best of our knowledge, the proposed algorithm (ViT-SCI) is the first video SCI reconstruction method built upon Transformers.
- We used a convolutional attention mechanism in order to exploit spatiotemporal information instead of global fully connected attention layers used in recent vision transformers.
- We provided detailed explanation of our architecture with detailed results and analysis which may be used as reference in future research works, especially on video transformers.
- ViT-SCI achieves strong results on DAVIS2017 training dataset in comparison with other video SCI reconstruction algorithms based on Deep Learning architectures and optimization methods.

IV.4. Overview of the proposed architecture; ViT-SCI

As illustrated in Figure IV.2, ViT-SCI consists of three main modules: low frequency feature extraction module, deep feature extraction module and a video reconstruction module. These 3 modules are preceded by a measurement normalization phase aiming to generate a preprocessed video. The entire training process is shown in Algorithm IV.1.

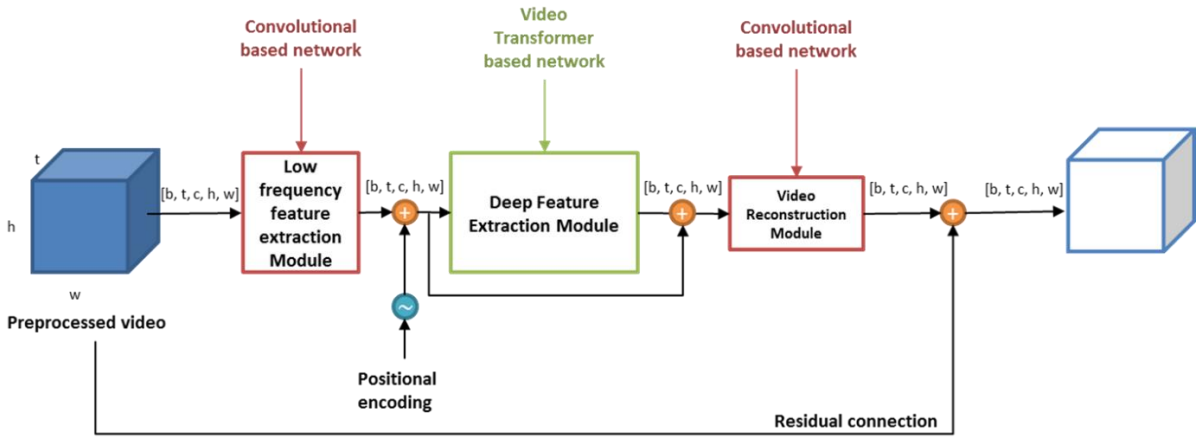


Figure IV.2: The architecture of the proposed ViT-SCI for video reconstruction in Video Snapshot Compressive Imaging.

Require: Measurement $\boldsymbol{Y}$, Coding Patterns $\boldsymbol{C_k}$

Randomly initialize all training parameters
while not done **do**
  for $i = 1{:}\,\boldsymbol{n_{epochs}}$ **do**
    for All training video sequences **do**
      Load $\boldsymbol{Y}$, $\boldsymbol{C_k}$, $\boldsymbol{G_t}$
      $\overline{\boldsymbol{Y}} \leftarrow \boldsymbol{Y} \sum_{k=1}^{B} \boldsymbol{C_k}$
      $\boldsymbol{I} \leftarrow [\overline{\boldsymbol{Y}} \circ \boldsymbol{C_1}, \dots, \overline{\boldsymbol{Y}} \circ \boldsymbol{C_B}]_3$
      $\boldsymbol{F_{low}} \leftarrow N_{fe}\,(\boldsymbol{I})$
      $\boldsymbol{F_{lowPE}} \leftarrow \boldsymbol{F_{low}} + \boldsymbol{PE}$
      for $d = 1{:}\,NumberOfAttentionLayers$ **do**
        for $h = 1{:}\,n_{heads}$ **do**
          $\boldsymbol{Att_h} \leftarrow Attention\,(\boldsymbol{F_{lowPE}})$
        end for
        $\boldsymbol{F_{deep}} \leftarrow FFN\,(\boldsymbol{Att_i})$
      end for
      $\boldsymbol{O_{rec}} \leftarrow N_{rec}\,(\boldsymbol{F_{deep}} + \boldsymbol{F_{lowPE}})$
      $\boldsymbol{O_{rec}} \leftarrow \boldsymbol{O_{rec}} + \boldsymbol{I}$
      Obtain loss: $\boldsymbol{\mathcal{L}} = \sum_{n=1}^{N} \sum_{k=1}^{B} \left\| \boldsymbol{O_{f_n,k}} - \boldsymbol{G_{t_n,k}} \right\|_2^2$
      Update all parameters via: $\boldsymbol{W} \leftarrow \boldsymbol{W} - Adam(\boldsymbol{\mathcal{L}})$
    end for
  end for
end while

Algorithm IV.1: ViT-SCI: the training process.

IV.4.1. Preprocessed Video and Measurement Energy Normalization

The output of the compressive sensing acquisition phase is the measurement matrix $\boldsymbol{Y} \in \mathbb{R}^{h \times w}$. Having the measurement matrix $\boldsymbol{Y}$ and the coding patterns (masks) $\boldsymbol{C} \in \mathbb{R}^{h \times w \times B}$, we preprocessed the training data before feeding our deep learning algorithm. One of the preprocessing techniques that have been applied recently [IV.14]-[IV.20]-[IV.33] is measurement energy normalization. In fact, the measurement matrix $\boldsymbol{Y}$ is not usually energy normalized which requires a normalization process to fit into the neural network. Technically, the energy-normalized measurement matrix $\overline{\boldsymbol{Y}}$ can be expressed as (IV.4):

$$\overline{\boldsymbol{Y}} = \boldsymbol{Y} \sum_{\boldsymbol{k=1}}^{\boldsymbol{B}} \boldsymbol{C_k}, \tag{IV.4}$$

where represents the matrix dot division. Figure IV.3, which describes the preprocessing approach (the illustrated frames are extracted from the training dataset), shows that the energy-normalized measurement matrix $\overline{\boldsymbol{Y}}$ presents more visual information than the initial measurement matrix $\boldsymbol{Y}$. Obviously, $\overline{\boldsymbol{Y}}$ can be defined as the estimated average of the original B high-speed frames $\boldsymbol{X} \in \mathbb{R}^{h \times w \times B}$. Then, in order to generate a preprocessed video from the energy-normalization measurement matrix $\overline{\boldsymbol{Y}}$ and the coding patterns $\boldsymbol{C} \in \mathbb{R}^{h \times w \times B}$, we processed the following concatenation along the 3rd dimension (IV.5):

$$\boldsymbol{I} = [\overline{\boldsymbol{Y}} \circ \boldsymbol{C_1}, \ldots, \overline{\boldsymbol{Y}} \circ \boldsymbol{C_B}]_{\boldsymbol{3}} \in \mathbb{R}^{h \times w \times B}. \tag{IV.5}$$

The preprocessed video $\boldsymbol{I}$, preserving the background and some main objects of the frames as illustrated in Figure IV.3, will feed the reconstruction network.

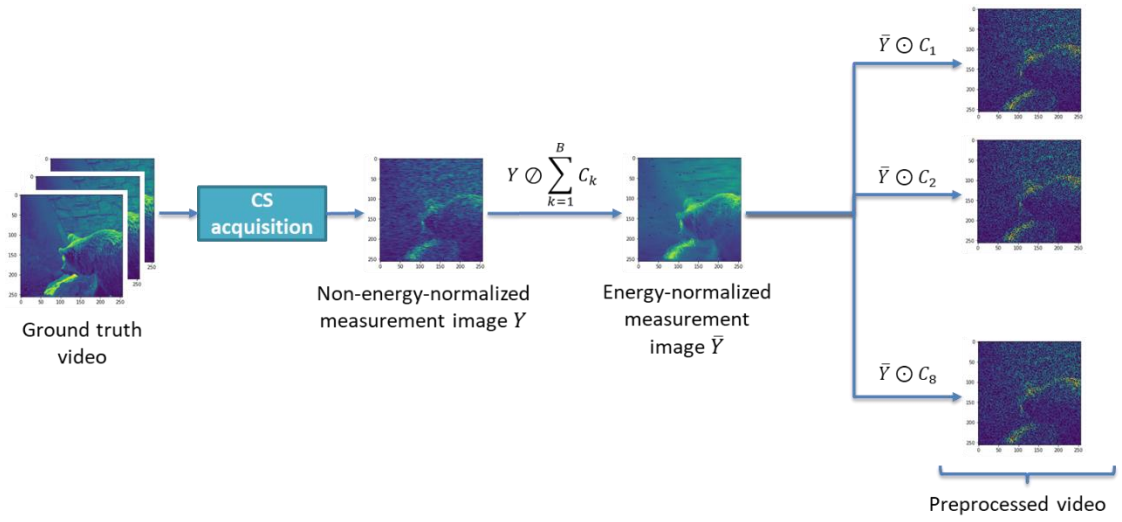


Figure IV.3: The preprocessing strategy.

IV.4.2. Low Frequency Feature Extraction Module

Given the preprocessed video $\boldsymbol{I} \in \mathbb{R}^{B \times c \times h \times w}$ (c denotes the number of channels), we used 5 residual blocks followed by LeakyReLU activation function in order to learn low frequency features $\boldsymbol{F_{low}} \in \mathbb{R}^{B \times c \times h \times w}$ as (IV.6):

$$\boldsymbol{F_{low}} = N_{fe}(\boldsymbol{I}), \tag{IV.6}$$

where N_{fe} denotes the network designed to extract low frequencies features from the input video $\boldsymbol{I}$.

This module aims to extract low frequencies in images which means the pixels that are changing slowly over space which enables to learn the background and the main shapes in the frames and to accelerate the learning process of the transformer-based module.

IV.4.3. Positional encoding

In contrast to standard neural networks, Transformer based models are permutation-invariant. However, ViT-SCI necessitates accurate position information. As a result, we added a fixed 3D positional encoding, including spatial and temporal information, to the features generated by the low frequency features extraction module of the input I. The 3D positional encoding ($\boldsymbol{PE}$) [IV.27] is defined as (IV.7):

$$PE_{3D}(pos, i) = \begin{cases} \sin(\boldsymbol{\beta_k}.\boldsymbol{pos}) \text{ for } \boldsymbol{i} = \boldsymbol{2k} \\ cos(\boldsymbol{\beta_k}.\boldsymbol{pos}) \text{ for } \boldsymbol{i} = \boldsymbol{2k+1}, \end{cases} \tag{IV.7}$$

where $\boldsymbol{\beta_k} = \frac{1}{10000^{\frac{6k}{d_c}}}$, pos is the position of the corresponding dimension, $\boldsymbol{d_c}$ represents the size of the channel dimension and $\boldsymbol{k} \in \mathbb{N}\ s.t.k \in \left[0, \frac{d}{6}\right]$.

IV.4.4. Deep Feature Extraction Module

We have specifically developed a new transformer encoder for video SCI recovery that achieves deep features extraction. The idea behind the deep feature extraction module is to build a network aiming to learn non-linear mapping to enable video reconstruction. This transformer encoder maps the input video space to a higher dimensional feature space. The deep features $\boldsymbol{F_{deep}} \in \mathbb{R}^{B\times c\times h\times w}$, extracted by the encoder, can be expressed as (IV.8):

$$\boldsymbol{F_{deep}} = N_{transformer}(\boldsymbol{F_{low}}), \tag{IV.8}$$

where $N_{transformer}$ denotes the application of the deep feature extraction module.

