UNIVERSITY OF CALIFORNIA SAN DIEGO

Unifying Physics and Semantics for Robust Sensor Time Series Analysis

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Xiyuan Zhang

Committee in charge:

Professor Rajesh K. Gupta, Co-Chair
Professor Jingbo Shang, Co-Chair
Professor Tarek Abdelzaher
Professor Julian McAuley
Professor Xinyu Zhang

2024

The Dissertation of Xiyuan Zhang is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This dissertation would not have been possible without the support of many incredible people I have had the fortunate to meet.

I extend my deepest gratitude to my advisors Rajesh K. Gupta and Jingbo Shang. I cannot thank you enough for giving me this opportunity to pursue PhD and for the endless support you have provided throughout the entire journey. Rajesh, your depth of thinking and intellectual rigor have profoundly shaped my development as an independent researcher. Thank you so much for giving me the academic freedom to explore my research interests, while continuously guiding me to stay in the right directions whenever I felt lost. Jingbo, I have learned so much from your invaluable insights and thoughtful discussions. Thank you so much for the countless hours of insightful meetings and all your detailed feedback. Your example as a role model has encouraged me to strive for excellence. I feel extremely fortunate to have both of you as my advisors. Thank you for all your support, encouragement and career guidance, and thank you for being the most caring advisors, not only for our research but also for our well-being as PhD students.

I am also thankful to Julian McAuley, Xinyu Zhang, Tarek Abdelzaher for serving on my thesis committee and providing valuable feedback to my thesis and presentations. Special thanks to Geoffrey Voelker, Rose Yu and Lily Weng for joining my research exam committee.

I want to thank the MESL alumni and my fellow lab mates. Thank Dezhi Hong for your valuable insights which have significantly enhanced my projects and refined my work. Thank Ranak Roy Chowdhury, Jiayun Zhang, Xiaohan Fu, Shuheng Li, Hsin-Yu Liu, Chengyu Dong, Keerthivasan Vijayakumar for your helpful suggestions and contributions to our research projects. The opportunity to learn from your research has greatly enriched my understanding of the field.

I am grateful to Qualcomm for awarding me the Qualcomm Innovation Fellowship, and to my mentors Diyan Teng and Rashmi Kulkarni for your detailed and constructive suggestions and feedback throughout our projects.

During my PhD, I am also extremely fortunate to intern twice at AWS AI. I am deeply grateful to my managers Bernie Wang and Danielle C. Maddix for offering me these amazing

opportunities and guiding me to work on these exciting projects. I also want to thank my mentors Abdul Fatir Ansari, Xiaoyong Jin, Hao Wang, Boran Han, Youngsuk Park, Gaurav Gupta, and Karthick Gopalswamy, for your inspiring discussions and valuable suggestions to our projects.

PhD journey is not only about research but also a wonderful life experience itself. I want to thank Minghua and Jojo for your support and for sharing numerous cherished moments. Thank Jiayun and Chenyang for all the laughter and unforgettable memories we shared. Thank Xiaofan, Sophia and Xintong for all the happy gatherings. Thank Shiqi for the memorable experiences we had together. Thank Shihan and Zhankui for always offering kind help. Special thanks to Tianlu for all the helpful academic and personal advice.

Finally, I want to thank my parents, for your unconditional love and continuous encouragement throughout my entire life.

I am also grateful to my co-authors who kindly approved the following publications and material to be included in my dissertation:

Chapter 2 incorporates material from the publication "Physics-Informed Data Denoising for Real-Life Sensing Systems", by Xiyuan Zhang, Xiaohan Fu, Diyan Teng, Chengyu Dong, Keerthivasan Vijayakumar, Jiayun Zhang, Ranak Roy Chowdhury, Junsheng Han, Dezhi Hong, Rashmi Kulkarni, Jingbo Shang, Rajesh Gupta, published in Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems (SenSys 2023). The dissertation author was primary investigator and the lead author of this paper.

Chapter 3 incorporates material from the publication "ESC-GAN: Extending Spatial Coverage of Physical Sensors", by Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, and Dezhi Hong, published in Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM 2022). The dissertation author was primary investigator and the lead author of this paper.

Chapter 4 incorporates material from the publication "Towards Diverse and Coherent Augmentation for Time-Series Forecasting", by Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, Dezhi Hong, published in ICASSP 2023-2023 IEEE International Confer-

ence on Acoustics, Speech and Signal Processing (ICASSP 2023). The dissertation author was primary investigator and the lead author of this paper.

Chapter 5 incorporates material from the publication "Large Language Models for Time Series: A Survey", by Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh Gupta, Jingbo Shang, published in the survey track of the 33rd International Joint Conference on Artificial Intelligence (IJCAI 2024). The dissertation author was primary investigator and the lead author of this paper.

Chapter 6 incorporates material from the publication "Unleashing the Power of Shared Label Structures for Human Activity Recognition", by Xiyuan Zhang, Ranak Roy Chowdhury, Jiayun Zhang, Dezhi Hong, Rajesh Gupta, Jingbo Shang, published in Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM 2023). The dissertation author was primary investigator and the lead author of this paper.

Chapter 7, in part is currently being prepared for submission for publication of the material. Zhang, Xiyuan; Teng, Diyan; Chowdhury, Ranak Roy; Li, Shuheng; Hong, Dezhi; Gupta, Rajesh K.; Shang, Jingbo. The dissertation author was the primary investigator and author of this material.

VITA

2020 B.S. in Computer Science, Zhejiang University

2024 Ph.D. in Computer Science, University of California San Diego

PUBLICATIONS

Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh Gupta, Jingbo Shang, "Large Language Models for Time Series: A Survey", In the 33rd International Joint Conference on Artificial Intelligence, *IJCAI 2024, Survey Track*

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, **Xiyuan Zhang**, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, Yuyang Wang, "Chronos: Learning the Language of Time Series", *Preprint*

Xiyuan Zhang, Ranak Roy Chowdhury, Jiayun Zhang, Dezhi Hong, Rajesh Gupta, Jingbo Shang, "Unleashing the Power of Shared Label Structures for Human Activity Recognition", In Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, *CIKM 2023*

Xiyuan Zhang, Xiaohan Fu, Diyan Teng, Chengyu Dong, Keerthivasan Vijayakumar, Jiayun Zhang, Ranak Roy Chowdhury, Junsheng Han, Dezhi Hong, Rashmi Kulkarni, Jingbo Shang, Rajesh Gupta, "Physics-Informed Data Denoising for Real-Life Sensing Systems", In Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems, *SenSys 2023*

Jiayun Zhang, **Xiyuan Zhang**, Xinyang Zhang, Dezhi Hong, Rajesh Gupta, Jingbo Shang, "Navigating Alignment for Non-Identical Client Class Sets: A Label Name-Anchored Federated Learning Framework", In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, *KDD 2023*

Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, Dezhi Hong, "Towards Diverse and Coherent Augmentation for Time-Series Forecasting", In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing, *ICASSP 2023*

Xiaofan Yu, Ludmila Cherkasova, Harsh Vardhan, Quanling Zhao, Emily Ekaireb, **Xiyuan Zhang**, Arya Mazumdar, Tajana Šimunić Rosing, "Async-HFL: Efficient and Robust Asynchronous Federated Learning in Hierarchical IoT Networks", In Proceedings of the 8th ACM / IEEE Conference on Internet of Things Design and Implementation, *IoTDI 2023*

Ranak Roy Chowdhury, Jiacheng Li, **Xiyuan Zhang**, Dezhi Hong, Rajesh Gupta, Jingbo Shang, "PrimeNet: Pre-training for Irregular Multivariate Time Series", In Proceedings of the AAAI Conference on Artificial Intelligence, *AAAI 2023*

Xiyuan Zhang, Xiaoyong Jin, Karthick Gopalswamy, Gaurav Gupta, Youngsuk Park, Xingjian Shi, Hao Wang, Danielle C. Maddix, Yuyang Wang, "First De-Trend then Attend: Rethinking Attention for Time-Series Forecasting", Advances in Neural Information Processing Systems, *NeurIPS 2022 All Things Attention Workshop*

Ranak Roy Chowdhury, **Xiyuan Zhang**, Dezhi Hong, Rajesh Gupta, Jingbo Shang, "Task-Aware Reconstruction for Time-Series Transformer", In Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, *KDD 2022*

Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, and Dezhi Hong "ESC-GAN: Extending Spatial Coverage of Physical Sensors", In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining, *WSDM 2022*

Xiyuan Zhang, Chengxi Li, Dian Yu, Samuel Davidson, and Zhou Yu, "Filling Conversation Ellipsis for Better Social Dialog Understanding", In Proceedings of the AAAI Conference on Artificial Intelligence, *AAAI 2020*

ABSTRACT OF THE DISSERTATION

Unifying Physics and Semantics for Robust Sensor Time Series Analysis

by

Xiyuan Zhang

Doctor of Philosophy in Computer Science

University of California San Diego, 2024

Professor Rajesh K. Gupta, Co-Chair
Professor Jingbo Shang, Co-Chair

The past decade has witnessed a significant growth of deployed sensors in our daily life, covering applications from healthcare, climate modeling to home automation and robotics. These sensors collect abundant time series data, which facilitate our understanding of various real-life processes. However, the inherent instability of these sensors, combined with the dynamic environments in which they operate, often results in the collection of data that is noisy, sparse, insufficient and varied. This presents a significant contrast to the data typically encountered in other domains such as computer vision and natural language processing. Consequently, current data-driven models trained under controlled experimental settings often fall short of their value

in accurate inferencing and analyses.

To address these limitations, we propose to bridge the inherent physical contextual knowledge and external semantic contextual knowledge of sensor time series to build a more robust analysis framework for sensor data. Specifically, we first exploit the inherent physics principles underlying time series data — ranging from mathematical models, spatio-temporal correlations to spectral properties — as inherent contextual knowledge. We leverage such physics knowledge to refine raw sensor time series and enhance data quality through denoising, imputation and augmentation. Additionally, sensor time series are often accompanied with external label names or metadata presented as text. Therefore, we build upon the recent advances in language modeling, to incorporate external semantic contextual knowledge from large language models or pre-train our own domain-specific foundation models. Such semantic knowledge further enriches sensor time series understanding and increases cross-domain robustness. Our framework is robust and demonstrates state-of-the-art performance in multiple tasks such as recognition, forecasting and navigation across sensing systems of various scales, from small-scale personal healthcare monitoring, smart home automation, to large-scale smart building control, energy management, climate modeling and beyond.

Chapter 1

Introduction

1.1 Background

Over the past decade, there has been a substantial increase in the integration of sensors into our daily lives, driven by the growing demand for applications such as healthcare monitoring, smart home automation, high-accuracy motion capture in video games, and the rising adoption of green energy technologies. According to recent news articles from Precedence Research, the global sensor market size was evaluated at USD 204.8 billion in 2022 and is expected to hit around USD 508.64 billion by 2032, growing at a Compound Annual Growth Rate (CAGR) of 8.40% from 2023 to 2032[1]. Recent analysis from IoT Analytics also predicts that by 2027, there will likely be more than 29 billion IoT connections[2]. Figure 1.1 visualizes the trend of global active IoT connections and sensor market size, whose increase benefits wide range of application domains from smart buildings, smart city, healthcare, to climate modeling, robotics and beyond.

Building on the expansive growth of sensor technology, its integration into our daily lives has become increasingly indispensable. These devices play critical roles in automating and enhancing decision-making processes from individual lives to large-scale industries such as agriculture and advanced manufacturing. The data collected by these sensors is typically in the form of time series, which is a sequence of data values indexed in time order. These time series

[1] https://www.precedenceresearch.com/sensor-market
[2] https://iot-analytics.com/number-connected-iot-devices

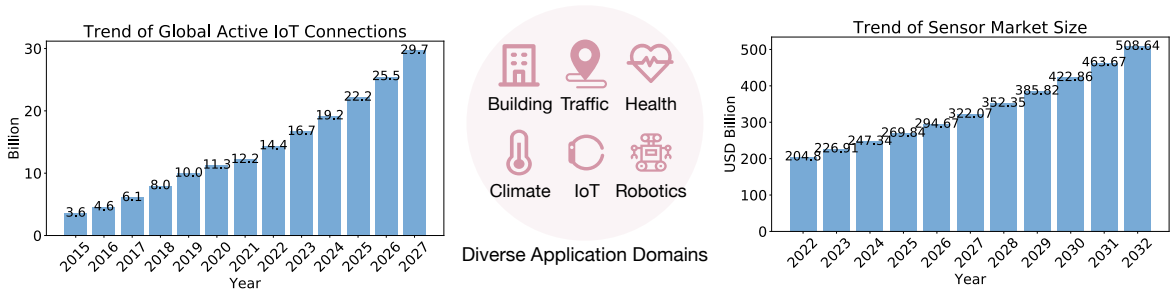


Figure 1.1. The number of sensors covering diverse application domains, as well as their market size has significantly increased over the years.

