UNIVERSITY OF CALIFORNIA,
IRVINE

Efficient Attention for Time Series, Image and Video Understanding

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Mathematics

by

Nhat Thanh Van Tran

Dissertation Committee:
Professor Jack Xin, Chair
Professor Knut Solna
Professor Yifeng Yu

2026

DEDICATION

To my colleagues, mentors, and family who have supported me.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

ACKNOWLEDGMENTS

I cannot adequately express how much unconditional support I received during my Ph.D. journey in Mathematics at UC Irvine. The knowledge, humor, and kindness of the people around me helped me through every stage of this journey.

First, I would like to thank Professor Jack Xin for being both my academic adviser and a mentor in life. I initially came to UCI intending to study mathematical biology, but I met Professor Xin while taking his Math 290C course. He presented one of his works on machine learning, and I immediately asked him to supervise a reading course for me. At the time, I knew nothing about machine learning, and I am grateful for his patience as he helped me take my first steps into the field. Until the final year of my Ph.D., I was uncertain about the direction of my research. However, Professor Xin has always been an advocate for efficient machine learning, and his influence inspired my interest in my current area of research. He provided the guidance, vision, and confidence that allowed me to carve my own path forward. His persistent pursuit of a mathematical understanding of deep neural networks has made valuable contributions to the field. Together, we wrote multiple papers. Beyond being my academic adviser, he supported me in Italy when I first presented my work, traveled to Banff, Canada, for the presentation of my second work, and shared a memorable conversation with me on the bus to the airport. He also helped me navigate the real world through his practical and grounded advice. Without him as my adviser, my academic work would not have been possible, and I would not have grown as much as a person.

I would like to thank Professor Knut Solna and Professor Yifeng Yu for serving on my dissertation committee. In particular, I thank Professor Solna for his courses on stochastic processes and mathematical finance, which provided my first exposure to mathematical finance. I also thank Professor Yu for inviting me to present my early research at the machine learning seminar.

The year 2020 was difficult for many of us, but it was especially difficult for new students because we were all isolated. I want to thank Anthony Zamora, Vignesh Iyer, and Yibiao Ruan for all our Zoom study sessions on algebra, real analysis, and complex analysis. I would especially like to thank John Palacios for the many long nights we spent studying for the qualifying examinations. He is one of the people with whom I can discuss anything, from ethics, politics, and computer science to food, religion, and relationships. I want to thank Stephen Robbins for our shared interests in finance and neural networks. He was a very early adopter of AI tools, and the knowledge he shared has shaped my interest in finance and helped me stay informed about the rapidly changing world of AI. I also want to thank William Black for sharing his understanding of the law and his insights into how to navigate the world.

I also want to thank my Ph.D. brothers: Yunling Zheng, for helping me begin my machine learning research when I knew nothing about the field and for his continued support, mentorship, and career advice; Zhijian Li, for helping me with my job search and welcoming me while I was interning in San Jose; Jongwon Kim, for all our conversations and for trusting me

to support him through difficult times; Elisha Dayag, for the research we conducted together on time series and medical imaging and for being my forward-deployed engineer; Ashwin Prabu, for all the thought-provoking questions; and Zhiqi (Kiara) Yu, for always being willing to take action and learn. Special thanks go to Kevin Bui for helping with the initial software setup for training neural networks and for collaborating with me on the optimization paper. I also want to thank my academic brothers from Qualcomm for their mentorship and feedback on my research and career development: Fanghui Xue, Shuai Zhang, and Jiancheng Lyu.

I want to thank the UCI Math Circle for giving me the wonderful opportunity to volunteer, learn, and grow as a person. I especially want to thank Professor Alessandra Pantano for all her guidance and collaboration. We worked together on the Math Circle throughout my six years at UCI.

Finally, I would like to thank my family - Nguyen Thao Tran, Nguyen Thanh Tran, Cuc Pham, and Theo Tran, for their unwavering support. I am especially grateful to them for taking care of me during my injury, cooking, cleaning, and handling countless daily responsibilities so that I could focus on my research.

Funding Acknowledgment: This dissertation was supported by NSF grants DMS-1952644, DMS-2151235, DMS-2219904, DMS-2309520, Faculty Endowed Fellowship and Graduate Scholar Success Fund from the University of California, Irvine.

VITA

Nhat Thanh Van Tran

EDUCATION

Doctor of Philosophy in Mathematics **2026**
University of California, Irvine *Irvine, CA*

Master of Science in Computational and Applied Mathematics **2020**
Colorado School of Mines *Golden, CO*

Bachelor of Science in Computational and Applied Mathematics **2015**
Colorado School of Mines *Golden, CO*

RESEARCH EXPERIENCE

Graduate Research Assistant **2023–2026**
University of California, Irvine *Irvine, CA*

WORK EXPERIENCE

Machine Learning Intern **Jul. 2024–Dec. 2024**
Mercedes Benz Research and Development North America *Sunnyvale, CA*

Software Developer **2015–2018**
Fast Enterprises *Denver, CO*

AWARDS

Kovalevsky Outstanding Ph.D. Thesis Award **2026**
Graduate Scholar Success Fund **2026**
Faculty Endowed Fellowship **2024-2025**
UCI Math Circle Academic Outreach Certicate **2022-2024**
UCI Math Circle Outstanding Service **2021**

REFEREED JOURNAL PUBLICATIONS

Fourier-mixed window attention for efficient and robust long sequence time-series forecasting **2025**
Frontiers in Applied Math and Statistics

A computational information criterion for particle-tracking with sparse or noisy data **2021**
Advances in Water Resources

REFEREED CONFERENCE PUBLICATIONS

SEMA: a Scalable and Efficient Mamba like Attention via Token Localization and Averaging **July 2026**
International Conference in Machine Learning

Deep Image Prior with L0 Gradient Regularizer for Image Smoothing **May 2026**
International Conference on Acoustic, Speech, and Signal Processing

FWin transformer for dengue prediction under climate and ocean influence **June 2024**
Proceeding of International Conference on Machinear Learning, Optimization, and Data Science

SOFTWARE

SEMA `https://github.com/nhatthanhtran/SEMA`
Python source code for SEMA model.

FWin `https://github.com/nhatthanhtran/FWin2023`
Python source code for FWin model.

ABSTRACT OF THE DISSERTATION

Efficient Attention for Time Series, Image and Video Understanding

By

Nhat Thanh Van Tran

Doctor of Philosophy in Mathematics

University of California, Irvine, 2026

Professor Jack Xin, Chair

In this thesis, we study efficient attention in the dominant transformer architecture from a theoretical perspective. We then use this theoretical understanding to guide the design of models for problems in time-series forecasting, image processing, and video understanding.

In Part I, we begin with the motivation for and an introduction to modern machine learning, starting with recurrent neural networks for processing sequential data and continuing to the current transformer-based approach. We discuss how image processing has progressed from relying mainly on convolutional neural networks, which respect image geometry, to the current paradigm of transformers in computer vision. We provide an overview of the chapters in this dissertation and explain how they connect to the core research question. We then present a unified theoretical analysis of the attention mechanism, laying the theoretical groundwork for the remainder of the dissertation.

In Part II, we consider applications to time-series forecasting. Chapter 3 presents the main work on developing FWin as an efficient model and comparing it with many state-of-the-art methods. In particular, we combine window attention with the Fourier transform as an efficient global mixing operator. Chapter 4 then validates the proposed FWin method on a more challenging dataset involving dengue cases in Singapore, where the data are nonstationary and quite limited. This demonstrates that FWin is robust enough to adapt to

other applications rather than only perform well on standard benchmarks with well-known properties.

In Part III, we develop a mathematically inspired efficient attention mechanism for image processing. We use arithmetic averaging as a global approximation of full attention while using window attention to mimic the local and recurrent nature of Mamba. This allows our proposed SEMA method to combine two well-known mechanisms in computer vision, retain their advantageous properties, and complement their weaknesses. We verify our design against state-of-the-art models on several challenging benchmarks. Following this success, we extend SEMA into a video model in the next chapter. We design a split space-time transformer in which SEMA processes the spatial component and standard softmax attention processes the temporal component. This design is efficient and scales to high-resolution videos.

We conclude the dissertation by discussing the limitations of the current line of research and the importance of the works presented in this thesis. We also point toward future research directions in efficient machine learning and discuss their importance for real-world applications.

Part I

Introduction

Chapter 1

Introduction: Efficient Local-Global Learning for Structured Data

1.1 Motivation

I want to begin this dissertation with a simple question: how do we design a learning model that can process a large amount of information efficiently, while still keeping enough of the information to make an accurate prediction? This question appears in many different forms. In time-series forecasting, the model needs to understand measurements across a long period of time. In an image, it needs to combine small local features such as edges and textures into a global object. In a video, it needs to understand both what appears in each frame and how the information changes across time. Even in a classical image-processing problem, the method needs to remove small details while preserving important structures such as boundaries and contours. Although these problems look different, they share a common difficulty: useful information may be local, global, or somewhere in between, and using all possible interactions is often too expensive.

Most modern learning problems begin with data pairs (X, Y), where X represents an input space and Y represents a desired output space. We assume that there is some complicated relationship $f : X \to Y$, and the goal is to construct a model $\widetilde{f}$ that approximates this relationship well. Neural networks provide a very flexible class of functions for this purpose. However, the fact that a large neural network can approximate a complicated function does not tell us how to construct the network, how much computation it requires, or what information it should preserve. In practice, the architecture matters. A good architecture does not only have enough parameters; it has a useful way to move and combine information.

For time series, language, images, and videos, the input is commonly converted into tokens. A sequence can be written as

$$x = (x_1, \ldots, x_n), \qquad x_i \in \mathbb{R}^d, \tag{1.1}$$

where n is the number of tokens and d is the hidden dimension. For time series, a token may represent a time step or a short temporal patch. For images, a token may represent a small spatial patch. For videos, the tokens have both spatial and temporal meanings. Once the input is represented in this way, the model needs to decide which tokens should communicate with each other.

The transformer answers this question by using attention, which allows each token to interact with every other token [82]. This is powerful because the model is not restricted to a fixed local neighborhood or a one-directional recurrence. Returning to a game-of-telephone example, attention does not require a message to pass from the first person to the second person, then from the second person to the third person, and so on. Instead, everyone may talk to everyone. If an important piece of information is far away, the model can still access it directly.

The difficulty is that everyone talking to everyone is expensive. Full attention forms an

interaction matrix with n^2 entries. When the sequence length, image resolution, or number of video patches becomes large, this quadratic cost can be a major limitation. One could reduce the cost by allowing each token to communicate with only nearby tokens, but then the model may lose long-range information. One could also use a simple global summary, but then the model may lose fine local details. Therefore, there is a natural tension between efficiency and accuracy:

Construct an architecture that is efficient while preserving the information needed for accurate prediction.

This is the central problem studied in this dissertation. The main approach is not to treat local and global information as competing choices. Instead, we combine them. A local operation is used to preserve fine and focused information, while a less expensive global operation allows information to move across the entire input. The exact choice depends on the structure of the problem. For time series, the global operation is Fourier mixing. For images, it can be arithmetic averaging placed inside a suitable architecture. For videos, spatial and temporal interactions can be separated and then combined.

1.2 From Sequential Models to Local-Global Models

Before attention, sequential data were commonly processed by recurrent neural networks. A recurrent model updates a hidden state one token at a time, so information from the past is compressed into a fixed-dimensional representation. State-space models, including Mamba, also use a recursive structure and can process long sequences with linear complexity [19]. These models are efficient because they do not explicitly compute every pairwise token

interaction. However, information must pass through a recurrence, and the way that the hidden state stores or forgets information becomes important.

Attention takes a different view. For a query, key, and value representation, a standard attention layer has the form

$$\mathrm{Attn}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d}}\right) V. \tag{1.2}$$

The attention matrix gives a direct interaction between tokens. This flexibility is one reason transformers have become useful across language, time series, images, and videos. However, a model that explicitly stores the full attention matrix pays quadratic memory and computational cost in the number of tokens.

A natural idea is to divide the tokens into small groups and compute attention only inside each group. This is window attention. It is similar to dividing people at a large conference into smaller discussion groups. Communication inside each group is manageable, but people in two different groups do not communicate directly. In image modeling, shifted windows allow information to move between nearby groups over multiple layers [50]. This works well in a sufficiently deep architecture, but it also shows the main question: after restricting attention locally, how should global information be recovered?

The works in this dissertation give several answers to this question. The answers are related, but they are not identical. Fourier mixing spreads local time-series information across the complete sequence. Arithmetic averaging represents the dispersed global behavior of attention in large image inputs. Split space-time attention uses the structure of videos rather than treating all video tokens as an unstructured sequence. These methods suggest the following general local-global form:

$$\mathcal{M}(x) = \mathcal{G}\big(\mathcal{L}(x)\big), \tag{1.3}$$

where $\mathcal{L}$ is a focused local operator and $\mathcal{G}$ is an efficient global operator. Depending on the architecture, the operators may also be applied in parallel, recursively, or over different dimensions. The goal is not necessarily to reproduce every entry of a full attention matrix. The goal is to retain the behavior that is important for the task, at a lower computational cost.

This viewpoint also provides a way to compare models that initially seem unrelated. Window attention, Fourier transforms, averaging, and state-space recurrence all move information differently. The important questions are: what information remains local, how information becomes global, what approximation is being made, and under what conditions the approximation is accurate? These questions motivate both the theoretical and experimental parts of this dissertation.

1.3 Time-Series Forecasting

Time series have a natural ordering, but useful dependencies may occur at very different time scales. Nearby observations often contain strong local information, while periodicity, trends, or delayed effects may require long-range information. Full attention can model these dependencies, but its cost becomes large for long sequence time-series forecasting. Sparse attention reduces the cost by selecting only part of the interaction, but its performance may depend on assumptions about the distribution or importance of the queries.

The first main work in this dissertation develops Fourier-Mixed Window attention, or FWin. FWin replaces a costly attention component with window attention followed by Fourier mixing. Window attention learns local relationships within small groups of time steps. The Fourier transform then mixes the window outputs across the entire sequence. This produces a simple division of responsibility: the window operation is local and learnable, while the

Fourier operation is global, parameter-free, and computationally efficient.

The method is motivated by architecture and also studied mathematically. Under a block diagonal invertibility condition on the attention matrix, the Fourier-mixed window representation can be equivalent to canonical full attention. This condition provides an explanation of when local attention followed by global mixing has enough information to recover the action of a full attention layer. The condition is then examined experimentally on benchmark data. The results show that FWin can improve or maintain prediction accuracy while accelerating inference, including on strongly nonstationary data.

The following dengue forecasting work studies FWin in a more specific application. Dengue incidence depends on variables such as temperature, precipitation, and large-scale ocean and climate behavior. These variables interact across time and may have delayed effects. The data are also limited and nonstationary, which makes the problem different from a standard benchmark with many repeated patterns. The dengue chapter is therefore not only another dataset for FWin. It examines whether the local-global design remains useful when the observations come from a real epidemiological system and when the prediction target is influenced by multiple environmental variables.

Together, these two works move from a general efficient forecasting architecture to an application where robustness is important. They also demonstrate a recurring principle of the dissertation: a useful efficient model should not only reduce FLOPs or inference time on a benchmark. It should preserve enough structure to remain accurate when the data distribution is complicated.

1.4 Image Modeling and the Dispersion of Attention

An image does not have a unique natural sequential ordering. It is a two-dimensional object with local spatial structure. Early convolutional neural networks use this structure directly through small kernels. A kernel first learns local patterns such as edges and textures, while deeper layers gradually form larger receptive fields and more global features. Vision transformers instead divide an image into patches and process them as tokens. This gives the flexibility of attention, but the number of tokens increases quickly with image resolution.

Window attention is well suited to images because nearby patches are strongly related, but a model also needs global information to recognize a complete object. This leads to the same local-global question as in time series. However, Fourier mixing is not the only possible global approximation. The SEMA work studies what happens to generalized attention as the number of tokens becomes large. It shows a dispersion phenomenon: under suitable assumptions, global attention approaches an equal distribution over tokens. In this asymptotic regime, a complicated global attention operation behaves like averaging.

This observation suggests a particularly simple global component. SEMA combines localized window attention with arithmetic averaging. Window attention preserves focused local information, while averaging captures the dispersed global behavior. The resulting attention module is placed in a Mamba-like macro-architecture, which provides an efficient multistage structure. SEMA is therefore not motivated only by replacing an expensive layer with a cheaper layer. Its global approximation is guided by a mathematical property of attention.

There is an interesting lesson here. A more complicated global interaction is not automatically a better approximation. If the attention weights disperse as the number of tokens increases, then arithmetic averaging may capture the large-scale behavior sufficiently well. At the same time, averaging alone cannot focus on important local features. The combination of a simple global operator and a focused local operator is essential. This is similar in spirit to FWin,

although the global operators and the theoretical explanations are different.

The experiments study image classification at different input resolutions and downstream vision tasks. Scaling the image resolution is important because it directly increases the number of tokens. A method that works only at a fixed small resolution does not completely solve the efficiency problem. SEMA is designed to remain effective as the input size grows, and its experiments examine both accuracy and computational behavior under this scaling.

1.5 From Images to Videos

A video introduces another dimension. If each frame is represented by many spatial patches, then a video contains a large number of tokens even when the clip is short. Full space-time attention allows every patch in every frame to interact with every other patch, but the computational cost grows rapidly with the number of frames and the spatial resolution. Compressing an entire frame into one token is efficient, but it can remove fine details. This may be especially problematic when a small object or a local motion is important.

A natural approach is to separate spatial and temporal processing. Spatial attention studies relationships between patches inside a frame, while temporal attention studies how information changes across frames. VideoSEMA follows this direction and extends the SEMA image backbone to video understanding. The model preserves local spatial information, introduces temporal interaction, and uses an efficient Mamba-like architecture rather than applying joint full attention to all space-time tokens.

This separation raises a theoretical question: when can split space-time attention represent the same interaction as full space-time attention? The VideoSEMA work studies rank conditions under which the factorized attention is equivalent to full attention. These conditions help describe what is gained and what may be lost by the factorization. The experiments then

evaluate the model on standard video-classification datasets and study its behavior as the spatial resolution increases.

VideoSEMA connects the time-series and image parts of the dissertation. A video is sequential across time and spatial within each frame. It therefore combines both types of structure studied earlier. The model should not ignore either structure, and it should not pay the cost of treating every interaction equally. This makes video a natural test of the local-global viewpoint.

1.6 Main Contributions of the Dissertation

The main contributions of this dissertation are summarized as follows.

- We develop FWin, an efficient local-global attention mechanism for long sequence time-series forecasting. FWin combines window attention with Fourier mixing and is supported by an equivalency result under a block diagonal invertibility condition.
- We study FWin on dengue prediction under climate and ocean influence. This application evaluates the method on limited, multivariate, and strongly nonstationary data, and shows the practical role of efficient long-range modeling beyond standard forecasting benchmarks.
- We establish a dispersion result for generalized attention and use it to motivate SEMA. SEMA combines localized attention with arithmetic averaging inside a Mamba-like vision architecture, providing a theoretically guided local-global approximation that scales to larger image resolutions.
- We extend this framework to videos through VideoSEMA. The model separates space and time in an efficient architecture, and the theoretical analysis gives rank conditions

under which split space-time attention can realize full space-time attention.

- Across these works, we identify a common design principle: use a focused local mechanism to preserve fine structure and an inexpensive global mechanism to communicate or enforce large-scale structure. The global mechanism does not need to reproduce every dense interaction if it preserves the behavior required by the task.

1.7 Organization of the Dissertation

The remainder of the dissertation is organized into an theoretical introduction section that encompass all of the main theoretical results used in the dissertation and two main parts.

The first part concerns time-series modeling. The chapter on Fourier-Mixed Window Attention introduces FWin, develops its theoretical formulation, and evaluates it on long sequence forecasting benchmarks and nonstationary datasets. The next chapter applies FWin to dengue forecasting using climate and ocean variables, with particular attention to the prediction task, model comparison, and the effect of window size.

The second part concerns computer vision. The SEMA chapter studies the dispersion of attention and constructs a scalable image model using token localization and averaging. The VideoSEMA chapter extends the same general idea to spatiotemporal data, develops split space-time attention, and evaluates the model on video-classification tasks.

Although the individual chapters address different data types and were developed as separate works, they are connected by one question: how much interaction is actually needed to solve a learning problem well? The results in this dissertation suggest that efficient learning does not require treating all information in the same way. Local structure, global structure, and the geometry of the task should each be handled by an operation suited to its role.

Chapter 2

A Unified Theoretical Analysis of Efficient Attention

2.1 Introduction

In this chapter, we collect the theoretical results used in FWin, SEMA, and VideoSEMA into a single framework. The three methods were developed for different applications, but they address the same question: when can an expensive full attention operation be replaced by a structured and more efficient operation? FWin uses local window attention followed by global Fourier mixing. SEMA uses window attention together with homogeneous mixing, motivated by the dispersion of attention over a large number of tokens. VideoSEMA separates a full space-time interaction into spatial and temporal interactions.

We use n for a general number of tokens and d for the feature dimension. For video, we use N for the number of spatial locations, T for the number of frames, and M for the number of queries. Therefore a flattened video has NT tokens. This allows the results in the later chapters to refer to one definition of attention rather than reintroducing it each time.

2.2 Unified Attention Notation

Given an input sequence $x \in \mathbb{R}^{n \times d_{\text{model}}}$, where n is the sequence length and d_{model} is the embedded dimension of the model. The input x is then converted into queries (Q), keys (K), values (V) as:

$$Q = xW_Q + b_Q,\ K = xW_K + b_K,\ V = xW_V + b_V,$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{attn}}}$ are the weighted matrix, and $b_Q, b_K, b_V \in \mathbb{R}^{n \times d_{\text{attn}}}$ are the bias matrix. In most cases, we will have $d_{\text{model}} = d_{\text{attn}} = d$, thus we will refer to d for the remaining of the dissertation.

Definition 2.1. Let $Q, K, V \in \mathbb{R}^{n \times d}$ be the query, key, and value matrices respectively. We denote their rows by $q_i, k_i, v_i \in \mathbb{R}^d$. Let $\psi_q, \psi_k : \mathbb{R}^d \to \mathbb{R}^d$ be continuous feature maps, and let $\phi : \mathbb{R} \to \mathbb{R}^+$ be a continuous positive normalization function. We define the unnormalized score and normalized attention coefficient by

$$e_{ij} = \psi_q(q_i)\psi_k(k_j)^T, \qquad p_{ij} = \frac{\phi(e_{ij})}{\sum_{l=1}^n \phi(e_{il})}. \tag{2.1}$$

The generalized ϕ-normalized attention matrix and full attention output are

$$P_\phi(Q, K) = (p_{ij})_{i,j=1}^n, \tag{2.2}$$

$$\mathcal{A}_\phi(Q, K, V) = P_\phi(Q, K)V = \begin{bmatrix} \frac{\sum_{l=1}^n \phi(\psi_q(q_1)\psi_k(k_l)^T)v_l}{\sum_{j=1}^n \phi(\psi_q(q_1)\psi_k(k_j)^T)} \\ \vdots \\ \frac{\sum_{l=1}^n \phi(\psi_q(q_n)\psi_k(k_l)^T)v_l}{\sum_{j=1}^n \phi(\psi_q(q_n)\psi_k(k_j)^T)} \end{bmatrix}. \tag{2.3}$$

This definition contains several familiar choices. If ψ_q and ψ_k are the identity maps, with the

factor $1/\sqrt{d}$ absorbed into either map, and $\phi(x) = \exp(x)$, then

$$\mathcal{A}_{\text{soft}}(Q, K, V) = \text{softmax}(QK^T/\sqrt{d})V, \tag{2.4}$$

which is the vanilla full attention [82]. If $\psi_q(x) = \psi_k(x) = \text{ELU}(x) + 1$ and $\phi(x) = x$, then (2.3) is a normalized linear attention [30, 39].

Definition 2.2. Let $Q, K, V \in \mathbb{R}^{n\times d}$ be the query, key, value matrices resp. Let $w \in \mathbb{N}$ be the window size such that w divides n. For token i, define its non-overlapping window by

$$J_w(i) = \{rw + 1, \ldots, (r + 1)w\}, \qquad r = \left\lfloor \frac{i-1}{w} \right\rfloor. \tag{2.5}$$

The window attention coefficient is

$$p_{ij}^{(w)} = \begin{cases} \dfrac{\phi(e_{ij})}{\sum_{l\in J_w(i)} \phi(e_{il})}, & j \in J_w(i), \\ 0, & j \notin J_w(i), \end{cases} \tag{2.6}$$

Define the window attention as:

$$P_{\phi,w}(Q, K) = (p_{ij}^{(w)})_{i,j=1}^{n}, \tag{2.7}$$

$$\mathcal{A}_{\phi,w}(Q, K, V; w) := \begin{bmatrix} \dfrac{\sum_{j\in J_w(1)} \phi(\psi_q(q_1)\psi_k(k_j)^T)v_j}{\sum_{i\in J_w(1)} \phi(\psi_q(q_1)\psi_k(k_i)^T)} \\ \vdots \\ \dfrac{\sum_{j\in J_w(n)} \phi(\psi_q(q_n)\psi_k(k_j)^T)v_j}{\sum_{i\in J_w(n)} \phi(\psi_q(q_n)\psi_k(k_i)^T)} \end{bmatrix} = P_{\phi,w}(Q, K)V. \tag{2.8}$$

For some index set $J(m)$, for example $J(m) = \{Mw + 1, \ldots, (M + 1)w\}$, where $M = \lfloor \frac{m-1}{w} \rfloor$.

The formulation of window attention here can be quite general. Since the index set $J(m)$ can be constructed so that it creates different patterns such as sliding window or dilated sliding window [2]. In particular, any attention mechanism limiting the query to only attend to a

fixed number of keys falls under definition 2.2. This is because that after positional encoding, we can relabel the indices of keys so that the final attention computation does not change.

For a vector $z = (z_1, \ldots, z_n) \in \mathbb{R}^n$, we define the following

$$\Phi_\phi(z)_j = \frac{\phi(z_j)}{\sum_{l=1}^{n} \phi(z_l)}. \tag{2.9}$$

Thus the i-th row of $P_\phi(Q, K)$ is $\Phi_\phi(e_i)$, where $e_i = (e_{i1}, \ldots, e_{in})$.

2.3 FWin: Mixed-Window and Full Attention

We first study whether the output of full attention can be recovered from window attention by a mixing operation. This result is algebraic. It does not claim that a fixed Fourier transform alone always equals full attention. Rather, it shows that window attention retains enough information under a block invertibility condition, and that an invertible global transform may be inserted without destroying this representational property.

2.3.1 Mixed Window

We start this section with a definition of the sufficient condition.

Definition 2.3 (Block diagonal invertibility)**.** Let $B \in \mathbb{R}^{n \times n}$ and let w divide n. For $r = 0, \ldots, n/w - 1$, let

$$B_r = B[rw + 1 : (r + 1)w,\ rw + 1 : (r + 1)w] \tag{2.10}$$

be the r-th $w \times w$ diagonal block. We say that B is block diagonally invertible with window size w, or BDI(w), if every B_r is invertible.

Definition 2.4 (Mixed window attention)**.** For a mixing matrix $G \in \mathbb{R}^{n\times n}$, define

$$\mathcal{A}_{\phi,mw}(Q,K,V;w,G) = G\mathcal{A}_{\phi,w}(Q,K,V;w). \tag{2.11}$$

Theorem 2.5 (Mixed-window representation of full attention)**.** *Let $Q,K,V \in \mathbb{R}^{n\times d}$, and let w divide n. Suppose the full attention matrix $P_\phi(Q,K)$ is BDI(w). Then there exists $G \in \mathbb{R}^{n\times n}$ such that*

$$\mathcal{A}_\phi(Q,K,V) = \mathcal{A}_{\phi,mw}(Q,K,V;w,G). \tag{2.12}$$

In particular, G can be constructed block by block from the full and window attention matrices.

Proof. We have the i-th row of full attention is

$$\mathcal{A}^i_\phi = \sum_{j=1}^{n} \frac{\phi(q_i^T k_j)v_j}{\beta_i}, \tag{2.13}$$

where $\beta_i = \sum_{j=1}^{L} \phi(q_i k_j^T)$. Let α_{im} be the i,m entry of G, the i-th row of mixed window attention is

$$\mathcal{A}^i_{\phi,mw} = \sum_{m=1}^{L} \alpha_{im} \sum_{j=Mw+1}^{(M+1)w} \frac{\phi(q_m k_j^T)v_j}{\gamma_m}, \tag{2.14}$$

where $\gamma_m = \sum_{j\in J(m)} \phi(q_m k_j^T)$, with $J(m) = \{Mw+1,\dots,(M+1)w\}$, where $M = \lfloor \frac{m-1}{w} \rfloor$.

Consider the following cases:

- If $i = m$ and $j \in \{Mw+1,\dots,(M+1)w\}$, set $\dfrac{1}{\beta_i} = \dfrac{\alpha_{ii}}{\gamma_m}$.
- If $i \neq m$ and $j \in \{Mw+1,\dots,(M+1)w\}$ and $j \in \{Iw+1,\dots,(I+1)w\}$, where $I = \lfloor \frac{i-1}{w} \rfloor$, set $\alpha_{im} = 0$.

- If $i \neq m$ and $j \in \{Mw+1, \ldots, (M+1)w\}$ and $j \notin \{Iw+1, \ldots, (I+1)w\}$, where $I = \lfloor \frac{i-1}{w} \rfloor$. For each j we set

$$\frac{\phi(q_i k_j^T)}{\beta_i} = \sum_{m \in J(j)} \frac{\alpha_{im} \phi(q_m k_j^T)}{\gamma_m}. \tag{2.15}$$

  For each i and set of $\{m, j\}$ pairs, we have to solve a system of w equations and unknowns. We now invoke the BDI assumption of $P_\phi(Q, K)$ to show that this system of equations has unique solution. Let C be the coefficient matrix of the right hand side of the system of equations in (2.15). We observe that C^T is invertible, because each row of C^T is a scaled version of a square sub matrix along the diagonal of $P_\phi(Q, K)$, and each square sub-matrix along the diagonal of $P_\phi(Q, K)$ is invertible. The invertibility of C then follows from that of C^T.