IV.4.4.1. Spatio-Temporal Convolutional Multi-Head Attention (ST-ConvMHA)

It has been proved in previous research works [IV.34]-[IV.35] that fully connected self-attention originally developed in [IV.21] is not suitable for computer vision tasks and especially for video reconstruction models. In fact, fully-connected self-attention is used to extract global interactions between different tokens which neglects local information. Also, it ignores the temporal dimension which is a crucial information in video processing related tasks. In addition, in [IV.34], it has been theoretically proved that fully-connected self-attention layers used for vision tasks may cause the vanishing gradient problem destabilizing the training process.

Bearing in mind the aforementioned limitation of the fully-connected self-attention layer, deep feature module, which enables to map the features to a series of continuous models, is mainly based on the Spatio-Temporal Convolutional Multi-Head Attention (ST-ConvMHA) layer designed to extract spatial-temporal information and the similarities between different tokens.

Our proposed ST-ConvMHA is a stack of parallel convolutional multi-head attention layers that allow a better understanding of the different aspects of the input feature maps $\boldsymbol{F_{low}} \in \mathbb{R}^{B\times c\times h\times w}$.

ST-ConvMHA is based on convolutional projections applied for Query(**Q**), Key(**K**) and Value(**V**) embeddings, respectively and a patch-wise non-local attention model using unfold and fold operations inspired from [IV.36].

The first step in calculating ST-ConvMHA is replacing the existing position-wise linear projections in the fully-connected self-attention mechanism [IV.21] with convolutional projections using three different convolutional layers with trainable elements $\boldsymbol{W_Q}$, $\boldsymbol{W_K}$ and $\boldsymbol{W_V}$. This embedding step, aiming to learn the spatial features of the different frames, can be expressed as follows (IV.9):

$$\begin{aligned} \boldsymbol{Q} &= EmbQ(\boldsymbol{F_{low}}) = \boldsymbol{W_Q} \odot \boldsymbol{F_{low}} \\ \boldsymbol{K} &= EmbK(\boldsymbol{F_{low}}) = \boldsymbol{W_K} \odot \boldsymbol{F_{low}} \\ \boldsymbol{V} &= EmbV(\boldsymbol{F_{low}}) = \boldsymbol{W_V} \odot \boldsymbol{F_{low}}, \end{aligned} \tag{IV.9}$$

where $\odot$ denotes the convolution operation and Emb is the embedding step.

The second step in the calculation process is using the unfold operation to extract sliding local tokens from $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ tensors. The kernel size used in this operation is $\boldsymbol{H_{patch}} \times \boldsymbol{W_{patch}}$ and the stride is $\boldsymbol{s} = \boldsymbol{H_{patch}}$ or $\boldsymbol{s} = \boldsymbol{W_{patch}}$. As illustrated in Figure IV.4, the output of the unfolding operation is three groups of 3D tokens. Each group contains N 3D tokens $\left(\boldsymbol{N} = \frac{\boldsymbol{BWH}}{\boldsymbol{W_{patch}} \times \boldsymbol{H_{patch}}}\right)$. Each token has the size of $\boldsymbol{dim_{patch}} = \boldsymbol{c} \times \boldsymbol{W_{patch}} \times \boldsymbol{H_{patch}}$.

This process is expressed as follows (IV.10):

$$\begin{aligned} \boldsymbol{Q_1}, \boldsymbol{Q_2}, \dots, \boldsymbol{Q_N} &= \theta(\boldsymbol{Q}) = \theta(\boldsymbol{W_Q F_{low}}) \\ \boldsymbol{K_1}, \boldsymbol{K_2}, \dots, \boldsymbol{K_N} &= \theta(\boldsymbol{K}) = \theta(\boldsymbol{W_K F_{low}}) \\ \boldsymbol{V_1}, \boldsymbol{V_2}, \dots, \boldsymbol{V_N} &= \theta(\boldsymbol{V}) = \theta(\boldsymbol{W_V F_{low}}), \end{aligned} \tag{IV.10}$$

where θ is the unfolding operation and $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{B \times c \times W_{patch} \times H_{patch}}$.

The third step is to reshape the Query and the Key tensors into 1D vectors of size $dim_{patch} \times \boldsymbol{N}$. The reshaping operator is subsequently denoted by Δ. Then, we calculated the score (the similarity matrix) by calculating the dot product of the matrix of the reshaped query and the matrix of the reshaped key. This score (similarity matrix) is related to all embedding patches of the video which guarantee the learning of the spatial-temporal details. Then, the obtained scores are divided by the square root of each patch in the current head layer since we are implementing a muti-head attention layer, motivated by [IV.21]. Then, we passed the result through a softmax operation. In fact, the softmax layer will determine the importance of patches corresponding to other patches.

The fourth step is to multiply each value vector by the output of the softmax layer. The third and fourth steps are expressed as follows (IV.11):

$$Attention(\boldsymbol{Q_i}, \boldsymbol{K_i}, \boldsymbol{V_i}) = softmax\left(\frac{\boldsymbol{Q_i}\boldsymbol{K_i}^{\boldsymbol{T}}}{\sqrt{\frac{\boldsymbol{dim_{patch}}}{\boldsymbol{n_{heads}}}}}\right)\boldsymbol{V_i}. \quad \text{(IV.11)}$$

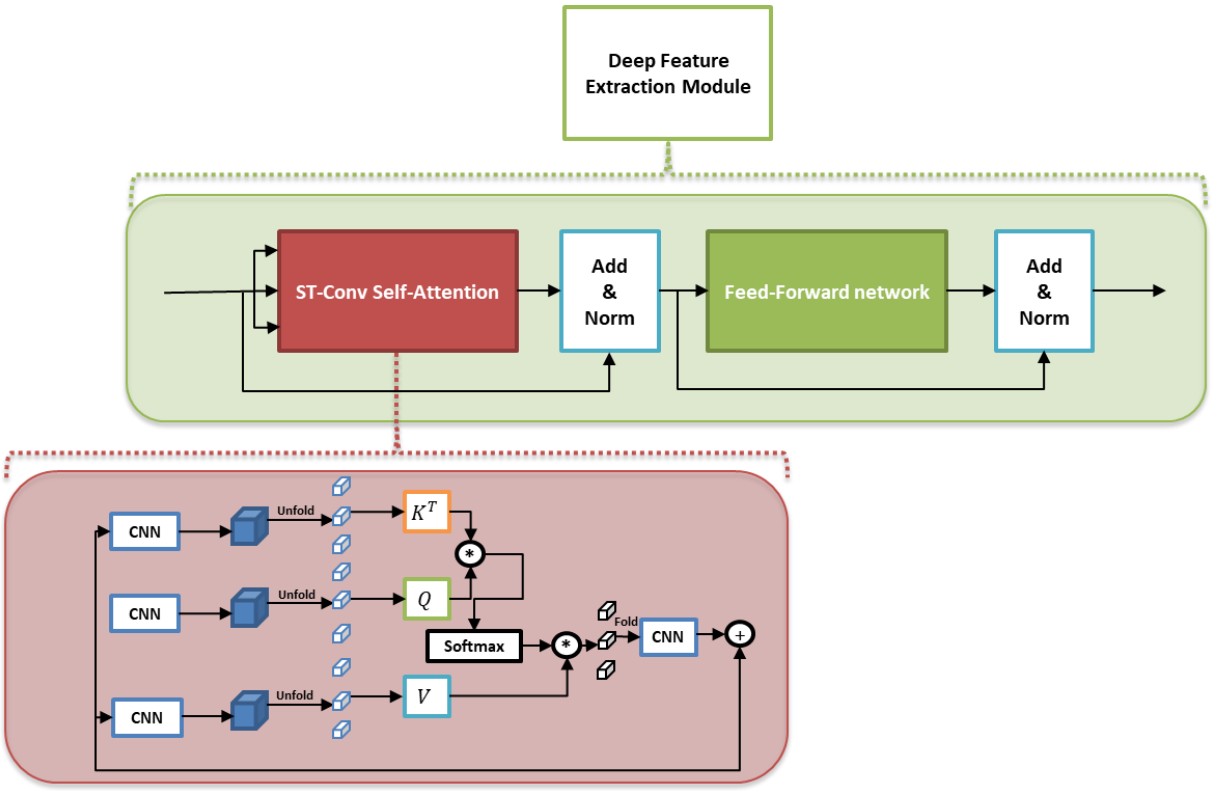


Figure IV.4: The Deep Feature Extraction Module.

Finally, we applied the folding operation Γ in order to combine the sliding local blocks of size $\boldsymbol{N} \times \boldsymbol{c} \times \boldsymbol{W_{patch}} \times \boldsymbol{H_{patch}}$ into one large containing tensor (feature map) of size $\boldsymbol{B} \times \boldsymbol{c} \times \boldsymbol{W} \times \boldsymbol{H}$. Then, we applied a convolutional layer $\boldsymbol{W_f}$ to generate the final feature map.

The i^{th} attention head process can be expressed as (IV.12):

$$head_{\boldsymbol{i}} = \boldsymbol{W_f^i}\Gamma\left(softmax\left(\frac{\boldsymbol{QK^T}}{\sqrt{\frac{\boldsymbol{dim_{patch}}}{\boldsymbol{n_{heads}}}}}\right)\boldsymbol{V}\right), \quad \text{(IV.12)}$$

i.e, when expanding the expressions of $\boldsymbol{Q_i}, \boldsymbol{K_i}$ and $\boldsymbol{V_i}$ (IV.13):

$$head_i = \boldsymbol{W_f^i}\Gamma\left(softmax\left(\frac{\theta\left(\Delta\left(\boldsymbol{W_Q^i F_{low}}\right)\right)\times\theta\left(\Delta\left(\boldsymbol{W_K^i F_{low}}\right)\right)^{\boldsymbol{T}}}{\sqrt{\frac{\boldsymbol{dim_{patch}}}{\boldsymbol{n_{heads}}}}}\right)\theta\left(\boldsymbol{W_V^i F_{low}}\right)\right). \quad \text{(IV.13)}$$

And the overall process of the St-ConvMHA layer is summarized as follows (IV.14):

$$ST - ConvMHA = Concat\,(head_1, \ldots, head_h)\boldsymbol{W^O}, \quad \text{(IV.14)}$$

where $\boldsymbol{h}$ is the number of heads or the number of parallel convolutional attention layers and $\boldsymbol{W^O}$ is a parameter matrix.

The implemented ST-ConvMHA enables to deeply learn spatial-temporal features in comparison with the fully-connected self-attention mechanism.

IV.4.4.2. Feed-Forward Network

As shown in Figure IV.4, the ST-ConvMHA layer is followed by a Feed Forward Network (FFN) [IV.21]. It is applied to every attention tensor to transform them into a form that can feed the next transformer encoder layer. In fact, the parallelization process is enabled by the FFN, because it processed all the attention tensors at one time.

IV.4.5. Video Reconstruction Module

In the reconstruction module, we recovered the video frames from processing the deep features generated by the transformer encoder as (IV.15):

$$O_{rec} = N_{rec}\left(\boldsymbol{F_{deep}} + \boldsymbol{F_{low}}\right), \quad \text{(IV.15)}$$

where N_{rec} is the reconstruction network. O_{rec} depends on $\boldsymbol{F_{deep}}$ and $\boldsymbol{F_{low}}$ to stabilize the training phase.