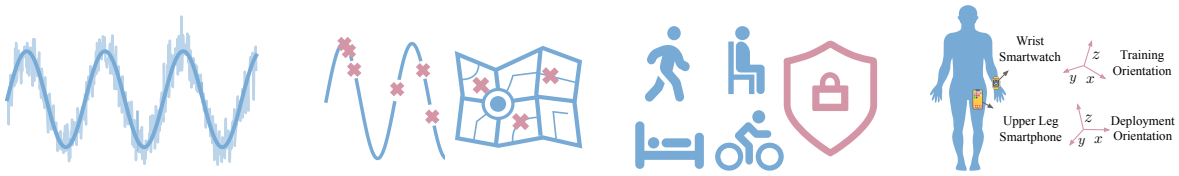
data, with its inherent temporal properties, enables a deeper understanding of the dynamics and patterns underlying the physical world.

1.2 Challenges

Despite the invaluable insights provided by sensor time series data, several challenges negatively impact their quality. First, the instability of sensors such as manufacturing variances and dynamic environmental conditions contributes to the inconsistency and unreliability of the data collected. Moreover, compared with large servers, such problems are especially prominent in edge-deployed sensors, which directly interact with humans and the physical world and are therefore in close proximity to the heterogeneity and instability of the real-world operating conditions. These two factors have complicated the collection of sensor time series and result in the following four major issues in terms of data quality (Figure 1.2):

- *Noise*. Sensor time series carry inherent sensor noise (Figure 1.2a). This includes random fluctuations generated by the sensor itself or by external environmental factors, such as electromagnetic interference or physical vibrations.

- *Sparsity*. This can be further categorized into temporal sparsity and spatial sparsity (Figure 1.2b). *Temporal sparsity* refers to data missing in the temporal dimension, resulting from factors such as sensor or storage failures, battery depletion and network disruption. *Spatial sparsity* refers to data missing in the spatial dimension, due to physical access



(a) Noise **(b)** Sparsity **(c)** Scarcity **(d)** Variability

Figure 1.2. Major challenges regarding sensor time series quality.

constraints or the high economic costs to construct and maintain sensors at certain locations.

- *Scarcity*. Sensor time series, especially those involving human subjects such as health related daily activity data, remain difficult to collect due to security or privacy concerns (Figure 1.2c), as human subjects involved in the collection process may not consent to data sharing or data transmission over the network.

- *Variability*. Sensor time series are inherently varied and display a wide variety of patterns. For instance, in the task of Human Activity Recognition (HAR), a smartphone placed on the upper leg exhibits distinct signal patterns compared with a smartwatch positioned on the wrist, and devices may show varying orientations during training and deployment phases (Figure 1.2d).

These challenges significantly undermine the quality of raw sensor data, leading to robustness and generalization problems of data-driven models trained in controlled experimental settings, such as those in computer vision (CV) and natural language processing (NLP), when applied to real-world sensing scenarios.

1.3 Contextual Knowledge

Although sensor time series present quality issues as mentioned in the previous paragraphs, they also have special properties or contextual knowledge that can be leveraged to effectively overcome these challenges. We further divide these contextual knowledge into inherent and external knowledge, as illustrated in Figure 1.3.

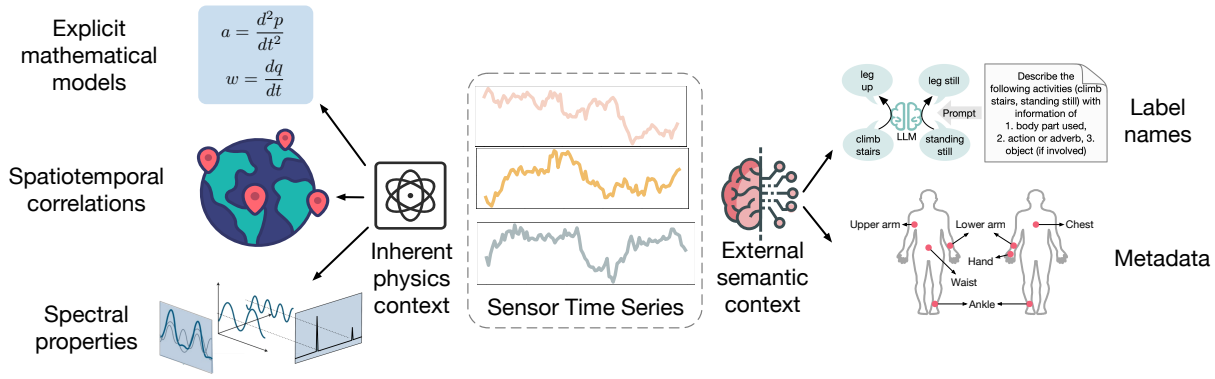


Figure 1.3. We can leverage inherent physical contextual knowledge (left) and external semantic contextual knowledge (right) to improve the robustness of sensor time series analysis.

Inherent knowledge refers to the physical attribute intrinsic to sensor time series data. Given that sensor time series model real-life physical processes, they are embedded with numerous explicit and implicit physical characteristics. Explicit physical principles may be represented through explicit mathematical equations or inequalities that describe the behavior of physical processes over time or across different variables. For example, the relationship between location (measured by GPS sensors) and acceleration (measured by accelerometers) is captured by the motion law, which equates the second-order derivative of location to acceleration. On the other hand, implicit physical properties might include spatial-temporal correlations or spectral properties that cannot be explicitly represented through mathematical formulations. For example, geo-sensors exhibit locally similar patterns in nearby locations as well as global trends, such as the decrease in temperature with increasing altitude. Another example is that transformation of time series data into the frequency domain can more accurately reflect seasonal patterns. Incorporation of these physics principles help denoise, impute and augment raw sensor time series and therefore improve the robustness of the models trained on these data.

External knowledge refers to the semantic context that is mostly extracted from other data modalities beyond time series itself. Sensor time series data are often accompanied by labels or metadata such as sensor locations in the form of textual information, offering an opportunity to employ state-of-the-art natural language processing techniques to elevate the level of abstraction

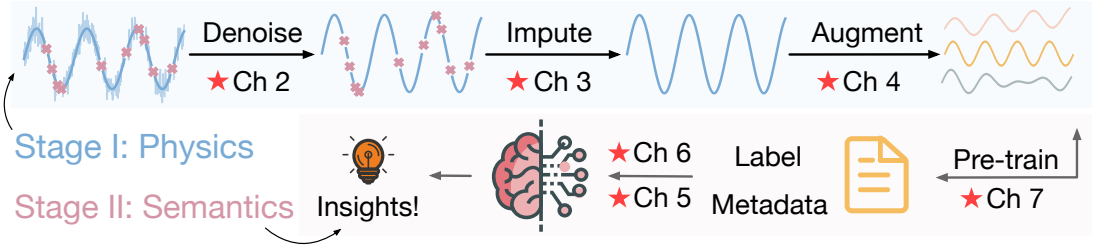


Figure 1.4. Our sensor time series analysis framework. Given input noisy and missing time series, stage I incorporates physical context to denoise, impute, augment raw data. Stage II leverages semantic context from LLMs or domain-specific pre-trained models to improve robustness.

and achieve uniform representations across different domains. For instance, sensor time series data from different domains may vary in sensor configurations and environmental conditions, yet share identical label names. Meanwhile, some domains may contain more sufficient samples to train data-driven machine learning models compared with other domains. By integrating semantic labels, we can identify commonalities across domains and enhance the model's ability to generalize from data-rich domains to domains with insufficient samples. Specifically, we leverage the latest advancements in large language models or pre-train our own domain-specific foundation models to improve the generalizability and robustness of sensor time series analysis.

1.4 Overall Framework

In light of these contextual knowledge, we present a robust sensor time series analysis framework as illustrated in Figure 1.4. In the first stage, we incorporate physical contextual knowledge to denoise [320] (chapter 2), impute missing values [317] (chapter 3) and augment [318] (chapter 4) sensor time series. The specific orders of these pre-processing steps are flexible. The first stage refines the quality of sensor time series, preparing it for subsequent analysis by data-driven models. In the second stage, we leverage the capabilities of existing large language models [315, 319] (chapter 5 and chapter 6) or pre-train our own domain-specific foundation models (chapter 7) to further extract insights using semantic contextual knowledge.

Table 1.1. Summary of contributions and applications of our proposed robust sensor time series analysis framework.

Context	Subcategory	Model	Chapter	Challenge	Application
Physics	mathematical model	PILOT	Ch 2	noise	inertial navigation, CO_2 monitoring, HVAC control
	spatio-temporal	ESC-GAN	Ch 3	sparsity	temperature, precipitation, air quality prediction
	spectral property	STAug	Ch 4	scarcity	electricity forecasting
Semantics	LLM	Survey	Ch 5	scarcity	general time series forecasting, classification, etc.
	LLM	SHARE	Ch 6	scarcity	human activity recognition
	foundation model	UniMTS	Ch 7	variability	human activity recognition

1.5 Contributions and Applications

We summarize our contributions and applications of the proposed framework for robust sensor time series analysis in Table 1.1. The technical contributions in this dissertation are supported by specific implementation tools that are available on the open-source Github repository as described in Appendix A. We also provide explanations of the notations used in each chapter in Appendix B. More specifically,

- PILOT[3] (Chapter 2) is the first generic, physics-informed denoising method that supports different sensing applications. PILOT offers a practical denoising solution for application scenarios where obtaining ground truth clean data or understanding the underlying noise distribution is challenging, better representing real-world sensing environments. Extensive experiments in three real-world applications of inertial navigation, CO_2 monitoring and HVAC control demonstrate that PILOT produces results that are closest to the clean data and most coherent to the laws of physics, achieving state-of-the-art denoising performance. We also deploy the model on edge devices and show real-time denoising capability.
- ESC-GAN[4] (Chapter 3) addresses challenges of lack of temporal dimension information in generating data for missing sensing stations. Extensive experiments on real-world temperature, precipitation and air quality datasets have demonstrated the superiority of ESC-GAN in spatio-temporal imputation over state-of-the-art spatio-temporal imputation methods, image and

[3]https://github.com/xiyuanzh/PILOT
[4]https://github.com/xiyuanzh/ESC-GAN

video inpainting methods.

- STAug[5] (Chapter 4) presents a diverse and coherent augmentation method for time series forecasting tasks that combines information from both the frequency and time domains. We evaluate STAug on real-world time series datasets, and the method demonstrates state-of-the-art performance compared with existing augmentation methods.
- A survey on LLM for time series[6] (Chapter 5) presents a taxonomy on various means to transfer knowledge from large language models for time series analysis. We categorize existing methodologies into five groups (prompting, time series quantization, aligning, vision as bridge, tool integration), where each group corresponds to one processing stage in typical LLM-driven NLP pipelines. We also compile a comprehensive list of existing multimodal text and time series datasets.
- SHARE[7] (Chapter 6) presents an effective Human Activity Recognition (HAR) framework which captures knowledge across activities by uncovering information from label structures. We also propose three label augmentation methods, each targeting at a different level, to more effectively identify shared structures across activities. We evaluate SHARE on seven HAR benchmark datasets and observe the new state-of-the-art performance. We also conduct experiments under few-shot settings and label imbalance settings and observe even more significant performance improvement.
- UniMTS (Chapter 7) introduces a unified pre-training procedure for motion time series that successfully generalizes to various device locations, device orientations and activities. We design a contrastive learning framework to align motion time series with corresponding semantic meanings for activity generalization. For device location generalization, we propose to synthesize motion time series covering various body locations and model their spatio-temporal correlations using graph convolutional neural networks. We also design rotation-invariant augmentation to make the model agnostic to different device orientations. Our

[5]https://github.com/xiyuanzh/STAug
[6]https://github.com/xiyuanzh/awesome-llm-time-series
[7]https://github.com/xiyuanzh/SHARE

pre-trained model demonstrates state-of-the-art performance across 18 real-world motion time series benchmark datasets, notably with performance improvement of 342.3% in the zero-shot setting, 16.3% in the few-shot setting, and 9.2% in the full-shot setting, compared with the respective best-performing baselines.