We completed the construction of G as claimed in the theorem.

□

Remark 2.6. The BDI assumption in Theorem 2.5 is a sufficient condition and not a necessary condition. We only need BDI to solve the system of equations 2.15 in the proof. This may be solvable even if BDI is not met. In this case, the solution will not be unique. In particular, the theorem holds true under no assumption when window size is 1. In latter chapters, we will show in practice that BDI is satisfied by most of the datasets studied, and window size of 1 provide competitive results.

In practice, invertibility alone does not describe numerical stability. The condition number of each diagonal block is therefore should be measured for numerical experiments.

2.3.2 Fourier mixing

The existence of G in Theorem 2.5 requires a $\mathcal{O}(n^2)$ computational complexity for mixing. We want to find a cheaper mechanism to do mixing. In this section, we will demonstrate that Fourier matrix as a mixing operator is efficient via Fast Fourier Transform.

Definition 2.7. Let $H \in \mathbb{C}^{n\times n}$ and Q, K, V, w be the same as defined in Definition 2.2, define the Fourier-mixed window attention as:

$$\mathcal{A}_{\phi,Fwin}(Q, K, V; w, H) := H\mathcal{F}(\mathcal{A}_{\phi,w}(Q, K, V; w)), \tag{2.16}$$

where $\mathcal{F}$ is the discrete Fourier transform.

Corollary 2.8. *Let Q, K, V and w be the same as defined in Theorem 2.5. If $P_\phi(Q, K)$ is BDI, then there exists a matrix $H \in \mathbb{C}^{n\times n}$ such that*

$$\mathcal{A}_\phi(Q, K, V) = \mathcal{A}_{\phi,Fwin}(Q, K, V; w, H). \tag{2.17}$$

Proof. From **Theorem 2.5**, there exists $B \in \mathbb{R}^{n\times n}$ such that

$$\mathcal{A}_\phi(Q, K, V) = \mathcal{A}_{\phi,mw}(Q, K, V; w, B). \tag{2.18}$$

Thus if we let $H = B\mathcal{F}^{-1}$, then we are done. □

The practical role of the Fourier transform is that it provides a fixed global mixing operation with fast implementation. The corollary is a representational result; the experiments determine whether the chosen architecture learns a useful approximation without explicitly forming the theoretical matrix H.

2.4 SEMA: Dispersion and Averaging

The FWin result asks whether a local representation can be globally mixed to recover full attention. SEMA begins from a different observation. When the number of keys becomes large and the logits remain bounded, every normalized attention coefficient becomes small. In this asymptotic regime, global attention loses its ability to focus strongly on one individual token. This motivates replacing the global part by homogeneous mixing while retaining window attention for local focus.

2.4.1 Dispersion in transformer layers

Theorem 2.9 (Dispersion of ϕ-normalized attention)**.** *Let $\mathcal{X} \subset \mathbb{R}^d$ be a compact input feature space, and let $X^{(n)} \in \mathcal{X}^n$ be a matrix of input features for n items. Let $e_{ij}^{(n)} = \psi_q(q_i)\psi_k((k_j^{(n)}))^T$ where $Q = \gamma(x_1^{(n)}, \ldots, x_n^{(n)})$ and $K^{(n)} = \kappa(x_1^{(n)}, \ldots, x_n^{(n)})$, where $\gamma : \mathcal{X}^n \to \mathbb{R}^{n\times d}$ and $\kappa : \mathcal{X}^n \to \mathbb{R}^{n\times d}$ are continuous functions, each expressible as a composition of L layers $g_L \circ f_L \circ \cdots \circ g_1 \circ f_1$ where each layer contains a feedforward component $f_i(z_1, \ldots, z_n)_k = f_i(z_k)$ and a self Φ-normalized attentional component $g_i(z_1, \ldots, z_n)_k = \sum_{1\le l\le n} \alpha_{lk} v_i(z_l)$ where $\alpha_{lk} \in [0, 1]$ are Φ-normalized attention coefficients and v_i is a feedforward network. Then, for any $\epsilon > 0$, there exists an $n \in \mathbb{N}$ such that $\Phi(e_i^{(n)})_j < \epsilon$ for all $1 \le i, j \le n$. That is the ϕ-normalized attention coefficients must disperse in all transformer heads.*

Proof. For a given head (h) of a transformer block, since $\mathcal{X}$ is compact and all feedforward layers f_i and v_i are continuous, thus resulting spaces are also compact. Every self Φ-normalized attention layer, g_i, computes a convex combination of the outputs of v_i and if outputs of v_i are on a compact space, the outputs of g_i remain on the same compact space. Similarly, γ and κ are continuous, so they preserve compactness. The dot product of two vectors $\psi_q((q_i^{(n)}))\psi_k((k_j^{(n)}))^T$ coming from compact spaces must be compact as well.

Thus, for all $1 \leq i, j \leq n$ the logits $e_{ij}^{(n)}$ must be bounded, that is there exists $M_h \in \mathbb{R}$ such that $\max_{i,j} |e_{ij}^{(n)}| < M_h$. Let $a, b \in [-M_h, M_h]$ such that $\phi(a) = \min_{x \in [-M_h, M_h]} \phi(x)$ and $\phi(b) = \max_{x \in [-M_h, M_h]} \phi(x)$. We have for all i, j,

$$\frac{\phi(a)}{n\phi(b)} \leq \frac{\phi(e_{ij}^{(n)})}{\sum_{l=1}^{n} \phi(e_{il}^{(n)})} \leq \frac{\phi(b)}{n\phi(a)}. \tag{2.19}$$

Let $\epsilon > 0$, then there exists $N(h, \epsilon) \in \mathbb{N}$ such that for all $n > N(h, \epsilon)$, we have

$$\frac{\phi(e_{ij}^{(n)})}{\sum_{l=1}^{n} \phi(e_{il}^{(n)})} \leq \epsilon. \tag{2.20}$$

Let $N = \max_{1 \leq h \leq H} N(h, \epsilon)$ where H is the total number of heads in the Transformer, thus for all $n > N$, we have

$$\frac{\phi(e_{ij}^{(n)})}{\sum_{l=1}^{n} \phi(e_{il}^{(n)})} \leq \epsilon. \tag{2.21}$$

This holds for all queries and keys in all of Transformer's heads. Hence we completed the proof. □

Remark 2.10. The compactness assumption can be relaxed to boundedness since we can take the closure of the set to get compactness by the Heine-Borel theorem. The boundedness assumption is strict, because if $\mathcal{X}$ is unbounded, then Theorem 2.9 may not hold. For example if we let $\phi(x) = \exp(x)$, and ψ_q, ψ_k as identity and $qk_j^T = \log(1/j^2)$, then

$$\frac{\phi(qk_j^T)}{\sum_{i=1}^{n} \phi(qk_i^T)} \geq \frac{\phi(qk_j^T)}{\sum_{i=1}^{\infty} \phi(qk_i^T)} = \frac{6}{\pi^2 j^2}. \tag{2.22}$$

Thus the query q attends to the first key with at least $6/\pi^2$ probability.

Softmax, normalized linear attention, focused attention, and other continuous normalized

variants fall under Theorem 2.9 when their features remain bounded. They may have very different dispersion coefficients, so one method may retain focus much longer than another. The theorem describes the common asymptotic behavior, not an identical finite-token behavior.

2.4.2 A probabilistic dispersion result

The deterministic theorem assumes bounded logits. We can also view normalized attention probabilistically by treating the positive unnormalized scores as random variables.

Proposition 2.11 (Dispersion under sparse positive covariance)**.** *Let $X_1, X_2, \dots$ be random variables (not necessarily i.i.d.) such that $X_i > 0$ with $\mathbb{E}[X_i] = \mu > 0$, $\sup_{i,j} \mathrm{Cov}(X_i, X_j) < \infty$, and for every n, $|\{(i,j) : \mathrm{Cov}(X_i, X_j) > 0 \text{ and } 1 \le i < j \le n\}| \le n^\alpha$ for $1 < \alpha < 2$. For $0 < \epsilon \ll \mu$, with at least (high) probability $1 - \epsilon$, and for sufficiently large n:*

$$\frac{X_i}{n(\mu+\epsilon)} \le \frac{X_i}{\sum_{j=1}^n X_j} \le \frac{X_i}{n(\mu-\epsilon)}. \tag{2.23}$$

Proof. Let $S_n := \sum_{i=1}^n X_i$ then we have

$$\begin{aligned}
\mathrm{Var}(S_n/n) &= \frac{1}{n^2}\mathrm{Var}(S_n) && (2.24)\\
&= \frac{1}{n^2}\left(\sum_{i=1}^n \mathrm{Var}(X_i) + 2\sum_{1\le i<j\le n} \mathrm{Cov}(X_i, X_j)\right) && (2.25)\\
&\le \frac{Cn}{n^2} + \frac{2Cn^\alpha}{n^2} \le 3C\frac{1}{n^{2-\alpha}}, && (2.26)
\end{aligned}$$

where $\sup_{i,j} \mathrm{Cov}(X_i, X_j) \le C$. For $0 < \epsilon \ll \mu$, and let $N = \lceil (\frac{3C}{\epsilon^3})^{\frac{1}{2-\alpha}} \rceil$ then by Chebyshev's inequality [84], we have

$$\mathbb{P}\left\{\left|\frac{S_N}{N} - \mu\right| > \epsilon\right\} \le \frac{\mathrm{Var}(S_N/N)}{\epsilon^2} = \frac{3C}{N^{2-\alpha}\epsilon^2} \le \epsilon. \tag{2.27}$$

Thus,

$$\mathbb{P}\left\{\left|\frac{S_N}{N}-\mu\right|\leq\epsilon\right\}\geq 1-\epsilon. \tag{2.28}$$

This implies with at least $1-\epsilon$ probability $S_N/N>\mu-\epsilon$ and $S_N/N<\mu+\epsilon$, and so

$$\frac{X_i}{N(\mu+\epsilon)}<\frac{X_i}{S_N}<\frac{X_i}{N(\mu-\epsilon)}. \tag{2.29}$$

Hence we completed the proof. □

If $X_j=\phi(e_{ij})$, Proposition 2.11 shows that the normalized attention coefficients disperse with high probability under the stated covariance condition.

Although the Proposition 2.11 requires sparse covariance condition, in the case when this does not meet, in practice the average is a good approximation. For example, in natural language processing, if most of the words in a input segment, e.g. paragraphs or documents, are highly correlated then we would expect that the unormalized attention score to be similar, thus the attention score will be similar thus average the value is a good approximation of the attention mechanism.

2.4.3 Why window attention and averaging are complementary

For fixed w, a nonzero window coefficient satisfies

$$p_{ij}^{(w)}=\frac{\phi(e_{ij})}{\sum_{l\in J_w(i)}\phi(e_{il})}, \tag{2.30}$$

whose denominator has only w terms. It does not go to zero merely because the total token count n increases. Thus window attention maintains local focus, but it cannot directly communicate across windows.

On the other hand, the global arithmetic average

$$\mathrm{Avg}(V) = \frac{1}{n}\sum_{j=1}^{n} v_j \tag{2.31}$$

has exactly homogeneous weights. Theorem 2.9 shows why this can be a meaningful asymptotic approximation of the global portion of normalized attention. SEMA combines the two behaviors: window attention preserves local high-frequency and focused information, while averaging provides an inexpensive global receptive field. Averaging alone is not meant to replace the local operator.

2.4.4 Mamba as recursive attention

In this section we draw a connection between Mamba recurrence and window attention. We start with the following proposition.

Proposition 2.12. *Let* $x_i, y_i, \Delta_i, D \in \mathbb{R}^{1\times c}$, $A_i, h_i, h_{i-1} \in \mathbb{R}^{d\times c}$, $B_i \in \mathbb{R}^{d\times 1}$, $C_i \in \mathbb{R}^{1\times d}$, *for* $i \in \{1, \dots, m\}$, *and*

$$h_i = A_i \odot h_{i-1} + B_i(\Delta_i \odot x_i), \quad y_i = C_i h_i + D \odot x_i \tag{2.32}$$

where h_0 *is given and* $\odot$ *denotes the elementwise (Hadamard) product. Then we have*

$$h_m = \left(\prod_{j=1}^{m} A_j\right) \odot h_0 + \sum_{i=1}^{m} \left(\left(\prod_{j=1}^{m-i} A_{m-(j-1)}\right) \odot (B_i(\Delta_i \odot x_i))\right) \tag{2.33}$$

where $\prod$ *is the elementwise multiple matrix product, under the convention that if the upper*

index is zero, the product acts as an identity. It follows that

$$y_m = C_m \left(\prod_{j=1}^{m} A_j \right) \odot h_0 + C_m \left(\sum_{i=1}^{m} \left(\prod_{j=1}^{m-i} A_{m-(j-1)} \right) \odot (B_i(\Delta_i \odot x_i)) \right) + D \odot x_m. \tag{2.34}$$

Proof. We prove by induction. First verify the base case,

$$h_1 = A_1 \odot h_0 + B_1(\Delta_1 \odot x_1). \tag{2.35}$$

Next for the inductive step, assuming

$$h_m = \left(\prod_{j=1}^{m} A_j \right) \odot h_0 + \sum_{i=1}^{m} \left(\left(\prod_{j=1}^{m-i} A_{m-(j-1)} \right) \odot (B_i(\Delta_i \odot x_i)) \right), \tag{2.36}$$

we have

$$\begin{aligned} h_{m+1} &= A_{m+1} \odot h_m + B_{m+1}(\Delta_{m+1} \odot x_{m+1}) && (2.37) \\ &= A_{m+1} \odot \left(\left(\prod_{j=1}^{m} A_j \right) \odot h_0 + \sum_{i=1}^{m} \left(\left(\prod_{j=1}^{m-i} A_{m-(j-1)} \right) \odot (B_i(\Delta_i \odot x_i)) \right) \right) \\ &\quad + B_{m+1}(\Delta_{m+1} \odot x_{m+1}) && (2.38) \\ &= \left(\prod_{j=1}^{m+1} A_j \right) \odot h_0 + \sum_{i=1}^{m+1} \left(\left(\prod_{j=1}^{m+1-i} A_{m+1-(j-1)} \right) \odot (B_i(\Delta_i \odot x_i)) \right). && (2.39) \end{aligned}$$

Substituting h_m into the y_m equation, we obtain

$$y_m = C_m \left(\prod_{j=1}^{m} \tilde{A}_j \right) \odot h_0 + C_m \left(\sum_{i=1}^{m} \left(\prod_{j=1}^{m-i} A_{m-(j-1)} \right) \odot (B_i(\Delta_i \odot x_i)) \right) + D \odot x_m. \tag{2.40}$$

The proof is complete.

□

The state space model is a map from input $x(t) \in \mathbb{R}$ to output $y(t) \in \mathbb{R}$ through a hidden state $h(t) \in \mathbb{R}^{d\times 1}$ that can be written as the following system of equations:

$$h'(t) = Ah(t) + Bx(t), y(t) = Ch(t) + Dx(t), \tag{2.41}$$

where $A \in \mathbb{R}^{d\times d}, B, h(t), h't(t) \in \mathbb{R}^{d\times 1}, C \in \mathbb{R}^{1\times d}, D \in \mathbb{R}$. After applying the zero-order hold discretization, and extending the map to high dimensional input $x \in \mathbb{R}^{n\times c}$, we obtain the following multi-dimensional system of discrete equations [25]:

$$h_i = \tilde{A}_i \odot h_{i-1} + B_i(\Delta_i \odot x_i),\ y_i = C_i h_i/1 + D \odot x_i, \tag{2.42}$$

where $x_i, y_i, \Delta_i, D, \in \mathbb{R}^{1\times c}, \tilde{A}_i, h_{i-1}, h_i \in \mathbb{R}^{d\times c}, B_i \in \mathbb{R}^{d\times 1}$, $C_i \in \mathbb{R}^{1\times d}$, / denotes elementwise division, and $\odot$ the elementwise product. In contrast, the causal linear attention in recursive form is ($S_0 = 0, Z_i := \sum_{j=1}^{i} k_j^T$):

$$S_i = 1 \odot S_{i-1} + k_i^T(1 \odot v_i), y_i = q_i S_i/q_i Z_i + 0 \odot x_i, \tag{2.43}$$

for given $Q, K, V \in \mathbb{R}^{n\times d}$, the queries, keys, and values respectively. This reveals the association between Eq. (2.42) and Eq. (2.43): $q_i \simeq C_i, k_i^T \simeq B_i$ and $v_i \simeq x_i$. Apply proposition 2.12, and it follows from Eq. (2.42) and $h_0 = 0$:

$$y_m = \sum_{i=1}^{m} q_m \tilde{k}_i^T \tilde{v}_i + D \odot v_m, \tag{2.44}$$

where $\tilde{k}_i^T \tilde{v}_i = \left(\prod_{j=1}^{m-i} \tilde{A}_{m-(j-1)}\right) \odot (B_i(\Delta_i \odot x_i))$, with $\prod$ as matrix product in the elementwise sense. The second term in (2.44) can be understood as a skip connection. The first term in (2.44) is the causal masked attention with $\phi(x) = x$ (without the normalization term). In Mamba, $\tilde{A}_i$'s are chosen so that every element falls between 0 and 1 [25], implying that the product of $\tilde{A}_i$ goes toward zero, i.e. the sum in equation (2.44) converges as $m \to \infty$. For

fixed query (q_m), increasing the number of keys above m does not change the value of the Mamba attention in equation (2.44). This implies that Mamba-like attention mechanism does not disperse. Equation (2.44) shows that there is exponential forgetting of previous keys in terms of Hadamard products of $\tilde{A}_i$. This is quite similar to **LagT** measure in HiPPO mechanism [20].

2.5 VideoSEMA: Space-Time Attention Factorization

We now consider video tokens $x_{ij} \in \mathbb{R}^d$, where $i \in \{1, \ldots, N\}$ is the spatial index and $j \in \{1, \ldots, T\}$ is the temporal index. For a fixed query q, full space-time attention is

$$\mathcal{A}_{ST}(q) = \sum_{j=1}^{T} \sum_{i=1}^{N} \eta_{ij} x_{ij}, \qquad \eta_{ij} = \frac{\phi(qk_{ij}^T)}{\sum_{r=1}^{T} \sum_{l=1}^{N} \phi(qk_{lr}^T)}. \tag{2.45}$$

The coefficients form a positive probability distribution over the NT space-time locations.

Split space-time attention has the form

$$\mathcal{A}_{T+S}(q) = \sum_{j=1}^{T} \alpha_j \left(\sum_{i=1}^{N} \beta_{i|j} x_{ij} \right), \tag{2.46}$$

where α_j is a temporal coefficient and $\beta_{i|j}$ is a spatial coefficient conditioned on frame j.

2.5.1 Marginal-conditional decomposition

Given the full coefficients η_{ij}, define

$$\alpha_j = \sum_{i=1}^{N} \eta_{ij}, \qquad \beta_{i|j} = \frac{\eta_{ij}}{\alpha_j}. \tag{2.47}$$

Since $\eta_{ij} > 0$, we have $\alpha_j > 0$. Moreover,

$$\sum_{j=1}^{T} \alpha_j = 1, \qquad \sum_{i=1}^{N} \beta_{i|j} = 1, \qquad \alpha_j \beta_{i|j} = \eta_{ij}. \tag{2.48}$$

Therefore

$$\mathcal{A}_{T+S}(q) = \mathcal{A}_{ST}(q). \tag{2.49}$$

For a single query, split space-time attention is exactly the marginal-conditional factorization of the full attention distribution. It remains to determine whether dot-product attention can realize the required α and β coefficients for all queries.

2.5.2 Exact realization under rank conditions

Let $m \in \{1, \ldots, M\}$ index the query rows. Denote the full coefficient by η_{mij} and define

$$\alpha_{mj} = \sum_{i=1}^{N} \eta_{mij}, \qquad \beta_{mi|j} = \frac{\eta_{mij}}{\alpha_{mj}}. \tag{2.50}$$

Theorem 2.13 (Exact rank condition for normalized ϕ)**.** *Assume $\phi : \mathbb{R} \to \mathbb{R}^+$ is invertible on its positive range. Choose positive constants c_m and c_{mj} so that the following quantities lie in the range of ϕ, and define*

$$R^{\phi}_{mj} = \phi^{-1}(c_m \alpha_{mj}), \qquad S^{\phi}_{m,(j,i)} = \phi^{-1}(c_{mj} \beta_{mi|j}). \tag{2.51}$$

If the temporal and spatial head dimensions satisfy

$$d_T \geq \operatorname{rank}(R^{\phi}), \qquad d_S \geq \operatorname{rank}(S^{\phi}), \tag{2.52}$$

then there exist query and key matrices

$$Q^\tau \in \mathbb{R}^{M\times d_T}, \quad K^\tau \in \mathbb{R}^{T\times d_T}, \quad Q^S \in \mathbb{R}^{M\times d_S}, \quad K^S \in \mathbb{R}^{NT\times d_S} \tag{2.53}$$

such that normalized temporal attention produces $\alpha_{m,:}$, normalized frame-wise spatial attention produces $\beta_{m:|j}$, and split space-time attention equals full space-time attention for every query.

Proof. The rank conditions imply the factorizations

$$R^\phi = Q^\tau (K^\tau)^T, \qquad S^\phi = Q^S (K^S)^T, \tag{2.54}$$

for example by compact singular value decomposition. Normalizing the temporal scores gives

$$\frac{\phi(R^\phi_{mj})}{\sum_{r=1}^T \phi(R^\phi_{mr})} = \frac{c_m \alpha_{mj}}{\sum_{r=1}^T c_m \alpha_{mr}} = \alpha_{mj}. \tag{2.55}$$

For every fixed (m, j), normalizing the spatial scores inside frame j gives

$$\frac{\phi(S^\phi_{m,(j,i)})}{\sum_{l=1}^N \phi(S^\phi_{m,(j,l)})} = \beta_{mi|j}. \tag{2.56}$$

Their product is $\alpha_{mj}\beta_{mi|j} = \eta_{mij}$, so the split and full outputs are equal. □

For softmax, $\phi(x) = \exp(x)$ and $\phi^{-1}(x) = \log(x)$. If the full logits are $e_{mij} = q_m k_{ij}^T$, one possible spatial score is e_{mij} within each frame, while the temporal score is the frame-level log-sum-exp

$$\tau_{mj} = \log\left(\sum_{i=1}^N \exp(e_{mij})\right). \tag{2.57}$$

This makes the marginal-conditional form explicit.

2.5.3 Low-rank approximation

For long sequences and high-resolution videos, the exact rank requirements may be strict. We therefore consider rank-constrained score approximations $\widehat{R}$ and $\widehat{S}$ with

$$\operatorname{rank}(\widehat{R}) \le d_T, \qquad \operatorname{rank}(\widehat{S}) \le d_S. \tag{2.58}$$

Let $\widehat{\alpha}$ and $\widehat{\beta}$ be the normalized ϕ coefficients obtained from these scores, and define

$$\widehat{\eta}_{mij} = \widehat{\alpha}_{mj}\widehat{\beta}_{mi|j}. \tag{2.59}$$

Theorem 2.14 (Error of low-rank split attention)**.** *Suppose*

$$\epsilon_T = \max_m \|\widehat{\alpha}_{m,:} - \alpha_{m,:}\|_1, \qquad \epsilon_S = \max_{m,j} \|\widehat{\beta}_{m:|j} - \beta_{m:|j}\|_1. \tag{2.60}$$

Then for every query m,

$$\sum_{j=1}^{T}\sum_{i=1}^{N} |\widehat{\eta}_{mij} - \eta_{mij}| \le \epsilon_T + \epsilon_S. \tag{2.61}$$

If $\|x_{ij}\|_2 \le B$, then

$$\|\widehat{\mathcal{A}}^{(m)}_{T+S} - \mathcal{A}^{(m)}_{ST}\|_2 \le B(\epsilon_T + \epsilon_S). \tag{2.62}$$

Proof. Add and subtract $\widehat{\alpha}_{mj}\beta_{mi|j}$:

$$|\widehat{\eta}_{mij} - \eta_{mij}| \le \widehat{\alpha}_{mj}|\widehat{\beta}_{mi|j} - \beta_{mi|j}| + |\widehat{\alpha}_{mj} - \alpha_{mj}|\beta_{mi|j}. \tag{2.63}$$

Summing over i and j, and using that $\widehat{\alpha}$ and β are probability distributions, gives

$$\sum_{j=1}^{T}\sum_{i=1}^{N}|\widehat{\eta}_{mij}-\eta_{mij}| \le \|\widehat{\alpha}_{m,:}-\alpha_{m,:}\|_1 + \sum_{j=1}^{T}\widehat{\alpha}_{mj}\|\widehat{\beta}_{m:|j}-\beta_{m:|j}\|_1 \tag{2.64}$$

$$\le \epsilon_T + \epsilon_S. \tag{2.65}$$

The output bound follows from the triangle inequality:

$$\|\widehat{\mathcal{A}}^{(m)}_{T+S}-\mathcal{A}^{(m)}_{ST}\|_2 \le \sum_{j=1}^{T}\sum_{i=1}^{N}|\widehat{\eta}_{mij}-\eta_{mij}|\|x_{ij}\|_2. \tag{2.66}$$

□

The score matrices may be approximated by their best rank-d_T and rank-d_S singular value decompositions. By the Eckart–Young theorem,

$$\|R^{\phi}-\widehat{R}\|_F^2 = \sum_{r>d_T}\sigma_r(R^{\phi})^2, \qquad \|S^{\phi}-\widehat{S}\|_F^2 = \sum_{r>d_S}\sigma_r(S^{\phi})^2. \tag{2.67}$$

Thus the quality of split attention is related to the singular-value tails of the target temporal and spatial scores, together with the sensitivity of the normalized ϕ map. When T is small, the temporal rank condition may be satisfied exactly even when the spatial factorization remains approximate.

2.6 Relationship Between the Three Results

The three theoretical results describe different ways to replace dense attention.

First, FWin is a finite-dimensional equivalency result. Under BDI(w), window attention is invertible as a block diagonal map, so a global mixing matrix can recover the full attention

output. The Fourier transform is useful because it is invertible, global, and fast. The result explains representational sufficiency, although the practical architecture uses a constrained and much cheaper mixing design.

Second, SEMA is an asymptotic result. If the token features remain bounded, every normalized global attention coefficient is of order $1/n$. This motivates homogeneous mixing for the global component. Since averaging cannot preserve local focus, SEMA combines it with window attention. Thus the SEMA result does not attempt an exact finite-token reconstruction of every full attention matrix. It describes the large-token behavior that the efficient operator should imitate.

Third, VideoSEMA is a structured factorization result. Full space-time attention is a joint distribution over spatial and temporal locations. Split attention represents the same distribution as a temporal marginal multiplied by a frame-conditional spatial distribution. Exact realization depends on the rank of the score matrices; otherwise the error is controlled by the temporal and spatial coefficient errors.

Together, these results suggest one common principle. Full attention can be simplified when we use the structure of the interaction rather than treat the attention matrix as an arbitrary dense matrix. For FWin, the structure is local blocks followed by global mixing. For SEMA, it is local focus together with asymptotically homogeneous global behavior. For VideoSEMA, it is the marginal-conditional structure of space and time. The later application chapters use these results without redefining the attention notation.

Part II

Time Series

Chapter 3

Fourier-Mixed Window Attention for Efficient and Robust Long Sequence Time-Series Forecasting

We study a fast local-global window-based attention method to accelerate Informer for long sequence time-series forecasting (LSTF) in a robust manner. While window attention being local is a considerable computational saving, it lacks the ability to capture global token information which is compensated by a subsequent Fourier transform block. Our method, named FWin, does not rely on query sparsity hypothesis and an empirical approximation underlying the ProbSparse attention of Informer. Experiments on univariate and multivariate datasets show that FWin transformers improve the overall prediction accuracies of Informer while accelerating its inference speeds by 1.6 to 2 times. On strongly non-stationary data, FWin outperforms Informer and recent SOTAs thereby demonstrating its superior robustness. The BDI is verified to hold with high probability on benchmark datasets experimentally.

3.1 Introduction

Recent progress in long sequence time-series forecasting (LSTF) has been led by either transformers with component (attention) upgrade such as sparse attention ([104] and references therein), attention in combination with signal processing (e.g. seasonal-trend decomposition [106], adopting auto-correlation to account for periodicity in the data [92], patching technique [65]) or architectural change [47]. In the category of advancing component (attention) efficiency without making architectural change, Fourier transform has been proposed as an alternative mixing tool in lieu of standard attention [82] to speed up prediction in natural language processing (NLP) tasks (FNet, [33]). Though Fourier transform is meant to mimic the mixing functions of multi-layer perceptron (MLP, [78]), it is not well-understood why it works and when assistance from attention layers remain necessary to maintain performance. In computer vision (CV), Fourier transform is also used as a filtering step in early stages of transformer (GFNet,[68]) to enhance a fully attention-based architecture. A recent advance in CV is to adopt window attention to reduce quadratic complexity of full attention [82]. In shifted window attention (Swin [49]), the attention is first computed on non-overlapping windows as a local approximation, then on shifted window configurations to spread local attention globally in the image domain. This local-global approximation of full attention occurs entirely in the image domain, and is repeated over multiple stages in the network. The advantage is that the recipe is independent of the data distribution. In contrast, the ProbSparse self-attention of Informer [104, 105] relies on long tail data distribution to select the top few queries.