The final output of our approach is the aggregation of the output of the reconstruction module O_{rec} while the input preprocessed video $\boldsymbol{I}$ (IV.16):

$$O_f = O_{rec} + \boldsymbol{I}. \quad \text{(IV.16)}$$

IV.4.6. Training Process and Loss Function

In our implementation (Algorithm IV.1), we optimized the parameters of ViT-SCI by minimizing the reconstruction error: the loss function used is the mean square error (MSE) (IV.17):

$$\boldsymbol{\mathcal{L}} = \sum_{\boldsymbol{n=1}}^{\boldsymbol{N}}\sum_{\boldsymbol{k=1}}^{\boldsymbol{B}}\left\|\boldsymbol{O_{f_{n,k}}} - \boldsymbol{G_{t_{n,k}}}\right\|_2^2, \quad \text{(IV.17)}$$

where $\boldsymbol{O_{f_{n,k}}}$ is the final output or the k^{th} reconstructed frame of the $\boldsymbol{n^{th}}$ training video using ViT-SCI, and $\boldsymbol{G_{t_{n,k}}}$ is the corresponding ground truth frame.

IV.5. Performance evaluation, comparison and discussion

In this section, we describe the implementation framework and compare the performances of the proposed reconstruction method with several state-of-the-art methods.

IV.5.1. Datasets

To train our algorithm, we used the DAVIS2017 [IV.37] dataset, designed for video object segmentation applications, since video SCI algorithms can be applied on any video scene and there is no specific dataset for the training phase. The original DAVIS2017 dataset has only 90 video scenes (6242 frames of size 854×480). For an efficient training in a video SCI context, we prepared the dataset by transforming and reformatting it. In fact, we generate 6516 video scenes of size 8 × 256 × 256 from the 90 videos of DAVIS2017. Then, we tested our trained model on six evaluation datasets: Aerial, Drop, Kobe, Runner, Traffic, and Vehicle.

IV.5.2. Data Augmentation

In order to deal with the problem of overfitting, data augmentation is a commonly used pre-processing technique aiming to generate more data than RNN and CNN based models, becoming greater in terms of size. Since transformer-based models in general require more data, augmenting the diversity of the training dataset will enhance the performances of our proposed ViT-SCI [IV.38].

After the data augmentation process consisting in cropping, rotating, and flipping input videos, the dataset becomes larger with 417024 video scenes (3 336 192 frames). The idea is to generate 417024 video scenes of size 8 × 64 × 64 from the 6516 video scenes of size 8 × 256 × 256. Data augmentation has significantly enhanced the performances of our model.

IV.5.3. Compared methods and performance metrics

This part is dedicated to presenting the VCS methods used in the comparison evaluation and the main considered metrics.

IV.5.3.1. Compared methods

Several state-of-the-art methods are used to evaluate the performances of our proposed approach for the video SCI reconstruction, including two iteration-based reconstruction algorithms:

- **GAP-TV** [IV.7].
- **DeSCI** [IV.9].

and a recent deep learning-based reconstruction algorithm:

- **BIRNAT** [IV.14].

IV.5.3.2. Performance metrics

To quantify the performances of the evaluated algorithms, we use well known frames quality evaluation metrics: the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index(SSIM) [IV.39].

IV.5.4. Implementation details

The ViT-SCI algorithm has been implemented using Pytorch framework [IV.40]. We used the Mean Square Error (MSE) as a loss function in the main implementation. To minimize the MSE function, we used Adam optimizer [IV.41] with an initial learning rate of 0.0003 (the learning rate is reduced by 5% every 5 epochs).

The performance evaluation of the different approaches is done on an NVIDIA RTX 2080 GPU (8GB GDDR6). Our method is trained for 100 epochs and it took about 190 hours to train the entire ViT-SCI network.

IV.5.5. Network architecture

In the ST-ConvMHA, we used three convolutional layers to learn the spatial information of each frame. The output of the ST-ConvMHA layer passes through a convolutional layer to generate the final feature map. To decrease the computational cost of our model, we used gray scale frames for the training process ($c = 1$). The low frequency feature extraction module has 5 residual blocks. The deep feature extraction module uses 4 transformer encoder layers. The final video reconstruction module has 30 residual blocks.

IV.5.6. Ablation study

In this section, we study the core implementation of ViT-SCI through a profound experimental study to demonstrate the effectiveness of our model design choices.

IV.5.6.1. Frame size

Table IV.1 reports that ViT-SCI has larger computational cost (Training Time) when having higher spatial resolution. Indeed, about 155 more hours is required to train our model on DAVIS dataset with a spatial resolution of 80 × 80 than on the same dataset with a spatial resolution of 64 × 64. This large computational cost can threaten the scalability of the model while maintaining efficiency. However, we notice from Figure IV.5 and Figure IV.6 that training ViT-SCI on smaller images with smaller spatial dimensions does not affect quality performances as much. Therefore, we believe that our model can be extended to process larger training datasets.

Table IV.1: Ablation study on varying the input frame size.

Input seq. video	Params	PSNR (dB)	SSIM	Training Time (s)
8×64×64	25.258.625	31.1859	0.9130	425100
8×72×72	32.624.785	31.5721	0.9225	768277
8×80×80	41.090.081	31.1393	0.9218	984910
8×88×88	Out of Memory			

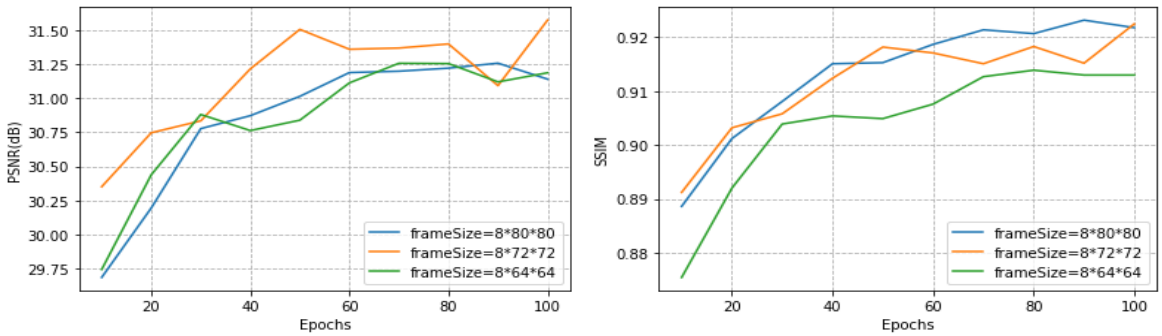


Figure IV.5: Ablation study on the effect of the frame size in training video clips: the average quality performances (Left: in terms of PSNR; Right: in terms of SSIM) on 6 test datasets.

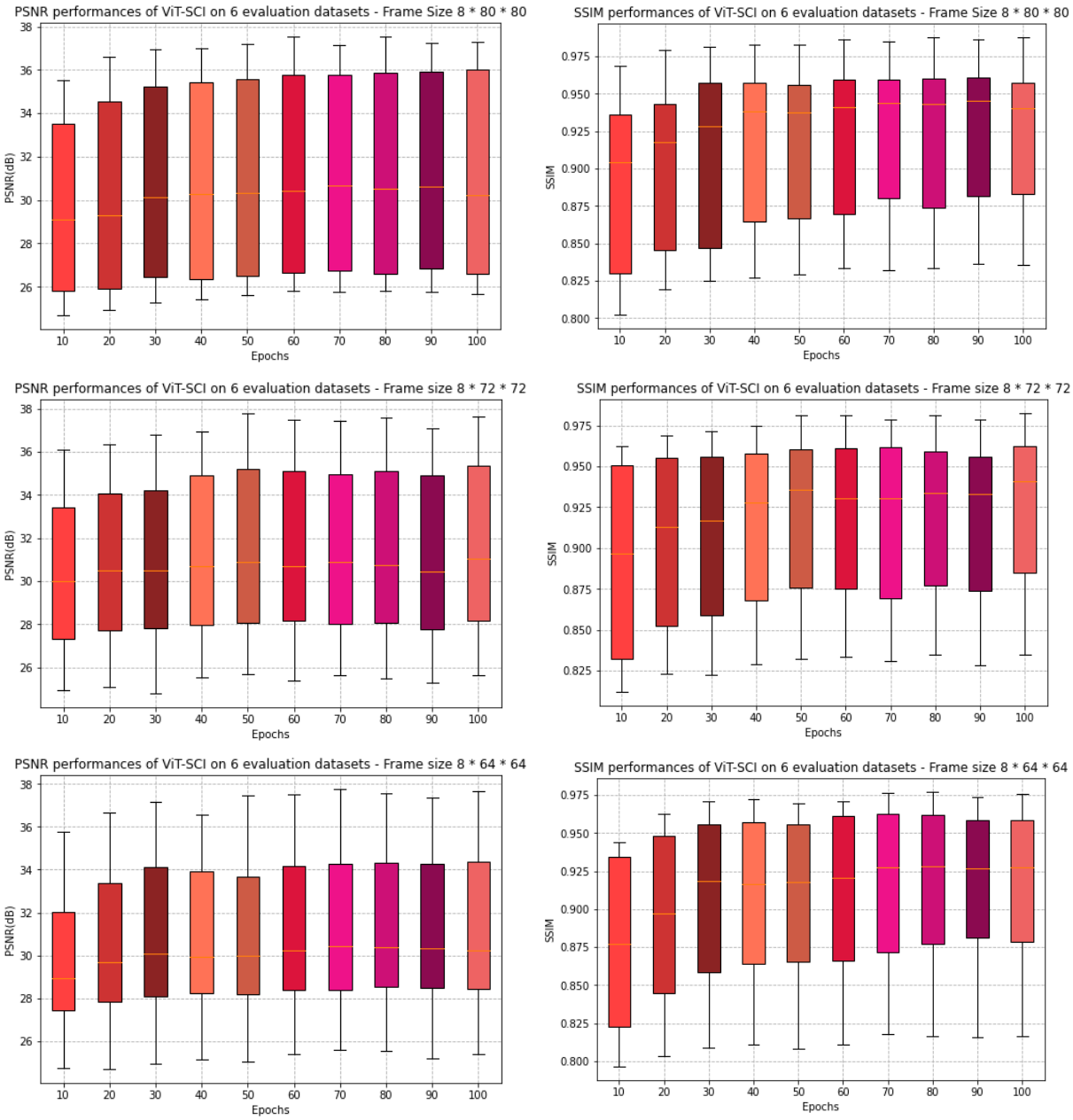


Figure IV.6: Ablation study on the effect of the frame size in training video clips: Box plots are used to visually show the distribution of PSNR and SSIM data and their skewness on 6 test datasets every 10 epochs, from 10 epochs to 100 epochs in the training process.

IV.5.6.2. Positional Embeddings

In Transformer based architectures, positional embeddings is of huge importance since all tokens are taken parallelly. The 3D positional encoding used in this implementation indicates the spatial and temporal positional embeddings which refers to the position in the video scene. To investigate the importance of our 3D positional embedding module, we conduct the following experiments (with 2 attention heads):

- No 3D positional embedding.
- 3D (Spatiotemporal) positional embedding.

The experiments, illustrated in Figure IV.7, shows that the model trained with 3D positional embeddings achieves better performances (+4.65% in terms of PSNR by passing from 30.1939 to 31.5969 and +1, 47% in terms of SSIM by passing from 0.9084 to 0.9218). This result proves that the positional information of every token is implicitly provided in the Transformer based architectures. However, it is important to enhance this positional information with explicitly implemented positional embeddings.