Overall, this dissertation effectively addresses the challenges of noise, sparsity, scarcity and variability inherent in sensor time series, significantly improving the robustness of sensor time series analysis systems. The resulting framework has wide applications across sensing systems of various scales, from small-scale personal healthcare monitoring, smart home automation, to large-scale smart building control, energy management, climate modeling and beyond.

Chapter 2

Physical Context of Explicit Mathematical Models

In this chapter, we introduce how to leverage physical context of explicit mathematical models to achieve the first step in our robust sensor time series analysis framework: denoising sensor data. While the physical context, or the physics of monitored phenomenon can be used for a variety of tasks, we first focus on denoising as an concrete practical example of using inherent contextual information for time series analysis.

Sensors measuring real-life physical processes are ubiquitous in today's interconnected world. These sensors inherently bear noise that often adversely affects performance and reliability of the systems they support. Traditionally, denoising is a signal-processing operation based on mathematical transformations on time series. Classic filtering-based approaches introduce strong assumptions on the time or frequency characteristics of sensor data, while learning-based denoising approaches typically rely on using ground truth clean data to train a denoising model, which is often challenging or prohibitive to obtain for many real-world applications.

We observe that in many scenarios, the relationships between different sensor measurements (e.g., location and acceleration) are analytically described by laws of physics (e.g., second-order differential equation). By incorporating such physics constraints, we can guide the denoising process to improve even in the absence of ground truth data. In light of this, we design a physics-informed denoising model that leverages the inherent algebraic relationships between

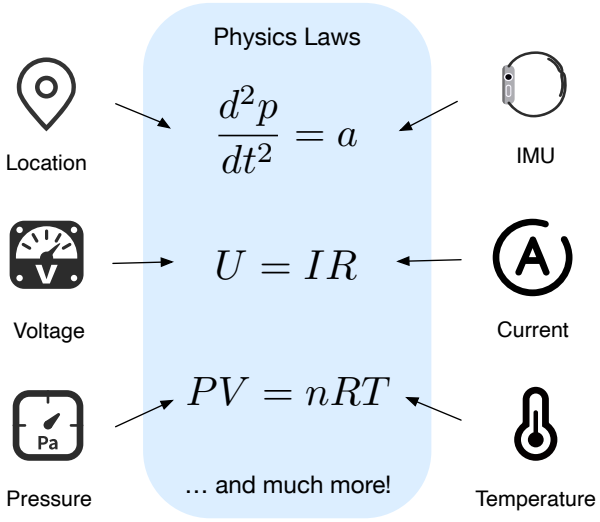


Figure 2.1. Real-life data captured by sensors are often governed by the laws of physics.

different measurements governed by the underlying physics. By obviating the need for ground truth clean data, our method offers a practical denoising solution for real-world applications. We conduct experiments in various domains, including inertial navigation, CO_2 monitoring, and HVAC control, and achieve state-of-the-art performance compared with existing denoising methods. Our method can denoise data in real time (4ms for a sequence of 1s) for low-cost noisy sensors and produces results that closely align with those from high-precision, high-cost alternatives, leading to more accurate, efficient and cost-effective sensor-based systems.

2.1 Introduction to Physics-Informed Sensor Data Denoising

Sensors measuring various real-life physical processes have permeated our daily lives. These sensors play a crucial role in acquiring a large amount of data in various applications such as environmental monitoring, healthcare, smart home and building, and transportation, enabling context inference, pattern recognition, and informed downstream decision-making. However, because of factors such as environmental interference, electrical fluctuations, and imprecision of the sensor itself, sensor data are naturally noisy. Such noise degrades data quality and adversely affects performance of downstream applications.

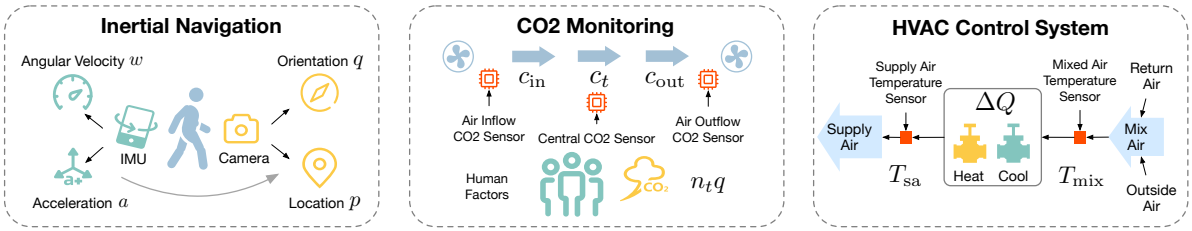


Figure 2.2. Three example applications of PILOT. Inertial navigation (left), CO_2 monitoring (middle), and HVAC control system (right).

To improve sensor data quality, there has been a line of research on noise reduction, from traditional filtering approaches that rely on prior knowledge of signal characteristics in time or frequency domain [57, 189], to more advanced machine learning methods [251, 310]. Existing machine learning based methods typically assume availability of ground truth clean data in order to train a denoising model in a supervised manner. However, given the inherently noisy nature of sensors, we often do not have access to ground truth *clean* data in real-world applications. In addition to supervised denoising methods, researchers have also developed self-supervised denoising methods, mostly in computer vision applications [128, 179]. These approaches often make assumptions for simplification (e.g., about noise distributions) that may not accurately reflect real-world sensor data with complex correlations. Moreover, they do not leverage the unique physical characteristics of sensor data distinct from typical image or text data.

In multiple applications, we observe that different measurement channels of sensor data collected from real-life physical processes are often correlated and characterized by equations rooted in well-understood underlying physics. We illustrate three example scenarios in Figure 2.1. In the first example, the relationship between location and acceleration is captured by the motion law, which equates the second-order derivative of location to acceleration. The second example connects the measurements by drawing upon Ohm's law – the ratio of voltage and current is a constant, resistance, of the circuit. In the third example, the ideal gas law establishes a link between pressure and temperature measurements. These three examples are merely a sample of the numerous physics principles that govern real-life sensing systems. We build upon the intuition that *such physics-based principles among measured channels can be used as constraints*

to improve sensor denoising techniques.

To this end, we propose a physics-informed denoising method, PILOT (Physics-Informed Learning for denOising Technique), by incorporating analytical model based on known physics as loss functions during training. Specifically, we introduce physics-derived equations as soft constraints as detailed in Section 2.4.1 that the denoised output from our model needs to satisfy. We illustrate the model architecture in Figure 2.4. We intentionally add noise to corrupt the original data collected from multiple sensors. Feeding this corrupted data as input, we train a denoising model to remove the added noise by minimizing a *reconstruction loss* between the model output and the original undistorted input. However, since the original sensor data are inherently noisy, a reconstruction loss alone is insufficient to remove such noise. Therefore, we introduce an additional *physics-based loss* to minimize physics misalignment of denoised output. For the example of inertial navigation, the corresponding physics equations include differential equations between position/orientation and acceleration/angular velocity. Therefore, in this example, the physics constraint would penalize discrepancy between the derivatives of location/orientation and the acceleration/angular velocity. Denoising processes for other applications follow similarly via a unified mathematical formulation that we will elaborate on.

To evaluate our approach, we conduct experiments on different real-world scenarios including inertial navigation, CO_2 monitoring, and HVAC (Heating, Ventilation, and Air Conditioning) control. For inertial navigation, PILOT generates denoised results that are most coherent to physics equations, leading to the best performance in downstream inertial navigation applications on the benchmark OxIOD dataset [36]. We collect data for the other two applications, detailed in Section 2.5.3 and Section 2.5.4. For CO_2 monitoring, we have two types of CO_2 sensors–one is low-cost and very noisy, and the other is more accurate but much more expensive. Denoised CO_2 measurements from the low-cost sensors by PILOT closely match those from the much more expensive CO_2 sensors. Consequently, our denoising technique offers significant cost savings. We observe similar advantage for HVAC control system, where our denoised temperature data collected from low-cost noisy sensors best match those collected from

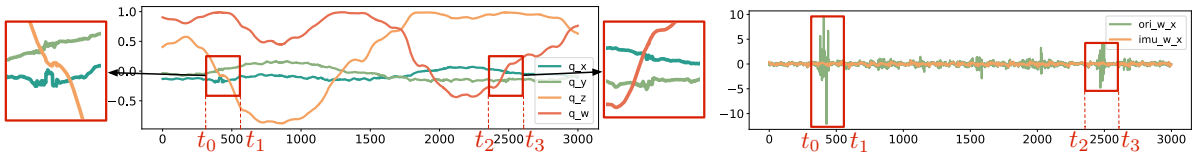
industry-grade sensors. In summary, this chapter makes the following contributions:

- We propose a novel denoising method, PILOT, trained under the guidance of physics constraints. To our best knowledge, this is the first *generic*, physics-informed denoising method that supports different sensing applications.
- PILOT offers a *practical* denoising solution for application scenarios where obtaining ground truth clean data or understanding the underlying noise distribution is challenging, better representing real-world sensing environments.
- We collect two sensor datasets on CO_2 monitoring and HVAC control, with pairs of noisy and accurate sensor data.
- Extensive experiments in three real-world applications demonstrate that PILOT produces results that are closest to the clean data and most coherent to the laws of physics, achieving state-of-the-art denoising performance. We also deploy the model on edge devices and show real-time denoising capability.

2.2 Related Work

General Physics-Informed Machine Learning. The concept of Physics Priors refers to our understanding of the inherent principles of the physical world, taking various representations such as differential equations, symmetry constraints, and common-sense knowledge, among others. An emerging field, Physics-Informed Machine Learning (PIML), aims to incorporate physics priors into machine learning models [91, 253, 116]. A line of research has been proposed for such integration, which can be categorized into different directions: incorporating physics priors as loss regularization [204, 208, 73, 118], imposing strict constraints into the model architectures [252, 232, 289] or hybrid approaches [76]. In general, physics-informed machine learning methods enhance efficacy, generalizability, and robustness of the models. However, most works in the PIML domain evaluate on simplified synthetic datasets.

Physics-Informed Machine Learning for Sensing Systems. Several works (such as



(a) Noisy Data **(b)** Two Angular Velocities

Figure 2.3. (a) Original orientations. (b) Comparison of measured (imu_w_x) and derived angular velocities from orientations (ori_w_x) in x-axis. X-axis: timestep, Y-axis: meter.

PIP [291], PACMAN [183], Reducio [30], and PhyAug [164]) employ physics in practical settings. However, most approaches are domain-specific [119, 43, 66, 93], and miss the consideration of physical relationships among different sensor measurement channels that can assist with the denoising process. Multimodal learning methods [107, 162, 261, 316] are also relevant topics as different modalities mutually compensate and offer enriched information, but existing multimodal methods mostly focus on fusing multiple modalities for downstream prediction tasks (e.g., human activity recognition) without addressing the denoising challenge. To address this gap, we present the first physics-informed machine learning model that incorporates these relationships into denoising sensor data, pushing the capability of what is currently achievable in Physics-Informed Machine Learning.

Denoising Methods. Sensor data are often noisy due to intrinsic characteristics such as thermal noise, operating point drifts, or environmental effects. Existing works have proposed methods to denoise sensor data or adapt the model to noisy measurements [67, 140]. The majority of these strategies, however, is custom-tailored to specific domains of application, such as COTS WiFi based motion sensing [330], image data processing under various weather conditions [41], speech recognition [154], and clinical data analysis [194]. Some researchers have also proposed more general denoising approaches through traditional estimation methods, or time-frequency domain analysis such as discrete wavelet transform [57] and empirical mode decomposition [189]. However, traditional estimation methods such as Kalman smoother rely heavily on the assumption of Gaussian noise and Markovian transition, but real-world physics models such as human motion typically have a higher-order temporal correlation. These methods

also rely on oversimplified assumptions that observed variables are conditionally independent over time. More recent deep learning approaches [310, 251] are supervised methods that require ground truth clean data, and they neglect the unique benefits of physics-rich real-life sensing systems. Self-supervised denoising methods have been recently proposed to train denoising model without access to ground truth clean data [243, 128, 103, 179, 280]. These methods are mainly designed for the computer vision domain and make assumptions about the noise.