We are interested in developing an efficient window-based attention to replace ProbSparse attention and accelerate Informer with no prior knowledge of data properties such as periodicity (seasonality) so that our method is applicable in a general context of time series. We also refrain from pre-processing data to increase performance as this step can be added later. The main issue is how to globalize the local window attention without performing shifts, since

Informer only has two attention blocks in the encoder and is unable to facilitate repeated window shifting as in a deeper network Swin [49].

We propose to replace ProbSparse attention of Informer via a (local) window attention followed by a Fourier transform (mixing) layer, a novel local-global attention which we call Fourier-Mixed window attention (FWin). The resulting network, FWin Transformer, aims to reduce the complexity of the full attention [82] and approximate its functionality by mixing the window attention. Instead of shifting windows for globalization [49], we employ the parameter free fast Fourier Transform (FFT) to generate connections among the tokens. The strategy allows the window attention layer to focus on learning local information, while the Fourier layer effectively mixes tokens and spreads information globally. In the ablation study, we find that the network prediction accuracy is lower if we replace ProbSparse by only Fourier mixing as in FNet [33] without the help of window attention. Besides conducting extensive experiments on FWin to support our methodology, we also provide a mathematical formulation of Fourier mixed window attention and prove that it is equivalent to the canonical attention.

Our main contributions in this paper are summarized below.

- We introduce FWin and replace the ProbSparse self attention block (Figure 3.1, left) via a window self-attention block followed by a Fourier mixing layer (Figure 3.1, right) in both the encoder and decoder of Informer [104, 105].
- We show experimentally that FWin either increases or maintains Informer's performance level while significantly accelerating its inference speed (by about 1.6 to 2 times) on both uni/multi-variate LSTF data. The training times of FWin are consistently lower than those of Informer across various data sets. Inference speeds of FWin exceed those of FEDformer [106] and Autoformer [92] by a factor of 5 with shorter training times overall. On highly non-stationary power grid data [6, 100], FWin out-performs Informer

and other recent SOTAs, showcasing its superior robustness.

- We propose FWin-S, a light weight version of FWin, by removing Fourier mixing layer in the decoder (Figure 3.1, right); and present its competitive performance with faster inference speed and surprising robustness (which often comes at the expense of speed instead) .

3.2 Background

3.2.1 Related Works

Several types of attention models are related to our work here.

First, MLP mixers relax the graph similarity constraints of the self-attention and mix tokens with MLP projections. The original MLP-mixer [78] reaches similar accuracy as self-attention in vision tasks. However, such a method lacks scalability as a result of quadratic complexity of MLP projection, and suffers from parameter inefficiency for high resolution input.

Next, Fourier-based mixers apply Fourier transform to mix tokens in NLP and vision tasks. FNet [33] resembles the MLP-mixer with token mixer being the classical discrete Fourier transform (DFT), without adaptive filtering on data distribution. Global filter networks (GFNs [68]) learn filters in the Fourier domain to perform depth-wise global convolution with no channel mixing involved. Also, GFN filters lack adaptivity that could negatively impact generalization. AFNO [21] performs global convolution with dynamic filtering and channel mixing for better expressiveness and generalization. However, AFNO network parameter sizes tend to be much larger than those of the light weight vision transformer (ViT) models such as GFN-T [68], shift-window mixer Swin-T [49] and hybrid convolution-attention models MOAT-T [94], and mobile ViT [57].

For long sequence time series forecasting, the Informer [104, 105] has a hybrid convolution-attention design with a probabilistic sparsity promoting function (ProbSparse) to reduce complexity of the standard self-attention and cross-attention. We shall adopt Informer as our baseline in this work, as it compares favorably vs. efficient transformers in recent years (see Table 1 and Table 2 in [104]). More recent improvements on benchmark data sets include Autoformer [92], FEDformer [106] and ETSformer [89], which are designed with certain prior-knowledge of datasets, e.g. using trend/seasonality decomposition and auto-correlation functions. Additionally, PatchTST [65] and iTransformer [47] focus on preserving variate information, employing independent channel processing or inverting dimensions of the multivariate inputs. However, they are not as robust on non-stationary time series as Informer (see Table 3.4). Informer's prediction is seen to generate spurious peaks on power grid data (3rd frame of Figure 3.2), while FWin predictions appear smoother and free from such distortions. Comparing Informer with full Informer in the bottom frame of Figure 3.2, we see that the cause of these distortions may be attributed to Probsparse. Glassoformer [100] uses group lasso penalty to enforce query sparsity and reduce complexity of full attention to speed up inference. Though this method works well on power grid data, its training time is higher than Informer since full attention is involved in network training.

3.2.2 Preliminary

In window attention, we compute attention on a window-by-window basis. Within each window, all the keys are multiplied with the corresponding query within that window. The output is the weighted sum of the values within the same window, rather than considering the entire set of values. This approach reduces the computational cost of computing attention on a sequence of length n from $\mathcal{O}(n^2)$ of full attention to $\mathcal{O}(nw)$, where w is a fixed window size. Figure 3.3 shows the overview of full attention versus window attention.

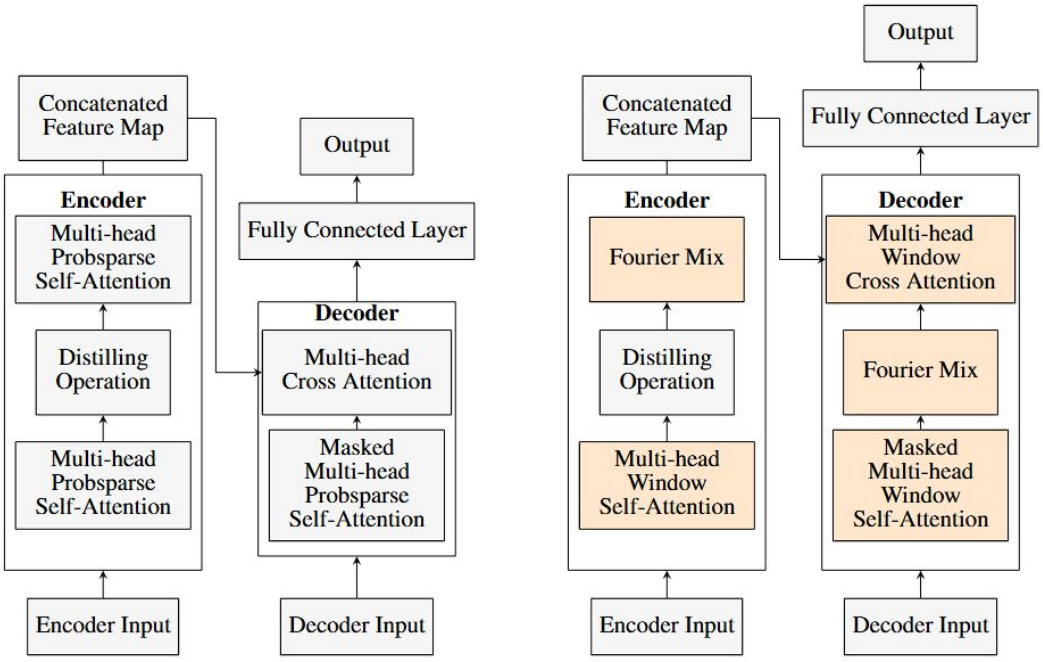


Figure 3.1: Model comparison: Informer (left), FWin (right, orange color denotes our contributions); FWin-S (FWin with its decoder's Fourier Mix block removed).

Window attention restricts the interaction between queries and keys by only allowing queries to attend to their local window keys. As a result, window attention provides limited or local information. On the other hand, full attention enables queries to attend to keys that are further away, allowing for global interaction. If we replace full attention with window attention, our model may lack a comprehensive understanding of global information. Therefore, it is desirable for our model to retain some level of global information when using window attention as a substitute for full attention. To incorporate global information, Swin Transformer [49] introduces shifted window attention. In this approach, before dividing the input sequence x into sub-sequences, one performs a circular shift of the indices of x by certain value. This shift allows the ending values of x to become the starting values of our input. The circular shifting process is repeated for each consecutive window attention layers.

Another way to promote global token interaction is by Fourier transform as proposed in FNet [33]. Given input $x \in \mathbb{R}^{L \times d_{\text{model}}}$, one computes Fourier transform along the model dimension

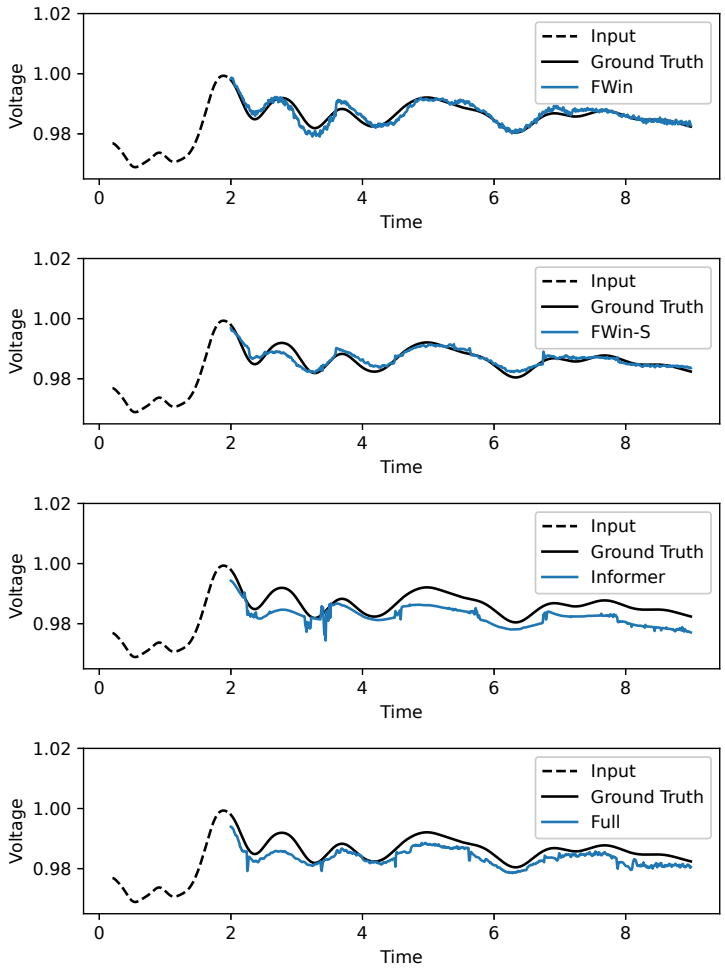


Figure 3.2: Univariate post fault prediction (voltage vs. time in second) on power grid data [6, 100]. FWin, FWin-S have "smooth" predictions while Informer has spurious jumps. Full in the bottom frame refers to Informer using full attention instead of probsparse. The dashed line to the left of 2 second is the input, to the right of which are the model predictions vs. the ground truth (in black).

(d_{model}), then along the time dimension (L), finally taking real part to arrive at:

$$y = \mathcal{R}(\mathcal{F}_{\text{time}}(\mathcal{F}_{\text{model}}(x))), \tag{3.1}$$

where $\mathcal{F}$ is 1D discrete Fourier transform (DFT), and $\mathcal{R}$ is the real part of a complex number. Here $\mathcal{F}_{\text{time}}, \mathcal{F}_{\text{model}}$ denote the Fourier transform apply along the sequence and hidden model dimension respectively. Since DFT is free of learning parameter, one would eventually pass the transformed sequence through a Feed Forward fully-connected layer (FC). This approach can

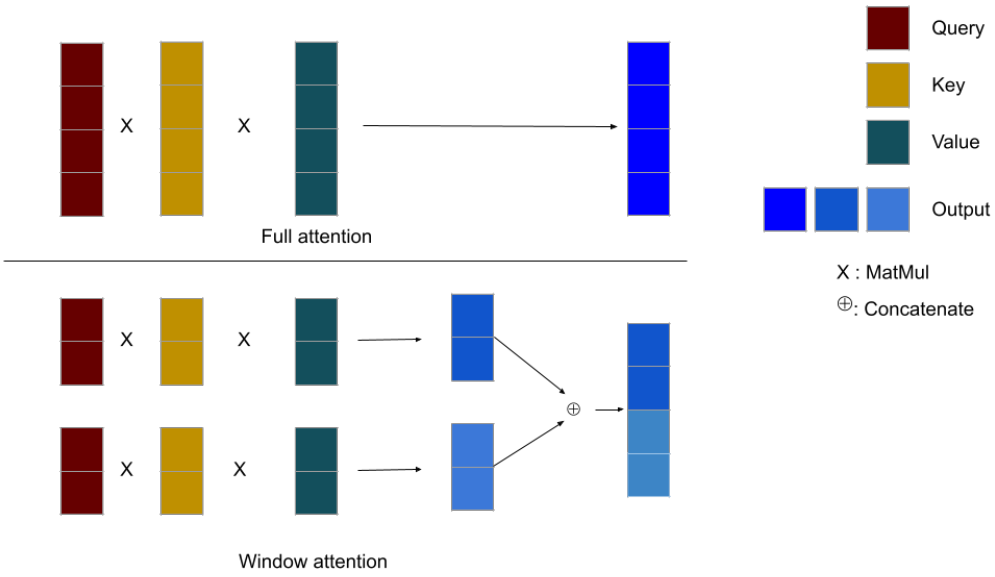


Figure 3.3: Overall structure of the full and window attention mechanisms. The symbol $\times$ denotes matrix multiplication between the query, key, and value matrices, while $\oplus$ represents the concatenation of the output matrices along the sequence dimension. In full attention, each query interacts with all of the keys and values, producing a dark blue output. In contrast, with window attention, the query interacts with only a subset of keys and values, resulting in a lighter blue output.

be interpreted as the Fourier transform being an effective mechanism for mixing sub-sequences (tokens) [33]. By applying the Fourier transform, the Feed Forward layer gains access to all the tokens.

3.3 Methodology

3.3.1 Informer Overview

The input $x \in \mathbb{R}^{n \times d_{\mathbf{data}}}$ passes through an embedding layer to encode the time scale information and return $X \in \mathbb{R}^{n \times d_{\text{model}}}$. In the encoder, each layer consists of an attention block followed by a distilling convolution with MaxPool of stride 2 and a down-sampling to halve dimension. Thus, with 2 encoder layers, the time dimension of the first attention block is n, while the second block input is $n/2$. For both efficiency and causality, the decoder attention has a masked multi-head ProbSparse self-attention structure, see Fig. 3.1 (left) for an overview. The ProbSparse self-attention [104] relies on a sparse query measurement function (an analytical approximation of Kullback-Leibler divergence) so that each key attends to only a few top queries for computational savings. The sparse query hypothesis or equivalently the long tail distribution of self-attention feature map is based on softmax scores in self-attention of a 4-layer canonical transformer trained on ETTh1 data set (Appendix C, [104]).

3.3.2 Our Approach

We introduce Fourier mixed window attention (FWin) to the self-attention blocks in the encoder and decoder of Informer. Specifically, we replace the ProbSparse self-attention blocks in the encoder and decoder with window attention followed by a Fourier mixing layer. We also replace multi-head cross-attention in the decoder by a window multi-head cross-attention. Fig.

3.1 (right) illustrates the key components of FWin Transformer. Different from ProbSparse attention, our FWin approach does not rely on whether the query sparse hypothesis holds on a data set.

We remark that our model differs from the FNet architecture in that Fourier transform is applied to the input along the model and the time dimensions without a subsequent Feed Forward layer. Partly this is due to the Feed Forward layer already present in the decoder of Informer before the output (Figure 3.1, left). We denote this specific component as **Fourier Mix** in our proposed FWin architecture, as depicted in Figure 3.1, right frame. If the Fourier Mix is removed from the decoder, we have a lighter model called FWin-S, which turns out to be a competitive design as well (see section 3.4).

Each encoder layer is defined as either a window attention layer or a Fourier Mix layer. The layers are interwoven, with the first layer being a window attention. Subsequent layers are connected by a distillation operation. For example, a 3-layer encoder will consist of a window attention layer, a distillation operation, and a Fourier Mix layer, another distillation operation, and finally another window attention layer. Figure 3.1 illustrates an encoder with 2 layers.

In the decoder, a layer composed of a masked window self-attention followed by a Fourier Mix and then window cross attention with layer normalization in between. Towards the end of the layer, convolutions and layer normalization are applied. Figure 3.1 shows a decoder with one layer.

In a self-attention layer, the query and key vectors have the same time dimension, allowing us to use the same window size to split these vectors. However, in the case of cross attention, this may not hold true, especially if the encoder includes dimension reduction layers. In such cases, the key and value vectors may have a smaller time dimension compared to the query vectors, which originate from the decoder. Additionally, the encoder and decoder inputs may

have different input time dimensions, as is the case in our specific problem. To ensure equal number of attention windows in cross attention, we divide the query, key, and value vectors based on the number of windows rather than the window size. This adjustment accounts for varying time dimensions and guarantees a consistent number of attention windows for the cross attention operation.

3.3.3 Complexity

With the replacements in the attention computation, our approach offers significant complexity reduction compared to Informer. In the encoder, the first attention layer partitions the time dimension of the input by a window of size w, by default $w = 24$, resulting in each window attention input having dimensions of $w \times d$. Thus the cost of this layer is $\mathcal{O}(nwd_{\text{model}})$. Furthermore, in the second attention layer, the computation of attention is completely replaced by the Fourier Mix layer, eliminating three linear projections for query, key, and value vectors. Since we apply the FFT over the time dimension and the model dimension, the total cost for the Fourier Mix layer is $\mathcal{O}(nd_{\text{model}} \log(nd))$. Overall Informer has complexity of $\mathcal{O}(n \log(n)d + n \log(n)d + n \log(n)d + n^2 d)$ and FWin is $\mathcal{O}(nwd + nd \log(nd) + nwd + nd \log(nd) + nwd)$. The $n^2 d$ cost of Informer comes from full cross attention in its decoder (Figure 3.1, left). In FWin, this cost is reduced to nwd by window cross attention (Figure 3.1, right).

Table 3.1: Model default parameters.

d_{model}	512	Window size	24
d_{ff}	2048	Cross Attn Window no.	4
n_heads	8	Epoch	6
Dropout	0.05	Early Stopping Counter	3
Batch Size	32	Initial Lr	1e-4
Enc.Layer no.	2	Dec.Layer no.	1

3.4 Experiments

3.4.1 Experimental Data and Details

The default setup of the model parameters is shown in Table 3.1. For Informer, we used their up to date code [1], which incorporated all the functionalities described recently in [105]. In our experiments, we average over 5 runs. The total number of epochs is 6 with early stopping. We optimized the model with Adam optimizer, and the learning rate starts at $1e^{-4}$, decaying two times smaller every epoch. For fair comparison, all of the hyper-parameters are the same across all the models which were trained/tested on a desktop machine with four Nvidia GeForce 8G GPUs.

Benchmark Datasets

Details of public benchmark datasets used in the main paper are described below:

ETT (Electricity Transformer Temperature)[2]: The dataset contains information of six power load features and target value "oil temperature". We used two hour datasets ETTh_1 and ETTh_2, and the minute level dataset ETTm_1. The train/val/test split ratio is 6:2:2.

Weather (Local Climatological Data)[3]: The dataset contains local climatological data

[1] https://github.com/zhouhaoyi/Informer2020
[2] https://github.com/zhouhaoyi/ETDataset
[3] https://www.ncei.noaa.gov/data/local-climatological-data

collected hourly in 1600 U.S. locations over 4 years from 2010 to 2013. The data consists of 11 climate features and target value “wet bulb". The train/val/test split ratio is 7:1:2.

ECL (Electricity Consuming Load) [4]: This dataset contains electricity consumption (Kwh) of 321 clients. The data convert into hourly consumption of 2 year and “MT_320” is the target value. The train/val/test split ratio is 7:1:2.

Exchange [5] [32]: The dataset contains daily exchange rates of eight different countries from 1990 to 2016. The train/val/test split ratio is 7:1:2.

ILI [6]: The dataset contains weekly recorded influenza-like illness (ILI) patients data from Centers for Disease Control and Prevention of the United States from 2002 to 2021. This describes the ratio of patients seen with ILI and the total number of the patients. The train/val/test split ratio is 7:1:2.

Additional Nonstationary Data

In addition to the benchmark datasets, we also validate our model on nonstationary dataset. Simulated New York/New England 16-generator 68-bus power system [6, 100]. The system has 88 lines linking the buses, and can be regarded as a graph with 68 nodes and 88 edges. The dataset has over 2000 fault events, where each event has signals of 10 second duration. These signals contain voltage and frequency from every bus, and current from every line. The train/val/test split is 1000:350:750.

[4] https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014.
[5] https://github.com/laiguokun/multivariate-time-series-data
[6] https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

3.4.2 Setup of experiments

For all the experiments to compute the errors, the encoder's input sequence and the decoder's start token are chosen from $\{24, 48, 96, 168, 336, 720\}$ for the ETTh_1, ETTh_2, Weather and ECL dataset, and from $\{24, 48, 96, 192, 288, 672\}$ for the ETTm dataset. The default window size is 24. We use a window size of 12 when the encoder's input sequence is 24. For the Exchange, ILI, and Traffic datasets, we use the same hyper-parameters as those provided in Autoformer. The encoder's input sequence is 96 and decoder's input sequence is 48. The window size is 24 for Exchange and Traffic. For ILI, we use an input length of 36 for the encoder and 18 for decoder, with window size of 6. The number of windows on the cross attention is set to 3. The models are trained for 6 epochs with learning rate adjusted by a factor of 0.5 every epoch.

For the power grid dataset [6, 100], the encoder and decoder inputs are set to 200. The prediction length is 700, and the window size is 25. The models are trained for 80 epochs with an early stopping counter of 30. The learning rate is adjusted every 10 epochs by a factor of 0.85. For the dengue dataset, we use the same hyper-parameters as in ILI dataset, because of its size and type.

3.4.3 Results and Analysis

Benchmarks

We present a summary of the univariate and multivariate evaluation results for all methods on 7 benchmark datasets in Tables 3.2 and 3.3. $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y-\hat{y}|$ and $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y-\hat{y})^2$ are used as evaluation metrics. The best results are highlighted in boldface, and the total count at the bottom of the tables indicates how many times a particular method outperforms others per dataset.

For the univariate setting, each method produces predictions for a single output over the time series. From Table 3.2, we observe the following:

(1) FWin and FWin-S achieve comparable performance.

(2) When comparing FWin and Informer, FWin outperforms the Informer by a margin of 50 to 16.

(3) FWin performs well on the ETT, Exchange, ILI datasets, it remains competitive for the Weather dataset. However, it performs not as well on the ECL dataset. This could be due to the differences caused by the dataset, which is common in time-series model [38], or Informer's ProbSparse hypothesis satisfied well for this dataset.

(4) The average MSE reduction is 19.60%, and MAE is about 11.88%, when comparing FWin with Informer.

For the multivariate setting, each method produces predictions based on multiple features over the time series. From Table 3.3, we observe that:

(1) The light model FWin-S leads the count at 34 total, followed closely by FWin with a total count of 30.

(2) In a head to head comparison, FWin outperforms Informer by a large margin (59 to 7).

(3) Though FWin and FWin-S have close accuracies in itemized-metric comparison on the benchmarks.

(4) The average MSE (MAE) reduction from Informer to FWin is about 16.33% (10.96%).

Table 3.2: Accuracy comparison on LSTF benchmark univariate data, best results highlighted in bold. Avg means the average results from all prediction lengths

Methods		Informer		FWin		FWin-S	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
$ETTh_1$	24	0.116	0.273	**0.060**	**0.196**	0.116	0.272
	48	0.170	0.333	**0.102**	**0.257**	0.141	0.302
	168	**0.149**	0.311	0.150	**0.308**	0.252	0.401
	336	0.160	0.323	**0.108**	**0.261**	0.397	0.519
	720	0.258	0.428	**0.105**	**0.253**	0.270	0.434
	Avg	0.171	0.334	0.105	0.255	0.235	0.386
$ETTh_2$	24	0.086	0.225	0.082	0.221	**0.075**	**0.212**
	48	0.164	0.318	**0.125**	**0.277**	0.140	0.299
	168	0.270	0.416	**0.220**	**0.375**	0.277	0.422
	336	0.324	0.458	**0.244**	**0.396**	0.277	0.425
	720	0.294	0.438	**0.261**	**0.412**	0.266	0.423
	Avg	0.228	0.371	0.186	0.336	0.207	0.356
$ETTm_1$	24	0.025	0.122	**0.015**	**0.096**	0.021	0.112
	48	0.055	0.181	**0.031**	**0.132**	0.032	0.135
	96	0.181	0.356	**0.050**	**0.175**	0.086	0.234
	288	0.279	0.450	0.179	0.341	**0.173**	**0.334**
	672	0.396	0.559	**0.133**	**0.286**	0.152	0.314
	Avg	0.187	0.334	0.082	0.206	0.093	0.226
Weather	24	0.113	0.249	**0.104**	**0.236**	0.111	0.243
	48	0.196	0.332	**0.172**	0.315	0.176	**0.310**
	168	0.257	0.376	0.259	0.391	**0.235**	**0.357**
	336	0.275	0.397	0.338	0.458	**0.262**	**0.388**
	720	0.259	0.389	0.291	0.421	**0.250**	**0.383**
	Avg	0.220	0.349	0.233	0.364	0.207	0.336
ECL	48	0.259	**0.359**	**0.249**	0.369	0.274	0.388
	168	**0.332**	**0.410**	0.408	0.479	0.403	0.472
	336	**0.378**	**0.441**	0.477	0.516	0.407	0.471
	720	**0.373**	**0.444**	0.568	0.577	0.376	0.455
	960	0.365	0.448	0.415	0.485	**0.359**	**0.448**
	Avg	0.341	0.420	0.423	0.485	0.363	0.447
Exchange	96	0.305	0.435	0.294	**0.409**	**0.281**	0.414
	192	1.345	0.902	0.679	0.622	**0.566**	**0.574**
	336	2.441	1.253	0.949	0.761	**0.785**	**0.703**
	720	1.933	1.106	**1.127**	**0.891**	1.253	0.897
	Avg	1.506	0.924	0.762	0.671	0.721	0.647
ILI	24	5.404	2.057	3.727	1.648	**3.491**	**1.597**
	36	4.384	1.849	3.124	1.534	**3.023**	**1.528**
	48	4.487	1.886	3.697	1.680	**3.293**	**1.605**
	60	5.179	2.035	4.141	1.788	**3.767**	**1.734**
	Avg	4.864	1.957	3.672	1.663	3.394	1.616
Count		8		32		26	

Power Grid

Informer's prediction accuracies on the benchmark datasets in Table 3.2 and Table 3.3 have been largely improved by recent transformers such as iTransformer [47], PatchTST [65] designed to prioritize variate information; and Autoformer [92], FEDformer [106] and ETSformer [89] designed with certain prior-knowledge of datasets, e.g. using auto-correlation

Table 3.3: Accuracy comparison on LSTF benchmark multivariate data, best results highlighted in bold. Avg means the average results from all prediction lengths

Methods		Informer		FWin		FWin-S	
Metric		MSE	MAE	MSE	MAE	MSE	MAE
$ETTh_1$	24	0.528	0.525	**0.483**	**0.499**	0.507	0.517
	48	0.764	0.665	**0.638**	**0.592**	0.695	0.626
	168	1.083	0.836	1.004	0.786	**0.885**	**0.742**
	336	1.270	0.920	1.094	0.821	**1.022**	**0.814**
	720	1.447	0.977	1.181	0.873	**1.087**	**0.846**
	Avg	1.018	0.785	0.880	0.714	0.839	0.709
$ETTh_2$	24	**0.455**	**0.508**	0.550	0.566	0.551	0.568
	48	2.368	1.241	**0.774**	**0.664**	0.792	0.680
	168	5.074	1.910	**2.309**	**1.136**	2.767	1.327
	336	3.116	1.460	**2.461**	**1.187**	2.663	1.337
	720	4.193	1.778	**2.847**	**1.286**	3.258	1.530
	Avg	3.041	1.379	1.788	0.968	2.006	1.088
$ETTm_1$	24	0.346	0.397	**0.305**	**0.375**	0.322	0.389
	48	0.480	0.482	**0.408**	**0.444**	0.436	0.467
	96	0.555	0.531	**0.517**	**0.517**	0.573	0.553
	288	0.943	0.746	0.831	**0.688**	**0.794**	0.691
	672	0.903	**0.729**	1.119	0.838	**0.890**	0.741
	Avg	0.645	0.577	0.636	0.572	0.603	0.568
Weather	24	0.335	0.388	**0.310**	**0.363**	0.316	0.369
	48	0.395	0.434	**0.379**	0.419	**0.379**	**0.415**
	168	0.625	0.580	0.561	0.539	**0.544**	**0.530**
	336	0.665	0.611	0.630	0.585	**0.598**	**0.574**
	720	0.657	0.604	0.686	0.614	**0.576**	**0.560**
	Avg	0.535	0.523	0.513	0.504	0.483	0.490
ECL	48	0.293	0.382	0.291	**0.371**	**0.287**	0.372
	168	**0.292**	0.386	0.302	0.379	0.295	**0.378**
	336	0.422	0.467	0.327	0.399	**0.307**	**0.388**
	720	0.600	0.559	0.344	0.408	**0.311**	**0.390**
	960	0.871	0.714	0.362	0.421	**0.314**	**0.393**
	Avg	0.496	0.502	0.325	0.396	0.303	0.384
Exchange	96	0.991	0.797	0.828	0.733	**0.762**	**0.694**
	192	1.175	0.859	**1.148**	**0.889**	1.178	**0.889**
	336	1.581	0.999	**1.301**	**0.960**	1.344	0.971
	720	2.643	1.356	2.071	1.196	**2.036**	**1.177**
	Avg	1.598	1.003	1.337	0.945	1.330	0.933
ILI	24	6.048	1.698	3.881	1.308	**3.849**	**1.298**
	36	5.871	1.681	4.036	1.331	**4.024**	**1.316**
	48	5.171	1.551	**4.334**	**1.404**	4.431	1.406
	60	5.273	1.553	**4.547**	1.443	4.600	**1.438**
	Avg	5.591	1.621	4.200	1.372	4.226	1.365
Count		4		30		34	

or trend/seasonality decomposition. Like Informer, FWin has no prior-knowledge based operation, which helps to generalize better on non-stationary time series where seasonality is absent. Such a situation arises in post-fault decision making on a power grid where predicting transient trajectories is important for system operators to take appropriate actions [34], e.g., a load shedding upon a voltage or frequency violation. We carry out experiments on a simulated New York/New England 16-generator 68-bus power system [6, 100]. Table 3.4

shows that FWin and FWin-S improve or maintain Informer's accuracy in a robust fashion while the five recent transformers pale in comparison. Figure 3.4 illustrates model predictions on the power grid dataset [6, 100]. FWin and Informer outperform FEDformer, Autoformer, iTransformer, and PatchTST.