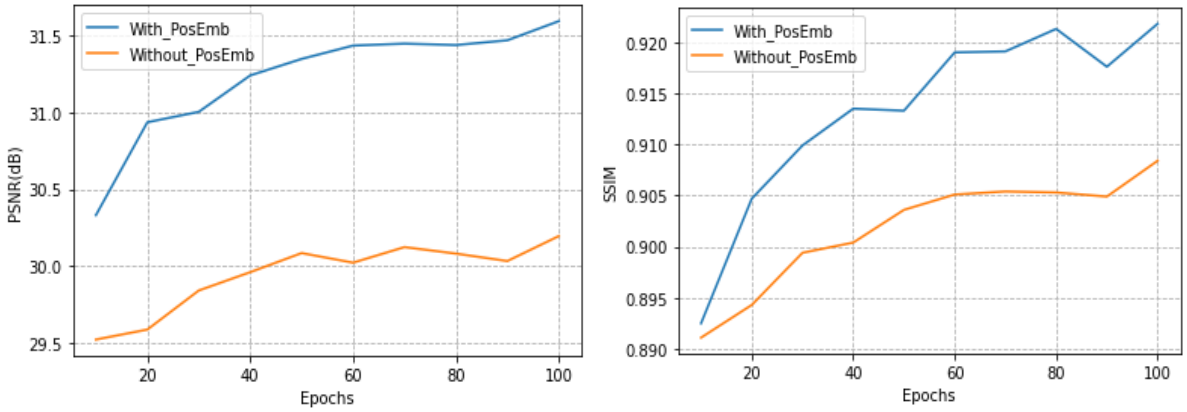


Figure IV.7: Ablation study on positional embeddings.

IV.5.6.3. Number of heads

We have carried out a series of experiments by training our model with one single attention head and with some independent attention layers applied in parallel to answer the famous question that has already been asked in [IV.42]: “Are Sixteen Heads Really Better than One?” or “is more than one head even needed?”. From the figures of Table IV.2, it is clear that the training time slightly increases with the number of heads, while the performances do not follow a monotonic behavior and even don’t show significant differences.

Table IV.2: Ablation on the number of attention heads.

Heads	PSNR (dB)	SSIM	Training Time (s)
1	31.0905	0.9118	421369
4	31.345	0.9162	423910
8	31.1859	0.9130	425100

We can further notice from Figure IV.8 and Figure IV.9 that the model is not sensitive to the number of attention heads. Therefore, one single attention head is sufficient, thus reducing the training computational cost. This may be explained by the fact that we have trained our approach on small sized video clips where the number of dynamic objects is limited and multiple heads are not needed to detect and learn syntactic relations between different objects. However, we believe that with higher temporal and spatial resolution datasets, pruning the attention heads in our model will result in significant performance degradation. Therefore, as suggested in [IV.43], it is advisable to retain more than one attention head and enhance formula (IV.14) to (IV.18):

$$ST - ConvMHA = \sum_{i} Concat_i\ (\boldsymbol{\lambda_i} head_i)\boldsymbol{W^O}, \qquad \text{(IV.18)}$$

where $\boldsymbol{\lambda_i}$ is a learnable parameter offering the capability to the neural network to learn more effective interactions between attention heads.

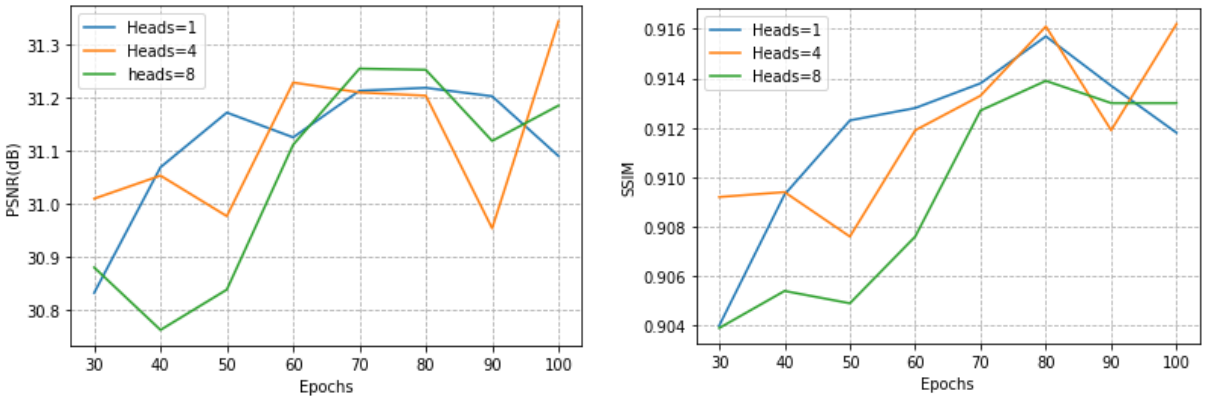
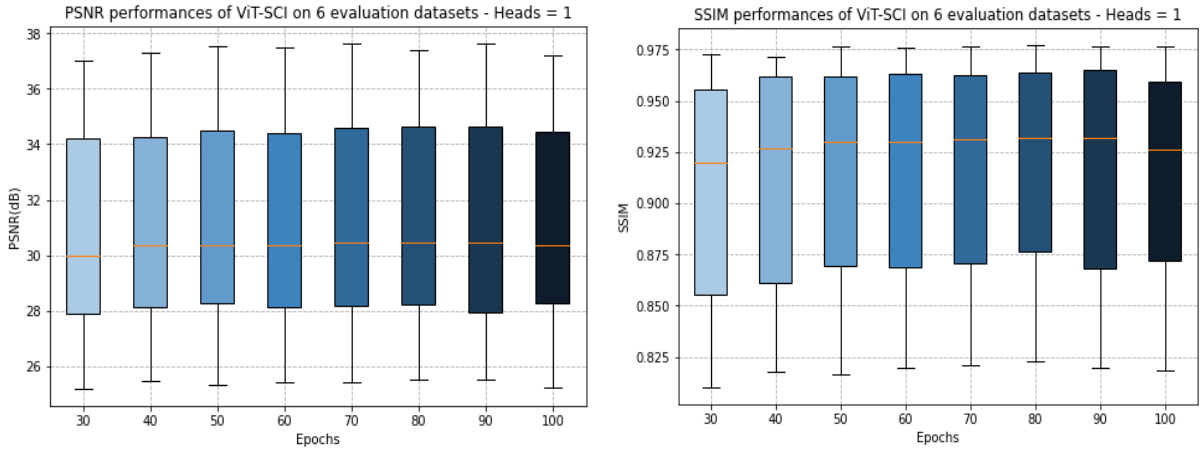


Figure IV.8: Ablation study on the effect of the number of attention heads: the average quality performance (Left: in terms of PSNR; Right: in terms of SSIM) on 6 test datasets.



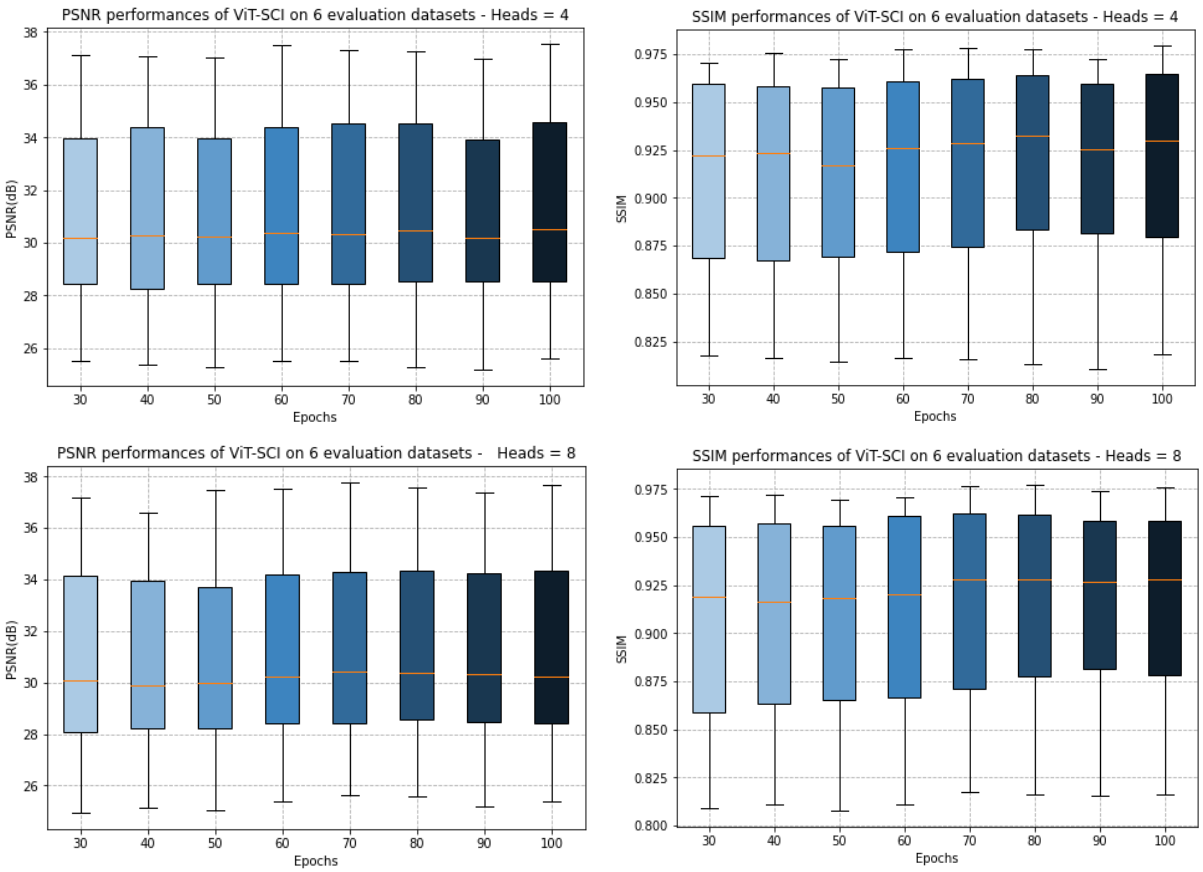


Figure IV.9: Ablation study on the effect of the number of attention heads: Box plots are used to visually show the distribution of PSNR and SSIM data and their skewness on 6 test datasets every 10 epochs, from 10 epochs to 100 epochs in the training process.

IV.5.6.4. Number of extraction blocks

The extraction network is important to extract the main features of input frames. Thus, we trained ViT-SCI with different numbers of extraction blocks to evaluate their impact on performances.

As reported in Table IV.3, the computational cost increases linearly with the number of extraction blocks. Indeed, about 11 more hours is needed when increasing the extraction blocks from 1 to 5. In addition, the average of the quality figures increases also when adding more extraction blocks (Figure IV.10). However, Figure IV.11 shows that the optimal number of extraction blocks can depend on the testing dataset. On the one hand, Aerial, Kobe and Traffic perform better with 5 extraction blocks. While on the other hand, Drop, Runner and Vehicle have better reconstruction quality with only 3 extraction blocks.

Table IV.3: Ablation study on extraction blocks.