Real-Life Sensing Systems. There exist numerous physics-rich real-life sensing applications. In this chapter, we focus on three applications: inertial navigation, CO_2 monitoring, and HVAC systems. Inertial navigation predicts location and orientation based on IMU data, which is prone to erroneous pose estimate rapidly over time [216]. Classic approaches incorporate external measurements such as GNSS and camera, to periodically correct the propagated pose value, at the cost of additional hardware and power. Recently, learning-aided approaches become popular [217, 158, 285, 35, 95, 155, 229, 151]. CO_2 monitoring monitors CO_2 level to ensure occupant comfort. Researchers have designed data collection [230] and generation [262] methods for CO_2 data. When modeling CO_2 data, existing works have developed data-driven, differential equation models [263]. Modeling CO_2 levels is beneficial for occupancy prediction and energy savings [10, 72]. HVAC Control System controls heating, ventilation and air conditioning based on temperature and airflow measurements [18, 184]. Apart from traditional model-based methods, recently deep reinforcement learning has also shown promising performance in modeling HVAC systems [124, 304, 281].

2.3 Preliminary

2.3.1 Problem Formulation

Clean Data. We denote the unobserved ground truth clean dataset as $\mathscr{D}_X = \{\mathbf{X}_i\}_{i=0}^N$ which consists of N samples. Each sample $\mathbf{X}_i \sim \mathscr{X}, \mathbf{X}_i \in \mathbb{R}^{c\times T}$, where $\mathscr{X}$ denotes the clean data space, T denotes the number of timesteps in the sensor data, and c denotes the number of

sensor measurement channels. Each sample $\mathbf{X}_i = [\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{ic}]^T$, where $\mathbf{x}_{ij} \in \mathbb{R}^T$ represents the jth sensor measurement channel for the ith sample in the ground truth clean dataset.

Noisy Data. We denote our observed noisy sensor dataset as $\mathscr{D}_Y = \{\mathbf{Y}_i\}_{i=0}^N$, composed of N samples. Each sample $\mathbf{Y}_i \sim \mathscr{Y}, \mathbf{Y}_i \in \mathbb{R}^{c \times T}$, and $\mathbf{Y}_i = [\mathbf{y}_{i1}, \mathbf{y}_{i2}, \cdots, \mathbf{y}_{ic}]^T$, where $\mathscr{Y}$ denotes the noisy data space, $\mathbf{y}_{ij} \in \mathbb{R}^T$ represents the jth sensor measurement channel for the ith sample in the noisy dataset.

Noise. Each noisy sample $\mathbf{Y}_i$ is corrupted from clean sample $\mathbf{X}_i$ by noise $\varepsilon_i \sim \mathscr{E}, \varepsilon_i \in \mathbb{R}^{c \times T}$, where $\mathscr{E}$ denotes the noise space. Specifically, $\mathbf{Y}_i = \mathbf{X}_i + \varepsilon_i, i = 1, 2, \cdots, N$.

Physics Equations. Equation $g(\cdot)$ describes the relationships between different sensor measurement channels in the ground truth clean data, i.e., $g(\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{ic}) = 0, i = 1, 2, \cdots, N$.

Denoising Model. The denoising model is parameterized by θ and maps noisy data to clean data by mapping $f(\cdot; \theta) : \mathscr{Y} \to \mathscr{X} \subset \mathbb{R}^{c \times T}$. We incorporate the physics equation $g(\cdot)$ as a constraint to optimize $f(\cdot; \theta)$ during training.

2.3.2 Physics Principles in Sensing Systems

In this section, we introduce the underlying physics principles of three example sensing applications: inertial navigation, CO_2 monitoring, and HVAC control. PILOT leverages these physics equations as constraints during training to better guide the denoising process. We evaluate the denoising performance of these three sensing applications in Section 2.5. The major difference across different applications is the specific form of the physics constraint. Consequently, to adapt our model to new applications, the only requirement is to formulate the equation representing the physics relationship between different sensors in the new application. Thereafter, we can seamlessly incorporate the constraint, treat it as regularization, and maintain a uniform approach across varying applications.

We select these three use cases to demonstrate the different capabilities of the framework to model various physics relationships. The use cases cover different sensor types, domains, and scales. In particular, we choose the three applications to validate that the proposed method

is robust across different scales — from single-human interactions to building air circulation models across many zones and users. Further, we seek to cover as many types of sensors (IMU, Camera, CO_2, airflow, temperature, etc.) as possible in these three experiments to demonstrate the potential to generalize to various types of sensors and noise levels. Our method can incorporate different physical laws with varying degrees of physics model complexity. We also pick the applications where physics models are informative but imperfect due to various modeling challenges. The selected applications are known for their dependencies on additional measurement modalities to compensate for the imperfect physics model, such as the visual information for navigation and IMU physics. This motivates the necessity for both learning component based models as well as physics-based models.

Application Scenario I: Inertial Navigation

Inertial navigation aims to estimate a moving device's position and orientation based on IMU sensor measurements, shown in Figure 2.2 (left). The inertial navigation model takes IMU measurements as input, and predicts locations and orientations. The ground truth location and orientation data are captured by cameras. However, both IMU and camera data carry noise due to factors like jittering or occlusion, posing challenges to training accurate inertial navigation models. If we denote location, orientation (in quaternion form), angular velocity, and acceleration as p, q, w, a respectively, then their relationships can be mathematically expressed by equations $g_1(\cdot), g_2(\cdot)$:

$$g_1(a, p, q) = a - R_q^T(\frac{d^2p}{dt^2} - g_0), \tag{2.1}$$

$$g_2(w, q) = \frac{dq}{dt} - \frac{1}{2}q \otimes w, \tag{2.2}$$

where $R_q, g_0, \otimes$ represent rotation matrix, gravity constant and quaternion multiplication. As a motivating example, we randomly select 30-second sensor data from the OxIOD dataset [36] and visualize the orientation measurements in Figure 2.3a. We note two major noise-heavy regions around the 500^{th} timestep and the 2500^{th} timestep (indicated by boxed regions). Applying

Equation 2.2, we derive the angular velocity from the orientation data, and compare it with the angular velocity directly measured by IMU gyroscope in Figure 2.3b. The misalignment between these two angular velocities appears around the same timesteps of noisy regions in the original orientation data. This highlights the role of known physics to guide the learning process of denoising models and its use in our model.

Application Scenario II: CO_2 Monitoring

CO_2 monitoring is of wide interest in smart building applications. An accurate zone CO_2 reading can enable transmission risk analysis [213], occupancy estimation [53], air quality monitoring [187] for energy-saving or residence safety purposes [72]. Figure 2.2 (middle) depicts one example placement of three CO_2 sensors proximate to the air intake, outtake, and the center of a standard office room. Assume $c_t, c_0, c_{\text{in}}^t, c_{\text{out}}^t$ represent the current average CO_2 concentration in the room, initial average CO_2 concentration in the room, CO_2 concentration for the input airflow, CO_2 concentration for the output airflow, respectively. Furthermore, let v, V, s, n_t represent the airflow velocity, room volume, CO_2 emission rate per individual, and the number of occupants in the room. Note that all these extra parameters are known information of the environment. They may change over time, but their values at each timestep are known. Then, we can write their relationship in the following equation $g(\cdot)$:

$$\begin{aligned} g(c_t, c_{\text{in}}^t, c_{\text{out}}^t) = c_t V - (c_0 V + \sum_t (c_{\text{in}}^t v \Delta t) \\ + \sum_t (n_t s \Delta t) - \sum_t (c_{\text{out}}^t v \Delta t)) \end{aligned} \tag{2.3}$$

Unfortunately, perfect CO_2 monitoring would require deploying multiple high-resolution and low-noise lab-level CO_2 sensors, which are prohibitively expensive for a typical commercial building consisting of hundreds of zones. By contrast, low-cost CO_2 sensors are readily available on the market with more noise and less accuracy (e.g., CCS881 as we will detail in Sec 2.5.3). We aim to incorporate the aforementioned physics equation to denoise CO_2 data collected from these

low-cost sensors, such that data quality after denoising can match the expensive counterparts.

Application Scenario III: Air Handling in HVAC Control System

HVAC control systems manage critical parameters such as temperature, humidity, and air quality of commercial or residential buildings to ensure thermal comfort and optimal air quality while maintaining energy efficiency. Air Handling Units (AHUs), the core of an HVAC control system, take in outside air and return air from zones in the building, recondition it via heating or cooling, de-humidification, and then supply it to spaces. Downstream equipment such as Variable Air Volumes (VAVs) may further condition the air based on factors including occupancy, time of day, and specific temperature desired of the space. Figure 2.2 illustrates an AHU in an example HVAC control system with both heating and cooling functionality. The AHU mixes a certain ratio of return air from the conditioned spaces and the fresh outside air, and reconditions the mixed air through cooling and heating coils and supply air to various spaces. We denote the enthalpy difference across the heating and cooling coils by ΔQ. The air's mass undergoing reconditioning is denoted as m, its specific heat capacity as c_h, the temperature of the supply air and mix air as T_{sa} and T_{mix}, respectively. We then derive the following heat transfer equation under the assumption of no additional heat source aside from the cooling and heating coils:

$$g(\Delta Q, m, c_h, T_{\text{sa}}, T_{\text{mix}}) = \Delta Q - mc_h(T_{\text{sa}} - T_{\text{mix}}) \tag{2.4}$$

Note that air mass m can be acquired through the airflow rate (measured in Cubic Feet per Minute, or CFM) going across the AHU. Furthermore, the air's specific heat capacity, c_h, is both humidity and temperature dependent.

In practice, both supply air temperature and mix air temperature measurements are subject to noise, which impedes accurate control in the HVAC system. Therefore, we incorporate the above equation to denoise the two temperature measurements, enhancing the accuracy and reliability of the HVAC control system.

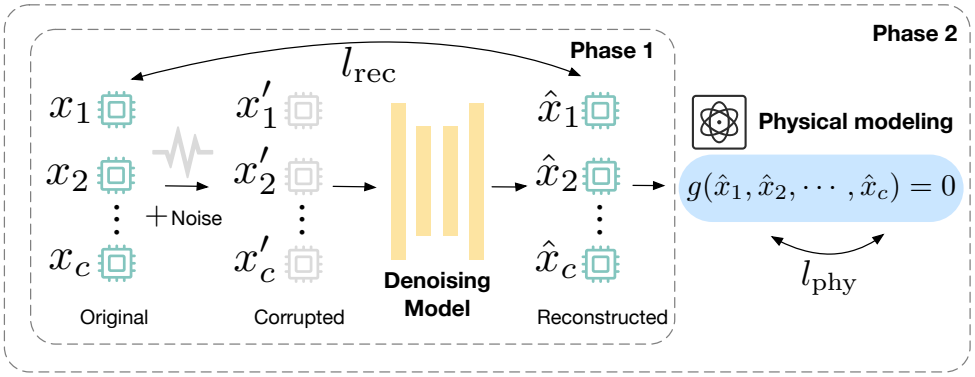


Figure 2.4. Physics-Informed denoising model framework. During training, we manually add noise to the original sensor data, and minimize a reconstruction loss to remove such noise. We also integrate a physics-based loss to ensure that reconstructed data obey the law of physics.

2.4 Methodology

2.4.1 Model Framework

Noise Injection. We first follow the setting of typical supervised denoising methods to manually inject noise and train the denoising model to remove the injected noise. Given the unknown nature of real-world noise distributions, we first randomly sample noise from pre-defined distributions as simulations. This process is insufficient to capture the complex noise distributions, so we augment the model training with physics modeling process illustrated later. When sampling the noise, we use Gaussian distribution with zero mean and variance of σ^2 as an example and compare with other noise distributions in Section 2.5.6 (Sensitivity Analysis). Mathematically, for each sample $\mathbf{Y}_i$ in our noisy dataset, we further sample Gaussian noise $\mathbf{n}_i$ and add it upon $\mathbf{Y}_i$ to obtain data $\mathbf{Z}_i$ with a higher degree of noise, i.e., $\mathbf{Z}_i = \mathbf{Y}_i + \mathbf{n}_i, \mathbf{n}_i \sim \mathcal{N}(0, \sigma^2)$. Note that our physics modeling component does not pose any constraints or assumptions on the noise type. The inclusion of Gaussian additive noise is purely for simulation purposes to facilitate the training of the denoising autoencoder.