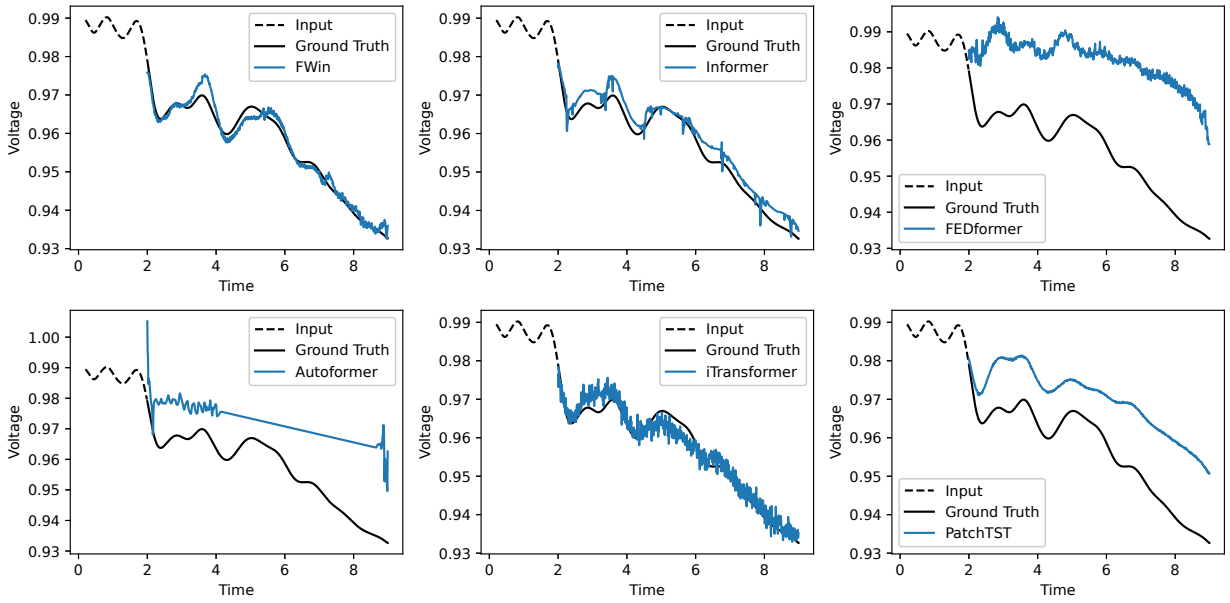


Figure 3.4: Multivariate post fault prediction comparison (voltage vs. time in second) on power grid data [6, 100]: {FWin,Informer} outperform (FED,Auto,iTrans)formers and PatchTST. The dashed line under 2 second duration is the input, to the right of which are the predictions vs. the ground truth (in black).

Table 3.4: Post-fault voltage prediction accuracy comparison on power grid dataset [6, 100] with input length of 200 and prediction of 700. Best results highlighted in bold.

Type	Multivariate		Univariate	
Method	MSE	MAE	MSE	MAE
FWin	0.063	0.141	**0.091**	0.141
FWin-S	**0.043**	**0.101**	0.092	**0.132**
Informer	0.049	0.113	0.111	0.170
FEDformer	0.245	0.311	0.272	0.310
Autoformer	0.390	0.403	0.367	0.388
ETSformer	0.339	0.381	0.303	0.322
iTransformer	0.071	0.135	0.101	0.165
PatchTST	0.211	0.278	0.138	0.188

Speed Up

Besides performance and robustness, the inference and training times of the models (in particular the former) are also of our interest and are summarized below:

(1) Compared to Informer, FWin achieves average speed up factors of **1.7** and **1.4** for inference and training times respectively (see Table 3.5). The ILI dataset has the lowest speed-up factor because both the input and the prediction lengths are small.

(2) FWin's inference and training times are very close to those of FWin-S. This indicates that the Fourier Mix layer in the decoder adds a minimal overhead to the overall model. The FWin-S model exhibits the fastest inference and training times, as expected because it is the model with smallest parameter size here.

(3) FWin has approximately 8.1 million parameters, whereas Informer has around 11.3 million parameters under default settings, resulting in a reduction about 28%. On the ETTh1 prediction length task of 720, Informer has 5.85 GFLOPs, while FWin has 5.32 GFLOPs under identical setting. This confirms our analysis that Informer's cross attention is full attention, whereas FWin's windowed cross attention is much more efficient.

(4) *The inference time for Informer increases with prediction length. FWin's inference time exhibits minimal growth.* The Exchange dataset demonstrates this effect as we used the same input length for all prediction lengths, and ran the models on a single GPU (see Supplemental material).

(5) *FWin's inference time is about 1.6 to 6 times faster compared to ETSformer, FEDformer, and Autoformer* (see Table 3.6).

Table 3.5: Average inference/training speed-up factors of FWin vs. Informer

Data	Feature	Inference	Train
$ETTh_1$	Multivariate	1.66	1.34
$ETTh_1$	Univariate	1.80	1.64
$ETTm_1$	Multivariate	1.70	1.30
$ETTm_1$	Univariate	1.68	1.34
Weather	Multivariate	1.70	1.28
Weather	Univariate	1.99	2.01
ECL	Multivariate	1.59	1.10
ECL	Univariate	2.01	1.53
Exchange	Multivariate	1.77	1.25
Exchange	Univariate	1.69	1.25
ILI	Multivariate	1.56	1.19
ILI	Univariate	1.62	1.29
Average		1.73	1.38

Table 3.6: Average inference speed-up factors of FWin vs. SOTAs

Data	Feature	Autoformer	FEDformer	ETSformer
$ETTh_1$	Multivariate	7.29	6.18	1.87
$ETTh_1$	Univariate	5.08	4.60	1.42
$ETTm_1$	Multivariate	5.77	5.34	1.27
$ETTm_1$	Univariate	6.14	5.86	1.38
ECL	Multivariate	5.65	5.08	1.67
ECL	Univariate	6.39	5.27	1.68
Average		6.05	5.39	1.55

3.4.4 Ablation Studies

Benefits of Combining Fourier and Window Attention

We examined the benefits of combining window attention and Fourier mixing by experiments on the ETTh1 and Weather dataset. Table 3.7 shows that Fourier-mixed window attention outperforms using Fourier mixing alone (FNet [33]) in accuracy. In addition, it also suggests that Fourier-mixed window attention is better overall than shifted window attentions in the encoder. In view of Table 3.2 and 3.3, FWin-S is better than Swin on metric 720 in MSE (comparable in MAE).

Table 3.7: FWin vs. FNet [33] (replacing ProbSparse attention of Informer by Fourier-Mix followed by an FC layer) and FWin vs. Swin [49] (replacing Fourier Mix in FWin by a shifted window attention) on multivariate data. Best results highlighted in bold.

Methods		FNet		FWin		Swin		FWin	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	0.490	0.502	**0.483**	**0.499**	0.567	0.540	**0.483**	**0.499**
	48	**0.562**	**0.543**	0.638	0.592	0.685	0.627	**0.638**	**0.592**
	168	1.052	0.806	**1.004**	**0.786**	1.016	0.810	**1.004**	**0.786**
	336	1.194	0.869	**1.094**	**0.821**	1.161	0.877	**1.094**	**0.821**
	720	1.349	0.948	**1.181**	**0.873**	**1.095**	**0.844**	1.181	0.873
Weather	24	0.331	0.378	**0.310**	**0.363**	**0.310**	**0.362**	**0.310**	0.363
	48	0.432	0.459	**0.379**	**0.419**	0.409	0.443	**0.379**	**0.419**
	168	0.596	0.561	**0.561**	**0.539**	0.630	0.576	**0.561**	**0.539**
	336	**0.609**	**0.580**	0.630	0.585	0.635	**0.580**	**0.630**	0.585
	720	0.724	0.640	**0.686**	**0.614**	**0.637**	**0.592**	0.686	0.614
Count		4		16		7		14	

Effect of Window Size Parameter

Window size is an important parameter for FWin. In this section we will explore the effect of window size to model performance across many datasets presented in the paper. We present the results in Table 3.8. We observe that across the datasets, window size of 6 provides the best results overall. Window size of 1 provides competitive results compare to the best window size of 6. This supports by the theoretical results presented in Theorem 2.5. In general, under various window sizes, the performance is consistent. Optimizing the window size for each data set may increase performance of our model. However, to keep the experiments consistent, we decide to keep the window size at a fixed constant 24. The time scale of a dataset may impact the choice of window size. Many of the datasets have hourly time scale, thus choosing a window size of 24 is meaningful in covering a daily observation. The flexible choice of window size enables the practical application of the method to various datasets with well-known temporal dependencies.

Table 3.8: Accuracy comparison of FWin model using different window sizes for various datasets on the multivariate task. Best results highlighted in bold.

Window Size		1		2		4		6		12		24	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24	0.468	0.486	0.474	0.490	0.481	0.498	**0.464**	**0.484**	0.489	0.505	0.506	0.515
	48	0.599	0.565	0.606	0.571	0.611	0.580	0.592	0.569	**0.575**	**0.554**	0.586	0.561
	168	0.915	0.741	0.930	0.748	0.917	0.752	**0.903**	**0.737**	0.904	0.741	1.076	0.822
	336	1.049	0.782	1.002	0.769	1.000	0.774	**0.978**	**0.766**	0.998	0.772	1.070	0.813
	720	1.096	0.837	1.090	0.840	**1.082**	**0.834**	1.107	0.844	1.133	0.859	1.205	0.884
Weather	24	0.310	0.361	0.310	0.365	0.311	0.364	**0.308**	**0.361**	**0.308**	0.362	0.309	0.362
	48	0.382	**0.419**	0.380	0.421	**0.378**	0.420	0.380	0.420	0.381	0.421	0.381	0.421
	168	0.561	0.542	0.561	0.542	0.570	0.547	**0.558**	0.539	0.550	**0.535**	0.561	0.539
	336	0.631	0.592	0.626	0.586	**0.610**	**0.577**	0.629	0.587	0.612	0.578	0.618	0.580
	720	0.705	0.619	0.699	0.623	0.702	0.627	0.685	0.617	**0.679**	**0.611**	0.691	0.620
Exchange	96	0.738	0.695	0.798	0.723	0.793	0.717	**0.714**	**0.683**	0.775	0.717	0.806	0.729
	192	1.195	0.910	**1.066**	**0.869**	1.297	0.951	1.261	0.941	1.132	0.891	1.136	0.886
	336	1.314	0.964	**1.244**	0.941	1.340	0.970	1.270	**0.940**	1.389	0.992	1.302	0.962
	720	1.916	1.134	2.034	1.166	1.944	1.137	**1.885**	**1.128**	2.225	1.225	2.055	1.176
Count		1		3		5		14		6		0	

FWin and Nonparametric Regression

The full self-attention function [82] can be conceptualized as an estimator in a non-parametric kernel regression problem in statistics [63]. Let the key vectors serve as the training inputs and the value vectors as the training targets. The key-value pairs $\{k_j, v_j\}$ for $j = 1, \ldots, n$, come from the model

$$v_j = f(k_j) + \epsilon_j, \tag{3.2}$$

where f is an unknown function to be estimated and ϵ_j are zero mean independent noisy perturbations. Let the key vectors $k_1, k_2, \ldots, k_N$ be i.i.d. samples from a distribution function $p(k)$, and the key-value pairs $(v_1, k_1), \ldots, (v_N, k_N)$ be i.i.d. samples from the joint density $p(k, v)$. Since $\mathbb{E}[v_j|k_j] = f(k_j)$, the classical Nadaraya-Watson method [59, 66, 69] approximates p by a sum of Gaussian kernels and gives the estimate of f below:

$$\hat{f}_\sigma(k) := \sum_{j=1}^{n} \frac{v_j \phi_\sigma(k - k_j)}{\sum_{l=1}^{n} \phi_\sigma(k - k_l)}, \tag{3.3}$$

where $\phi_\sigma(\cdot)$ is the isotropic multivariate Gaussian density function with diagonal covariance matrix $\sigma^2\mathbf{I}_D$. In particular if $k = q_i$, and the k_j's are normalized, one obtains

$$\hat{f}_\sigma(q_i) = \frac{\sum_{j=1}^n v_j \exp(q_i k_j^T/\sigma^2)}{\sum_{l=1}^n \exp(q_i k_l^T/\sigma^2)} = \sum_{j=1}^n \text{softmax}(q_i k_j^T/\sigma^2) v_j. \tag{3.4}$$

Letting $\sigma^2 = \sqrt{d_{\text{model}}}$, where d_{model} is the dimension of q_i (k_j), turns estimator (3.4) into the softmax self-attention.

To build a window attention, we allow the query vector q_i to interact only with nearby key and value vectors. Thus the window version of the softmax estimator is

$$\bar{f}_\sigma(q_i) := \frac{\sum_{j\in J(i)} v_j \exp(q_i k_j^T/\sigma^2)}{\sum_{l\in J(i)} \exp(q_i k_l^T/\sigma^2)} = \sum_{j\in J(i)} \text{softmax}(q_i k_j^T/\sigma^2) v_j$$

where $J(i)$ is the index set that corresponds to the set of keys the query q_i interact with. In view of the fully connected (MLP) layer after the Fourier mixing layer and before the output in Figure 3.1, we define the analogous FWin estimator for the regression model as

$$\tilde{f}_\sigma(q_i) := A \cdot \mathcal{R}(\mathcal{F}(\bar{f}_\sigma(q_i))), \tag{3.5}$$

where $\mathcal{R}$ takes the real part, $\mathcal{F}$ is the discrete Fourier transform, $\cdot$ represents matrix multiplication, and A is a real matrix to be learned from the training data by minimizing the sum of squares error (MSE) of the regression model (3.2).

Kernel Regression Experiment To examine the differences among the three estimators $\hat{f}$, $\bar{f}$, and $\tilde{f}$, we opt for the Laplace distribution function $f = \exp(-\alpha|x|)$, for $\alpha = 0.01$, as the ground truth nonlinear function acting componentwise on the input to the regression model (3.2). We use a set of query, key, value vectors from Informer† [104] on ETTh_1 multivariate data set with prediction length (metric) of 720. We choose this particular data set because Informer† [104] with full softmax self-attention has the best performance there. The key

vectors may not satisfy the theoretical i.i.d. assumption [63]. In this experiment, we have a set of 168 query and key vectors from each of the 8 heads. Denote query q_i, and key k_j vectors for $i, j = 1, 2, \ldots, 168$. The value vectors are $v_j = f(k_j)$. We divide the data into 136 vectors for training and the remaining 32 for testing. The mean square error (MSE) in testing of the estimator $\hat{f}(q_{i_n})$ for $n = 1, 2, \ldots, 32$, are labeled full estimator in Figure 3.5. Similarly, we compute MSEs for $\bar{f}_\sigma(q_{i_n})$, and $\tilde{f}(q_{i_n})$, using window size of 4, where A is learned by solving a least squares problem. Figure 3.5 compares the MSE of the three estimators over data from 8 heads. We observe that the FWin estimator consistently outperforms the window attention estimator, and approaches the full softmax attention. In heads 0/1/4, FWin outperforms the full softmax attention estimator which is not theoretically optimal for the regression task [63]. In conclusion, the regression experiment on the three estimators indicates that FWin is a simple and reliable local-global attention structure with competitive capability, lending added support to its robust performance.

3.4.5 Condition Number of Attention Matrix

Theorem 2.5 requires the attention matrix to be BDI. In this section, we verify that in practice BDI is satisfied by many of the datasets here. The experimental set-up is:

- Run simulations on the Informer model using full attention instead of Probsparse.
- Collect the full attention matrix of the first encoder block of the Informer.
- Given a window size w, compute the condition numbers of $w \times w$ sub-matrices along the diagonal of the attention matrix.
- If the condition numbers are finite, then BDI is true and this instance is collected for a histogram plot, otherwise it is counted as a failure.

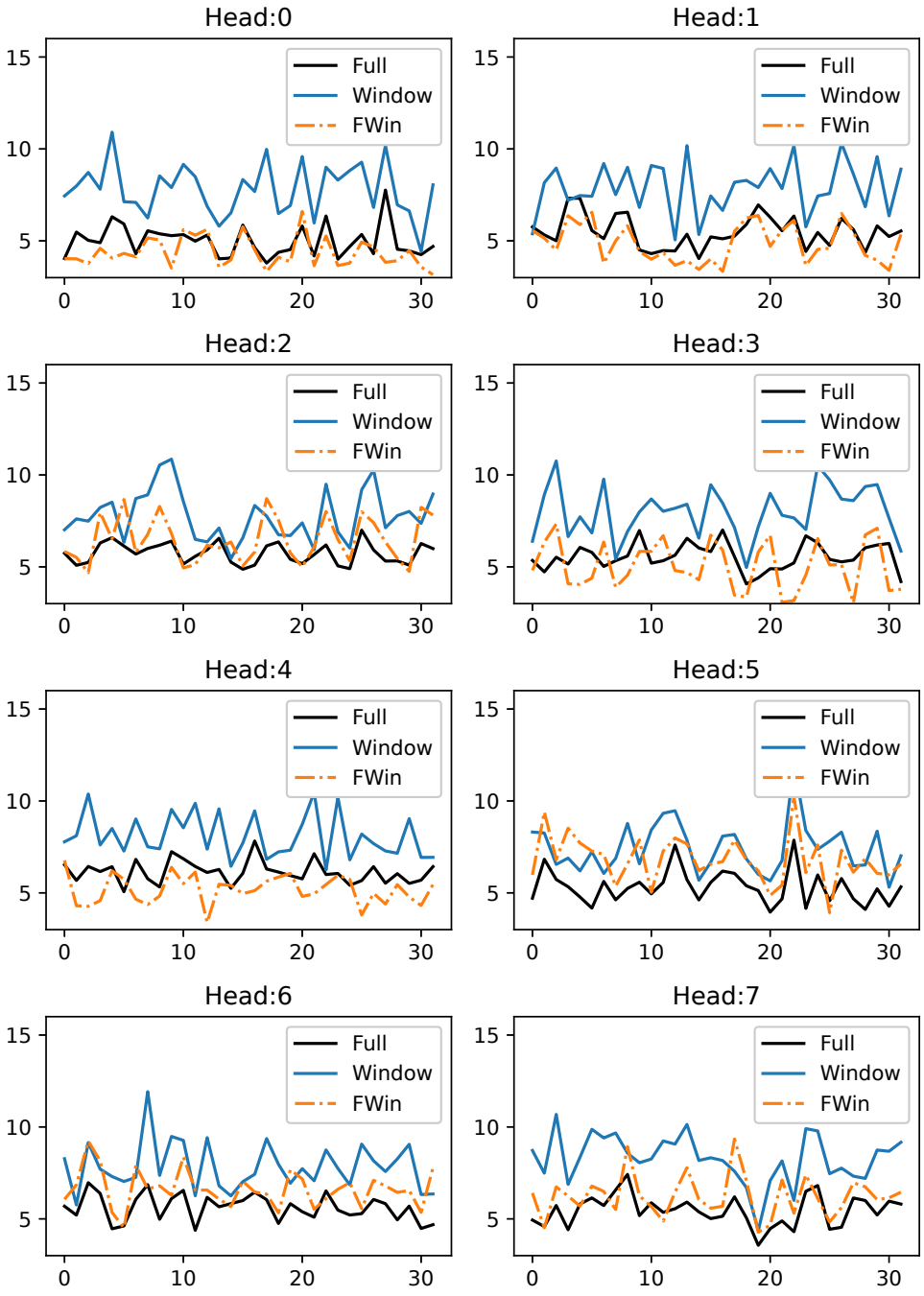


Figure 3.5: MSE error vs. query number comparison of full softmax attention (black), window attention (blue) and FWin attention (orange) in the non-parametric regression model (3.2) based on key vectors of a full attention layer of Informer† [104] trained from the ETTh_1 data set.

Using the procedure above, we plot all condition numbers for all simulations of the ETTh_1, Weather, and ILI dataset. For all the simulations, we use the same hyper-parameters as in

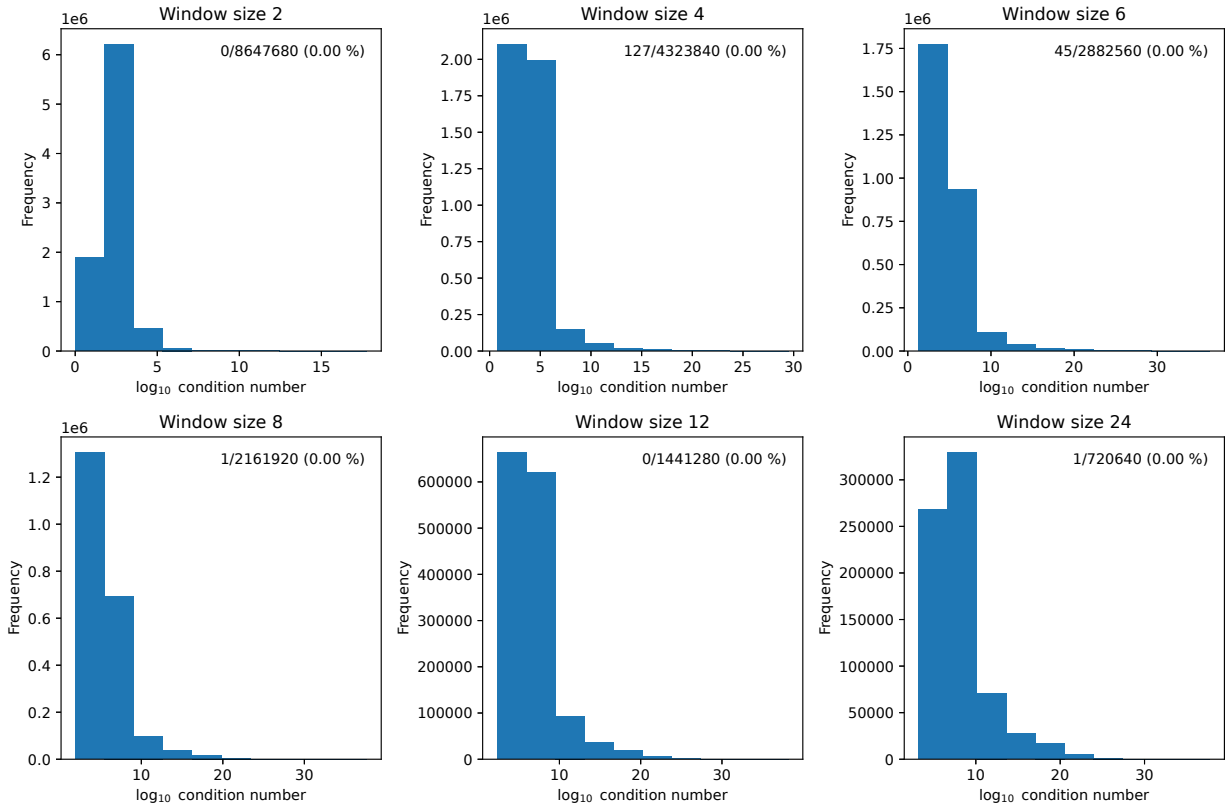


Figure 3.6: Condition numbers under various window sizes for ETTh1 dataset. In the top-right corner of each subplot, the label "n/m (k%)" denotes the number of infinite condition numbers (n) over the total number of condition numbers (m), together with the corresponding percentage (k).

the experiment sections. Due to memory space constraint, we report the first 11 batches of the test datasets.

From Figure 3.6, 3.7, and 3.8, we observe that ETTh1, Weather, ILI datasets satisfy the assumption of the theorem relatively well. In particular, ETTh1 has a few infinite condition number out of millions. For ILI, approximately 0.4–0.5% of the condition numbers are infinite, while the Weather dataset has around 3% infinite condition numbers. We also noted that the condition numbers increase with the window size for WTH dataset. We selected these three datasets to demonstrate the robustness of the assumption, as they span different temporal granularities from minutes in WTH, to hours in ETTh1, and weeks in ILI. Combining this observation with the results from Table 9, we suggest using a small window

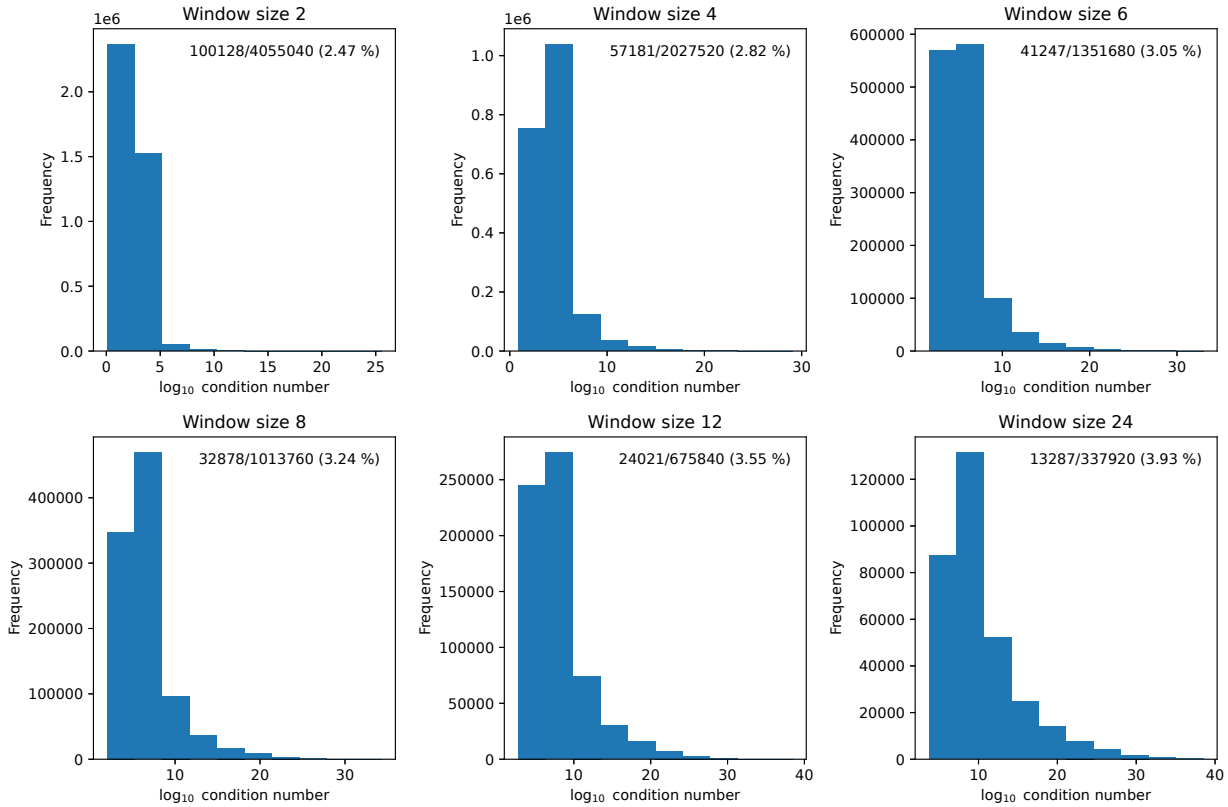


Figure 3.7: Condition numbers under various window sizes for WTH dataset. In the top-right corner of each subplot, the label "n/m ($k\%$)" denotes the number of infinite condition numbers (n) over the total number of condition numbers (m), together with the corresponding percentage (k).

size, as it maintains competitive performance compared to larger window sizes while ensuring that the BDI condition is satisfied in practice.