$Extract_{blocks}$	Params	PSNR (dB)	SSIM	Training Time (s)
1	24.963.201	30.571	0.9027	388767
3	25.110.913	30.9192	0.907	407600
5	25.258.625	31.1859	0.913	425100

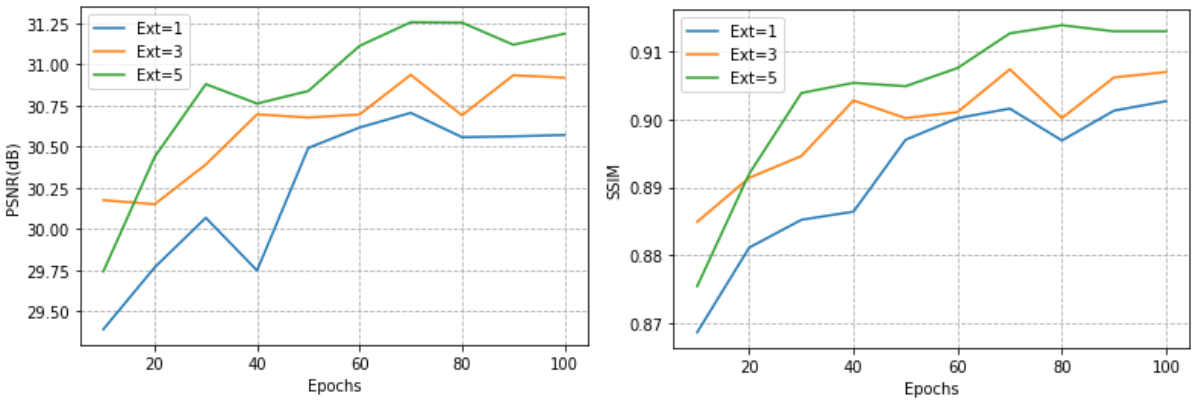
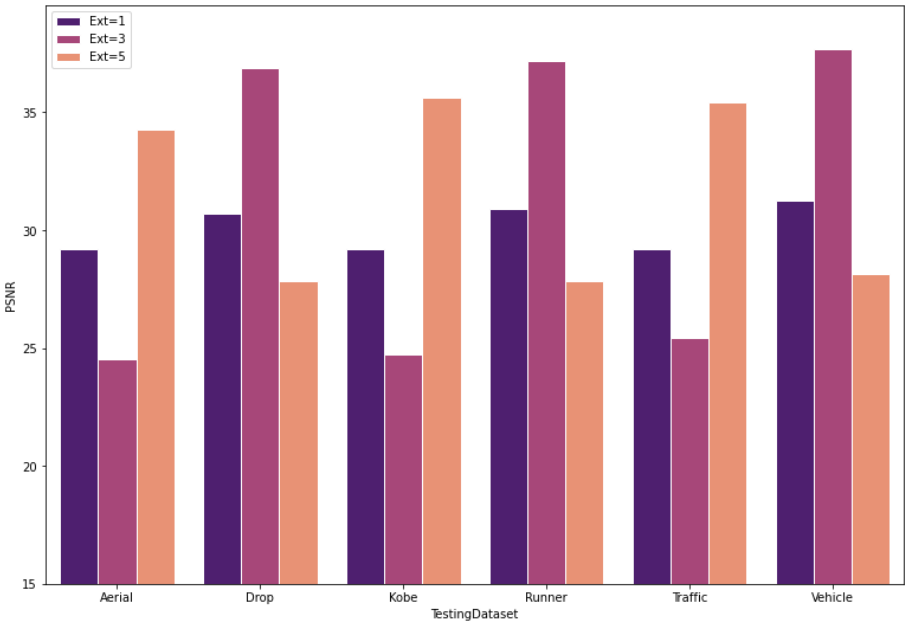


Figure IV.10: Ablation study on the effect of the number of extraction blocks: the average quality performances (Left: in terms of PSNR; Right: in terms of SSIM) on 6 testing datasets.



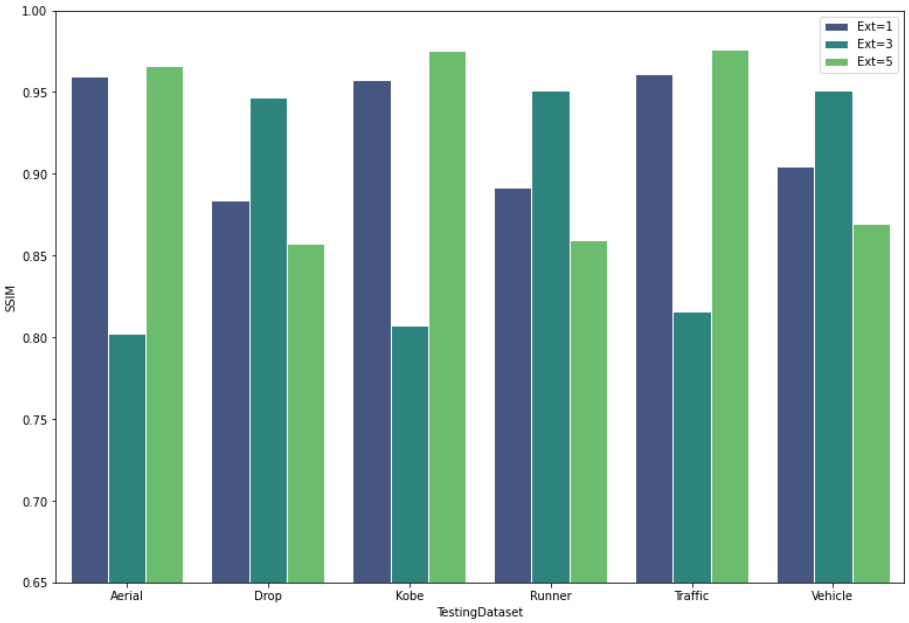


Figure IV.11: Ablation study on the effect of the number of extraction blocks: Bar plots are used to visually show PSNR (upper plot) and SSIM (lower plot) results represented with rectangular bars that are proportional to their values for the 6 test datasets: Aerial, Drop, Kobe, Traffic and Vehicle.

IV.5.6.5. Number of reconstruction blocks

The reconstruction module can also significantly impact the quality performances of ViT-SCI. Therefore, finding an optimal trade-off between the computational cost and the number of reconstruction blocks can be very challenging. So, we trained our model on 3 different numbers of reconstruction blocks and we compared our model's quality performances. It is obvious that the training time increases when increasing the number of reconstruction blocks since about 1.000.000 more model parameters must be learned when adding 10 reconstruction blocks. The average PSNR and SSIM results on 6 test sets, presented in Figure IV.12, show ViT-SCI performs well with 30 reconstruction blocks. However, when we study each dataset separately in Figure IV.13, we notice that Aerial, Kobe and Traffic need 30 reconstruction blocks for better PSNR and SSIM performances while Drop, Runner and Vehicle are well reconstructed with only 20 reconstruction blocks (Table IV.4).

Table IV.4: Ablation study on reconstruction blocks.

Rec_{blocks}	Params	PSNR (dB)	SSIM	Training Time (s)
10	23.781.505	30.7795	0.908	258528
20	24.520.065	29.7892	0.8974	342588
30	25.258.625	31.1859	0.913	425100

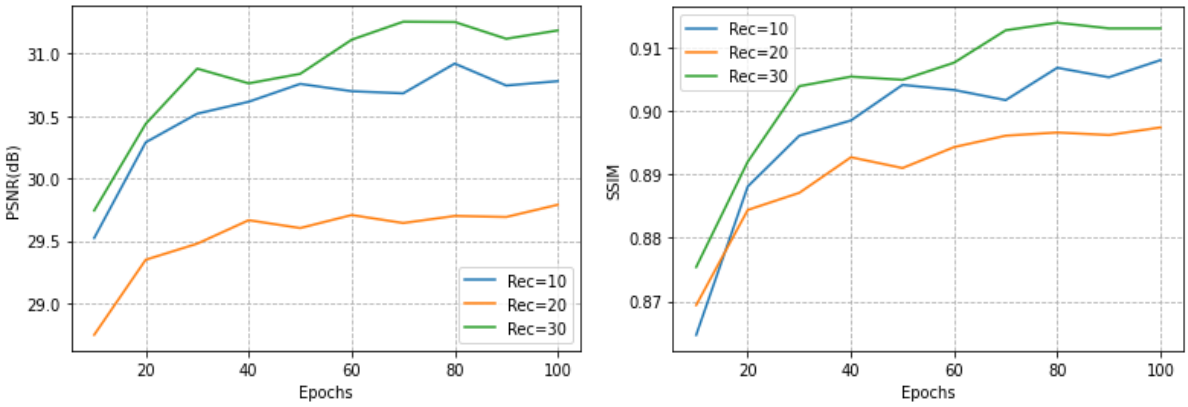


Figure IV.12: Ablation study on the effect of the number of reconstruction blocks: the average quality performances (Left: in terms of PSNR; Right: in terms of SSIM) on 6 testing datasets.

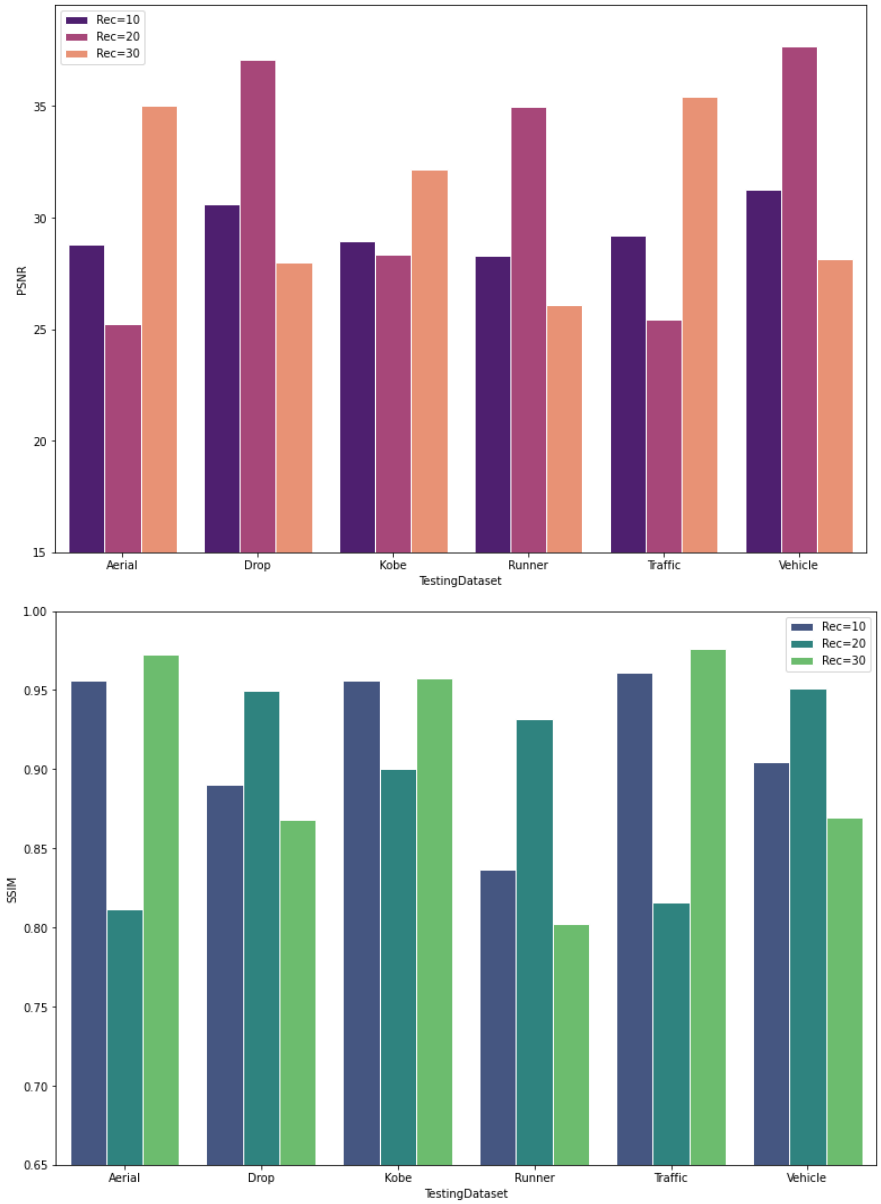


Figure IV.13: Ablation study on the effect of the number of reconstruction blocks: Bar plots are used to visually show PSNR (upper plot) and SSIM (lower plot) results represented with rectangular bars that are proportional to their values for the 6 test datasets: Aerial, Drop, Kobe, Traffic and Vehicle.

IV.5.6.6. Number of ST-ConvMHA Attention layers or depths

To explore the impact of the number of ST-ConvMHA attention layers on the performances of ViT-SCI, we trained our model with different attention layers or depths. Each layer has 8 attention heads.