Naive Denoising Model. The naive denoising model performs optimization without modeling physical relationships. The denoising model takes data with manual noise injected $\mathbf{Z}_i$ as input, and the training objective is to output data with such manual noise removed, by

minimizing the reconstruction loss, i.e.,

$$\text{argmin}_\theta \mathbb{E}_{\mathbf{Y}_i \sim \mathscr{Y}, \mathbf{n}_i \sim \mathscr{N}} \ell_{\text{rec}}(f(\mathbf{Y}_i + \mathbf{n}_i; \theta), \mathbf{Y}_i), \tag{2.5}$$

$$\ell_{\text{rec}}(f(\mathbf{Y}_i + \mathbf{n}_i; \theta), \mathbf{Y}_i) = ||f(\mathbf{Y}_i + \mathbf{n}_i; \theta) - \mathbf{Y}_i||_2^2. \tag{2.6}$$

The naive denoising model learns how to remove noise generated from our pre-defined noise distribution, e.g., Gaussian distribution. This is effective if the real noise follows the same noise distribution. However, in practice, the noise distribution is often much more complex and cannot be simply approximated by predefined patterns. As we will theoretically show in Section 2.4.2, the naive denoising method becomes suboptimal when the real noise distribution deviates from our pre-defined noise distributions. Therefore, we incorporate physics constraint to enhance denoising performance, enabling the model to adapt to more complex, real-world noise distributions.

Physics-Informed Denoising Model. Apart from the reconstruction loss in the naive denoising model, we incorporate another *physics-based loss* to overcome the limitation of the naive model. Recall in Section 2.3.1 (Problem Formulation) we mentioned that ground truth clean data satisfy physics constraint $g(\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{ic}) = 0, i = 1, 2, \cdots, N$. In an ideal scenario, if our denoising model can perfectly eliminate all noise, then the resultant output of this optimal denoising model $f(\cdot; \theta^*)$ will also adhere to the physics constraint:

$$g(f(\mathbf{z}_{i1}; \theta^*), f(\mathbf{z}_{i2}; \theta^*), \cdots, f(\mathbf{z}_{ic}; \theta^*)) = 0, i = 1, 2, \cdots, N, \tag{2.7}$$

which can be simplified as $g(f(\mathbf{Z}_i; \theta^*)) = 0$ or $g(f(\mathbf{Y}_i + \mathbf{n}_i; \theta^*)) = 0$. Therefore, we can exploit the physics constraints intrinsic to an optimal denoising model and formulate the constrained

optimization problem as

$$\begin{aligned} &\text{argmin}_{\theta}\mathbb{E}_{\mathbf{Y}_i\sim\mathscr{Y},\mathbf{n}_i\sim\mathscr{N}}\ell_{\text{rec}}(f(\mathbf{Y}_i+\mathbf{n}_i;\theta),\mathbf{Y}_i), \\ &\text{s.t. } g(f(\mathbf{Y}_i+\mathbf{n}_i;\theta))=0. \end{aligned} \tag{2.8}$$

Enforcing the physics equation as strict constraints makes the model difficult to optimize. Instead, we include the constraint as a *soft regularization term in the loss function*. Specifically, the loss function for the physics-informed denoising model is a combination of reconstruction loss and physics-based loss:

$$\text{argmin}_{\theta}\mathbb{E}_{\mathbf{Y}_i\sim\mathscr{Y},\mathbf{n}_i\sim\mathscr{N}}\ell(f(\mathbf{Y}_i+\mathbf{n}_i;\theta),\mathbf{Y}_i), \tag{2.9}$$

$$\ell=\ell_{\text{rec}}+\lambda\ell_{\text{phy}}, \tag{2.10}$$

$$\ell_{\text{phy}}(f(\mathbf{Y}_i+\mathbf{n}_i;\theta))=||g(f(\mathbf{Y}_i+\mathbf{n}_i;\theta))||_2^2. \tag{2.11}$$

The physics-based loss offers additional prior knowledge about the ground truth data and better guides the learning process of the denoising model, especially when we don't have access to the ground truth distributions. We find in the experiment section that the physics constraint works for both first-order terms and higher-order terms. Here, λ is the ratio that balances the two losses. We find that an adaptive balancing strategy that keeps these two losses always at the same orders of magnitude during training is empirically more effective than setting a fixed ratio, as we will illustrate in Section 2.5.6.

PILOT Framework. We design a two-phase training framework for the final denoising model. Specifically, we first optimize the model only with reconstruction loss as a pre-training phase. The two-stage training ensures that training is not biased by the physics loss at the beginning, as the initial exclusive focus on optimizing the physics loss can lead to overly simplistic and trivial solutions, such as all-zero predictions. Two-stage training mitigates this by serving as a warm-up period and providing better denoising model initialization. By learning to

reconstruct the data, the model better adapts to the underlying data distribution. In the second phase, we combine both reconstruction loss and physics-based loss to optimize the model. The physics-based loss corrects and guides the model for enhanced denoising performance. To choose the denoising model backbone, we keep both efficacy and efficiency in mind, ensuring that denoising models can run on edge devices with limited compute. Therefore, we use one-dimensional Convolutional Neural Networks (CNN) as an example, as they are relatively lightweight with superior capability in extracting sensor features. We provide details about code implementation on Github[1].

2.4.2 Theoretical Analysis

In this section, built upon previous works in [331, 103], we analytically show that when the inherent noise of our model input has a non-zero mean, training with naive denoising model would be insufficient to recover the ground truth data distribution.

Corollary 1. *Let* $\mathbf{y}$ *be observed noisy data corrupted from clean data* $\mathbf{x}$ *by noise* ε *with non-zero mean* η*:* $\mathbf{y} = \mathbf{x} + \varepsilon, \varepsilon \sim \mathscr{E}$*. Let* $\mathbf{z}$ *be manually corrupted data from* $\mathbf{y}$ *by pre-defined noise* $\mathbf{n}$ *with zero mean:* $\mathbf{z} = \mathbf{y} + \mathbf{n}, \mathbf{n} \sim \mathscr{N}$*. Assume* $\mathbb{E}_{\mathbf{z}|\mathbf{x}}(\mathbf{z}) = \mathbb{E}_{\mathbf{y}|\mathbf{x}}(\mathbf{y}) = \mathbf{x} + \eta$*, and the variance of* $\mathbf{y}$ *is* $\sigma_{\mathbf{y}}^2$*. Then the following equation holds true:*

$$
\begin{aligned}
&\mathbf{E}_{\mathbf{x}\sim\mathscr{X},\varepsilon\sim\mathscr{E},\mathbf{n}\sim\mathscr{N}} ||f(\mathbf{x}+\varepsilon+\mathbf{n};\theta) - \mathbf{x}||_2^2 \\
&= \mathbf{E}_{\mathbf{x}\sim\mathscr{X},\varepsilon\sim\mathscr{E},\mathbf{n}\sim\mathscr{N}} ||f(\mathbf{x}+\varepsilon+\mathbf{n};\theta) - (\mathbf{x}+\varepsilon)||_2^2 \\
&- \sigma_{\mathbf{y}}^2 + 2\eta \mathbf{E}_{\mathbf{x}\sim\mathscr{X},\varepsilon\sim\mathscr{E},\mathbf{n}\sim\mathscr{N}} (f(\mathbf{x}+\varepsilon+\mathbf{n};\theta) - \mathbf{x}).
\end{aligned}
\tag{2.12}
$$

Since $\eta \neq 0$, optimizing $\mathbf{E}_{\mathbf{x}\sim\mathscr{X},\varepsilon\sim\mathscr{E},\mathbf{n}\sim\mathscr{N}} ||f(\mathbf{x}+\varepsilon+\mathbf{n};\theta) - (\mathbf{x}+\varepsilon)||_2^2$ is not equivalent to the ideal goal of optimizing against ground truth, i.e., $\mathbf{E}_{\mathbf{x}\sim\mathscr{X},\varepsilon\sim\mathscr{E},\mathbf{n}\sim\mathscr{N}} ||f(\mathbf{x}+\varepsilon+\mathbf{n};\theta) - \mathbf{x}||_2^2$. In summary, the corollary states that the naive denoising model's optimization target fails to

[1] https://github.com/xiyuanzh/PILOT

completely mitigate the inherent noises. Therefore, we incorporate physics-based loss to augment the denoising performance, particularly when underlying noise distributions are inaccessible.

2.5 Evaluation

2.5.1 Experimental Setup

We use a one-dimensional convolutional neural network with four layers as the backbone for our denoising model. The respective kernel sizes for each layer are set to $7, 5, 3, 3$. We use Adam optimizer with learning rate 0.0001 and batch size 16. For each dataset $\mathscr{D}_X = \{\mathbf{X}_i\}_{i=0}^{N}$, we first compute how well each sample aligns with the physics equation. Specifically, the alignment for the i-th sample is the L2 norm $a_i = ||g(\mathbf{X}_i)||_2^2$. We select samples with the top 50% smallest a_i as our training set. These samples align more coherently with physics equations and potentially carry less noise. The remaining samples are used as the test set. The applications we have studied are mostly stationary systems, so the model should be able to generalize to new test data in practice. We conduct the experiments in PYTORCH with NVIDIA RTX A6000 (with 48GB memory), AMD EPYC 7452 32-Core Processor, and Ubuntu 18.04.5 LTS. We also implement our method on edge device (Raspberry Pi 4) and evaluate its efficiency in Section 2.5.7. We tune the hyper-parameters of both PILOT and baselines to minimize loss on the training set, and then evaluate on the test set after hyper-parameter tuning.

We compare PILOT with both statistical methods and recent deep learning denoising methods as follows (including two self-supervised methods DIP and N2N):

- **Gaussian Filter** employs a Gaussian filter with standard deviation σ for the kernel to smooth the sensor data.
- **DWT** [57] decomposes the signal using Discrete Wavelet Transform (DWT) and chooses the most energetic coefficients to reconstruct the denoised signal.
- **DnCNN** [310] is a convolutional neural network based denoising model that exploits residual learning and batch normalization to boost the denoising performance.

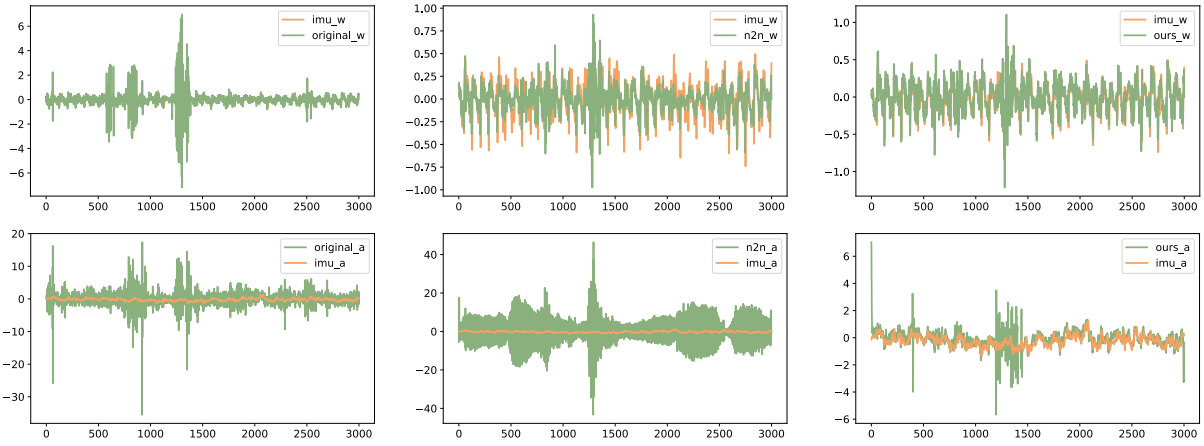
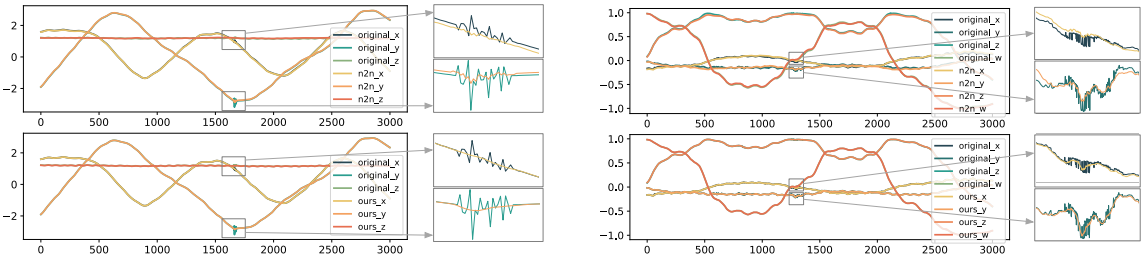


Figure 2.5. Top: Angular velocity from IMU and from first-order derivative calculation of orientation based on original data. Bottom: Acceleration from IMU and from second-order derivative calculation of location based on original data. X-axis: timestep, Y-axis: meter.