3.5 Conclusion

We introduced FWin Transformer and its light weight version FWin-S to successfully reduce the complexity, and improve/maintain the accuracy of Informer by replacing its ProbSparse and full attention layers with window attention and Fourier mixing blocks in both encoder and decoder. The FWin attention approach does not rely on sparse attention hypothesis or periodic like patterns in the data, hence also achieves robustness especially on highly non-

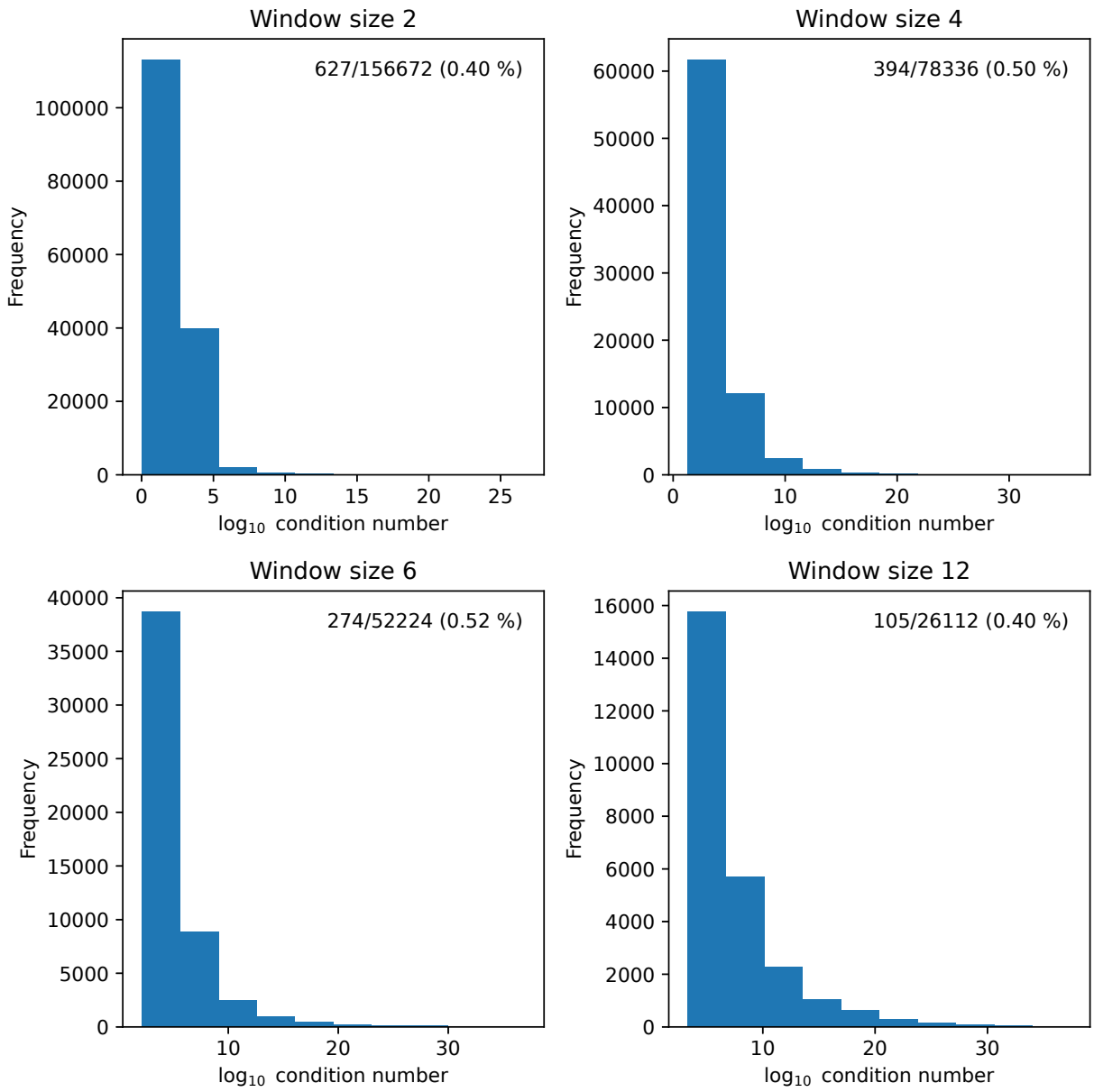


Figure 3.8: Condition numbers under various window sizes for ILI dataset. In the top-right corner of each subplot, the label "n/m ($k\%$)" denotes the number of infinite condition numbers (n) over the total number of condition numbers (m), together with the corresponding percentage (k).

stationary data. The experiments on uni/multi-variate datasets and theoretical guarantees demonstrated FWin's merit in fast and reliable inference on LSTF tasks.

In future work, we plan to (1) embed FWin in an encoder only architecture (e.g. iTrans-

former [47]) for acceleration and improved generalization; (2) optimize the FWin approach towards accurate, robust and fast transformers in challenging non-stationary real-world LSTF applications.

Chapter 4

FWin transformer for dengue prediction under climate and ocean influence

Dengue fever is one of the most deadly mosquito-born tropical infectious diseases. Detailed long range forecast model is vital in controlling the spread of disease and making mitigation efforts. In this chapter, we examine methods used to forecast dengue cases for long range, in particular apply FWin on challenging problem. The dataset consists of local climate/weather in addition to global climate indicators of Singapore from 2000 to 2019. We utilize newly developed deep neural networks to learn the intricate relationship between the features. The baseline models in this chapter are in the class of recent transformers for long sequence forecasting tasks. We found that a Fourier mixed window attention (FWin) based transformer performed the best in terms of both the mean square error and the maximum absolute error on the long range dengue forecast up to 60 weeks.

4.1 Introduction

The modeling and forecasting of vector-borne diseases (VBD) are scientifically challenging and important to the society at large due to their complex non-local temporal dependencies in the data as well as external climate factors. VBD originated from tropical countries pose serious public health risks to both local and global communities. Among them, malaria and dengue fever are the two most deadly mosquito-borne tropical infectious diseases, with about 240 million malaria cases globally and 440,000 malaria deaths annually, and 50-100 million dengue cases. Currently, no tested vaccine or treatment is available to stop or prevent all types of dengue fever. Thus modeling dengue disease evolution is particularly significant.

The correlation of weather/climate with VBD evolution is well-documented ([7, 53, 58, 62, 103] among others). With global warming upon us, higher temperatures create more habitats for mosquitoes to infect unexposed human populations and spread diseases. Just in July 2023, about 80 million Americans experienced a heat index of at least 105 degrees according to the National Weather Service. The extreme heat waves prompt irregular typhoons in Asia and flash floods in north America. Such intense precipitation and flooding events become more frequent and longer (unless humans essentially stop adding carbon dioxide to the atmosphere), behaving as *strongly non-stationary* instead of traditional seasonal signals. They can favor mosquito breeding and survival to further complicate VBD evolution. Besides temperature, ocean currents can also influence the infection dynamics of VBD [7, 53].

Knowledge of weather and population susceptibility cycles is known to help prediction, see [41, 56] and references therein. A new challenge of VBD modeling is to extract critical information from complex correlations and multiple factors *in the presence of intense transients lasting for week long periods due to extreme climate events.* Ensemble machine learning (ensemble support vector machines [56]), recurrent neural networks (RNN) and regression [41], among other tools [71] have been utilized in the past to study the effects of climate and

seasonal weather. However due to either stationary hypothesis or simple decision boundary or one time unit ahead recursion, these existing tools are not well-suited for predicting the aftermath of intense non-stationary behavior in the data.

In this chapter, we study a local-global attention based efficient transformer [80] for non-stationary VBD modeling and prediction in a specific spatial region. For simplicity, we leave the spatial synchrony issue [4] (coupling among neighboring regions) to a future study. Transformer networks equipped with attention blocks [82] have powered the recent breakthroughs of natural language processing (NLP and Open AI's Chat-GPT) where long range temporal correlations exist. Since VBD data are not as abundant as in NLP, light weight and efficient transformer networks are more suitable and will be our main objects of study here. A successful strategy to reduce computational complexity of transformers is to approximate the full attention by a local attention (e.g. window attention [49]) globalized by a subsequent mixing step (e.g. through shifting [49] or shuffling [29] windows among other treatments). Recently, Fourier transform based mixing has been found competitive in accuracy and efficiency in both training and inference on long sequence time series forecasting tasks. On standard benchmarks [104] as well as highly non-stationary power grid data [100], Fourier-mixed window attention (FWin, [80]) out-performs prob-sparse attention [104] and other recent models such as Autoformer [92], FEDformer [106], ETSformer [90] and PatchTST [65]. We aim to continue this line of inquiry here on multi-variate dengue and climate data in Singapore which provide a real-world VBD data set to help understand and evaluate transformers as a new tool for advancing public health.

The innovations of this chapter include a comprehensive transformer based generative model to encompass essential driving mechanisms of VBD evolution and make fast prediction of non-stationary dynamic behavior over a longer temporal duration than existing methods. The ideas of attention and multi-variate data fusion have been applied before in the context of infectious disease prediction, e.g. hand-foot-mouth disease and hepatitis beta virus [99],

influenza and dengue [61, 107] etc. However, 1) the prior approaches of using a composition of a linear projection and softmax normalization as attention [99, 107] or a special form of attention in machine translation [55] adopted in [61] is not robust compared to the canonical quadratic (Q,K,V) form [82]; 2) these methods have not been evaluated on the public long sequence time-series forecast benchmarks [104]; and 3) the models by design can only make one time unit predictions which is not enough for any early warning. Though the attention enhanced LSTM in [61] was extended from 1 month ahead to 2/3/6 month ahead predictions, the performance was increasingly worsening. Methods being a generative transformer can naturally make multi-time unit predictions.

The rest of the chapter is organized as follows. In section 2, we discuss the Singapore dengue data set and two prediction tasks. In section 3, we give an overview of transformer models in this paper including FWin transformer. In section 4, we verify the hypothesis of the FWin equivalence theorem. In section 5, we compare results from the transformer models on the two prediction tasks and give our interpretations. In section 6, we provide an ablation study on the choice of window lengths of FWin transformer, and effect of shorter inputs. Concluding remarks are in section 7.

4.2 Dataset and Task

4.2.1 Data

The dataset contains 1000 weeks of Singapore's weekly dengue data spanning from 2000 to 2019. The dataset's features include the following variables: the cases number, average temperature, precipitation, Southern Oscillation Index (SOI), Oceanic Niño Index (ONI) total, ONI anomaly, Indian Ocean Dipole (IOD), IOD East, NINO1+2, NINO3, NINO4, NINO3.4, and the respective NINO anomaly. We provide descriptions of important features here: 1)

SOI is the difference from average air pressure in the western Pacific, measured in Darwin, Australia, to the difference from average pressure in the central Pacific, measured at Tahiti [44], 2) ONI is running 3-month average sea surface temperatures in the east-central tropical Pacific between 120W-170W [43], 3) IOD is the surface temperature difference between the western and eastern tropical Indian ocean [70], 4) NINO1+2, 3, 3.4, 4 represent the sea surface temperature corresponds with the region across the Pacific ocean with coordinates (0-10S, 90W-80W), (5N-5S, 150W-90W), (5N-5S, 170W-120W), (5N-5S, 160E-150W) respectively. Many of the features are reported in a daily or monthly interval. Whenever data are available at a coarser temporal resolution than weekly, we select the data corresponding to the month of the first day of that week. In cases where data is available at a finer temporal resolution than weekly, we calculate the average data for that week. The train/val/test split ratio is 6/2/2. We present a sample of the features of the dataset with the corresponding split ratio in Fig. 4.1. Data normalization applies to the entire dataset before passing it to the model. This means that each feature in the dataset will have zero mean and variance equal to 1. We label the data set as Singapore Dengue (SD). The processed data is available upon request.

4.2.2 Prediction Task

In this chapter, we are interested in predicting the dengue cases number using all provided features. The appropriate task for this is Multivariate to Univariate (MS). For this prediction task, we utilize information from all features from the past m time steps, and predict the number of dengue cases for the next n time steps. In the models, we set m to be 36 by default, and $n \in \{24, 36, 48, 60\}$.

Since we do not need to predict the other features, e.g. precipitation, we introduce a new prediction task where the inputs consist of $m + n$ time steps of the predictor variables (PV). However, the input will only contain m time steps of the response variable (RV), with the

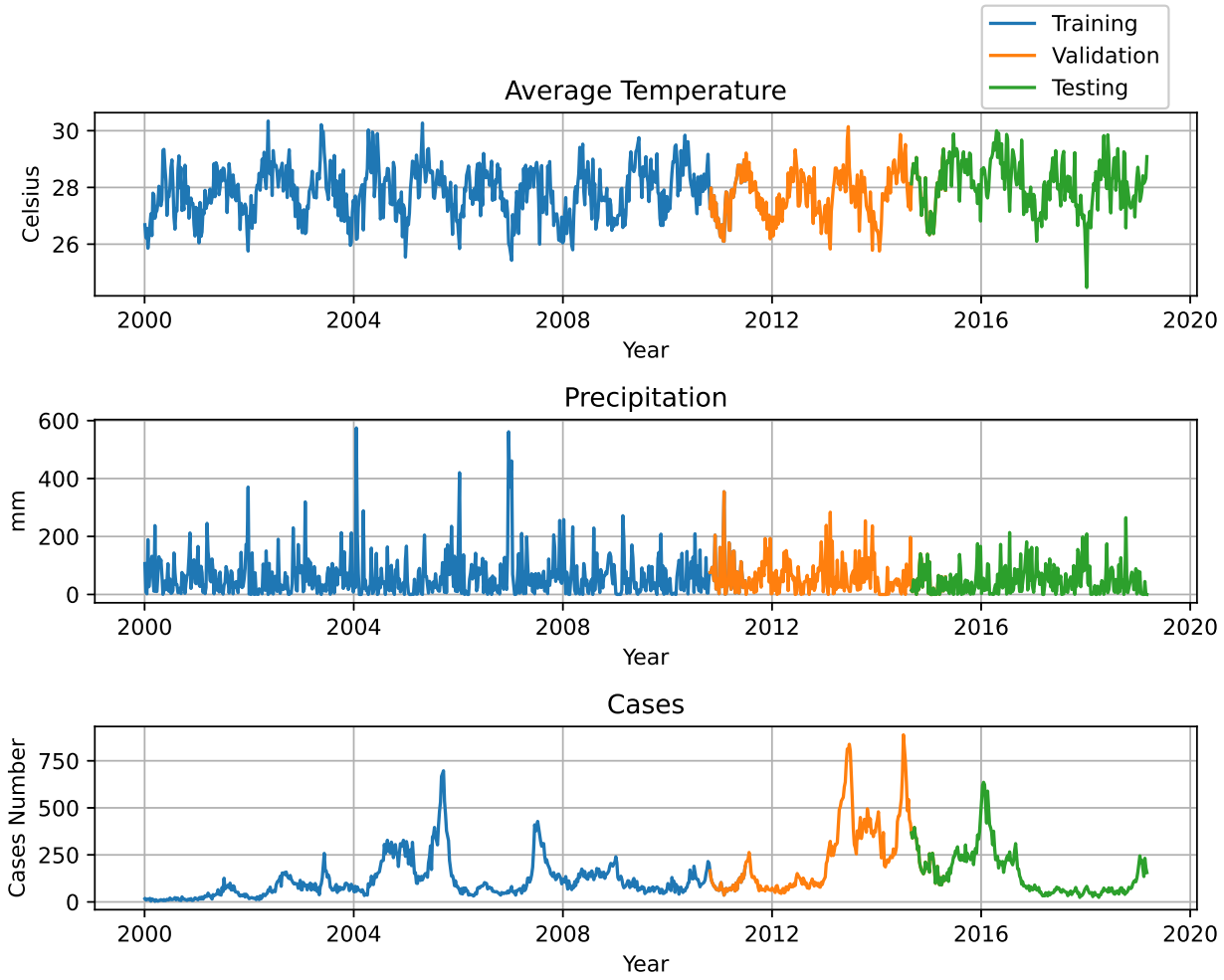


Figure 4.1: Sample features of the dataset. The plot includes the average temperature and precipitation from 2000 to 2019 with the dengue cases number. Here blue, orange, green indicate training, validation and testing split respectively.

RV at the remaining n time steps set to 0. We refer to this task as Modified Multivariate to Univariate (MM). The rationale behind this task is that if we have accurate forecasts for the PV, then we can leverage this information to improve the prediction of the RV. Later, we will show that this modification increases the prediction power of the models. Fig. 4.2 provides an overview of the overall structure of these tasks.

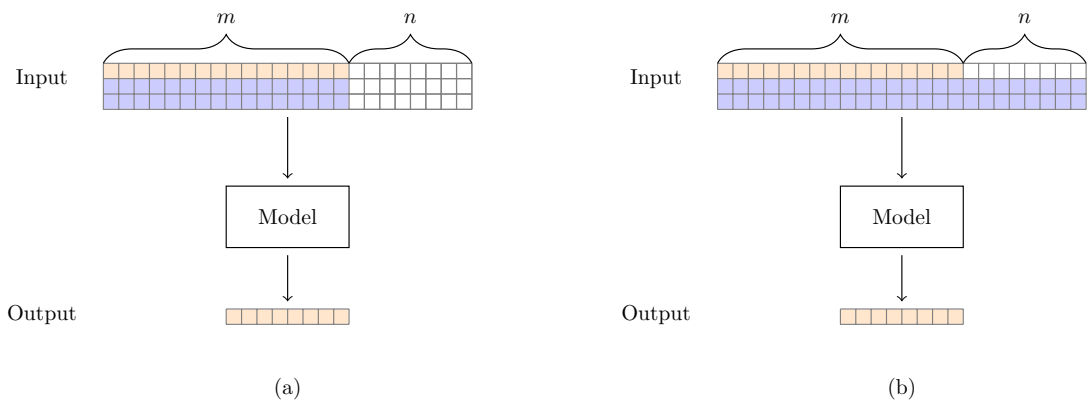


Figure 4.2: (a) Input-Output structure of MS task. (b) Input-Output structure of MM task. Here orange shaded cells indicate non-zero value of the RV, blue shaded cells indicate non-zero value of the PV, and white cells indicate zero padded value.

4.3 Models

Time scale latency between the effect of weather features (such as the abundance of water for larvae, and symptoms of disease in the host) varies depending on the location. This latency could range from up to 6 months of delay to as short as just 6 weeks [77]. Thus choosing a lag order (an integer parameter) in a standard statistical model such as ARIMA or VAR is non-trivial for our dataset. Moreover, traditional methods require choosing significant features, i.e. Pearson's correlation, before passing them in to the model [60]. Also features are assumed to be linearly correlated, which may not be true in general. In cases where a nonlinear model is necessary, we must have some functional form to describe the relationship between the climate features and cases number. To address this issue, we will utilize recent deep neural networks, in particular attention based neural networks. We will discuss in detail the structure in the following sections. Below are some reasons why a neural network may resolve some of the issues we have with traditional methods. First, a deep neural network has the ability to extract important features through learning. Second, we treat each time step of the input as a token passing into the neural network. Given a certain time step information (token), the attention mechanism allows a particular token to focus on other tokens, this

alleviates the need to pre-define a lag order. Of course, if one has a prior knowledge of a lag order, then it can be incorporated into the model such that one only allows tokens in that particular lag window to affect the final prediction. We will show this as an advantage of FWin. Lastly, other than choosing an architecture for a deep neural network, since in most cases it is a universal approximator, we do not need to have an exact equation to describe the nonlinear relationship between predictor and response variables.

In order to accomplish the dengue forecasting task, we will utilize some of the recently developed deep neural network models. We will compare the following models: FWin [80], Informer [104], FEDformer [106], Autoformer [92], ETSformer [90], and PatchTST [65]. We only include transformer based models in this paper, because these models are the current SOTAs for time series tasks. They demonstrated to out perform statistical models, RNN and CNN on multiple benchmarks [65, 104, 106].

4.3.1 Background

Transformer

Transformer is a basis for many of the newly developed models. Transformer has an encoder-decoder structure. The encoder maps an input x to a representation y. Then the decoder accepts y as an input to generate an output z. Encoder composes of stacks of self attention and feed forward layers. Similarly, decoder composes of stacks of masked self attention, cross attention, and feed forward layers [82].

4.3.2 Other Models

FEDformer uses an encoder-decoder structure as well, however, instead of canonical attention, it uses the frequency enhanced attention. It applies Fourier transform to the input, then select a few modes to compute attention. In addition, it incorporates a seasonal-trend decomposition layers to capture the global properties of time series [106]. Similarly, Autoformer uses Series Decomposition and Auto-Correlation instead [92]. ETSformer uses similar structure with frequency attention and exponential smoothing attention to extract growth and seasonal information from the inputs. Here the attention is weighted with an exponential term that favors nearby tokens. Then it uses such information selection in the decoder to forecast the future horizon [90]. Lastly, PatchTST only uses the encoder structure of the Transformer. It first divides the input into patches, and passes each feature independently through the Transformer's encoder, then concatenate the outputs to form the final prediction [65].

Some of the models only make multivariate prediction. Therefore, to generate univariate predictions, we simply extract the response feature from the model's output. This approach is standard for these types of problems.

4.3.3 Models Hyperameters

For all of the models in this chapter, we used their default hyper-parameters. For FWin we use a window size of 12, and cross attention window number is 3. The total number of epochs is 6 with early stopping. We used the Adam optimizer, and the learning rate starts at $1e^{-4}$, decaying two times smaller every epoch. For ETSformer we used the initial learning rate of $1e^{-3}$ in the exponential with a warm up learning rate schedule. For PatchTST we used the constant learning rate of $2.5e^{-3}$ and 100 training epochs with early stopping.

4.4 FWin Equivalency Condition

In chapter 2, we present the theoretical results of FWin equivalency condition. The assumption of the Theorem require BDI condition. Thus in this section, we will verify the assumption on the dengue dataset. To verify the condition of the theorem we will follow the procedure: 1) Run simulations using the Informer model with full attention instead of probsparse on the testing set. 2) Collect the full attention matrix of the first encoder block of the Informer. 3) For a fixed window size w, compute the condition number of each block sub-matrix of size $w \times w$. 4) If the condition number is finite, then the sub-matrix satisfies the condition.

From Fig. 4.3, we verify that dengue dataset satisfies Theorem 2.5. All of the condition numbers are finite under various window sizes, most of them are relatively small (less than 10^4). We noted that as window size increases, the condition numbers increase. This supports the design of using small window size for efficiency and theoretical guarantee.

4.5 Results and Discussion

We present a summary of all the prediction tasks on the Singapore dataset in Tab. 4.1. MAE $= \frac{1}{n}\sum_{i=1}^{n}|y-\hat{y}|$ and MSE $= \frac{1}{n}\sum_{i=1}^{n}(y-\hat{y})^2$ serve as evaluation metrics. The results are the average of five independent simulations. The best results are highlighted in boldface, and the total count at the bottom of the table indicates how many times a particular method outperforms the others per metric per task. From Tab. 4.1, we observe that FWin has the best performance on both tasks.

In addition, we also demonstrate that including future PV information (e.g. climate and ocean current features) in the model increases FWin performance (on dengue cases) significantly. We observed that for longer time forecasting, i.e. metrics of 36, 48, 60, the error for MM

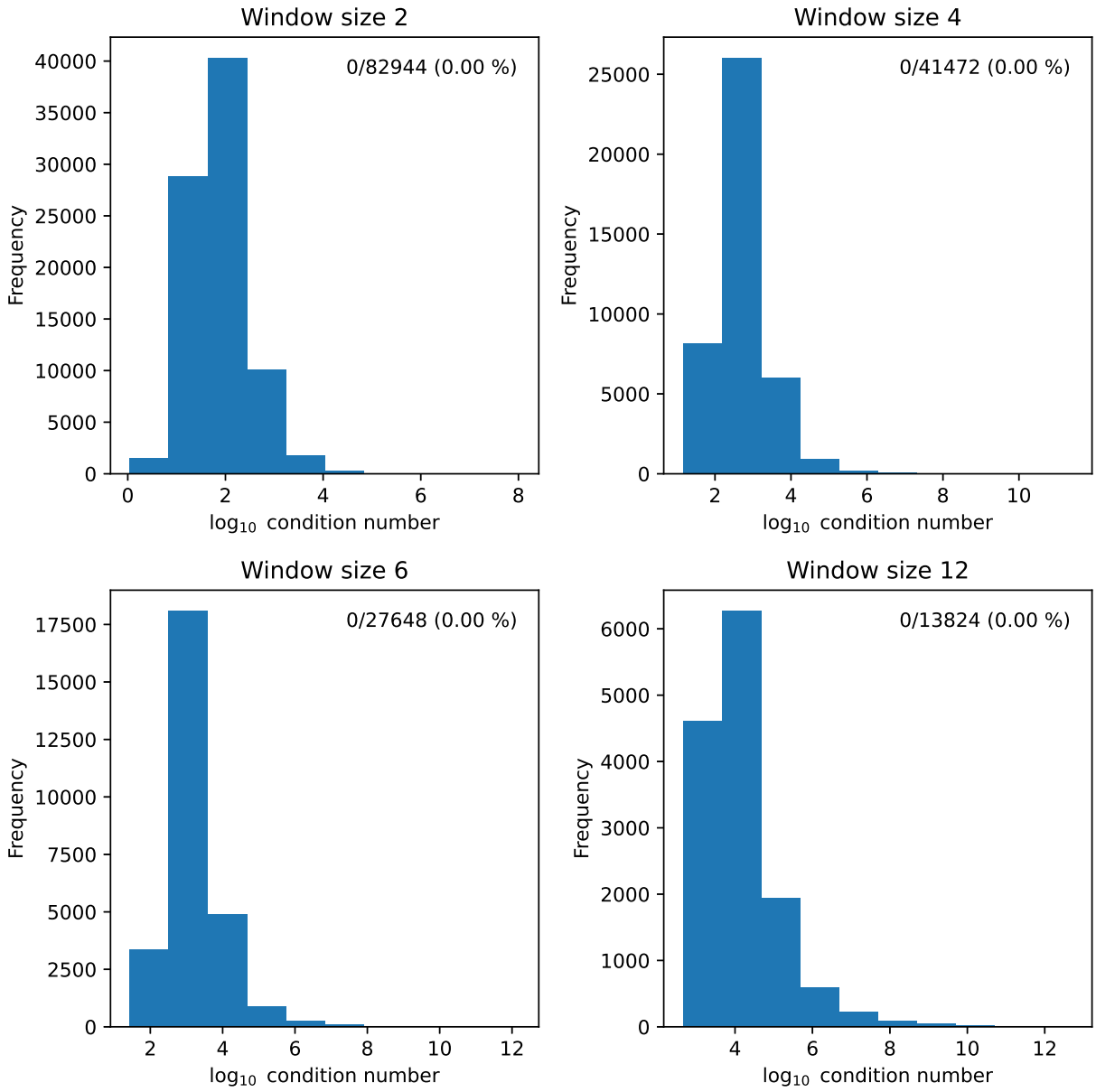


Figure 4.3: Condition numbers of block sub-matrices with various sizes of attention matrix obtained from the first encoder block of Informer model using full attention on the testing data. On the top right corner of each subplot, there is a label "n/m (k%)"; here m is the total number of condition numbers, n is the infinite condition numbers, and k is the corresponding percentage.

task is lower than MS task. However, for Informer and PatchTST, the models' performances decrease with additional information. An explanation for the degradation in performance is as follows: 1) the Informer's probsparse attention only chooses a number of queries to compute

attention, which means if the input length increases then a few top queries may not capture the majority of the contribution. 2) For PatchTST, it initially patches the input with overlaps, thus increasing the sequence length may cause too much overlap, leading to an overflow of information. In addition, it treats each feature independently, thus adding additional information for the PV may not affect the RV. In case of Autoformer and ETSformer, including future predictor (e.g. climate and ocean current) information is beneficial to model prediction power. In particular, ETSformer's error reduces significantly for MM task compared to the MS task. However, FEDformer was not able to perform the MM task because the requirement of the input length of the decoder need to be half of the input of the sequence.

Furthermore, we compared the vector autoregression (VAR) to DNN models. VAR depends on a choice of lag order, which is non-trivial. Opting for a large lag order can lead to the model making wild predictions, resulting in large errors. On the other hand, selecting a small lag order can significantly reduce errors, but the prediction may appear too smooth, which is unrealistic given the behavior of the dataset. Additionally, the overall error was higher than FWin, thus we do not present the full VAR results obtained from calling the VAR library of the package statsmodels [72].

We present a sample of the prediction of various models in Fig. 4.4 for the MM tasks. The $x-$axis represents the timescale in weeks, and $y-$axis is the normalized number of cases. FWin performs the best in terms of metrics presented and visual. It was able to predict a large drop in number of cases while many other models were unable to do so. We noted that in the MM task, the model input is richer, containing the most information. This confirm the results we obtained in Tab. 4.1.

Table 4.1: Accuracy comparison on Singapore data with input length of 36, best results highlighted in bold. Here MS, MM are multivariate to univariate, and modified multivariate to univariate respectively. SD is short for Singapore Dengue, and "-" indicates that the method is inapplicable for the task.