Table IV.5 shows that the number of learnable parameters increases linearly when increasing the number of the encoder layers and the training becomes computationally heavier. This ablation study aims to find the smallest number of attention layers that gives better quality performances to ensure the trade-off between the output quality and the computational cost. Figure IV.14 shows that 2 attention layers performs well, outperforming the same model configuration but with 1 and 4 attention layers on the average reconstruction quality on the 6 testing datasets. However, we notice from Figure IV.15 that these better-quality performances are valid on Drop, Runner and Vehicle. For Aerial, Kobe and Traffic, 4 attention layers are needed for better reconstruction quality. These results prove that for some datasets deeper is better but it is not always the case for every dataset. We notice also, from Table IV.5 and Figure IV.14 and Figure IV.15, that the difference in performance is very small because we train our model on very short videos of 8 frames so we believe that with larger datasets the difference can be more noticeable.

Table IV.5: Ablation study on Transformer depth or the number of ST-ConvMHA attention layers.

Depth	Params	PSNR (dB)	SSIM	Training Time (s)
1	8.343.617	31.4758	0.9187	327614
2	13.981.953	31.5969	0.9218	355962
4	25.258.625	31.1859	0.913	425100

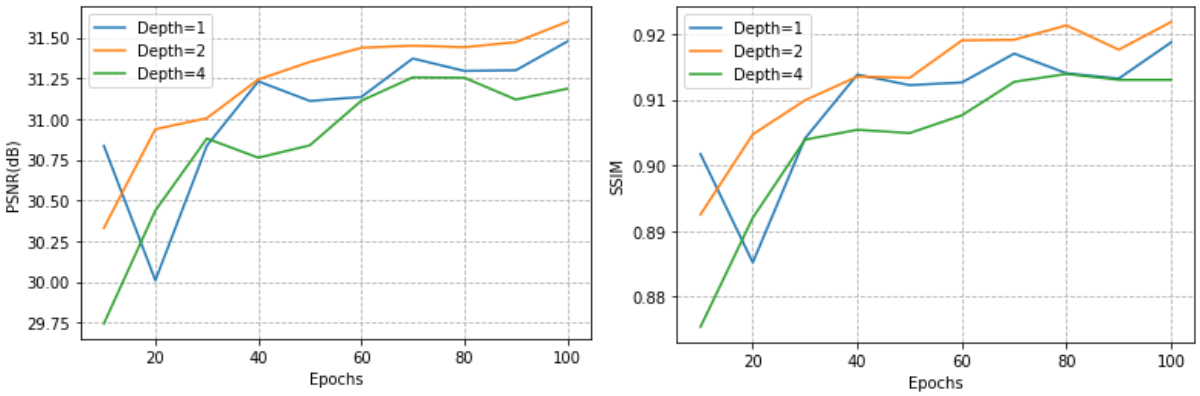


Figure IV.14: Ablation study on the effect of the number of ST-ConvMHA layers: the average quality performances (Left: in terms of PSNR; Right: in terms of SSIM) on 6 testing datasets.

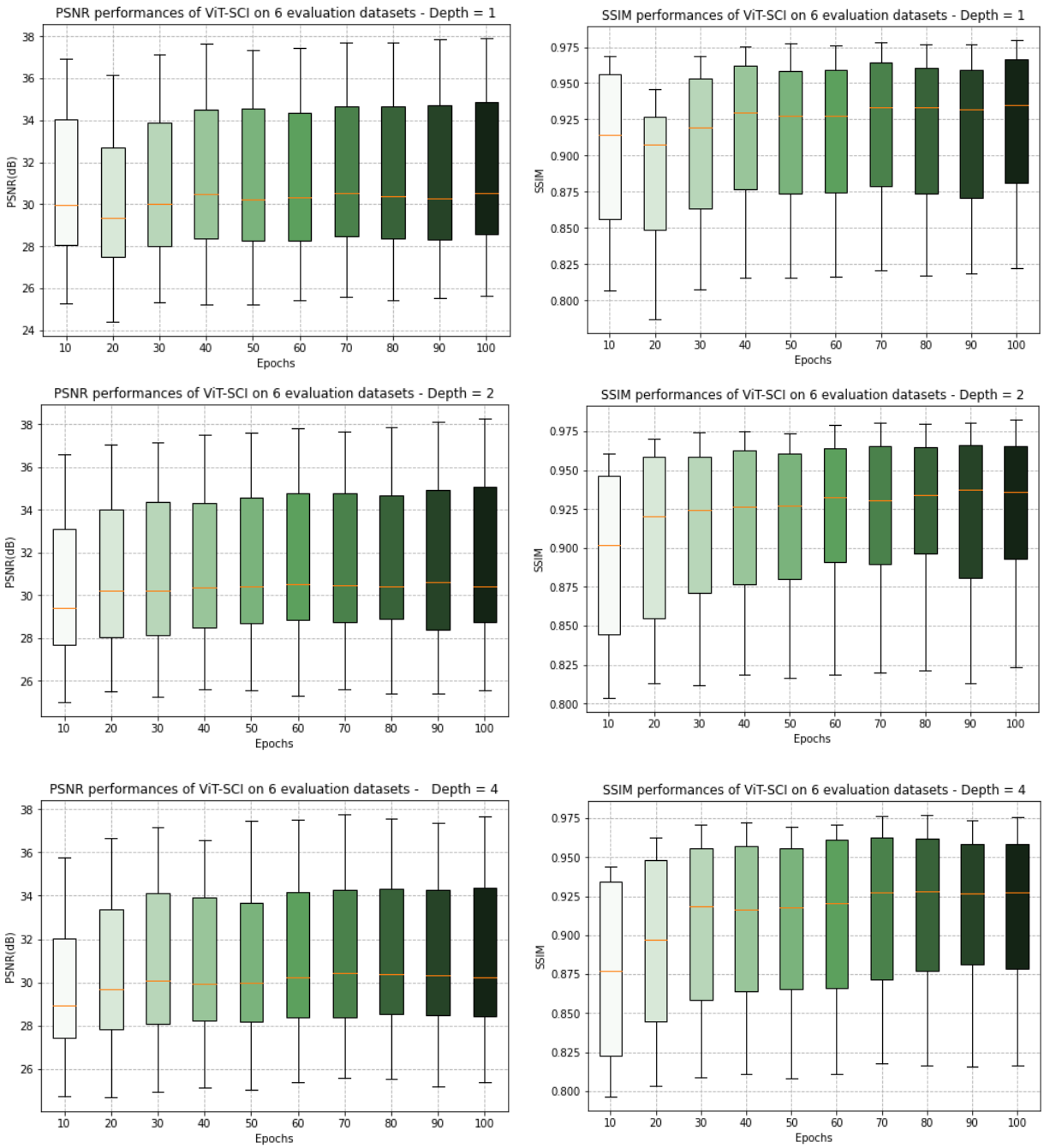


Figure IV.15: Ablation study on the effect of the number of ST-ConvMHA layers: Box plots are used to visually show the distribution of PSNR and SSIM data and their skewness on 6 test datasets evry 10 epochs, from 10 epochs to 100 epochs in the training process.

IV.5.7. Main simulation results

After confirming our design choices by means of the ablation study, we perform experiments to compare our proposed ViT-SCI algorithm with the state-of-the-art approaches on video compressive sensing. The quantitative results are summarized in Table IV.6, from which we compared the reconstruction quality of different reconstruction models and their recovery speed. For PSNR and recovery speed measured, our ViT-SCI achieves the best results among the video reconstruction methods with good SSIM results. On Arial dataset, ViT-SCI slightly outperforms BIRNAT in terms of the reconstruction quality (+1.49% and +0.5% for PSNR and SSIM, respectively) and largely outperforms GAP-TV (+32.23% and +10.19% for PSNR and

SSIM, respectively) and DeSCI (+28.33% and +8.37% for PSNR and SSIM, respectively) on the same metrics. On Kobe dataset, a limited improvement is noticed over DeSCI (+0.55%) in terms of PSNR. DeSCI performs better in terms of SSIM on Kobe dataset. The quantitative results prove the efficiency of our proposed approach, based on an attention mechanism, on complex background datasets. Table IV.6, also shows that DeSCI has better PSNR and SSIM performances on Drop, Runner and Traffic datasets over our Transformer based approach. These results can be justified because our training dataset rarely includes high speed motions. So, our model is not well trained (BIRNAT also) to reconstruct video scenes with very high-speed motions.

Table IV.6: Average PSNR (dB), SSIM and run time (in sec) per measurement for different approaches on 6 evaluation datasets. Best results are in bold, second best results are in gray.

Algorithms	Aerial	Drop	Kobe	Runner	Traffic	Vehicle	Average
GAP-TV	22.09 0.8719 8.0	27.73 0.9141 8.0	25.74 0.7909 8.1	31.29 0.9177 8.1	24.17 0.7515 8.3	24.72 0.8700 8.2	25.95 0.8526 8.12
DeSCI	22.76 0.8866 6168.2	**36.51** **0.9840** 6336.9	31.08 **0.9278** 6396.5	**38.48** **0.9609** 6331.5	**31.59** **0.9138** 6215.3	26.05 **0.9140** 6258.8	31.07 **0.9311** 6284.5
BIRNAT	28.74 0.9560 0.1050	32.77 0.9626 0.1097	28.96 0.8594 0.1056	35.41 0.9337 0.1057	26.49 0.8199 0.1087	**28.23** 0.9019 0.1132	30.10 0.9056 0.1079
Ours	**29.21** **0.9608** **0.0092**	35.40 0.9759 **0.0090**	**31.25** 0.9047 **0.0089**	37.67 0.9509 **0.0079**	28.15 0.8696 **0.0089**	25.40 0.8161 **0.0080**	**31.18** 0.9130 **0.0086**

In Figure IV.16, we show the qualitative results of our ViT-SCI compared with the-state-of-the-art. Our ViT-SCI could synthesize finer details and clearer edges on the six evaluation datasets which confirm the quantitative results and illustrate the effectiveness of the ST-ConvMHA module on the reconstruction process. Further, considering a real-time application, the most interesting performances of the experimental results remains those of the recovery time. Our algorithm is able to reconstruct a video scene of size $8 \times 64 \times 64$ in about one centisecond which is faster than BIRNAT by 12 times and much faster than the leader of the optmization based methods DeSCI by $730 \times 10+3$ times. Specifically, ViT-SCI can achieve good results in only 8ms. So, it is able to perform real-time reconstruction of up to 125 measurements per second.

Both the quantitative and qualitative results prove the ViT-SCI can be used as a reconstruction model in a video compressive sensing framework in real-time applications because of the good quality performances and especially the excellent recovery time.