(a) Location (Top: N2N, Bottom: PILOT)

(b) Orientation (Top: N2N, Bottom: PILOT)

Figure 2.6. We compare PILOT with the best performing baseline N2N for qualitative denoising performance of (a) location and (b) orientation. We also zoom in for regions with higher noise levels for better illustration. X-axis: timestep, Y-axis: meter.

- **TSTNN** [251] is a Transformer-based model for end-to-end speech denoising. It is composed of a feature-mapping encoder, a two-stage Transformer model to extract local and global information, a masking module and a decoder to reconstruct denoised speech.
- **Deep Image Prior (DIP)** [243] exploits the structure of the neural network as priors for denoising data. The method fits the model to the noisy data with early exit, so the network captures dominant patterns while ignoring the random noise.
- **Neighbor2Neighbor (N2N)** [103] applies random neighbor sub-sampler on noisy data to generate input and target for the network. It also proposes a regularizer as additional loss for performance augmentation.

Table 2.1. Physics alignment results for OxIOD dataset. We bold the best and underline the second best. PILOT aligns the best with both acceleration and angular velocity.

Model	Acceleration (m/s^2)		Angular Velocity (rad/s)	
Metrics	MSE	MAE	MSE	MAE
Original	762.6	3.7862	2.6219	0.2376
Gaussian	363.2	3.2295	1.6277	0.2161
DWT	854.9	5.4534	2.6034	0.2701
DnCNN	312.5	8.3830	0.3470	0.1896
TSTNN	3272.0	30.513	0.4184	0.4836
DIP	2153.6	33.938	0.3788	0.4013
N2N	118.7	4.5749	0.3565	0.1756
PILOT	**1.8695**	**0.6372**	**0.0380**	**0.0690**

We evaluate the performance from three perspectives. First of all, if we have collected data of higher quality using other sources (e.g., more expensive and accurate sensors), we would regard these higher-quality data as approximate ground truth and compute Mean Square Error (MSE) and Mean Absolute Error (MAE) between the denoised data $\hat{\mathbf{X}}$ and the higher-quality data $\mathbf{X}$ as the first metric, noted as "Reconstruction Performance". Specifically, Reconstruction MSE $= ||\hat{\mathbf{X}} - \mathbf{X}||_2^2$, and Reconstruction MAE $= ||\hat{\mathbf{X}} - \mathbf{X}||_1$. The second metric investigates alignment of denoised data with the governing physics equations, noted as "Physics Alignment". Specifically, Physics MSE $= ||g(\hat{\mathbf{X}})||_2^2$, and Physics MAE $= ||g(\hat{\mathbf{X}})||_1$. Lastly, if the denoised data are further applied in downstream applications (e.g., inertial navigation system), we also evaluate "Downstream Performance" based on the metrics of interest for the corresponding downstream task. Measuring downstream performance helps us understand the full impact of denoising on real-world applications.

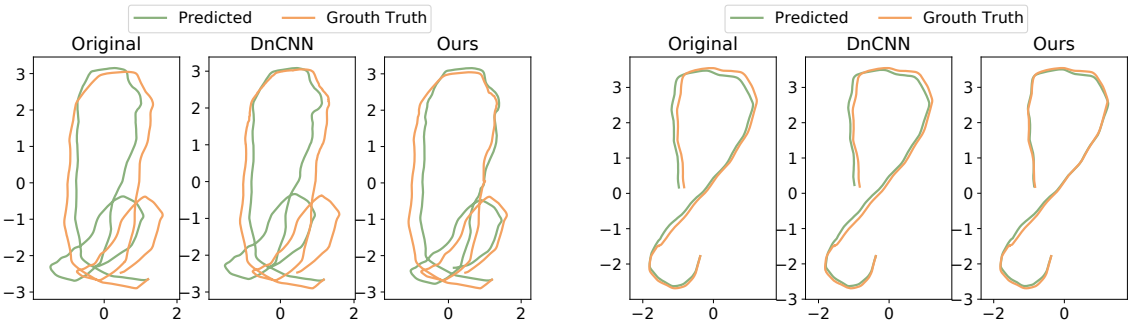
2.5.2 Application I: Inertial Navigation

Dataset

We use OxIOD [36] dataset for inertial navigation experiments. OxIOD collects accelerometer and gyroscope data (100 Hz) mostly by IMUs (InvenSense ICM20600) in iPhone 7 plus. A Vicon motion capture system (10 Bonita B10 cameras) is used to record the locations and

Table 2.2. Inertial navigation performance on OxIOD dataset with two inertial navigation models (IONet and RoNIN). We bold the best and underline the second best.

Model	IONet [35]						RoNIN [95]					
Metrics	vx (m/s)	vy (m/s)	vz (m/s)	mean v (m/s)	ATE (m)	RTE (m)	vx (m/s)	vy (m/s)	vz (m/s)	mean v (m/s)	ATE (m)	RTE (m)
Original	0.0207	0.0642	0.0093	0.0314	0.3076	0.8194	0.0180	0.0621	0.0090	0.0297	0.2472	0.6337
Gaussian	0.0249	0.0496	0.0145	0.0297	0.6111	1.8727	0.0242	0.0498	0.0147	0.0296	0.5988	1.8427
DWT	0.0266	0.0732	0.0094	0.0364	0.3142	0.8079	0.0243	0.0714	0.0091	0.0349	0.2665	0.7023
DnCNN	0.0200	0.0235	0.0144	0.0193	0.3001	0.7891	0.0177	0.0213	0.0139	0.0176	0.2476	0.6598
TSTNN	0.2857	0.3250	0.0935	0.2348	0.6496	1.6575	0.2865	0.3253	0.0938	0.2352	0.6256	1.5794
DIP	0.1971	0.2650	0.0105	0.1576	0.5759	1.5108	0.1926	0.2570	0.0101	0.1533	0.3989	1.0358
N2N	0.0246	0.0144	0.0183	0.0191	0.3151	0.8317	0.0224	0.0122	0.0182	0.0176	0.2605	0.6956
PILOT	**0.0102**	**0.0095**	**0.0031**	**0.0076**	**0.2998**	**0.7875**	**0.0081**	**0.0078**	**0.0017**	**0.0059**	**0.2413**	**0.6309**



(a) Case 1 **(b)** Case 2

Figure 2.7. Downstream inertial navigation results on OxIOD dataset with RoNIN model. RoNIN trained on denoised data of PILOT reconstructs trajectories closet to the ground truth trajectories. X-axis: meter, Y-axis: meter.

orientations. The total walking distance and recording time of the dataset are 42.5 km and 14.72 h. The data collection protocol is designed to simulate natural pedestrian movement within an indoor environment equipped with motion capture system. When collecting the data, a pedestrian walks naturally inside a room with motion capture system, carrying the phone in hand, in the pocket, in the handbag, or on the trolley.

Physics Alignment

We compare the physics alignment performance in Table 2.1. Specifically for this application, we compute the misalignment for acceleration and angular velocity. Adapting our experimental approach for the specificities of inertial navigation, we only denoise the location and

orientation data, and leave IMU data in its original form. This is because the downstream inertial navigation model's training requires integration over the IMU, thereby inherently smoothing and denoising the IMU data. Since our model only denoises the location and orientation data, we model the physics relationships between the denoised location, orientation, and the original IMU. More specifically, acceleration MSE in Table 2.1 is computed as the discrepancy between accelerometer data from IMU (a) and the second-order derivative of the denoised location subtracted by gravity constant and multiplied by rotation matrix ($R_q^T(d^2p/dt^2 - g_0)$), as shown in Equation 2.1. Similarly, angular velocity MSE is computed as the difference between gyroscope data from IMU multiplied by orientation ($q \otimes w/2$) and the first-order derivative of denoised orientation (dq/dt), as in Equation 2.2.

From Table 2.1, we can see that while most denoising methods reduce the physics MSE compared with original data without denoising, *PILOT significantly reduces the physics MSE by two orders of magnitude*. Accompanied by these quantitative results, we also offer qualitative illustrations in Figure 2.5. We present a visual comparison of the direct IMU measurements and derived accelerations and angular velocities. When comparing PILOT with original and denoised data from the best-performing baseline, we observe that PILOT-denoised data provide the best physics alignment for both angular velocity and acceleration.

Reconstruction Performance

For the inertial navigation application, we don't have access to ground truth location and orientation data. Therefore, we qualitatively compare the reconstruction performance of PILOT and the best-performing baseline in Figure 2.6. We compare the reconstruction performance for both location and orientation, with detailed zoom-ins on regions exhibiting higher noise levels for more clear illustration. We observe that PILOT effectively reduces the noise in the original data, while simultaneously reconstructing results that show greater alignment and coherence with the original locations and orientations.

Downstream Performance

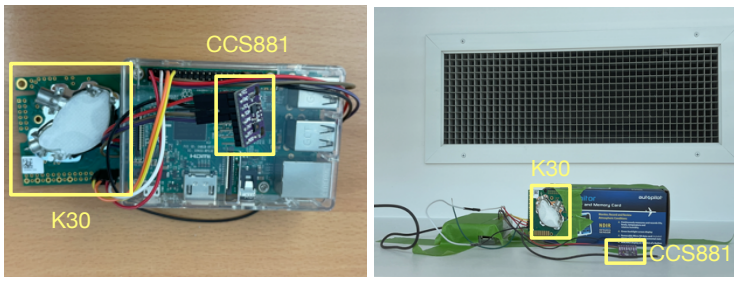
We also use the denoised data to train downstream inertial navigation model, and compare PILOT and baselines in terms of downstream task performance. We use IMU data, denoised location and orientation data as training pairs for the following inertial navigation models:

- **IONet** [35]: IONet uses deep recurrent neural networks to learn location transforms in polar coordinates from raw IMU data, and construct inertial odometry.
- **RoNIN** [95]: RoNIN regresses velocity given IMU with coordinate frame normalization and robust velocity losses.

We measure the downstream performance by the following three metrics, following previous works [95].

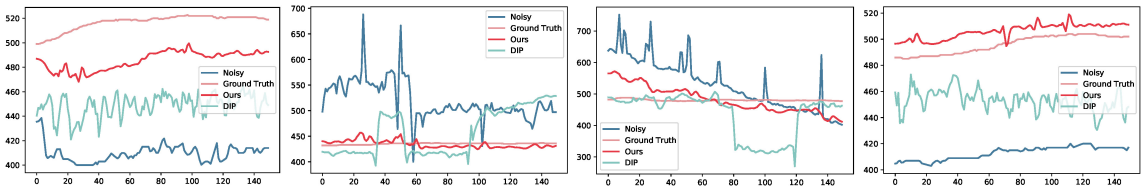
- **Velocity**: We compute MSE of predicted velocity (x, y, and z-axis, along with the mean velocity of all three axes).
- **Absolute Translation Error (ATE)**: ATE is defined as the Root Mean Squared Error (RMSE) between the entire estimated and ground truth location trajectory.
- **Relative Translation Error (RTE)**: RTE is defined as the Root Mean Squared Error (RMSE) over a fixed time interval. We choose 1 minute in our experiment as in previous works [95].

We compare PILOT with baselines on downstream inertial navigation task using IONet and RoNIN (Table 2.2) as the downstream models. *We can observe that PILOT best predicts*



(a) Deployment on desk **(b)** Deployment by vent

Figure 2.8. We connected a pair of two different CO_2 sensors (K30 and CCS 881 respectively) on a Rasberry Pi. We deploy (a) one pair on the desk in the center of the office and (b) the other pair close to the exhaust vent to monitor CO_2 level.



(a) Example 1 **(b)** Example 2 **(c)** Example 3 **(d)** Example 4

Figure 2.9. We compare the CO_2 denoising results of PILOT and the best performing baseline DIP. PILOT denoises CCS881 results closet to the ground truth (K30 measurements). X-axis: timestep, Y-axis: ppm.

the velocity and locations, regardless of the specific inertial navigation model. In Figure 2.7, we also qualitatively compare PILOT with the best-performing baseline using RoNIN as the inertial navigation model. We can see PILOT's denoised data offer RoNIN the best trajectory reconstruction relative to both the original data and that from the best-performing baseline.