Methods		FWin		Informer		FEDformer		Autoformer		ETSformer		PatchTST	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
SD (MS)	24	**1.170**	**0.684**	1.842	0.877	2.090	1.085	2.237	1.127	1.611	0.856	1.273	**0.684**
	36	**1.450**	**0.725**	1.974	0.942	2.455	1.187	2.441	1.198	1.823	0.938	1.664	0.815
	48	**1.782**	**0.830**	2.043	0.958	2.912	1.334	2.927	1.370	2.271	1.063	1.887	0.915
	60	**1.518**	**0.826**	1.784	0.910	2.938	1.358	3.015	1.395	2.379	1.128	2.390	1.107
SD (MM)	24	**1.303**	**0.787**	2.137	1.002	-	-	1.951	1.055	1.480	0.885	1.734	0.833
	36	**1.156**	**0.680**	2.043	0.968	-	-	2.136	1.105	1.305	0.774	1.994	0.937
	48	**1.251**	**0.693**	2.228	1.033	-	-	2.238	1.152	1.355	0.825	2.352	1.062
	60	**1.124**	**0.710**	1.911	0.946	-	-	2.680	1.272	1.295	0.781	2.579	1.117
Count		16		0		0		0		0		1	

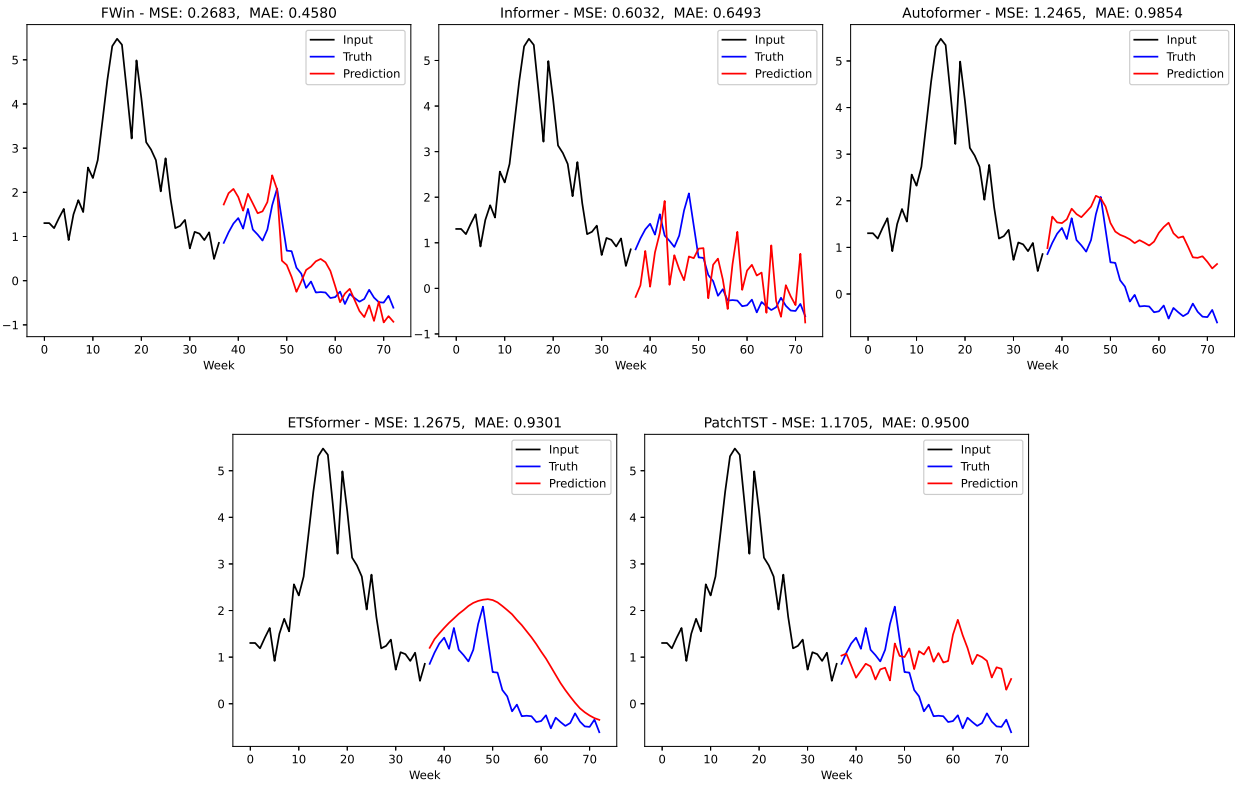


Figure 4.4: Sample models' prediction for MM task. The x-axis represents time in weeks, and y-axis is the normalized cases number. The suptitle includes the model name and the prediction errors (MSE and MAE). In black is the case number input, blue is the ground truth and red is the prediction of the model.

4.6 Ablation Study

In this section, we will examine the effect of window size of the FWin model on its performance. We can think of the window size of FWin as prior knowledge of the time delay effect of weather/climate on the dengue cases. Due to the restricted interaction within each window, the cases number in a particular window only attends to local weather information. For this experiment, we utilize FWin with window sizes of 1, 2, 4, 6, 12, and 18.

From Tab. 4.2, we observe that the biggest window size of the task gives the best performance in most cases. This is intuitively consistent with the design of FWin where we expect that the larger the window size, the better the overall performance. In addition, we observe that for the smallest window size of 1, the errors are significantly lower than naively expected. In particular, under the MAE metric, the window size of 18 performs the best, while under the MSE metric, a window size of 1 is the best for MS task. MSE amplifies the effects of large errors while suppressing those from the small errors. Thus under this metric, the smallest window size of 1 does not result in many large errors overall. On the other hand, MAE penalizes all error types equally, thus implying that the window size of 18 mostly results in small errors. We observe from the proof of FWin attention equivalency with the full attention that the theorem is true without any assumption on the structure of the attention matrix if the window size is 1. This provides an explanation for why we encountered low MSE error for window size of 1, i.e. the model does not make many large errors in the prediction. On the other hand for MM task, since we incorporate more information in the input, the largest window size of 12 performs the best. Moreover, except for the metric (prediction length) of 24, MM task performs better than MS task. This indicates that additional information is useful. An explanation for this phenomena is if the dependency between the RV (dengue cases) and the PV (precipitation) is short, i.e. 6 weeks, then the model performance may degrade at later time steps.

Table 4.2: Accuracy comparison of FWin model using different window sizes with input length of 36. Here MS, MM are multivariate to univariate, and modified multivariate to univariate respectively. Best results highlighted in bold. "-" denotes window size is inapplicable.

Window Size		1		2		4		6		12		18	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
SD (MS)	24	1.195	0.705	1.282	0.720	1.284	0.877	1.223	0.703	**1.181**	0.678	1.196	**0.667**
	36	**1.432**	0.742	1.507	0.754	1.517	0.755	1.492	0.747	1.458	0.725	1.440	**0.708**
	48	**1.673**	0.828	1.854	0.866	1.821	0.859	1.836	0.862	1.765	0.824	1.756	**0.817**
	60	**1.538**	0.848	1.552	**0.805**	1.555	0.810	1.622	0.830	1.592	0.813	1.598	0.809
SD (MM)	24	**1.354**	**0.795**	1.367	0.802	1.368	0.802	1.351	0.798	**1.354**	0.807	-	-
	36	1.306	0.726	1.326	0.740	1.331	0.738	1.286	0.726	**1.185**	**0.695**	1.216	0.718
	48	1.523	0.815	1.502	0.808	1.504	0.807	1.534	0.839	**1.270**	**0.719**	-	-
	60	1.339	0.802	1.338	0.805	1.328	0.800	1.312	0.796	**1.140**	**0.724**	-	-
Count		5		1		0		0		8		3	

In addition, we performed further experiments to understand the effect of shorter input sequence length to model performance. For this purpose, we reduced the input sequence length of the model from the default value of 36 to 18. From Tab. 4.3 we observe that for the shortest prediction length, PatchTST performs the best. However, for longer prediction lengths, FWin exhibits the best results. In general, the errors remain similar to their counterparts when the input length is 36. Therefore FWin is robust under the variation to shorter input lengths. Moreover, the MM tasks perform better than their counterparts on longer prediction lengths for all models except Informer.

Table 4.3: Accuracy comparison on Singapore data with input length of 18, best results highlighted in bold. Here MS, MM are multivariate to univariate, and modified multivariate to univariate respectively..

Methods		FWin		Informer		FEDformer		Autoformer		ETSformer		PatchTST	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
SD (MS)	24	1.388	0.673	1.899	0.888	1.650	0.890	1.734	0.937	1.476	0.839	**1.040**	**0.621**
	36	**1.444**	**0.698**	1.903	0.921	2.169	1.021	2.267	1.064	1.722	0.908	1.542	0.772
	48	**1.564**	**0.771**	2.045	0.983	2.863	1.241	2.930	1.275	2.147	1.048	1.972	0.932
	60	**1.281**	**0.666**	1.811	0.928	2.992	1.282	3.030	1.297	2.169	1.073	2.282	1.050
SD (MM)	24	1.669	0.863	2.250	1.017	-	-	2.075	1.189	1.807	0.944	**1.340**	**0.765**
	36	**1.479**	**0.839**	2.225	1.004	-	-	2.451	1.309	1.995	1.009	1.650	0.872
	48	**1.511**	**0.822**	2.307	1.058	-	-	2.687	1.364	1.726	0.890	1.800	0.940
	60	**1.187**	**0.696**	1.952	0.966	-	-	2.803	1.312	1.462	0.844	1.570	0.864
Count		12		0		0		0		0		4	

4.7 Conclusions

We evaluated various attention-based deep neural networks for predicting dengue cases in Singapore. The dataset contains multiple features, including average temperature, precipitation, and global climate indices such as Southern Oscillation Index, Oceanic Niño Index, and Indian Ocean Dipole. Among the models investigated in the study, FWin demonstrated the highest prediction accuracy overall. Moreover, incorporating future climate/weather/ocean information in the modified multivariate to univariate task generally improved dengue case prediction for many models considered.

Part III

Computer Vision

Chapter 5

SEMA: a Scalable and Efficient Mamba like Attention via Token Localization and Averaging

Attention is the critical component of a transformer. Yet the quadratic computational complexity of vanilla full attention in the input size and the inability of its linear attention variant to focus have been challenges for computer vision tasks. Motivated by the dispersion property and recent development of Mamba form of attention, we design Scalable and Efficient Mamba like Attention (SEMA) which utilizes token localization to avoid dispersion and maintain focusing, complemented by theoretically consistent arithmetic averaging to capture global aspect of attention. We support our approach on Imagenet-1K where classification results show that SEMA is a scalable and effective alternative beyond linear attention, outperforming recent vision Mamba models on increasingly larger scales of images at similar model parameter sizes.

5.1 Introduction

Attention models of linear complexity in the input image size have been actively pursued in computer vision. One line of inquiry started with Swin [50] where window attention and shifting form a local-global approximation of vanilla attention [82] in the image domain. Another approach is the recursive attention of linear complexity, or a selective state space model known as Mamba [19], which has been applied to multi-directional scanning paths across an image for key-query computation (e.g. VMamba [48]). Observing the similarity of Mamba and causal linear attention, [25] employed linear attention [30] in the macro-architecture of Mamba. Surprisingly, the resulting model MILA out-performed VMamba on 224^2 images of ImageNet-1K and some downstream tasks. Since linear attention is known to lack expressive power or focusing capability ([23] and references therein), it is mysterious that such a design works well. As the ablation study (Tab. 6 in [25]) showed, advanced and improved linear attention modules in standalone or other contexts (e.g. Flatten attention [23]) did worse in the MILA environment.

In this chapter, we use asymptotic behavior of attention in the simplest manner, we adopt the arithmetic averaging as a global approximation, in conjunction with a localized (window) attention to maintain the desired focusing property of the vanilla softmax attention. We then place such a local-global approximation in Mamba like transformer and Swin architecture as MILA [25]. Our design is thus explainable, besides being scalable and efficient as will be seen.

The key difference between our proposed method and others [23, 24, 83, 95] is that we utilize the theoretically confirmed dispersion property to approximate full attention. We prove in chapter 2 that any global attention mechanism inevitably disperses.

Our main contributions in this chapter are summarized below.

- We use theoretical guidance from the long range token asymptotic limit to match

dispersive behavior with arithmetic averaging for capturing the global aspect of attention layer in transformer models. Together with window attention to preserve the local aspect of attention on high frequency features, we put forth a novel scalable and efficient local-global attention (SEMA) in a Mamba type macro-structure [20, 25].

- We demonstrate in ImageNet-1K classification that SEMA is effective while image sizes scale up. It is flexible in finetuning on larger image input, improving accuracy and efficiency over recent scalable vision models [48]. SEMA is also competitive in downstream tasks.

5.2 Related Works

Transformer [82] with its softmax attention mechanism has been successful in language models. Its architecture is extended to applications in computer vision [14, 50]. The ability of having direct access to all of the tokens via the attention mechanism is essential. However, there are a few challenging problems. The first is the quadratic computation complexity in terms of the input scale, while the second is the hidden ability in the attention to focus on the relevant information.

Many works resolve the quadratic complexity of vanilla attention [82] such as window attention [2, 13, 50], and linear attention [30, 39] simplifying the softmax into a separable form and applying the associative property of matrix multiplication. Linear attention has been improved by others such as [23, 25]. Mamba [19, 48] is a linear complexity state space alternative to transformer. Inspired by Mamba, MILA [25] designed a linear attention block in a macro structure of the state space model. However, linear attention has the non-injectivity issue addressed in [24].

The attention selection mechanism suffers greatly as the number of tokens increases. In very

long context, the softmax attention is found unable to attend to important keys [83, 95]. Lately [83] describes this phenomenon as dispersion. There are existing works attempting to resolve the dispersion problem observed in the softmax attention head as the context length increases. [83] proposed an adaptive temperature fix to increase the sharpness of softmax attention, whereas [95] suggested to compute the attention as the difference of two attention matrices to remove the noises in the attention matrices, and allow a query to attend to useful keys. The dispersion effect has been observed in linear attention in situations where softmax does not exhibit (See Figure 4 of [24] and Figure 3 of [23]). In order to increase the focus of linear attention, [23] proposed to enhance the large attention coefficients while suppressing the smaller ones, while [24] adopted subtraction instead of division for normalization of linear attention. Both works have convolution to help maintain focus.

5.3 SEMA Models via Token Localization and Averaging

In order to resolve dispersion in softmax full attention, [83] suggested an adaptive temperature as an ad hoc technique for improving sharpness. In our notation, their choice is $\phi(x) = \exp(x/\theta)$ for some optimal θ. We observe that as $\theta \to 0$, the softmax attention converges to a hardmax, i.e. all of the attention is given to a single key, the remaining keys receive zero attention. The functional form of θ is chosen manually with each dataset. This limits the ability of the model to generalize on new datasets. Instead of resolving the dispersion property of attention, we fully embrace the property since we proved that any suitable choice of ϕ will cause dispersion.

We aim to design a model where attention computation is local and does not lead to dispersion. A way to achieve this is to limit the number of keys that a query attends to, i.e. window attention or some form of sparse attention, which resembles exponential forgetting in Mamba-like recursive attention. Since window attention has a narrow receptive field, it prevents

Table 5.1: Performance reported on standard 224^2 image size input of ImageNet-1K.

Method	Type	# Params	FLOPs	Top-1 (%)
ConvNeXt-T [52]	CNN	29M	4.5G	82.1
MambaOut-T [96]	CNN	27M	4.5G	82.7
VAN-B2 [22]	CNN	27M	5.0G	82.8
MixFormer-B4 [5]	CNN+Transformer	35M	3.6G	83.0
Swin-T [50]	Transformer	29M	4.5G	81.3
PVTv2-B2 [87]	Transformer	25M	4.0G	82.0
CSwin-T [13]	Transformer	23M	4.3G	82.7
Inline-CSwin-T [24]	Transformer	21M	4.3G	83.2
Flatten-CSwin-T [23]	Transformer	21M	4.3G	83.1
NAT-T [27]	Transformer	28M	4.3G	83.2
VMamba-T [48]	Mamba	31M	4.9G	82.6
LocalVMamba-T [28]	Mamba	26M	5.7G	82.7
DefMamba-S [46]	Mamba	32M	4.8G	83.5
MILA-T [25]	Mamba-like	25M	4.2G	83.3
SEMA-T (Ours)	Mamba-like	26M	4.3G	**83.7**
ConvNeXt-S [52]	CNN	50M	8.7G	83.1
MambaOut-S [96]	CNN	48M	9.0G	84.1
PVTv2-B3 [87]	Transformer	45M	7.9G	83.2
CSwin-S [13]	Transformer	35M	6.9G	83.6
VMamba-S [48]	Mamba	50M	8.7G	83.6
DefMamba-B [46]	Mamba	51M	8.5G	84.2
MILA-S [25]	Mamba-like	43M	7.3G	84.4
SEMA-S (Ours)	Mamba-like	46M	7.6G	**84.6**
ConvNeXt-B [52]	CNN	89M	15.4G	83.8
NAT-B [27]	Transformer	90M	13.7G	84.3
VMamba-B [48]	Mamba	89M	15.4G	83.9
MILA-B [25]	Mamba-like	96M	16.2G	85.3
SEMA-B (Ours)	Mamba-like	102M	16.9G	**85.4**

the model from accessing global information. To alleviate this problem, one may adopt either a shifted window [50] or a mixing mechanism after each attention layer. Each such fix has its own weakness however. Shifted window requires manual design and multiple layers [50]. Though a mixing mechanism recovers global receptive field to a large extent, the exact form is often unclear if one also aims at efficiency. We showed in chapter 2 that full softmax attention will disperse, i.e. $\text{softmax}(QK^T) = \Theta(1/n)$, where n is the number

Table 5.2: Model top-1 (%) comparison with and without finetuning on larger size input images from ImageNet-1K.

Method	Type	Image Size	# Params	FLOPs	No tuning	Finetune
VMamba-T	Mamba	384^2	31M	14G	82.4	83.5
SEMA (Ours)	Mamba-like	384^2	26M	13G	**83.1**	**84.1**
VMamba-T	Mamba	672^2	31M	43G	77.9	83.6
SEMA (Ours)	Mamba-like	672^2	26M	40G	**78.9**	**84.1**
VMamba-T	Mamba	768^2	31M	57G	74.7	83.5
SEMA (Ours)	Mamba-like	768^2	26M	52G	**75.4**	**84.2**
VMamba-T	Mamba	1024^2	31M	100G	57.4	83.4
SEMA-T (Ours)	Mamba-like	1024^2	26M	93G	**64.7**	**83.8**

Table 5.3: Model average inference time of 1000 runs on NVIDIA RTX A6000.

Method	Type	Image Size	Runtime (ms)
MILA-T	Mamba-like	224^2	25.86
SEMA-T (Ours)	Mamba-like	224^2	23.83
MILA-T	Mamba-like	512^2	26.10
SEMA-T (Ours)	Mamba-like	512^2	23.87
MILA-T	Mamba-like	1024^2	60.73
SEMA-T (Ours)	Mamba-like	1024^2	40.33
MILA-T	Mamba-like	2048^2	244.35
SEMA-T (Ours)	Mamba-like	2048^2	161.85

Table 5.4: Mask R-CNN 1x and 3x tasks on COCO dataset using input image of resolution (1280×800). Transformer(T), Mamba(M), Mamba-like(ML), **bold best-2**.

				(a) Mask R-CNN 1x						**(b) Mask R-CNN 3x**					
Method	Type	# Params	FLOPs	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ConvNeXt-T	CNN	48M	262G	44.2	66.6	48.3	40.1	63.3	42.8	46.2	67.9	50.8	41.7	65.0	44.9
PVTv2-B2	T	45M	309G	45.3	67.1	49.6	41.2	64.2	44.4	47.8	69.7	53.0	43.1	66.8	46.7
CSWin-T	T	42M	279G	46.7	68.6	51.3	42.2	65.6	45.4	**49.0**	70.7	**53.7**	43.6	67.9	46.6
LocVMamba-T	M	45M	291G	46.7	68.7	50.8	42.2	65.7	45.5	48.7	70.1	53.0	43.4	67.0	46.4
VMamba-T	M	50M	271G	**47.3**	69.3	**52.0**	**42.7**	**66.4**	**45.9**	48.9	70.6	53.6	43.7	67.7	**46.8**
MILA-T	ML	44M	255G	46.8	**69.5**	51.5	42.1	**66.4**	45.0	48.8	**71.0**	53.6	**43.8**	**68.0**	**46.8**
SEMA-T (Ours)	ML	46M	275G	**47.2**	**69.9**	**52.2**	**42.4**	**66.6**	**45.6**	**49.2**	**71.0**	**54.1**	**43.9**	**68.3**	**47.1**

of keys. Thus one can mimic the full softmax attention by simply averaging out all the tokens (as a low pass filtering): for a query (row vector) q, key matrix K, and value matrix V, approximate row-wise in the large n regime as: $\mathbf{softmax}(q\,K^T)V \approx \frac{1}{n}\sum_{j=1}^{n} v_j$. We call

Table 5.5: Semantic Segmentation on ADE20K. SS and MS denote single-scale and multi-scale, resp. FLOPs are calculated with an input size of 512×2048, **bold best-2**.

Backbone	Params	FLOPs	mIOU (SS)	mIOU (MS)
ResNet-50	67M	953G	42.1	42.8
DeiT-S + MLN	58M	1217G	43.8	45.1
Swin-T	60M	945G	44.5	45.8
ConvNeXt-T	60M	939G	46.0	46.7
NAT-T	58M	934G	47.1	48.4
VMamba-T	62M	949G	**47.9**	**48.8**
SEMA-T	56M	956G	**48.2**	**48.5**

this averaging a *homogeneous mixing*, which reduces the computational complexity of full attention coefficients from $O(n^2\, d)$ to $O(1)$, as the matrix product $Q\, K^T$ is down to a scalar factor $1/n$, and d the hidden dimension. The algorithms for computing window attention and such a homogeneously mixed window attention (referred to as SEMA attention from here on) are summarized in Algorithm 1.

Concretely, for given $Q, K, V \in \mathbb{R}^{n \times d}$ and $w \in \mathbb{N}$, such that w divides n, define:

$$SEMA(Q, K, V) := \mathcal{A}_{\phi,w}(Q, K, V; w) + \left[\frac{1}{n}\sum_{j=1}^{n} v_j\right], \tag{5.1}$$

where $[\cdot]$ broadcasts the row n times to permit matrix addition. In all our experiments, we use $\psi_q(x) = \psi_k(x) = x$, and $\phi(x) = \exp(x)$, or the window softmax attention. The first term in Eq. 5.1 processes information locally via window attention and the second term aggregates the global information to all of the tokens. We present an overview comparison of window attention and SEMA attention in Fig. 5.1. For the overall architecture of the model, we will utilize the design of MILA. We present the details architecture in Appendix 5.4. We choose MILA because (1) the design uses a simple linear attention that we will replace with SEMA attention; (2) our choice of window attention inspired by the forgetting property of Mamba naturally lands SEMA into the Mamba-like category. An overview of our model is in Fig.

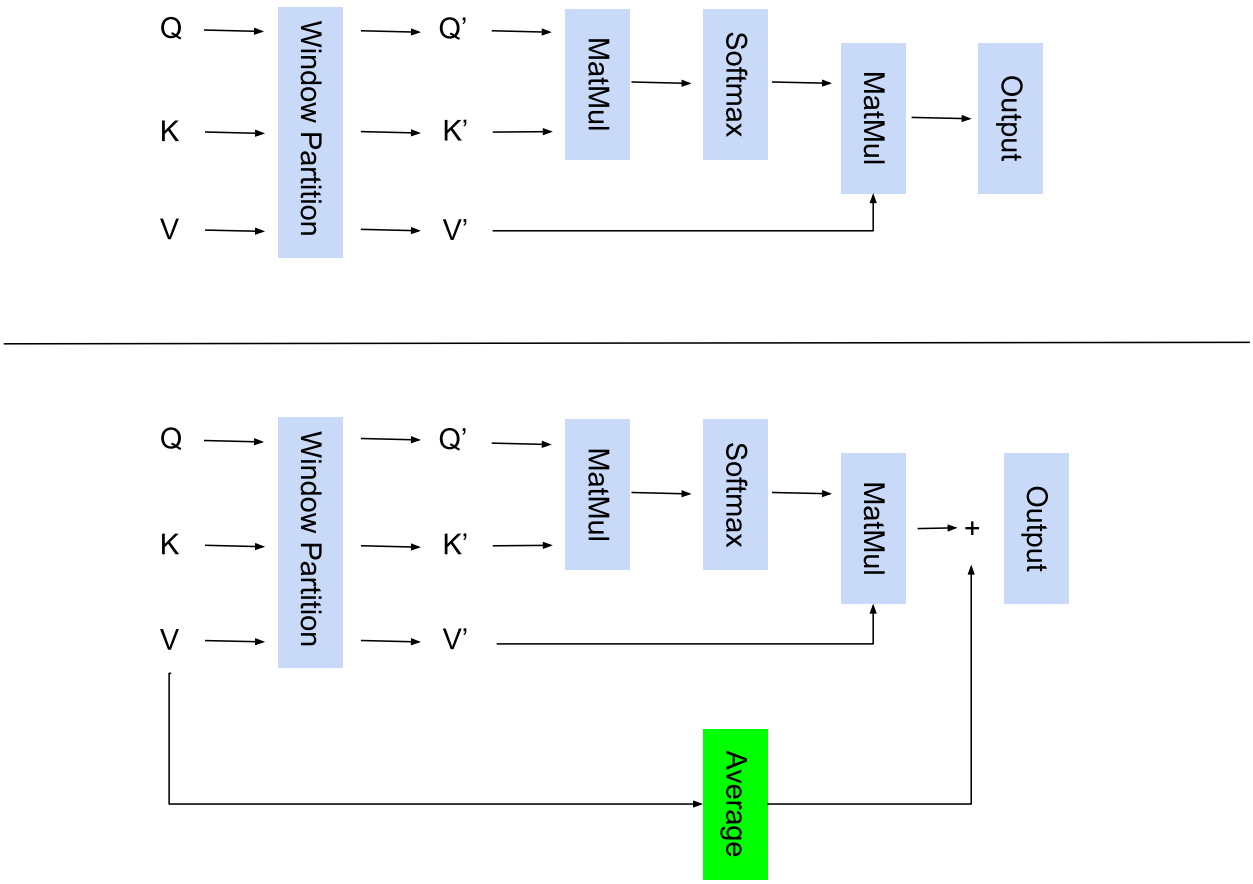


Figure 5.1: Homogeneous mixing (averaging) in SEMA attention (bottom) vs. window attention (top).

5.2. The main difference lies on the Attention Block, where we used SEMA attention in lieu of Linear attention for MILA. We noted that MILA does not use a linear projection for the values V (inspired by Mamba) while our formulation uses a linear projection for the values. For this reason, SEMA's parameter size is slightly higher than MILA's.

The latter part of the formulation in Eq. 5.1 is applied independently to each attention head. Consequently, a single key representation is shared within each head. This design choice is closely related to KV cache mechanism. In particular, Multi-Query Attention [73] shares a single set of keys and values across all attention heads, thereby reducing memory and bandwidth requirements during inference. In contrast, Multi-Head Latent Attention [45] stores keys and values in the KV cache using a low-rank latent representation, then

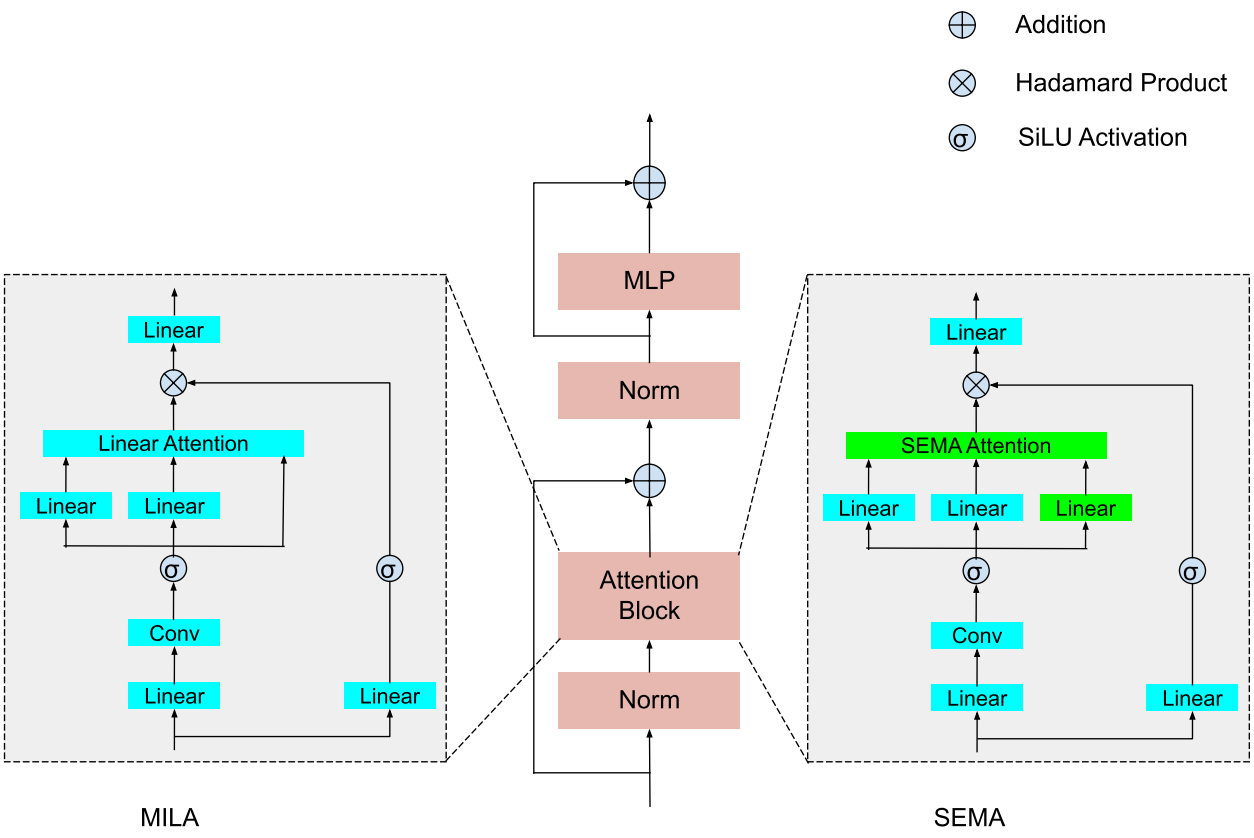


Figure 5.2: Mamba like transformers: MILA vs. SEMA, green blocks show our innovations.

reconstructs them via low-rank projection during attention computation. When the original keys are similar, this compression-projection (CP) process may produce nearly identical keys and values for attention, effectively behaving as an averaging operation. In this regime, the resulting attention computation is equivalent to the SEMA formulation. In general, the CP process resembles a sparsely weighted average which may be a future extension for SEMA.

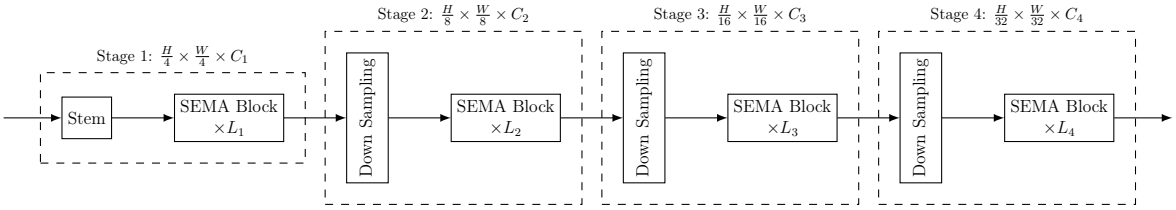


Figure 5.3: The 4 stage architecture of SEMA similar to Swin [50] and MILA [25].