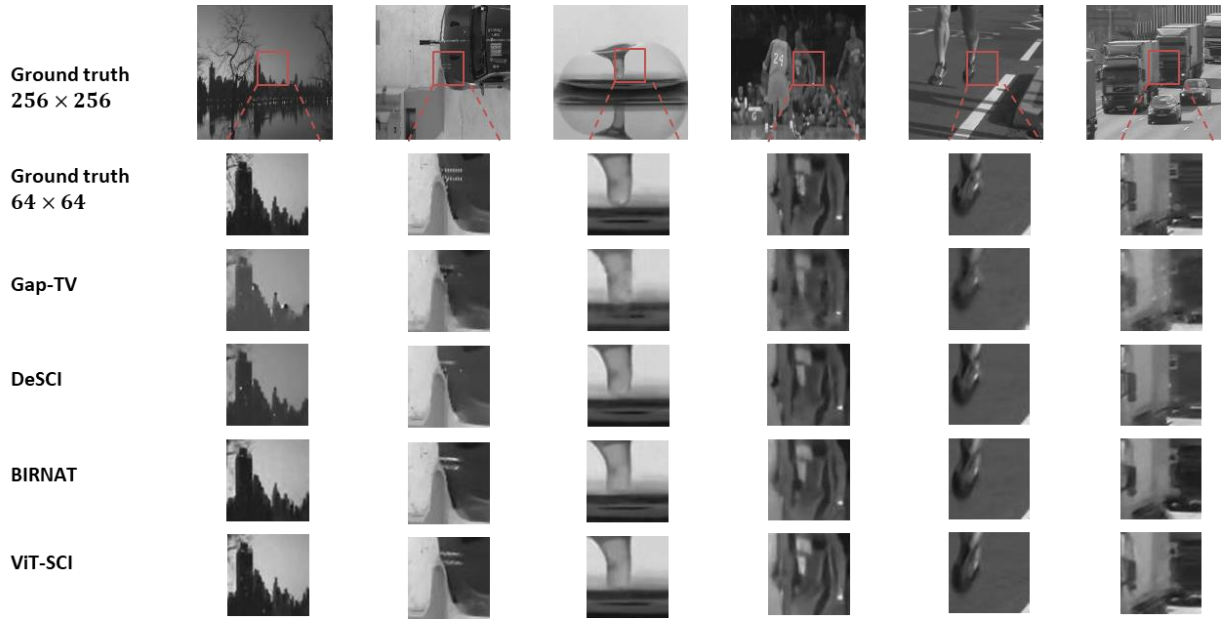


Figure IV.16: Reconstructed frames of GAP-TV, DeSCI, E2E-CNN, BIRNAT and ViTSCI on six simulated video SCI datasets.

IV.5.8. Discussion

As supported by our ablation study, we want to highlight that optimizing the hyperparameters of our proposed architecture is non-trivial and strongly depends on the dynamics and information content of the input videos. Owing to limitations in computational resources, we could not achieve a fully satisfying optimization of the hyperparameters. Furthermore, the videos size had to be restricted to afford the training process. These limitations prevented our algorithm from reaching its best potential. Future researches should tackle these limitations by designing a memory optimized architecture. Finally, more computational resources must be provided when training Deep Learning-based models. Otherwise, training several experiments would take several days, even months, on limited spatial dimensional datasets.

IV.6. Conclusion

Designing efficient video compressive sensing reconstruction algorithms has been very challenging in inverse problems. Inspired by recent advances in Deep learning and motivated by the huge success of Transformer-based architectures in NLP, we proposed the first video SCI reconstruction algorithm built upon Transformers. in this model, the recovery approach is viewed as an end-to-end decoding task. The proposed approach, trained on DAVIS dataset, achieves state-of-the-art quality performance on 6 different simulation datasets. Also, it is much faster than all existing approaches since it is able to perform real-time acquisition and reconstruction of up to 125 measurements per second. A complete ablation study is provided to justify the choice of some hyperparameters. We strongly believe that our algorithm will pave the way for more research work on video compressive sensing based on recent advances in Deep Learning. Also, we assume that ViT-SCI is now ready to be widely exploited in energy-efficient real-time applications.

References

[IV.1] Yuan, Xin, David J. Brady, and Aggelos K. Katsaggelos. “Snapshot compressive imaging: Theory, algorithms, and applications.” *IEEE Signal Processing Magazine 38*, no. 2 (2021): 65-88.

[IV.2] Llull, Patrick, Xuejun Liao, Xin Yuan, Jianbo Yang, David Kittle, Lawrence Carin, Guillermo Sapiro, and David J. Brady. “Coded aperture compressive temporal imaging.” *Optics express 21*, no. 9 (2013): 10526-10545.

[IV.3] Koller, Roman, Lukas Schmid, Nathan Matsuda, Thomas Niederberger, Leonidas Spinoulas, Oliver Cossairt, Guido Schuster, and Aggelos K. Katsaggelos. “High spatio-temporal resolution video with compressed sensing.” *Optics express 23*, no. 12 (2015): 15992-16007.

[IV.4] Reddy, Dikpal, Ashok Veeraraghavan, and Rama Chellappa. “P2C2: Programmable pixel compressive camera for high speed imaging.” In *CVPR* 2011, pp. 329-336. IEEE, 2011.

[IV.5] Sun, Yangyang, Xin Yuan, and Shuo Pang. “Compressive high-speed stereo imaging.” *Optics express 25*, no. 15 (2017): 18182-18190.

[IV.6] Jalali, Shirin, and Xin Yuan. “Snapshot compressed sensing: Performance bounds and algorithms.” *IEEE Transactions on Information Theory 65*, no. 12 (2019): 8005-8024.

[IV.7] Yuan, Xin. “Generalized alternating projection based total variation minimization for compressive sensing.” In 2016 *IEEE International Conference on Image Processing (ICIP)*, pp. 2539-2543. IEEE, 2016.

[IV.8] Yang, Jianbo, Xin Yuan, Xuejun Liao, Patrick Llull, David J. Brady, Guillermo Sapiro, and Lawrence Carin. “Video compressive sensing using Gaussian mixture models.” *IEEE Transactions on Image Processing 23*, no. 11 (2014): 4863-4878.

[IV.9] Liu, Yang, Xin Yuan, Jinli Suo, David J. Brady, and Qionghai Dai. “Rank minimization for snapshot compressive imaging.” *IEEE transactions on pattern analysis and machine intelligence 41*, no. 12 (2018): 2990-3006.

[IV.10] Yuan, Xin, Yang Liu, Jinli Suo, and Qionghai Dai. “Plug-and-play algorithms for large-scale snapshot compressive imaging.” In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1447-1457. 2020.

[IV.11] Iliadis, Michael, Leonidas Spinoulas, and Aggelos K. Katsaggelos. “Deep fully-connected networks for video compressive sensing.” *Digital Signal Processing 72* (2018): 9-18.

[IV.12] Qiao, Mu, Ziyi Meng, Jiawei Ma, and Xin Yuan. “Deep learning for video compressive sensing.” *Apl Photonics 5*, no. 3 (2020): 030801.

[IV.13] Xu, Kai, and Fengbo Ren. “CSVideoNet: A real-time end-to-end learning framework for high-frame-rate video compressive sensing.” In 2018 *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1680-1688. IEEE, 2018.

[IV.14] Cheng, Ziheng, Ruiying Lu, Zhengjue Wang, Hao Zhang, Bo Chen, Ziyi Meng, and Xin Yuan. “BIRNAT: Bidirectional recurrent neural networks with adversarial training for video snapshot compressive imaging.” In *European Conference on Computer Vision*, pp. 258-275. Springer, Cham, 2020.

[IV.15] Kulkarni, Kuldeep, Suhas Lohit, Pavan Turaga, Ronan Kerviche, and Amit Ashok. “Reconnet: Non-iterative reconstruction of images from compressively sensed measurements.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 449-458. 2016.

[IV.16] Meng, Ziyi, Shirin Jalali, and Xin Yuan. “Gap-net for snapshot compressive imaging.” arXiv preprint arXiv:2012.08364 (2020).

[IV.17] Yuan, Xin, and Yunchen Pu. “Parallel lensless compressive imaging via deep convolutional neural networks.” *Optics express 26*, no. 2 (2018): 1962-1977.

[IV.18] Ma, Jiawei, Xiao-Yang Liu, Zheng Shou, and Xin Yuan. “Deep tensor admm-net for snapshot compressive imaging.” In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10223-10232. 2019.

[IV.19] Iliadis, Michael, Leonidas Spinoulas, and Aggelos K. Katsaggelos. “Deepbinary-mask: Learning a binary mask for video compressive sensing.” *arXiv* preprint arXiv:1607.03343 (2016).

[IV.20] Cheng, Ziheng, Bo Chen, Guanliang Liu, Hao Zhang, Ruiying Lu, Zhengjue Wang, and Xin Yuan. “Memory-efficient network for large-scale video compressive sensing.” In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16246-16255. 2021.

[IV.21] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention is all you need.” *Advances in neural information processing systems 30* (2017).

[IV.22] Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. “Bert: Pre-training of deep bidirectional transformers for language understanding.” *arXiv* preprint arXiv:1810.04805 (2018).

[IV.23] Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. “Improving language understanding by generative pre-training.” (2018).

[IV.24] Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani et al. “An image is worth 16x16 words: Transformers for image recognition at scale.” *arXiv* preprint arXiv:2010.11929 (2020).

[IV.25] Wang, Xiaolong, Ross Girshick, Abhinav Gupta, and Kaiming He. “Non-local neural networks.” In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7794-7803. 2018.

[IV.26] Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. “End-to-end object detection with transformers.” In *European conference on computer vision*, pp. 213-229. Springer, Cham, 2020.

[IV.27] Wang, Yuqing, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. “End-to-end video instance segmentation with transformers.” In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8741-8750. 2021.

[IV.28] Liu, Rui, Hanming Deng, Yangyi Huang, Xiaoyu Shi, Lewei Lu, Wenxiu Sun, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. “Fuseformer: Fusing fine-grained information in transformers for video inpainting.” In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14040-14049. 2021.

[IV.29] Liu, Ze, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. “Video swin transformer.” *arXiv* preprint arXiv:2106.13230 (2021).

[IV.30] Saideni, Wael, David Helbert, Fabien Courreges, and Jean-Pierre Cances. “An Overview on Deep Learning Techniques for Video Compressive Sensing.” *Applied Sciences 12*, no. 5 (2022): 2734.

[IV.31] Hochreiter, Sepp. “The vanishing gradient problem during learning recurrent neural nets and problem solutions.” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6*, no. 02 (1998): 107-116.

[IV.32] Li, Yawei, Kai Zhang, Jiezhang Cao, Radu Timofte, and Luc Van Gool. “Localvit: Bringing locality to vision transformers.” *arXiv* preprint arXiv:2104.05707 (2021).

[IV.33] Wang, Zhengjue, Hao Zhang, Ziheng Cheng, Bo Chen, and Xin Yuan. “Metasci: Scalable and adaptive reconstruction for video compressive sensing.” In Proceedings of the *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2083-2092. 2021.

[IV.34] Cao, Jiezhang, Yawei Li, Kai Zhang, and Luc Van Gool. “Video super-resolution transformer.” *arXiv* preprint arXiv:2106.06847 (2021).

[IV.35] Wu, Haiping, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. “Cvt: Introducing convolutions to vision transformers.” In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22-31. 2021.

[IV.36] Mou, Chong, Jian Zhang, Xiaopeng Fan, Hangfan Liu, and Ronggang Wang. “COLA-Net: Collaborative attention network for image restoration.” *arXiv* preprint arXiv:2103.05961 (2021).

[IV.37] Pont-Tuset, Jordi, Federico Perazzi, Sergi Caelles, Pablo Arbelàez, Alex Sorkine-Hornung, and Luc Van Gool. “The 2017 davis challenge on video object segmentation.” *arXiv* preprint arXiv:1704.00675 (2017).

[IV.38] Touvron, Hugo, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herv´e J´egou. “Training data-efficient image transformers distillation through attention.” In *International Conference on Machine Learning*, pp. 10347-10357. PMLR, 2021.

[IV.39] Hore, Alain, and Djemel Ziou. “Image quality metrics: PSNR vs. SSIM.” In 2010 *20th international conference on pattern recognition*, pp. 2366-2369. IEEE, 2010.

[IV.40] Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen et al. “Pytorch: An imperative style, high-performance deep learning library.” *Advances in neural information processing systems 32* (2019).

[IV.41] Kingma, Diederik P., and Jimmy Ba. “Adam: A method for stochastic optimization.” *arXiv* preprint arXiv:1412.6980 (2014).

[IV.42] Michel, Paul, Omer Levy, and Graham Neubig. “Are sixteen heads really better than one?.” *Advances in neural information processing systems 32* (2019).