2.5.3 Application II: CO_2 Monitoring

Dataset Collection

In this experiment, we deployed two pairs of CO_2 sensors (K30 and CCS811) in a typical graduate student office environment. K30 is a highly accurate Non-Dispersive Infra-Red (NDIR) CO_2 sensor manufactured by SenseAir which costs $100 per unit. It features an accuracy of $\pm$30 ppm or $\pm$3% of measurements and a high repeatability of $\pm$20 ppm or $\pm$1% of measurements [245]. The NDIR technology [192] utilizes the unique property of CO_2 molecules — their significant absorption of infrared (IR) light in the vicinity of 4.2 μm wavelength. When a gas sample is illuminated with light of this particular wavelength, the concentration of CO_2 can be deduced by examining the fraction of light absorbed. We regard K30 readings as the ground truth of CO_2 concentration.

In comparison, CCS811 [4] is a substantially cheaper Metal-Oxide (MOX) gas sensor, priced at less than $10 per unit, which computes equivalent CO_2 readings based on hydrogen gas readings. A MOX gas sensor works by measuring and analyzing changes in the conductivity of the gas-sensitive MOX semiconductor layer(s) at various gas exposure [54]. It has no verifiable

Table 2.3. Reconstruction and physics alignment results on CO_2 dataset. We bold the best and underline the second best.

Model	Recons (1×10^6 ppm)		Physics (1×10^6 ppm)	
Metrics	MSE	MAE	MSE	MAE
Original	1.5654	0.0020	0.1082	0.1908
Gaussian	0.6265	0.0016	0.0278	0.1019
DWT	1.5076	0.0020	0.1045	0.1846
DnCNN	1.5381	0.0020	0.1064	0.1897
TSTNN	0.0956	0.0007	0.0027	0.0498
DIP	<u>0.0841</u>	<u>0.0006</u>	<u>0.0018</u>	<u>0.0308</u>
N2N	0.4396	0.0018	0.0085	0.0789
PILOT	**0.0371**	**0.0004**	**0.0012**	**0.0200**

accuracy or repeatability guarantee and is usually not recommended for laboratory use.

For each pair of these two sensors, we connect both of them to a single Raspberry Pi board, which collects data through UART protocol and I2C interface from K30 and CCS811 respectively, at an interval of 4 seconds. The collected readings are transmitted to a remote InfluxDB instance for storage and further analysis. We place these two pairs of CO_2 sensors as follows. One pair is adjacent to the HVAC exhaust vent and the other pair is at the center of the room, to capture the relevant variables in Equation 2.3 as illustrated in Figure 2.8. We set inflow CO_2 concentration in Equation 2.3 to 440 ppm based on empirical observation.

Reconstruction and Physics Performance

For the application of CO_2 monitoring, data collected from high-precision K30 sensors serve as the benchmark for calculating the reconstruction performance. We also evaluate the physics alignment by how well our denoised data match Equation 2.3. As shown in Table 2.3, *PILOT outperforms all baselines, showcasing the lowest reconstruction and physics alignment errors*. We also qualitatively compare PILOT with the best-performing baseline in Figure 2.9. We can observe that the denoised data by PILOT align the closest to the ground truth (K30 data) compared with original noisy data as well denoised data from the best-performing baseline.

(a) ΔQ **(b)** ΔT

Figure 2.10. Sensor deployments in HVAC systems. (a) We measure heat difference ΔQ through the water temperature difference. (b) We measure the air temperature differences with a copper thread averaging sensor.

2.5.4 Application III: Air Handling in HVAC Control System

Dataset Collection

We collect data points of relevant variables in Equation 2.4 from an AHU serving a lecture hall on campus. We choose this particular AHU based on the following two criteria. Firstly, this AHU is equipped with both heating and cooling thermal submeters (Figure 2.10a), ensuring the availability of ΔQ in Equation 2.4. Secondly, none of the zones served by this AHU has any reheat unit. In theory, this guarantees that our assumption holds true that heating and cooling coils are the only heat sources. We collect the data through the campus building management system (BMS) for 18 consecutive days, with readings taken at 1-second intervals. We plan to collect more long-term data as a future work.

The mixed air temperature and supply air temperature are measured by copper-thread averaging duct sensors (costing $150 for each) within the respective supply and exhaust ducts of this AHU, as shown in Figure 2.10b. These sensors give the average temperature based on the fact that the electrical resistance of copper changes in a predictable way with temperature changes. However, this method is prone to inaccuracy and lacks repeatability due to challenges

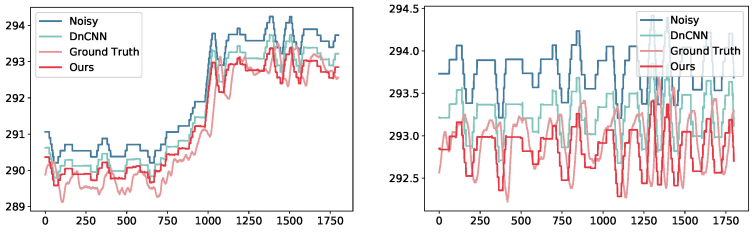
Table 2.4. Reconstruction and physics alignment on HVAC dataset. We bold the best and underline the second best. PILOT best denoises HVAC data in terms of both reconstruction accuracy as well as physics alignment.

Model	Reconstruction (K)		Physics (K)	
Metrics	MSE	MAE	MSE	MAE
Original	0.9479	0.8841	50.302	6.7184
Gaussian	0.8687	0.8782	49.528	6.7003
DWT	0.8553	0.8689	48.782	6.6471
DnCNN	<u>0.3284</u>	<u>0.4786</u>	50.380	6.7241
TSTNN	2.2980	1.1980	32.012	3.7260
DIP	3.7336	1.5098	<u>29.747</u>	3.7410
N2N	1.1830	0.9522	31.748	<u>3.6990</u>
PILOT	**0.1994**	**0.3454**	**14.081**	**3.1600**

associated with maintenance and cleaning, as well as the uneven distribution of the thread within the duct. A feasible yet costly alternative solution is to use multiple single-unit temperature sensors evenly distributed throughout the duct. These sensors are more expensive and harder to deploy. For evaluation, we carry this setup by deploying two dual-probe high-accuracy, industry-level temperature sensors in the supply duct, each costing $200, and take the average of readings from each probe to get a highly reliable ground truth temperature measurement.

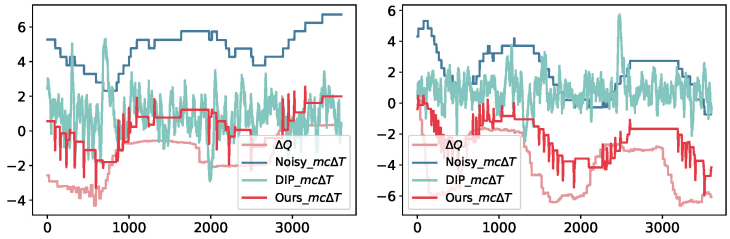
Reconstruction and Physics Performance

For the HVAC control system application, we measure the denoising performance by both reconstruction performance and physics alignment. For reconstruction performance, we deploy high-precision temperature sensors and utilize the collected data as ground truth temperature to assess the reconstruction performance of different denoising methods. For physics alignment, we compute the MSE and MAE between ΔQ and $mc\Delta T$ in Equation 2.4. As shown in Table 2.4, *PILOT achieves state-of-the-art performance and shows the lowest errors for both reconstruction performance and physics alignment*. We also provide qualitative comparisons to supplement our evaluation. Figure 2.11 showcases two example reconstructions of supply air temperature data. Denoised data from PILOT exhibits the best alignment with the ground truth data compared with



(a) Example 1 **(b)** Example 2

Figure 2.11. Two example reconstructions for supply air temperature. PILOT's outputs best align with the ground truth compared with original noisy measurements and the best- performing baseline. X-axis: timestep, Y-axis: K.



(a) Example 1 **(b)** Example 2

Figure 2.12. Two example alignment between ΔQ and $mc\Delta T$. PILOT achieves the closet alignment compared with original data and the best baseline. X-axis: timestep, Y-axis: K.

original data, as well as the best-performing baseline DnCNN. In Figure 2.12, we also present two examples of physics alignment. We see that denoised data of PILOT best align with the physics equation compared with original noisy data and the best-performing baseline DIP.

2.5.5 Ablation Study

To study the effects of different components of our model, we separately remove the physics-based loss, reconstruction loss, and pre-training phase, and evaluate PILOT in Table 2.5. For Inertial Navigation System (INS), we compare the physics alignment of both acceleration (MSE_a, MAE_a) and angular velocity (MSE_w, MAE_w). For CO_2 monitoring and HVAC control, we compare both reconstruction (MSE_rec, MAE_rec) and physics alignment (MSE_phy, MAE_phy).

We note that by removing the physics-based loss, PILOT degenerates to a naive denoising model, which is not sufficient to capture the complex noise distribution in real-world sensor data. Therefore, compared with PILOT, MSE and MAE of both reconstruction and physics

Table 2.5. Ablation Study of PILOT. We bold the best and underline the second best. Removing either physics-based loss, reconstruction loss or pre-training phase would negatively affect the performance, demonstrating the effectiveness of all three components.

Task	Metrics	w/o l_{phy}	w/o l_{rec}	w/o pre-train	PILOT
INS	MSE_{a}	316.5	259.1	20.70	**1.8695**
	MAE_{a}	8.342	8.759	2.2930	**0.6372**
	MSE_{w}	0.3579	0.2845	0.1380	**0.0380**
	MAE_{w}	0.1899	0.3352	0.1291	**0.0690**
CO_2	MSE_{rec}	0.4641	0.0568	0.0724	**0.0371**
	MAE_{rec}	0.0139	0.0051	0.0074	**0.0047**
	MSE_{phy}	0.0194	0.0021	0.0023	**0.0012**
	MAE_{phy}	0.0781	0.0264	0.0330	**0.0200**
HVAC	MSE_{rec}	0.3314	0.5314	0.2686	**0.1994**
	MAE_{rec}	0.4709	0.6128	0.4670	**0.3454**
	MSE_{phy}	50.22	14.69	14.469	**14.081**
	MAE_{phy}	6.713	3.263	3.203	**3.1600**

alignment increase after removing the physics-based loss. We also observe that a higher degree of precision in the physics model and higher frequency sensor data sampling (resulting in fewer synchronization errors), lead to larger performance improvements. For example, the relationships in inertial navigation physics are more precisely captured than with CO_2 or temperature systems, and the IMU sensors are sampled at higher frequencies than CO_2 sensors. Correspondingly, we observe greater performance gains from employing physics equations than other applications. Secondly, reconstruction loss is important as it facilitates the model's ability to learn data distribution. In its absence, the model may potentially generate trivial outputs that, while satisfying physics constraints, neglect the actual data distribution (e.g., outputs of pure zeros). Lastly, the pre-training phase serves as an essential warm-up period to help the model better adapt to the underlying data distribution. To summarize, the physics-based loss, reconstruction loss, and pre-training phase collectively contribute to the overall efficacy of the model, and their presence is crucial for optimal performance.

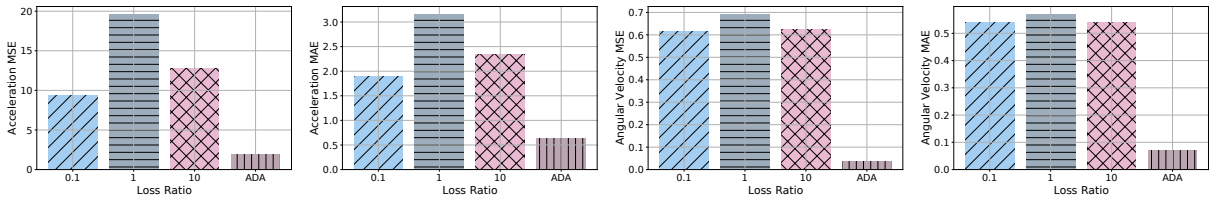


Figure 2.13. Performance analysis on the ratio of reconstruction loss over physics-based loss. Adaptive loss ratio yields better performance compared with fixed ratios.

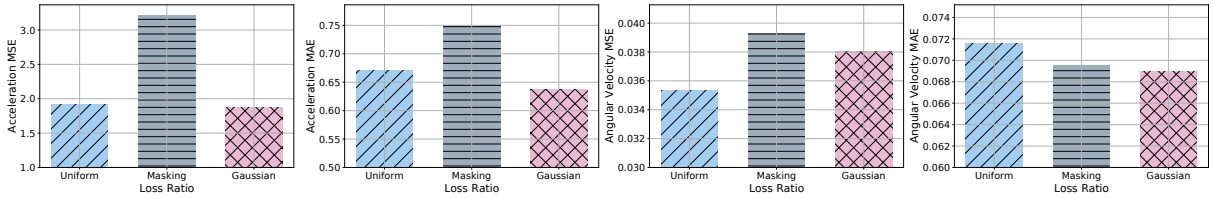


Figure 2.14. Performance analysis on the types of manually injected noise for the reconstruction process. Applying random noise performs better than zero masking.