5.4 Model Architecture

We present the architecture of our SEMA model in Fig. 5.3 and summarize the detailed structure in Tab. 5.6 which gives the parameter values in Fig. 5.3: $L_1 = 2$, $L_2 = 4$, $L_3 = 8$, $L_4 = 4$ for SEMA-T, among others. The 4-stage framework to build SEMA is similar to Swin [50] and MILA [25].

5.5 Experiments

In this section, we will test SEMA on classical benchmarks in computer vision such as ImageNet-1K, COCO, and ADE20K. Due to limited resources available, we are unable to perform a thorough tuning of hyper-parameters and so have adopted those of MILA [25]. We also report performance based on a single run in most of the experiments, this is typical for this line of research under physical resource constraints [25, 48]. Notably, we observed that the performance of SEMA-T on Imagenet-1K, as presented in Tab. 5.1, was consistent across

stage	output	SEMA-T	SEMA-S
1	56×56	stem, 64 $\begin{bmatrix} \text{dim } 64 \\ \text{head } 2 \\ \text{window size } 7 \end{bmatrix} \times 2$	stem, 64 $\begin{bmatrix} \text{dim } 64 \\ \text{head } 2 \\ \text{window size } 7 \end{bmatrix} \times 3$
2	28×28	downsampling, 128 $\begin{bmatrix} \text{dim } 128 \\ \text{head } 4 \\ \text{window size } 7 \end{bmatrix} \times 4$	downsampling, 128 $\begin{bmatrix} \text{dim } 128 \\ \text{head } 4 \\ \text{window size } 7 \end{bmatrix} \times 6$
3	14×14	downsampling, 256 $\begin{bmatrix} \text{dim } 256 \\ \text{head } 8 \\ \text{window size } 7 \end{bmatrix} \times 8$	downsampling, 256 $\begin{bmatrix} \text{dim } 256 \\ \text{head } 8 \\ \text{window size } 7 \end{bmatrix} \times 21$
4	7×7	downsampling, 512 $\begin{bmatrix} \text{dim } 512 \\ \text{head } 16 \\ \text{window size } 7 \end{bmatrix} \times 4$	downsampling, 512 $\begin{bmatrix} \text{dim } 512 \\ \text{head } 16 \\ \text{window size } 7 \end{bmatrix} \times 6$

Table 5.6: Architecture of SEMA model.

several runs.

5.5.1 ImageNet-1K

To evaluate SEMA on ImageNet-1K classification [11], we trained our models under the same settings as MILA [25] and Swin Transformer [50]. The dataset consists of 1.28M training images and 50K validation images with a total of 1000 classes. We trained all of our models from scratch for 300 epochs using AdamW [54] optimizer with a cosine learning rate decay schedule of 20 warm-up epochs and weight decay of 0.05. The initial learning rate is set to 4×10^{-3}. We apply standard augmentation and regularization as RandAugment [9], Mixup [98], CutMix [97], and random erasing [102]. For window attention, we used the standard window size of 7. We compared the results with current SOTAs. SEMA-T achieves a top-1 accuracy of 83.7, outperforming 83.3 of MILA-T [25], and DefMamba [46]. Similarly SEMA-S/B achieves a top-1 accuracy of 84.6/85.4, surpasses 84.4/85.3 of MILA-S/B. We summarized the result in Tab. 5.1.

5.5.2 ImageNet-1K on Larger Resolutions

Classical tokenization of images, such as patching in Swin, converts every 4×4 pixels patch into a token via convolutions. Thus, increasing image resolution while maintaining the patch size increases the number of tokens. For example, under the standard settings, an image resolution of 224^2 yields $n = 56^2$ tokens in stage one (Fig. 5.3) where an image of size 1024^2 generates $n = 256^2$ tokens. We evaluate models on higher image resolutions $(384^2, 672^2, 768^2, 1024^2)$ of ImageNet-1K. Such experiments have been scarcely reported, except in VMamba [48]. When conducting no tuning or finetuning classification experiments on higher resolution ImageNet-1K, we follow the similar configurations as VMamba [48]. For no tuning, we load the pretrained VMamba models and conduct inference on $384^2, 672^2, 768^2, 1024^2$ using its validation scripts. For finetuning, the number of epochs is set to be 10 to ensure the training loss converges. During training, an AdamW optimizer is applied, with a learning rate of 0.001 and a weight decay of 0.05. We finetune the whole VMamba network, which is different from the linear finetuning in the original settings. Similarly, for SEMA we finetune for 30 epochs with a weight decay of 10^{-8} and a base learning rate of 10^{-5}. We pick a window size of 8 for all of the resolutions since this is the closest divisors. The remaining finetuning strategy mirrors the approach used during the backbone training with a 224^2 input resolution.

We start with generalization under no-tuning. As increasing resolutions lower the performance, a prevalent phenomenon in existing networks [48], we mainly compare the differences among the models. We observe that for image resolution of $384^2, 672^2$ and 768^2, VMamba-T is about 1% behind SEMA-T. However, VMamba-T performs significantly worse than SEMA-T at 1024^2. The gap is 7.3%. This reveals that VMamba-T struggles to process large images, while SEMA-T is much more resilient as image sizes scale up. We then finetune the models pre-trained on 224^2 images with few epochs retraining on larger images (the corresponding size of the input in the generalization test). The models all recover and surpass their respective performance levels at 224^2. Table 5.2 shows that SEMA-T is 0.4-0.7% better than VMamba-T

on all resolutions.

5.5.3 COCO

We also evaluate SEMA for the instance segmentation task on MSCOCO2017 [42] via the 1x and 3x Mask R-CNN training setting in Swin [50], with pretrained SEMA-T backbone on the ImageNet-1K. Due to resource limitations, we only conducted experiments using a couple of window size options. Our best results were obtained from window size of 12. We expect that finetuning the window size would yield better result. We compare SEMA-T to other SOTAs in Table 5.4. SEMA-T shows superiority in all measures for 3x task and stays in top-2 across all measures for 1x task. SEMA-T demonstrates performance that is comparable to or surpassing that of MILA-T across all metrics within Mamba-like architectures.

5.5.4 ADE20K

To evaluate semantic segmentation on ADE20K, we use UPerNet [93] as segmentation framework in combination with MMSEG [8] and follow the similar setting as Swin [50]. We use a batch size of 8, and a window size of 14 for our model. We train using ADAM with learning rate of 6×10^{-5}, with $\beta \in [0.9, 0.999]$ and weight decay of 0.01. Table 5.5 shows that SEMA-T surpasses VMamba-T in the single scale regime with an mIOU of 48.2, while remaining competitive in the multi-scale regime with an mIOU of 48.5.

Overall, SEMA's performance on standard benchmarks such as ImageNet-1K, COCO, and ADE20K demonstrates two key findings. First, SEMA serves as a robust replacement for existing attention mechanism in the regime of relatively small number of tokens, i.e. 56^2 for ImageNet-1K. In the large token regime, its theoretical guarantees provide an additional safety net. Second, as shown in Tab. 5.2, VMamba-T exhibits weaknesses when applied to

large-resolution image inputs, whereas SEMA maintain significantly stronger adaptability and robustness.

We compare the runtime of SEMA against its counterpart MILA. In Tab. 5.3, we observe that at low resolution of $224^2, 512^2$, SEMA is about 10% faster than MILA with similar runtime. In the large resolution regime of $1024^2, 2048^2$ SEMA is about 33% faster than MILA, demonstrating the scalability of the proposed approach.

5.6 Ablation Study

In this section, we conduct experiments to show the benefit of modifications to the overall performance of the model. First, at the standard resolution of 224^2, removing the value projection significantly reduces top-1 accuracy by about -0.8%, while removing the averaging component has only a minor impact (about -0.2%). This indicates that, in this regime, the performance gain primarily comes from the value projection (corresponding to the additional linear layer), while the averaging component-related to dispersion contributes little. Since the number of tokens in our network (see Fig. 5.3 in the Appendix) is small in the later stages, the typical window attention is nearly full attention, esp. stage 4 has input resolution of 7×7 and the window size is 7, thus window and full attention are identical. This explains why we have a minor improvement under the input resolution of 224^2. We demonstrated SEMA's stronger performance under larger input resolution in Tab. 5.2.

To further examine the role of the averaging (homogeneous mixing) component, we provide an additional ablation across different resolutions in Tab. 5.8. At 224^2, the averaging term contributes only $+0.2\%$ top-1 accuracy. However, as the input resolution increases to 672^2, its contribution becomes more pronounced: $+0.4\%$ after fine-tuning and $+2.2\%$ without fine-tuning.

These results are consistent with our theoretical analysis. When the input space $\mathcal{X}$ is fixed (Theorem 2.9), the dispersion coefficient is fixed, and the sequence length becomes the primary factor affecting dispersion. As the sequence length increases (e.g., higher resolution), dispersion becomes more significant, and the averaging term, serving as a global approximation, plays a larger role. Therefore, while the improvement at 224^2 is not driven by dispersion, the gains at larger resolutions are increasingly attributable to the averaging component. After fine-tuning, we expect the input space $\mathcal{X}$ to shift, which can partially mitigate dispersion.

Although the benefit of homogeneous mixing in standard ImageNet-1K is marginal, [10] shows that homogeneous mixing is highly effective when apply to medicals image segmentation tasks. In particular, as the image size increase the $F1$ score gain is much more significant. Under different dataset with various properties, we showed that adding a simple average help model capability. Thus demonstrate the robustness of the design.

Table 5.7: Ablation study on images of 224^2 resolution of ImageNet-1K.

Method	Average	Value Projection	Top-1
SEMA-T	✓		82.9
SEMA-T		✓	83.5
SEMA-T	✓	✓	83.7

Table 5.8: Ablation study of homogeneous mixing (averaging) on various images resolution of ImageNet-1K.

Method	Average	224^2	672^2 Finetune	672^2 No tune
SEMA-T		83.5	83.7	76.7
SEMA-T	✓	83.7	84.1	78.9

5.7 Computing Resource

We train and test all of the models on 8 NVIDIA RTX A6000 GPUs, each with 46G of memory. For standard image size of 224^2 on ImageNet-1K, we use a batch size of 256 per GPU. For larger size image input $384^2, 672^2, 768^2$, 1024^2, the corresponding batch sizes are $128, 32, 32, 16$ per GPU respectively. For COCO dataset, we use an identical setup as in MILA [25]. For ADE20K, we used similar setup as in Swin [50].

Algorithm 1 SEMA integrates window attention and homogeneous mixing (averaging), to be placed in a Mamba macro-setting so that long range global features are captured efficiently with local features.

Input: Input x

Parameter: window size w

Output: Attention

1: Compute $Q = \text{Linear}(x)$, $K = \text{Linear}(x)$, $V = \text{Linear}(x)$.

2: Divide Q, K, V into Q_w, K_w, V_w with window size w.

3: Apply RoPE to Q_w, K_w.

4: Compute window attention $WA = \text{softmax}(Q_w K_w^T) V_w$.

5: Compute $V_L = \text{LePE}(V)$.

6: Compute $V_A =$ Average V along the sequence dimension.

7: **Return** $WA + V_L + V_A$

5.8 Conclusion

SEMA uses window attention to avoid dispersion and averaging to capture global information. The simple averaging step is based on the large token asymptotic analysis of generalized attention both deterministically and probabilistically, hence theoretically rooted. Experiments on ImageNet-1K, COCO, ADE20K showed the effectiveness, scalability and robustness of

SEMA. In future work, we plan to (1) derive sharper bounds on dispersion coefficients, (2) extend averaging to learned and weighted averaging as well as sparsely weighted averaging in case of extremely long tokens, and (3) apply our framework to scientific data problems involving long tokens such as large size medical images [91].

Chapter 6

VideoSEMA: a scalable and efficient Mamba-like attention for video understanding

A natural extension of the line of research presented in this dissertation would be combine time series and image. In this chapter, we present for video understanding (classification) a split space-time attention model, VideoSEMA, consisting of a scalable and efficient Mamba-like attention (SEMA) block in space and a softmax temporal attention in time. In each frame, SEMA attention applies a local window attention in parallel with a global averaging in a Mamba macro-architecture, which is called Mamba-like. On benchmark K400 data sets, VideoSEMA out-performs heavier vision transformer and Mamba models. On benchmark SSv2 data, VideoSEMA leads in top-1 accuracy among models of similar parameter sizes. As image resolution scales up from standard 224^2 to 1024^2 on K400 and without fine-tuning, VideoSEMA degrades much more gracefully than VideoMamba in accuracy.

6.1 Introduction

Understanding complex spatiotemporal patterns in videos remains a fundamental challenge in computer vision, with application ranging from classification and segmentation to world modeling for autonomous system such as robotics and self-driving vehicles. With the widespread of usage of multimodal models in combination with language models, recent advances have leveraged transformer based architecture and large foundation models to capture long range dependencies, achieving state of the art performance [1, 3, 15, 40, 85, 88]. Despite their success, these models predominately rely on Vision Transformer (ViT) backbones, which incur high computational costs for large video inputs. To alleviate this problem, some works explores Mamba [19] as backbone, which reduces the computation cost for long sequences, such as in VideoMamba [37]. Linear attention is another approach to reduce the cost of classical attention mechanism and WLiT demonstrates that it is an effective way to process video [75]. Lately, Mamba-like attention models have been developed as an efficient replacement of classical softmax attention in image application [26]. Mamba-like means to keep the macro-architecture of Mamba yet embed in it light weight non-Mamba attention blocks. Along this line, SEMA [81] is designed to mimic the exponential forgetting property of Mamba by an asymptotically guided global approximation of softmax attention. For a duality view and treatment of softmax and efficient attentions, see [64, 101]. An operator splitting and integro-differential equation perspective of transformer is in [76]. In this work, we explore the efficacy of SEMA backbone in video applications.

Our main contributions are:

- We develop a novel video model in the split space-time attention framework based on a Mamba-like scalable and efficient image backbone (SEMA,[81]).
- We demonstrate on K400 (SSv2) classification tasks that VideoSEMA is an efficient

model, outperforming much larger (comparable) parameter and flops size transformer and Mamba video models to date. Post-training and without fine-tuning on K400, it is much more robust than VideoMamba [37] as video frames scale up in size.

6.2 Related Works

Video representation learning has evolved from CNN to attention based models to hierarchical and efficient sequence architectures such as Mamba [19]. TimeSformer [3] demonstrated that factorized space time attention can replace 3D convolutions for video recognition, while MViT and MViTv2 [15, 40] introduced multiscale hierarchical Transformer that improves efficiency and scalability across image and video tasks. More recently, state space models have been explored for long range temporal modeling, with VideoMamba achieving competitive performance on video understanding tasks [37]. At scale, foundation models such as InternVideo2 unify self-supervised, constrastive, and generative objectives for multimodal video understanding [88]. Other efforts focus on efficiency and prediction such as adaptive token strategies improving training and inference flexibility [85], while V-JEPA 2 advanced self-supervised world modeling for motion prediction and long horizon reasoning in video [1]. However, many of these rely on ViT as a vision backbone, whereas our work explores a light weight Mamba-like model (SEMA [81]) as an alternative backbone to capture spatial and temporal features efficiently.

6.3 Methodology

6.3.1 Proposed Method - VideoSEMA

Given an input video $x \in \mathbb{R}^{C\times T\times h\times w}$. VideoSEMA first tokenizes using a 3D convolution (e.g. kernel=(1,4,4)) to project the input video into $X \in \mathbb{R}^{C\times T\times H\times W}$ of non-overlapping spatiotemporal patches where $H = h/4, W = w/4$. Second, we append a learnable classifier token at the end of the sequence. The last step of pre-processing is to add learned positional embedding and then temporal embedding. Concretely,

$$X = \text{3DConv}(x) \tag{6.1a}$$

$$X = [X, X_{cls}] + \text{Emb}_{pos} + \text{Emb}_{temp}, \tag{6.1b}$$

where $X_{cls} \in \mathbb{R}^{H\times W\times C}$ is classifier token, $\text{Emb}_{pos} \in \mathbb{R}^{H\times W\times C}$ and $\text{Emb}_{temp} \in \mathbb{R}^{T\times C}$ are learned positional and temporal embedding respectively. Here the addition is broadcast along the appropriate dimension to enable matrix addition.

Next, we will process the spatial and temporal information of the input via the VideoSEMA block. The VideoSEMA block is constructed as

$$y = Spatial\_SEMA(X), \tag{6.2a}$$

$$y' = LayerNorm(y), \tag{6.2b}$$

$$z' = Temp\_Attn(y'), \tag{6.2c}$$

$$z = X + z', \tag{6.2d}$$

where spatial function operates over the $H \times W$ dimension of the input, while the temporal

function operates over the T dimension. To be precise, for $Q, K, V \in \mathbb{R}^{T\times H\times W\times C}$, we have

$$\begin{aligned} Spatial\_SEMA(q_{t,h,w}) := \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W} v_{t,i,j} \\ + \sum_{l,p\in I(h,w)} \frac{\exp(q_{t,h,w}k_{t,l,p}^T)}{\sum_{i,j\in I(h,w)}\exp(q_{t,h,w}k_{t,i,j}^T)} v_{t,l,p}, \end{aligned} \tag{6.3}$$

where $I(h, w)$ is the index set for the window, e.g. window size of 7. And temporal attention

$$Temp\_Attn(q_{t,h,w}) := \sum_{i=1}^{T} \frac{\exp(q_{t,h,w}k_{i,h,w}^T)}{\sum_{j=1}^{T}\exp(q_{t,h,w}k_{j,h,w}^T)} v_{i,h,w}. \tag{6.4}$$

The alternating spatial and temporal attention treatment in (6.2a) and (6.2c) can be viewed as an efficient operator splitting approximation of the full SEMA obtained by directly extending SEMA to space and time. For video frames of moderate lengths considered here, a softmax attention in (6.2c) is affordable and effective as supported by our ablation study, making linear complexity approximations unnecessary in time. As seen later, VideoSEMA outperforms much heavier networks (Tab. 6.1). As a future direction for handling longer videos, a sparse or dilated attention [12] in the temporal attention step is a promising alternative to leverage continuity of information in time at reduced computational costs.

VideoSEMA's macro-structure is shown in Fig. 6.1, and is repeated in the network. Between two adjacent repeats, we use a convolutional layer to down-sample the spatial dimension of the video signal to produce a hierarchical network (similar to [26, 50, 81]). Lastly in Algorithm 2, the representation of the $[CLS]$ token is normalized before passing through a linear classification head.

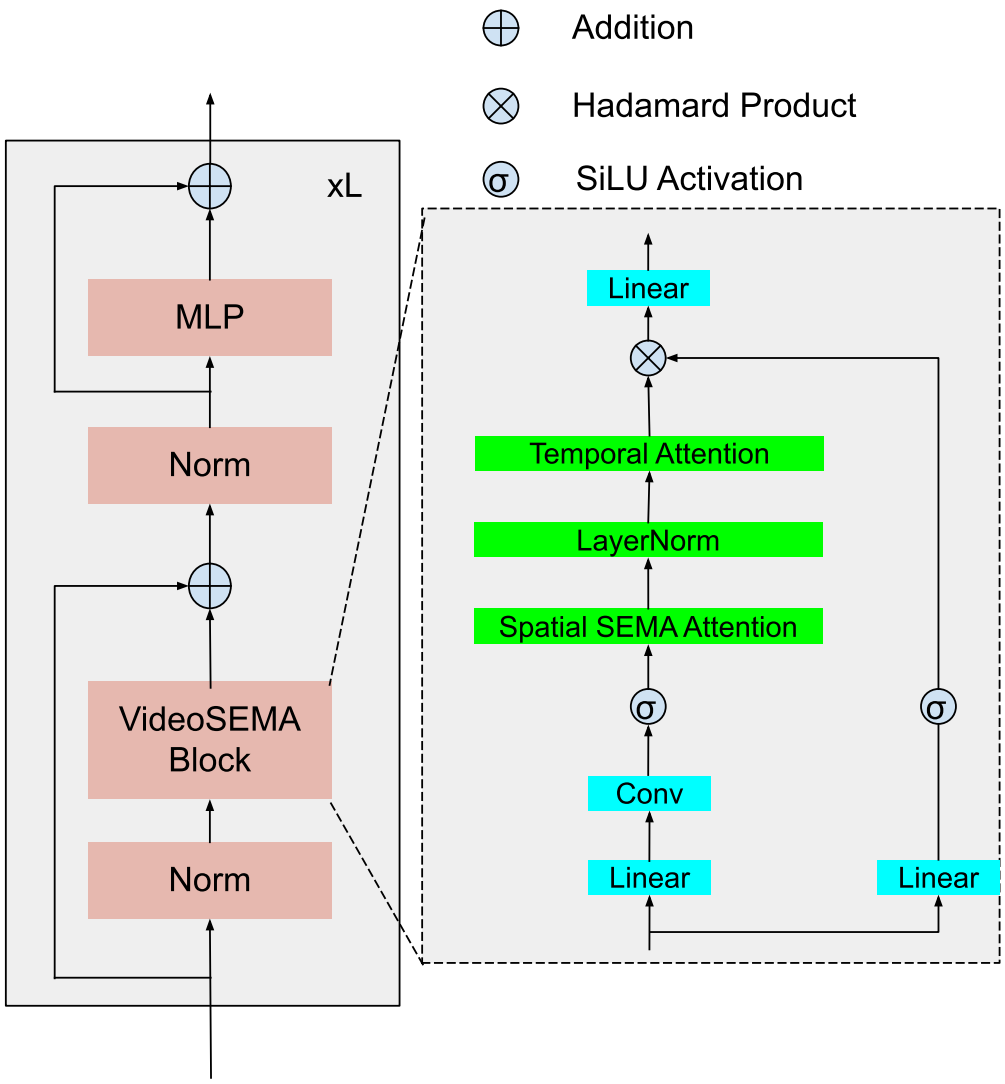


Figure 6.1: Overview of VideoSEMA macro-structure.

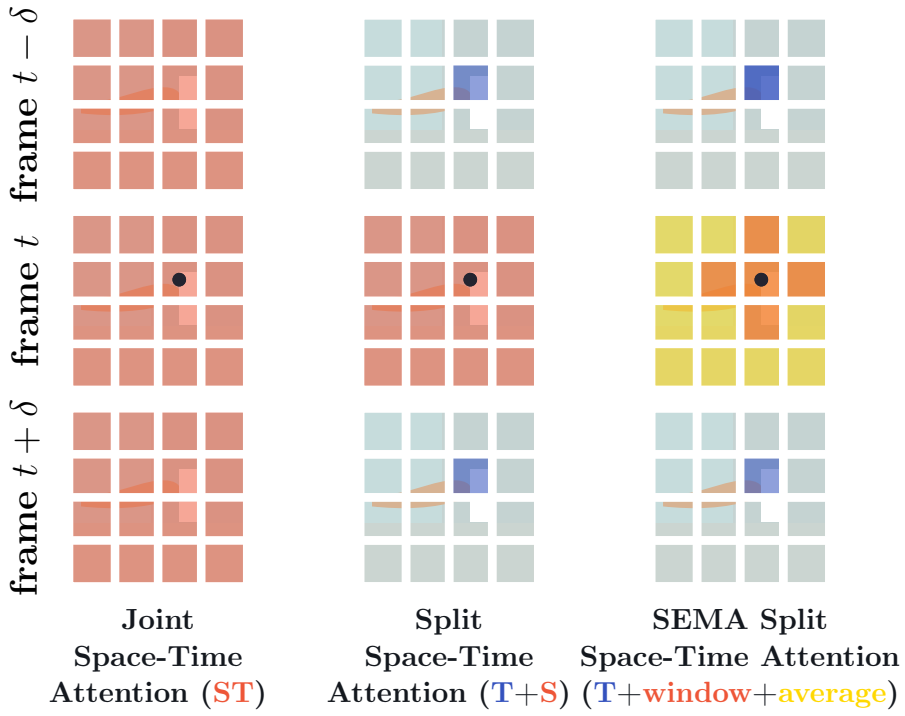


Figure 6.2: Visualization of space time attention types. For illustration, the dark dot denotes the query patch and colored patches show its self-attention space-time neighborhood under each scheme. Patches without color are not used for the self-attention computation of the query patch. Multiple colors within a scheme denote attentions separately applied along different dimensions, e.g., space and time for $(T + S)$. Note that self-attention is computed for every single patch in the video clip, i.e., every patch serves as a query. Although the attention pattern is shown for only two adjacent frames, it extends in the same fashion to all frames of the clip.

Algorithm 2 Proposed VideoSEMA algorithm. Here cls, temp, pos denote classifier tokens, temporal embedding, and positional embedding respectively. When there are dimensions mismatch in binary operations, there are implicit reshaping/broadcasting to match the dimensions. On the right hand side, we denote the current running shape of x.

Input: Input x $(C \times T \times H \times W)$

Learnable Params: cls $\in \mathbb{R}^{H \times W \times C}$, temp $\in \mathbb{R}^{T \times C}$, pos $\in \mathbb{R}^{HW \times C}$

$x = \text{reshape}(x)$ $(T \times H \times W \times C)$
$x = \text{concat}((x, \text{cls}), \text{dim}=0)$ $((T+1) \times H \times W \times C)$
$x = \text{reshape}(x)$ $((T+1) \times HW \times C)$
$x = x + \text{pos}$ $((T+1) \times HW \times C)$
$x = x + \text{temp}$ $((T+1) \times HW \times C)$
$x = \text{reshape}(x)$ $((T+1) \times HW \times C)$
$x = \text{VideoSEMA}(x)$ $((T+1) \times HW \times C)$
    Process spatial information via SEMA
    Process temporal information via classical attention
$x = \text{AvgPooling}(x)$ $((T+1) \times C)$
Return $x[-1, :]$ $(1 \times C)$

6.3.2 Complexity

Given an input video $x \in \mathbb{R}^{T \times H \times W \times C}$, the joint space-time attention treats the video as a sequence of length $n = THW$. Therefore, the attention matrix has size $n \times n$, yielding computational complexity $\mathcal{O}((THW)^2 C)$. In practice, this formulation is prohibitively expensive for videos due to the quadratic memory and compute cost of the attention matrix. Consequently, fully joint space time attention is rarely used in large scale video models [3].

The split space time attention approach works with a factorized (or separate spatial and temporal) attention, thereby lower the computational costs similar to operator splitting [17, 74]. In particular, TimeSformer [3] applies a split 2-step procedure: (1) spatial attention independently in each frame, and (2) temporal attention independently across frames at each spatial location. The spatial attention complexity in (1) is $\mathcal{O}(T(HW)^2 C)$ as each of the T frames performs full attention over the HW spatial tokens. The temporal attention complexity in (2) is $\mathcal{O}(T^2 HWC)$ as each spatial location performs full attention over T tokens. So the total complexity of the split attention design becomes $\mathcal{O}((T^2 HW + T(HW)^2)C)$.

In contrast, VideoSEMA replaces full spatial attention with SEMA, whose complexity scales linearly with spatial dimension $\mathcal{O}(THWwC)$ where w denotes the window size. The temporal attention is still performed globally, thus the total computational complexity of VideoSEMA is $\mathcal{O}((T^2HW + THWw)C)$, linear in spatial resolution HW. In the datasets of this paper, the number of video frames is at most 32 (i.e. $T \leq 32$ in Tab. 1, and $T \leq 16$ in Tab. 2 and Tab. 3). The main contribution to the complexity comes from spatial resolution $HW = (224)^2$. With a linear complexity in HW, VideoSema made considerable computational savings from TimeSformer [3], as seen in Table 6.1 on 16×224^2 resolution of K400 dataset where VideoSema's parameter size is a fraction $31/121$ of TimeSformer's while achieving better accuracies. We present visualization of each space-time attention in Fig. 6.2.

6.4 Experiments

6.4.1 Dataset

We evaluate our approach on two widely used large-scale video action recoginition benchmarks: Kinetic-400 (K400) [31] and Something-Something v2 (SSv2) [18]. K400 contains around 240000 training, 20000 validation, and 40000 testing videos spanning 400 human action categories, with clips source from YouTube and trimmed to focus on a single action that lasted around 10 seconds. The dataset is focused on human action from diverse scenes, actors, and viewpoints, making it a standard benchmark for learning high-level semantic. In contrast, SSv2 consists of around 220000 videos, with approximately 170000 training, 25000 validation and 27000 testing videos that last around 2-6 seconds. SSv2 places a strong temporal reasoning and fine-grained actions. Together, these datasets provide complementary evaluation settings.

6.4.2 Setting

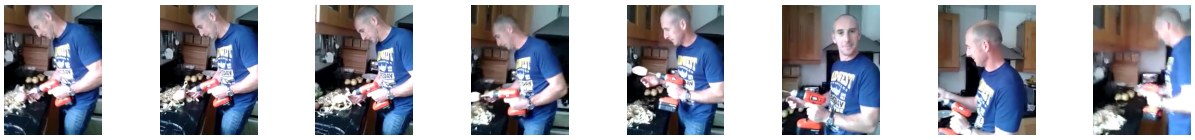
A common strategy to train a video model is to pretrain a model with an image-based architecture [3, 37, 51, 79] on ImageNet-1K or ImageNet-21K, and then inflate the image model into a video model. For fair comparison with VideoMamba, we pretrain the SEMA model on ImageNet-1K, then incorporate the temporal component and train the model on K400 and SSv2 to obtain the VideoSEMA results. For SEMA, we use the same training setup as described in chapter 5. We use the setting of the T model variant, i.e. 4 stages with hidden dimension of 64, 128, 256, and 512 respectively. To train VideoSEMA, we use a similar setup to VideoMamba. In particular, for K400 we use a set of 5 warmup epochs, 50 total epochs, a 0.35 stochastic depth rate, and 0.05 weight decay, with an initial learning rate of 2×10^{-4} using the AdamW optimizer. For SSv2, we use a set of 5 warmup epochs, 30 total epochs, a 0.35 stochastic depth rate, and 0.05 weight decay, with initial learning rate of 4×10^{-4}. We train and test VideoSEMA and VideoMamba on 8 NVIDIA RTX A6000 GPUs, each with 46G of memory.