[IV.43] Voita, Elena, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. “Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned.” *arXiv* preprint arXiv:1905.09418 (2019).

Conclusion and Future Work

A. Conclusion

Data today is generated at exponentially growing rates which creates unbearable demands on the sensing, storage, and processing devices. Indeed, thousands of data centers are built worldwide to store this huge amount of data which leads to extremely high power that is consumed on acquiring and processing.

Indeed, IoT based applications in smart cities require a considerable amount of heterogeneous intelligent tools and devices to capture, communicate, and visualize environmental data in order to monitor urban conditions and empower several services. The large number of intelligent devices is creating a huge amount of redundant data that would be the origin of an avoidable network congestion which would degrade the overall network performances. Another challenge faced by research nowadays is data transmission using limited computational and storage resources.

Therefore, as long as we generate more data, there is an urgent need for novel data acquisition and processing concepts such as compressive sensing. Furthermore, we propose in this thesis to study and explore Deep Learning-based approaches in a Video Compressive Sensing context to reduce the amount of data gathered while maintaining the quality performances of the collected videos, thus enhancing the overall system potentials.

This dissertation started with describing the foremost applications of Compressive Sensing and the main Deep Learning architectures exploited in the different video compressive sensing approaches studied and developed in this work.

Chapter 2 provides an overview on the theory of Compressive Sensing, starting from the fundamental model of data acquisition to the standards that have to respected while implementing the sensing matrix, as well as listing the most recent Video Compressive Sensing frameworks used in the literature. Furthermore, this chapter compare the qualitative and quantitative performances of these recent VCS algorithms to provide a clear comparative study to researchers and businesses. Their choice will obviously depend on the specifications of their various applications.

In Chapter 3, we design a novel video compressive sensing framework based on a video prediction paradigm. For that, we started by designing and implementing a novel video prediction called “Robust Spatio-Temporal Colvolutional Long-Short Term Memory” (Robust-ST-ConvLSTM) which is suitable for with data with spatial and temporal correlations such as video sequences. Experimental results on two video datasets for random digits and human motions, show the advantages of the presented algorithm. We end up this chapter by a complete discussion on the main limitations of the designed VCS framework and the proposed research direction that will be developed in Chapter 4.

In Chapter 4, the Deep Learning paradigm for Video Compressive Sensing was extended to a novel VCS algorithm built upon Transformers called ViT-SCI. We introduced the first VCS framework based on Transformers and explored the advantages of a convolutional multi-head attention mechanism. We started this research work by presenting the main benefits of Transformers over the well-known convolutional and recurrent reconstruction models. Then, we detailed the core architecture of our proposed model. The proposed approach, trained on DAVIS dataset, achieves state-of-the-art quality performance on 6 different simulation datasets.

Another fascinating result about ViTSCI is its running time. It is much faster than all existing approaches which prove that it could be deployed for real-time applications

B. Future Work

The Video Compressive Sensing architecture designed and implemented in this research work have a significant number of hyper-parameters that can be fine-tuned to improve their performances. These performances can be enhanced by further theoretical studies on the different parameters or extended ablation studies which definitely will need more computational resources.

- **Hybrid Systems and Edge Computing**

Obviously, there is a tremendous intellectual progress in compressive sensing and sparse representation systems. Therefore, many mathematical concepts such as probability theory, convex optimization and reconstruction algorithms become an essential toolbox for many researchers and engineers to design and develop real-world applications.

Hence, in the future, we are going to talk about designing hybrid systems that integrate hardware and software, where these two systems are implemented simultaneously from the beginning using the mathematical concepts described above.

Also, a new research direction has appeared with deploying a video compressive sensing system with edge computing to optimize the memory storage and bandwidth [1]. In addition, theoretical studies on detection algorithms directly from the snapshot compressed measurement have already started [2]. Finally, we can say that compressive sensing allows us to think about data, complexity, algorithms, and hardware at the same time. In a nutshell, the answer will be an algorithm with better flexibility, accuracy, and speed.

- **Reinforcement Learning based VCS**

Reinforcement Learning (RL) [3] is an online machine learning algorithm originally designed to develop behavioral policies by rewarding desired behaviors and penalizing undesired ones. In this case, the model is trained from its own interactions with the environment not from historical data. Indeed, we can potentially work on an RL based model for VCS to adapt the compression ratio (CR). In our research work, we reconstruct a fixed number of frames, captured by a low-speed camera, from one single measurement frame. However, we can adapt the number of the reconstructed frames for different scenes. In fact, the compression ratio will be determined by an RL strategy. For that, an object detection algorithm can be used to increase or decrease the CR. Then, the detection rate and some image quality performance metrics of the reconstructed frames will be sent to the RL algorithm to adjust the CR for different scenes. The idea is to adapt the CR with scenes where we have many moving objects to detect and others where we have static backgrounds. Also, we adapt the CR with fast and slow scenes i.e. with the movement velocities of different moving objects.

- **Efficient Transformers based reconstruction model**

We already mentioned in this thesis that Transformers have become an essential piece in modern deep learning architectures. However, to train these models on high dimensional data, huge memory resources are required. Indeed, in our research, in particular in Chapter 4, we were forced to evaluate the Transformer-based architecture on limited spatial and temporal dimensions of the training video dataset. This may affect the performances of the model.

Therefore, the most obvious direction for further research is making improvements around computational and memory efficiency. Many Transformer variants have been proposed recently, such as X-former models [4]-[5], to enhance the memory usage at the training phase.

These recent works can be an inspiration to design a novel memory efficient Transformer able to reconstruct videos in VCS contexts.

- **Applying VCS in a Massive MIMO transmission problem**

One of the most significant problems in wireless telecommunication systems is multipath propagation affecting different wireless channels. To overcome this issue, Multiple Input and Multiple Output (MIMO) [6] is proposed as effective channels that lead to notable increase in link range and data throughput. These features enable to transmit high dimensional data, i.e. videos, with good reliability.

Therefore, another interesting perspective is the design of a new strategy to accurately transmit compressed video information through MIMO channels using multiple antenna techniques. This approach will aim to optimize both physical and application layers. In fact, a video may be source coded by a VCS paradigm where videos are represented by numerical measurement vectors. Then, these vectors would be transmitted simultaneously over the available antennas. The next step would be to design a power allocation strategy integrated into the transmission scheme. The energy would be allocated to the antenna in proportion to the amount of transmitted data.

References

[1] Liu, S.; Liu, L.; Tang, J.; Yu, B.; Wang, Y.; Shi, W. Edge computing for autonomous driving: Opportunities and challenges. *Proc. IEEE 2019, 107*, 1697–1716. doi:10.1109/JPROC.2019.2915983.

[2] Lu, S.; Yuan, X.; Shi, W. An integrated framework for compressive imaging processing on CAVs. In *Proceedings of the ACM/IEEE Symposium on Edge Computing (SEC)*, San Jose, CA, USA, 12–14 November 2020.

[3] Kaelbling, Leslie Pack, Michael L. Littman, and Andrew W. Moore. "Reinforcement learning: A survey." *Journal of artificial intelligence research 4* (1996): 237- 285.

[4] Kitaev, Nikita, Lukasz Kaiser, and Anselm Levskaya. "Reformer: The efficient transformer." *arXiv* preprint arXiv:2001.04451 (2020).

[5] Beltagy, Iz, Matthew E. Peters, and Arman Cohan. "Longformer: The long-document transformer." *arXiv* preprint arXiv:2004.05150 (2020).

[6] Yang, Jian, and Sumit Roy. "On joint transmitter and receiver optimization for multiple-input-multiple-output (MIMO) transmission systems." *IEEE Transactions on Communications 42*, no. 12 (1994): 3221-3231.

Publications

- **International conferences:**

Saideni, Wael, Fabien Courreges, David Helbert, and Jean Pierre Cances. “End-to-End Video Snapshot Compressive Imaging using Video Transformers.” In *International Conference on Image Processing Theory*, Tools and Applications. Salzburg 2022

Saideni, Wael, David Helbert, Fabien Courreges, and Jean Pierre Cances. “A Novel Video Prediction Algorithm based on Robust Spatiotemporal Convolutional Long Short Term Memory.” In *International Congress on Information and Communication Technology*. London 2022

- **Journal papers:**

Saideni, Wael, Fabien Courreges, David Helbert, and Jean-Pierre Cances. “ViT-SCI: Video Transformer is all you need for Video Compressive Sensing” under consideration in Computer Vision and Image Understanding

Saideni, Wael, Fabien Courreges, David Helbert, and Jean-Pierre Cances. “Robust Spatiotemporal Convolutional Long Short-Term Memory Algorithm for Video Prediction.” under consideration in Signal Processing: Image Communication Journal

Saideni, Wael, David Helbert, Fabien Courreges, and Jean-Pierre Cances. “An Overview on Deep Learning Techniques for Video Compressive Sensing.” *Applied Sciences 12*, no. 5 (2022): 2734

Résumé :

La technique de compressive sensing joue un rôle important dans le traitement des données vu que l'acquisition et la compression se font simultanément grâce à un processus de prise de mesures. Cette technique optimise les capacités de stockage des systèmes ainsi que la vitesse et le coût d'acquisition. Récemment, cette technique est devenue de plus en plus utilisée grâce à l'optimisation des algorithmes de reconstruction en utilisant les architectures du Deep Learning. L'objectif principal de cette thèse est de tirer profit des architectures de Deep Learning pour optimiser la technique de compressive sensing en l'appliquant sur des signaux vidéo et par la suite optimiser l'acquisition, la transmission et la reconstruction des vidéos dans les systèmes numériques modernes. Ainsi, la stratégie adoptée au cours de ces travaux de recherche consiste à commencer par établir une étude comparative sur les approches de vidéo compressive sensing (VCS) basées sur le Deep Learning en évaluant la qualité et la vitesse de reconstruction ainsi que les différentes architectures. Puis, deux environnements de VCS ont été conçus : le premier se base sur la prédiction des frames vidéo en implémentant une approche basée sur un nouveau réseau récurrent et le deuxième exploite les dernières performances réalisées avec les Transformers et le mécanisme d'attention. Alors, la démarche adoptée dans ces deux approches repose sur une analyse de l'état de l'art suivie d'une explication de chaque architecture et une validation expérimentale. Les différentes contraintes rencontrées au cours de ces travaux sont discutées et des solutions appropriées sont proposées.

Mots-clés : Acquisition Comprimée, Apprentissage profond, Traitement de vidéo, Vision par ordinateur

Abstract:

Compressive Sensing, commonly used to approximate solutions for underdetermined linear systems of equations, is gaining a lot of attention as an efficient acquisition and compression paradigm that combines nonlinear reconstruction algorithms and random sampling on sparse basis. It enables to optimize the storage capacity of the wireless systems as well as the speed and cost of acquisition. Recently, Deep Learning architectures have frequently been exploited to optimize the reconstruction phase. The main objective of this thesis is to take advantage of Deep Learning architectures to optimize the compressive sensing technique by applying it on video signals and subsequently optimize the acquisition, transmission, and reconstruction of videos in modern digital systems. Therefore, the strategy adopted during this research work consists in establishing a comparative study on video compressive sensing (VCS) approaches based on Deep Learning by evaluating the quality and the speed of reconstruction as well as the different architectures. Then, two VCS environments have been designed: the first one is based on the prediction of video frames by implementing an approach based on a new recurrent network and the second one exploits the latest performances achieved with the Transformers and the attention mechanism. So, the approach adopted is based on a state-of-the-art analysis followed by an explanation of each architecture and an experimental validation. The different constraints encountered during this work are discussed and appropriate solutions are proposed.

Keywords: Compressive Sensing, Deep Learning, Video Processing, Computer Vision