6.4.3 Experimental Results

K400

We present the overall performance of VideoSEMA on K400 dataset in Table 6.1. Compared to VideoMamba under a similar computational budget, VideoSEMA's top-1 accuracies are 1-2% better across different input resolutions. Compared to other state of the art methods under similar training methodologies and computational budgets shown in the first row group of Table 6.1, VideoSEMA performs significantly better in both top-1 and top-5 metrics. For example, at an input resolution of 16×224^2, it is 1.4% better than MViTv2-S in top-1 and 0.8% in top-5.

label: peeling potatoes
VideoSEMA prediction: peeling potatoes, P=0.9017
VideoMamba prediction: peeling apples, P=0.5198



(a) example depicts a person peeling potatoes, where VideoSEMA correctly predicts the action with high ($P = 0.9$), while VideoMamba incorrectly classifies it as peeling apples.

label: drinking shots
VideoSEMA prediction: drinking shots, P=0.5002
VideoMamba prediction: tasting beer, P=0.4820

(b) example shows a person drinking shots, which VideoSEMA correctly recognizes with moderate confidence $P = 0.5$.

label: ripping paper
VideoSEMA prediction: ripping paper, P=0.2000
VideoMamba prediction: folding napkins, P=0.4225

(c) example contains a video of a person ripping paper played in reverse time; VideoSEMA correctly identifies the action with low confidence, whereas VideoMamba fails to classify it correctly despite assigning moderate confidence to its prediction.

Figure 6.3: Qualitative comparison of VideoSEMA and VideoMamba on the K400 test set. Shown are representative test samples along with the predicted labels and corresponding highest confidence scores.

Notably, the second row group of Table 6.1 includes models with substantially larger parameter counts and FLOPs. Despite operating under much smaller computational budget, VideoSEMA consistently achieves superior performance. This demonstrates that VideoSEMA is a not only effective, but also highly efficient video model capable of outperforming significantly larger counterparts.

In addition, we present a few visual example comparisons between VideoSEMA and Video-

Table 6.1: Performance reported on standard K400 dataset. Here $F \times m \times n$ denotes by F the total FLOPs, m the number of testing segments, n the number of testing crops. Res: resolution, T: transformer. P(M): parameter size in millions. T1: top-1, T5: top-5. C: CNN, CT: CNN +T, M: Mamba, Ml: Mamba-like.

Method	Type	P(M)	FLOPs (G)	Res	T1(%)	T5(%)
X3D-XL [16]	C	20	$194 \times 3 \times 10$	16×224^2	80.4	94.6
Swin-T [51]	T	28	$88 \times 3 \times 4$	32×224^2	78.8	93.6
MViTv1-B [15]	CT	37	$70 \times 1 \times 5$	32×224^2	80.2	94.4
MViTv2-S [40]	CT	35	$64 \times 1 \times 5$	16×224^2	81.0	94.6
Uniformer-S [36]	CT	21	$42 \times 1 \times 4$	16×224^2	80.8	94.7
WLiT [75]	T	22	$21 \times 1 \times 4$	8×224^2	74.6	92.0
TimeSformer-L [3]	T	121	$2380 \times 3 \times 1$	16×224^2	80.7	94.7
Uniformer-S [36]	CT	311	$3992 \times 3 \times 4$	16×224^2	81.3	94.7
Mformer-HR [67]	T	311	$959 \times 3 \times 10$	16×336^2	81.1	95.2
VideoMamba-M [37]	M	74	$202 \times 3 \times 4$	16×224^2	81.9	95.4
VideoMamba-S [37]	M	26	$34 \times 3 \times 4$	8×224^2	79.3	94.2
VideoMamba-S [37]	M	26	$68 \times 3 \times 4$	16×224^2	80.8	94.8
VideoMamba-S [37]	M	26	$135 \times 3 \times 4$	32×224^2	81.5	95.2
VideoSEMA (Ours)	Ml	31	$46 \times 3 \times 4$	8×224^2	**81.3**	**94.9**
VideoSEMA (Ours)	Ml	31	$87 \times 3 \times 4$	16×224^2	**82.4**	**95.4**
VideoSEMA (Ours)	Ml	31	$172 \times 3 \times 4$	32×224^2	**82.6**	**95.2**

Mamba on somewhat challenging predictions in Fig. 6.3. Example (A) depicts a video of a person peeling potatoes. VideoSEMA correctly predicts the action with a high probability of 90%, while VideoMamba incorrectly classifies it as the similar label peeling apples. The global action of peeling is correct for both models; however, this particular video is challenging due to the local distinction between apples and potatoes, which occupy only a small region of the image. This could be explained by the window attention of SEMA, which keeps the fine details. Example (B) shows a person drinking shots. VideoSEMA correctly identifies the action with a medium probability of 50%, while VideoMamba misidentifies it as tasting beer. Drinking shots involves rapidly consuming a small amount of liquid, while tasting beer is a slower action. This shows that the attention in time between frames allows VideoSEMA to understand quick changes in motion between frames, while VideoMamba processes these tokens in sequential order, thus reducing its temporal capability. Lastly, example (C) shows a video of a person ripping paper recorded in reverse time. VideoSEMA classifies it correctly

with low confidence of 20%, while VideoMamba predicts folding napkins. Because the video plays in reverse, the model needs strong temporal understanding. VideoSEMA with temporal attention allows the model to access frames in both directions simultaneously, which helps the model prediction. In contrast, VideoMamba's forward-direction processing observes the person putting the paper together and thus predicts folding napkins.

Table 6.2: Performance (ablation study) of VideoSEMA using various temporal processing units on K400 dataset. Here $F \times m \times n$ denotes by F the total FLOPs, m the number of testing segments, n the number of testing crops.

Time Component	# Params (M)	FLOPs (G)	Res	Top-1 (%)
3DConv (ker=7)	26	$40 \times 3 \times 4$	8×224^2	80.0
3DConv (ker=7)	26	$76 \times 3 \times 4$	16×224^2	79.3
3DConv (ker=13)	26	$76 \times 3 \times 4$	16×224^2	80.8
Mamba	36	$47 \times 3 \times 4$	8×224^2	76.3
Attention	31	$46 \times 3 \times 4$	8×224^2	**81.3**

SSv2

Table 6.3: Performance reported on standard SSv2 dataset. Here $F \times m \times n$ denotes by F the total FLOPs, m the number of testing segments, and n the number of testing crops. Res: resolution, T: transformer. P(M): parameter size in millions. T1: top-1, T5: top-5. C: CNN, CT: CNN +T, M: Mamba, Ml: Mamba-like.

Method	Type	P(M)	FLOPs (G)	Res	T1(%)	T5(%)
CT-Net$_{R50}$ [35]	C	21	$75 \times 1 \times 1$	16×224^2	64.5	89.3
TDN$_{R50}$ [86]	C	26	$75 \times 1 \times 1$	16×224^2	65.3	**91.6**
WLiT [75]	T	22	$50 \times 3 \times 1$	16×224^2	66.3	91.5
VideoMAE [79]	T	22	$57 \times 2 \times 3$	16×224^2	66.8	90.3
MViTv1-B [15]	CT	37	$71 \times 3 \times 1$	16×224^2	64.7	89.2
VideoMamba-S [37]	M	26	$34 \times 3 \times 2$	8×224^2	65.2	89.6
VideoMamba-S [37]	M	26	$68 \times 3 \times 2$	16×224^2	66.0	90.2
VideoSEMA (Ours)	Ml	31	$46 \times 3 \times 2$	8×224^2	**67.2**	91.0
VideoSEMA (Ours)	Ml	31	$87 \times 3 \times 2$	16×224^2	**67.3**	90.4

We present the overall performance of VideoSEMA on the SSv2 dataset in Table 6.3. We observe that with 8 frames inputs, VideoSEMA achieve a 2% improvement over VideoMamba-S, and with 16 frames inputs, it outperforms VideoMAE by 0.5%. Notably, using only 8

frames, VideoSEMA is able to outperform other models that utilize all 16 frames. This demonstrates that VideoSEMA achieves strong efficacy and temporal efficiency. We also note that the performance improvement with 16 frames is smaller compared to that with 8 frames. One possible explanation is the short temporal duration of videos in SSv2, thus limiting the benefit of longer input sequences.

6.4.4 Ablation Study

We extend the SEMA model from image domain to processing videos by adding a temporal processing unit. There are many standard choices for this component such as convolution, attention, and Mamba. In this subsection, we examine the performance of each choice. For convolution, to maintain relatively low parameter count, we employed a depthwise convolution. For Mamba, we use a single directional Mamba from VideoMamba [37]. And lastly, for attention we use softmax full attention. From Table 6.2, we observe that on K400, convolution performs relatively well, and as the number of frames increases, we need larger temporal convolutional kernel. Mamba on the other hand, performs poorly as a temporal processing unit. One possible explanation is that Mamba is applied only in the temporal dimension, resulting in independent operations on spatial positions. Lastly, since attention provides the best trade-off between accuracy and efficiency, we use attention as a temporal processing unit for VideoSEMA on the datasets here.

6.4.5 Scalability to Higher Resolution Videos

Table 6.4: Top-1 accuracy (%) at higher input resolutions to models pretrained on 16×224^2 videos of K400, without fine-tuning.

Method	16×224^2 (baseline)	16×512^2	16×1024^2
VideoMamba	80.8	74.4	40.8
VideoSEMA	82.4	75.3	54.6

SEMA is designed to handle high-resolution frames. In this subsection, we examine the adaptability of the model when applied to larger input resolutions. We use the model pretrained on 224^2 and evaluate it on larger input sizes of 512^2 and 1024^2 with 16 frames on K400. From Tab. 6.4, we observe that VideoMamba and VideoSEMA perform similarly on 224^2; however, at 1024^2, the performance of VideoMamba degrades much more quickly compared to VideoSEMA. In particular, the performance gap between VideoMamba and VideoSEMA is about 13.8 percentage points. This demonstrates the robustness and scalability of VideoSEMA for large input resolutions.

6.5 Conclusion

We introduced a space-time attention model (VideoSEMA) consisting of a scalable and efficient Mamba-like attention (SEMA) in space and a regular attention in time. For moderate number of video frames, the model out-performs recent space-time transformers and video-mamba of larger (comparable) sizes on benchmark K400 (SSv2) datasets.

Chapter 7

Concluding Remarks

The pace of research in this field moves at light speed, so many of the proposed methods do not stand the test of time. Many model designs are based on engineering intuition and lack theoretical guarantees. In this dissertation, we attempted to provide some mathematical justification for our design choices. Neural networks can be very complicated in nature, so simple theoretical results may not work well in practice. Thus, for the theoretical results proposed in this dissertation, we tried to make weak assumptions, and consequently the results may be mild. This allows our approach to be more practical in nature. For example, working with well-known i.i.d. distributions would provide strong bounds for stochastic results; however, for real datasets, these assumptions may be too strong. In addition, there are multiple layers in deep neural networks. Assuming a simplified setting may allow us to write down exact mathematical expressions; however, this limits the expressiveness of our theoretical results. A natural direction for these works would be to delve into more complicated mathematical settings that match deep neural network structures, such as positional embeddings, skip connections, normalization, and multi-head attention.

On the topic of efficiency, the field is moving toward larger problems, such as processing

video, language, and audio together using Vision-Language Models (VLMs). These problems demand larger models in terms of the number of parameters. For example, some Qwen models have billions of parameters, and some LLMs have reached trillions of parameters. As the input length increases through the use of more input media, a model will require tremendous computation to create an output. This increases the demand for efficient models so that they are economical to run at a large scale, as reflected, for example, in token pricing. Thus, the field of efficient models will continue to expand, even though these models can utilize large server infrastructures. On the other hand, a model also needs to be useful on edge devices such as phones, laptops, cars, and robots. These applications can range from photo editing, sending texts, and summarizing emails and documents to autonomous driving and working in hazardous environments. This requires models to work with very limited resources on edge devices under strict latency and memory constraints. Besides quantization and pruning, models need to be natively designed to be efficient prior to these steps. Lastly, models are mostly trained and run inference on NVIDIA chips through CUDA. However, many edge devices are built differently and may not accommodate many common operations. Thus, an efficient model on an edge device may be different from an efficient model on a server. Utilizing simple built-in operations on edge devices is therefore very important and requires a specialized understanding of each chip. For example, the Mamba kernel was designed to take advantage of the memory-handling architecture of modern GPUs to speed up computation, but it may not be as friendly to many edge devices. This is one of the reasons we invested a great deal of time in this dissertation designing the SEMA mechanism to mimic Mamba.

I would personally like to end this dissertation with the first lesson my advisor, Prof. Jack Xin, gave me: "Don't fight your grandfather's battle; fight your own battle." This fits perfectly with current AI research, as the entire world is eager to figure out how things work. I am blessed to have had the chance to make my own mark.

Bibliography

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

[2] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020.

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning (ICML)*, July 2021.

[4] O. Bjørnstad, R. Ims, and X. Lambin. Spatial population dynamics: analyzing patterns and processes of population synchrony. *Tree*, 14(11):427–432, 1999.

[5] Qiang Chen, Qiman Wu, Jian Wang, Qinghao Hu, Tao Hu, Errui Ding, Jian Cheng, and Jingdong Wang. Mixformer: Mixing features across windows and dimensions. *CVPR*, 2022.

[6] J. Chow and G. Rogers. Power system toolbox. Available at `https://www.ecse.rpi.edu/~chowj/`, 2008. Accessed: 2023-07-19.

[7] T-W Chuang, L. F. Chaves, and P-J Chen. Effects of local and regional climatic fluctuations on unprecedented dengue outbreaks in southern Taiwan. *PLoS One*, 12(7) e0181638, 2017.

[8] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. `https://github.com/open-mmlab/mmsegmentation`, 2020.

[9] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18613–18624. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/d85b63ef0ccb114d0a3bb7b7d808028f-Paper.pdf`.

[10] Elisha Dayag, Nhat Thanh Tran, and Jack Xin. USEMA: a scalable efficient mamba like attention for medical image segmentation, 2026. URL https://arxiv.org/abs/2605.11131.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Li Kai, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[12] J. Ding, S. Ma, L. Dong, X. Zhang, S. Huang, W. Wang, N. Zheng, and F. Wei. Longnet: Scaling transformers to 1,000,000,000 tokens. *arXiv preprint arXiv:2307.02486*, 2023.

[13] Xiaoyi Dong, Jianmin Bao, Dongdong Chen, Weiming Zhang, Nenghai Yu, Lu Yuan, Dong Chen, and Baining Guo. Cswin transformer: A general vision transformer backbone with cross-shaped windows. *CVPR*, 2022.

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

[15] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6824–6835, October 2021.

[16] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[17] R. Glowinski, S. J. Osher, and W. Yin. Splitting methods in communication, imaging, science, and engineering. *Springer*, 2017.

[18] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5843–5851, 2017. doi: 10.1109/ICCV.2017.622.

[19] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

[20] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Re. Hippo: Recurrent memory with optimal polynomial projections. *NeurIPS*, 2020.

[21] J. Guibas, M. Mardani, Z. Li, A. Tao, A. Anandkumar, and B. Catanzaro. Efficient token mixing for transformers via adaptive Fourier neural operators. *International Conference on Learning Representations*, 2022.

[22] M. Guo, C. Lu, Z. Liu, M. Cheng, and S. Hu. Visual attention network. *Computational Visual Media*, 9(4):733–752, 2023.

[23] Dongchen Han, Xuran Pan, Yizeng Han, Shiji Song, and Gao Huang. Flatten transformer: Vision transformer using focused linear attention. *ICCV*, 2023.

[24] Dongchen Han, Yifan Pu, Zhuofan Xia, Yizeng Han, Xuran Pan, Xiu Li, Jiwen Lu, Shiji Song, and Gao Huang. Bridging the divide: Reconsidering softmax and linear attention. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=RSiGFzQapl`.

[25] Dongchen Han, Ziyi Wang, Zhuofan Xia, Yizeng Han, Yifan Pu, Chunjiang Ge, Jun Song, Shiji Song, Bo Zheng, and Gao Huang. Demystify Mamba in Vision: A Linear Attention Perspective. *NeurIPS*, 2024.

[26] Dongchen Han, Ziyi Wang, Zhuofan Xia, Yizeng Han, Yifan Pu, Chunjiang Ge, Jun Song, Shiji Song, Bo Zheng, and Gao Huang. Demystify Mamba in Vision: A Linear Attention Perspective. *NeurIPS*, 2024.

[27] Ali Hassani, Steven Walton, Jiachen Li, Shen Li, and Humphrey Shi. Neighborhood attention transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6185–6194, June 2023.

[28] Tao Huang, Xiaohuan Pei, Shan You, Fei Wang, Chen Qian, and Chang Xu. LocalMamba: Visual state space model with windowed selective scan. *arXiv:2403.09338*, 2024.

[29] Zilong Huang, Youcheng Ben, Guozhong Luo, Pei Cheng, Gang Yu, and Bin Fu. Shuffle transformer: Rethinking spatial shuffle for vision transformer. *arXiv:2106.03650*, 2021.

[30] A. Katharopoulos, A. Vyas, N. Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. *ICML*, 2020.

[31] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv:1705.06950*, 2017.

[32] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks, 2018.

[33] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon. Fnet: Mixing tokens with Fourier transforms. *Proc. Conf. North American Chapter of Association Comput. Linguistics: Human Language Tech.*, pages 4296 – 4313, 2022.

[34] J. Li, M. Yue, Y. Zhao, and G. Lin. Machine-learning-based online transient analysis via iterative computation of generator dynamics. *in Proc. IEEE SmartGridComm*, 2020.

[35] Kunchang Li, Xianhang Li, Yali Wang, Jun Wang, and Yu Qiao. CT-Net: Channel tensorization network for video classification. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=UoaQUQREMOs`.

[36] Kunchang Li, Yali Wang, Gao Peng, Guanglu Song, Yu Liu, Hongsheng Li, and Yu Qiao. Uniformer: Unified transformer for efficient spatial-temporal representation learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nBU_u6DLvoK`.

[37] Kunchang Li, Xinhao Li, Yi Wang, Yinan He, Yali Wang, Limin Wang, and Yu Qiao. VideoMamba: State space model for efficient video understanding. *arXiv:2403.06977, in ECCV*, 2024.

[38] M. Li, X. Zhao, R. Liu, C. Li, X. Wang, and X. Chang. Generalizable memory-driven transformer for multivariate long sequence time-series forecasting. *arXiv preprint arXiv:2207.07827*, 2022.

[39] Rui Li, Jianlin Su, Chenxi Duan, and Shunyi Zheng. Linear attention mechanism: An efficient attention for semantic segmentation. *arXiv:2007.14902*, 2020.

[40] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. MViTv2: Improved multiscale vision transformers for classification and detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4804–4814, June 2022.

[41] Z. Li, J. Xin, and G. Zhou. An integrated recurrent neural network and regression model with spatial and climatic couplings for vector-borne disease dynamics. In *Proc. of Int. Conf. Pattern Recognition Applications Methods*, pages 505–510, 2022.

[42] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1.

[43] Rebecca Lindsey. Climate variability: Oceanic niño index, 2009. URL `https://www.climate.gov/news-features/understanding-climate/climate-variability-oceanic-nino-index`. Accessed: 2024-03-01.

[44] Rebecca Lindsey. Climate variability: Southern oscillation index, 2009. URL `https://www.climate.gov/news-features/understanding-climate/climate-variability-southern-oscillation-index`. Accessed: 2024-03-01.

[45] Aixin Liu, Bei Feng, Bin Wang, Bingxuan Wang, Bo Liu, Chenggang Zhao, Chengqi Dengr, Chong Ruan, Damai Dai, Daya Guo, and et al. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model, 2024. URL https://arxiv.org/abs/2405.04434.

[46] Leiye Liu, Miao Zhang, Jihao Yin, Tingwei Liu, Wei Ji, Yongri Piao, and Huchuan Lu. DefMamba: Deformable visual state space model. *arXiv:2504.05794*, 2025.

[47] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long. itransformer: Inverted transformers are effective for time series forecasting. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

[48] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. VMamba: Visual State Space Model. *NeurIPS*, 2024.

[49] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, page 12009–12019, 2022.

[50] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, October 2021.

[51] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3202–3211, June 2022.

[52] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s . In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11966–11976, Los Alamitos, CA, USA, June 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.01167. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.01167.

[53] P. Liyanage, Y. Tozan, H. Overgaard, H. Tissera, and J. Rocklöv. Effect of El Niño-southern oscillation and local weather on Aedes vector activity from 2010 to 2018 in Kalutara district, Sri Lanka: a two-stage hierarchical analysis. *Lancet Planetary Health*, 6:e577–e585, 2022.

[54] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[55] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[56] S. McGough, L. Clemente, J. N. Kutz, and M. Santillana. A dynamic, ensemble learning approach to forecast dengue fever epidemic years in Brazil using weather and population susceptibility cycles. *J. R. Soc. Interface*, 18:20201006, 2021.

[57] S. Mehta and M. Rastegari. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. *ICLR*, 2022.

[58] C. Morin, A. Comrie, and K. Ernst. Climate and dengue transmission: evidence and implications. *Environ. Health Perspect.*, 121:1264–1272, 2013.

[59] E. Nadaraya. On estimating regression. *Theory of Probability and its Applications*, 9: 141–142, 1964.

[60] F. Nejad and K. Varathan. Identification of significant climatic risk factors and machine learning models in dengue outbreak prediction. *BMC Medical Informatics and Decision Making*, 21, 2021.

[61] H. Nguyen, T. Tran, J. Mulhall, M. Hoang, Q. Duong, C. Nguyen, N. Nguyen, M. Hoang L. Vu and, C. Do, L. Tran B. Nguyen, Q. Nguyen, T. Nguyen, N. Ngu, A. Le, D. Phan, H. Nguyen, and S. Mai. Deep learning models for forecasting dengue fever based on climate data in Vietnam. *PLOS Neglected Tropical Diseases*, 2022. URL `https://doi.org/10.1371/journal.pntd.0010509`.

[62] L. Nguyen, H. Le, D. Nguyen, H. Ho, and T-W Chuang. Impact of climate variability and abundance of mosquitoes on dengue transmission in central Vietnam. *Journal of Environmental Research and Public Health*, 17(7):2453, 2020.

[63] T. Nguyen, M. Pham, T. Nguyen, K. Nguyen, S. Osher, and N. Ho. Fourierformer: Transformer meets generalized Fourier integral theorem. *Advances in Neural Information Processing Systems*, 2022.

[64] Tan Nguyen, Tam Nguyen, Nhat Ho, Andrea Bertozzi, Richard Baraniuk, and Stanley Osher. A primal-dual framework for transformers and neural networks. *in Proc. of ICLR*, 2023.

[65] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=Jbdc0vTOcol`.

[66] E. Parzen. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33:1065–1076, 1962.

[67] Mandela Patrick, Dylan Campbell, Yuki Asano, Ishan Misra, Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and Joao F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=mfQxdSMWOF`.

[68] Y. Rao, W. Zhao, Z. Zhu, J. Lu, and Jie Zhou. Global filter networks for image classification. *Advances in Neural Information Processing Systems*, 34:980–993, 2021.

[69] M. Rosenblatt. Remarks on some nonparametric estimates of a density function. *Annals of Mathematical Statistics*, page 832–837, 1956.

[70] N. Saji, B.N. Goswami, P.N. Vinayachandran, and T. Yamagata. A dipole mode in the tropical Indian ocean. *Nature*, 1999.

[71] O. Santangelo, V. Gentile, D. Giordano S. Pizzo, and F. Cedrone. Machine learning and prediction of infectious disease: A systematic review. *Mach. Learn. Knowl. Extr.*, 5(1):175–198, 2023.

[72] Skipper Seabold and Josef Perktold. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*, 2010.

[73] Noam Shazeer. Fast transformer decoding: One write-head is all you need, 2019. URL `https://arxiv.org/abs/1911.02150`.

[74] G. Strang. On the construction and comparison of difference schemes. *SIAM Journal on Numerical Analysis*, 5(3):506–517, 1968.

[75] Ruochen Sun, Tian Zhang, Yiming Wan, Feng Zhang, and Jian Wei. WLiT: Windows and linear transformer for video action recognition. *Sensors (Basel, Switzerland)*, 23 (3):1616, 2023. doi: 10.3390/s23031616. URL `https://doi.org/10.3390/s23031616`.

[76] Xue-Cheng Tai, Hao Liu, Lingfeng Li, and Raymond H Chan. A mathematical explanation of transformers. *arXiv:2510.03989*, 2025.

[77] Olav Titus Muurlink, Peter Stephenson, Mohammad Zahirul Islam, and Andrew W. Taylor-Robinson. Long-term predictors of dengue outbreaks in Bangladesh: A data mining approach. *Infectious Disease Modelling*, 3:322–330, 2018. ISSN 2468-0427.

[78] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Peter Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. MLP-mixer: An all-MLP architecture for vision. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=EI2KOXKdnP`.

[79] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 10078–10093. Curran Associates, Inc., 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/416f9cb3276121c42eebb86352a4354a-Paper-Conference.pdf`.

[80] Nhat Thanh Tran and Jack Xin. Fourier-mixed window attention for efficient and robust long sequence time-series forecasting. *Frontiers in Applied Mathematics and Statistics*, Volume 11 - 2025, 2025. ISSN 2297-4687. doi: 10.3389/fams.2025.1600136. URL https://www.frontiersin.org/journals/applied-mathematics-and-statistics/articles/10.3389/fams.2025.1600136.

[81] Nhat Thanh Tran, Fanghui Xue, Shuai Zhang, Jiancheng Lyu, Yunling Zheng, Yingyong Qi, and Jack Xin. SEMA: a scalable and efficient mamba like attention via token localization and averaging. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=H3vDDSHWos.

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

[83] Petar Veličković, Christos Perivolaropoulos, Federico Barbero, and Razvan Pascanu. Softmax is not enough (for sharp size generalisation). In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=S4JmmpnSPy.

[84] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, Cambridge, 2 edition, 2026.

[85] Chenting Wang, Kunchang Li, Tianxiang Jiang, Xiangyu Zeng, Yi Wang, and Limin Wang. Make your training flexible: Towards deployment-efficient video models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 23880–23891, October 2025.

[86] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. Tdn: Temporal difference networks for efficient action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1895–1904, June 2021.

[87] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. *Computational Visual Media*, 8(3):415–424, 2022. doi: 10.1007/s41095-022-0274-8. URL https://doi.org/10.1007/s41095-022-0274-8.

[88] Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng, Zun Wang, Yansong Shi, Tianxiang Jiang, Songze Li, Jilan Xu, Hongjie Zhang, Yifei Huang, Yu Qiao, Yali Wang, and Limin Wang. Internvideo2: Scaling foundation models for multimodal video understanding. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 396–416, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-73013-9.

[89] G. Woo, C. Liu, D. Sahoo, A. Kumar, and S. Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv:2202.01381*, 2022.

[90] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv:2202.01381*, 2022.

[91] H. Wu and et al. A whole-slide foundational model for digital pathology from real-world data. *Nature*, 630:181–188, 2024.

[92] H. Wu, J. Xu, J. Wang, and M. Long. Autoformer: Decomposition transformers with Auto-Correlation for long-term series forecasting. In *NeurIPS*, 2021.

[93] Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[94] C. Yang, S. Qiao, Q. Yu, X. Yuan, Y. Zhu, A. Yuille, H. Adam, and L-C. Chen. Moat: Alternating mobile convolution and attention brings strong vision models. *arXiv preprint arXiv:2210.01820*, 2022.

[95] Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. Differential transformer. *arXiv:2410.05258*, 2024.

[96] Weihao Yu and Xinchao Wang. Mambaout: Do We Really Need Mamba for Vision? *CVPR*, 2025.

[97] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6022–6031, 2019. doi: 10.1109/ICCV.2019.00612.

[98] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=r1Ddp1-Rb`.

[99] P. Zhang, Z. Wang, G. Chao, Y. Huang, and J. Yan. An oriented attention model for infectious disease cases prediction. *Theory and Practices in Artificial Intelligence; Lecture Notes in Computer Science book series*, 13343, 2022.

[100] Y. Zheng, C. Hu, G. Lin, M. Yue, B. Wang, and J. Xin. Glassoformer: a query-sparse transformer for post-fault power grid voltage prediction. *Proc. of IEEE Inter. Conf. on Acoustics, Speech, and Signal Processing*, pages 3968–3972, 2022.

[101] Y. Zheng, Z. Xu, F. Xue, B. Yang, J. Lyu, S. Zhang, Y. Qi, and J. Xin. AFIDAF: Alternating Fourier and Image Domain Adaptive Filters as an Efficient Alternative to Attention in ViTs. *International Symposium of Visual Computing, Reno, NV*, 15046: 17–30, 2024.

[102] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34 (07):13001–13008, Apr. 2020. doi: 10.1609/aaai.v34i07.7000. URL `https://ojs.aaai.org/index.php/AAAI/article/view/7000`.

[103] G. Zhou, N. Minakawa, A. Githeko, and G. Yan. Association between climate variability and malaria epidemics in the east African highlands. *Proceedings of the National Academy of Sciences*, 101(8):2375–2380, 2004.

[104] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *in Proc. of AAAI*, 35:11106–11115, 2021.

[105] H. Zhou, J. Li, S. Zhang, S. Zhang, M. Yan, and H. Xiong. Expanding the prediction capacity in long sequence time-series forecasting. *Artificial Intelligence*, 318:103886, 2023. ISSN 0004-3702.

[106] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. *ICML*, 2022.

[107] X. Zhu, B. Fu, Y. Yang, Y. Ma, J. Hao, S. Chen, S. Liu, T. Li, S. Liu, W. Guo, and Z. Liao. Attention-based recurrent neural network for influenza epidemic prediction. *BMC Bioinform.*, 20-S(18):art. no. 575: 1–10, 2019.