2.5.6 Sensitivity Analysis

Balance between Reconstruction Loss and Physics-Based Loss

In our experiments, we adopt an adaptive strategy to adjust the ratio between reconstruction loss (l_{rec}) and physics-based loss (l_{phy}) during training. Denote the ratio as $\lambda = l_{\text{rec}}/l_{\text{phy}}$. More specifically, for each iteration, we compute the mean value of the reconstruction loss, and adaptively adjust λ such that $(\lambda \cdot l_{\text{phy}})$ aligns with the same order of magnitude as l_{rec}. Consequently, this adaptive approach avoids overemphasizing either the reconstruction loss or physics-based loss throughout the entire training process. To validate the effectiveness of adaptive loss ratios, we compare PILOT with its counterparts trained with fixed loss ratios, varying from 0.1, 1 to 10. As shown in Figure 2.13, adaptive loss ratio (denoted as "ADA") yields the best performance compared with different fixed ratios.

Manually Injected Noise Types

We also study the effect of different noise types injected for optimizing the reconstruction loss. In our experiments, we use Gaussian noise as an example injected noise. We compare PILOT with its counterparts trained with manually injected uniformly distributed noise, as well

Table 2.6. Efficiency Analysis on Raspberry Pi 4.

Metrics	Params	Size	Inference Time	CPU Usage
Efficiency	270K	284 KB	4 ms	25%

as manually injected zero masks. As shown in Figure 2.14, injecting random noise (Gaussian or Uniform noise) performs better than applying zero masking, as random noise aligns closer to real-world noise distribution compared with zero masking. Moreover, the model is less sensitive to the particular noise types applied, such as Gaussian noise or uniformly distributed noise.

2.5.7 Efficiency Analysis for Edge Devices

As many denoising methods for sensor data run on edge devices, we explore the time and memory efficiency of PILOT in Table 2.6 using Raspberry Pi 4 as an example edge device. We deploy the inertial navigation denoising application through the Tensorflow Lite framework on Raspberry Pi 4. For time efficiency, PILOT can denoise a sequence of 100 readings (corresponding to a sequence of 1 second) within 4 milliseconds, performing real-time denoising. To ensure the reliability of our evaluation, this experiment has been repeated a thousand times to obtain the average inference time. For memory efficiency, PILOT is a lightweight CNN model with just 270K parameters and 284 KB model size. Moreover, it demonstrates an average CPU utilization of only 25% during the entire inference process. In summary, PILOT is both time efficient and memory efficient for real-time operations on edge devices.

2.6 Discussions and Future Work

In this section, we discuss potential future directions of PILOT.

Broader Applicability to Other Sensing Systems. For PILOT to work, we, of course, need a well-understood and verified physical model that accurately models relationships among important observed variables. Such models may be complex as the "order" of physical models increases with their complexity to model various conditions and corner cases. It is, however,

possible to combine such physics-based models with well-characterized numerical simulation models that achieve similar effect: for instance, model of semiconductor device versus a BSIM3 simulation model used in SPICE simulators [46]. We note that apart from the three applications we have explored, in practice numerous sensing systems have underlying physics relationships, and we recognize this opportunity for extended exploration in future work. We discuss a few potential use cases as some examples for the model's wider applicability:

- **Power System.** In a power system, the most classical Ohm's law describes the relationships between power, current and voltage. Electronic devices measuring current or voltage are often subject to environmental disruptions, such as temperature fluctuations, resulting in measurement noise. Utilizing physics priors like Ohm's Law, PILOT can offer a promising solution to mitigate these noise-related inaccuracies.
- **Weather Monitoring.** Weather monitoring and forecasting involve calculations with respect to air pressure, temperature, wind speed, etc. These properties can be modeled as fluid dynamics problems, which are often characterized by the Navier-Stokes equations linking velocity and pressure. Environmental sensors responsible for capturing data like wind speed or pressure are susceptible to noise originating from environmental influences or sensor sensitivity. PILOT can leverage physics-based priors such as the Navier-Stokes equations to denoise these collected data, enhancing the accuracy of weather monitoring.
- **Localization in Autonomous Systems.** In autonomous systems, we employ a combination of GPS, Lidar, Radar and IMU sensors to localize autonomous driving cars or drones. These sensor readings can be noisy due to atmospheric effects, multipath propagation, etc. PILOT is capable of applying the laws of motion (e.g., acceleration is the second derivative of location) to denoise location data collected from these multiple sources.

Uncertainty Quantification. Our future work also includes incorporating uncertainty quantification (UQ) into the denoising process. The integration of UQ aims to provide a holistic understanding of noise structure, delivering not just a point estimate for the denoised data but also the model's confidence in that estimate, avoiding overconfident and potentially inaccurate

inferences. The integration may also help quantify the effects of denoising with respect to the precision of the model or synchronization errors. Uncertainty quantification can be achieved by adopting a probabilistic approach to extend the current model architecture to output not just a single denoised signal but a distribution over possible denoised signals.

FPGA Acceleration. We have implemented our denoising method on edge devices like Raspberry Pi and demonstrated its time and memory efficiency. To take this step further, we plan to explore hardware acceleration as Field-Programmable Gate Arrays (FPGA) co-processors to explore the possibility and cost of such edge inference. With FPGA's reprogrammable nature, we have the flexibility of prototyping and testing different configurations and parameters for the best possible performance and efficiency. Furthermore, FPGA allows us to seamlessly transition from prototype to subsequent production of Application-Specific Integrated Circuit (ASIC), which enable us to scale up our solution while reducing power consumption and production costs.

2.7 Summary

In this chapter, we present a physics-informed denoising method, PILOT, as the first step in our robust sensor time series pipeline. We build upon the insight that measurements from different sensors are intrinsically related by known underlying physics principles. This approach allows the model to harness these physics constraints as guidance during training to improve denoising process. This paves the way for a more practical denoising solution, especially given the frequent challenges associated with acquiring ground truth clean data in sensing systems or understanding underlying noise patterns. Extensive experiments show the efficacy of PILOT in removing sensor noise across three representative real-world sensing systems. PILOT produces denoised results for low-cost sensors that align closely with high-precision and high-cost sensors, leading to a cost-effective denoising approach. PILOT is also lightweight and can enable real-time denoising on edge devices.

Chapter 2 incorporates material from the publication "Physics-Informed Data Denoising

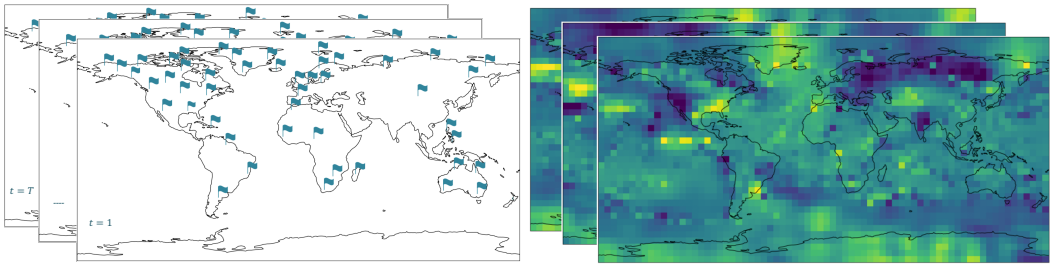
for Real-Life Sensing Systems", by Xiyuan Zhang, Xiaohan Fu, Diyan Teng, Chengyu Dong, Keerthivasan Vijayakumar, Jiayun Zhang, Ranak Roy Chowdhury, Junsheng Han, Dezhi Hong, Rashmi Kulkarni, Jingbo Shang, Rajesh Gupta, published in Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems (SenSys 2023). The dissertation author was primary investigator and the lead author of this paper.

Chapter 3

Physical Context of Spatio-Temporal Correlations

After removing the noise, the sensor time series might still contain missing values which negatively affect the model robustness. In this chapter, we present the second step of imputing missing values in our framework by leveraging physical context of spatio-temporal correlations.

Scientific discoveries and studies about our physical world have long benefited from large-scale and planetary sensing, from weather forecasting to wildfire monitoring. However, the limited deployment of sensors in the environment due to cost or physical access constraints has lagged behind our ever-growing need for increased data coverage and higher resolution, impeding timely and precise monitoring and understanding of the environment. Therefore, we seek to *extend the spatial coverage* of analysis based on existing sensory data, that is, to "generate" data for locations where no historical data exists. This problem is fundamentally different and more challenging than the traditional spatio-temporal imputation that assumes data for any particular location are only partially missing across time. Inspired by the success of Generative Adversarial Network (GAN) in imputation, we propose a novel ESC-GAN. We observe that there are *local* patterns in nearby locations, as well as trends in a *global* manner (e.g., temperature drops as altitude increases regardless of the location). As local patterns may exhibit at different scales (from meters to kilometers), we employ a multi-branch generator to aggregate information of different granularity. More specifically, each branch in the generator contains 1)



(a) Original Data Coverage **(b)** Extended Data

Figure 3.1. Our task of extending spatial coverage: based on the sparse sensory measurements over time in (a), we aim to extend its coverage to the entire globe as shown in (b).

randomly masked 3D partial convolutions at different resolutions to capture the local patterns and 2) global attention modules for global similarity. Next, we adversarially train a 3D convolution based discriminator to distinguish the generator's output from the ground truth. Experiments on three geo-sensor datasets demonstrate that ESC-GAN outperforms state-of-the-art methods on extending spatial coverage as well as traditional spatio-temporal imputation.

3.1 Introduction to Extending Spatial Coverage of Physical Sensors

Wide-scale deployment of environmental geo-sensors have advanced our understanding of our ecosystem and its evolution. These sensors enable us to observe the very fabric of our surrounding physical world. For example, outdoor thermometers detect seasonal patterns and annual shift in temperature to help understand global warming [55]; rain gauges measure the precipitation level for hydrological modeling, flood forecasting, and agricultural purposes [2]; magnetometers monitor the earth's magnetic field, helping advance magnetosphere studies [79].

While valuable, a critical limitation of these geo-sensors is that each of them covers only a fraction of the total area, and it is impossible for a majority of them to cover the entire planetary surface. Constructing and maintaining sensing stations incurs high costs, and many locations are often inaccessible due to physical access constraints such as harsh environmental conditions and urban development. Consequently, sensors are sparsely distributed across the globe, limiting

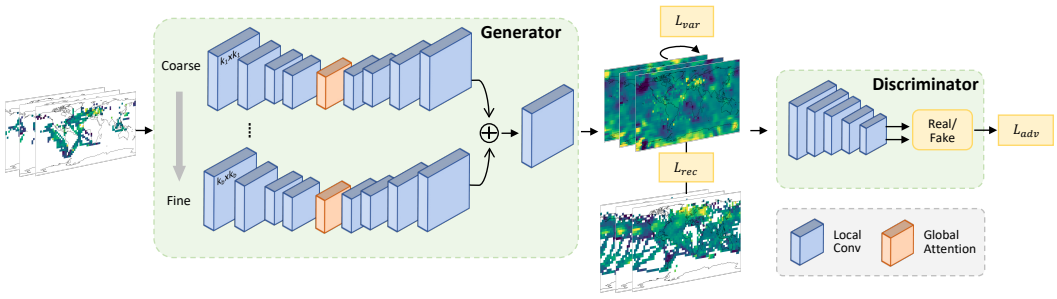


Figure 3.2. An overview of our ESC-GAN: The multi-branch generator takes as input missing maps, and the generator produces grid maps with all the missing grid cells recovered. We feed the recovered and ground-truth maps to the discriminator for a real or fake classification.

deeper understandings of large-scale phenomena. Figure 3.1a shows an example of the limited availability of magnetic field monitoring stations on the earth.

In this chapter, we seek a cost-effective approach to extend the spatial coverage of existing planetary sensory data without deploying additional sensors. We name this task as *extending spatial coverage* (ESC) of sensor data. To be specific, our goal is to "generate" data at locations where no historical values are ever recorded and extend the spatial coverage to the entire globe (as illustrated in Figure 3.1b). To formulate the ESC problem, we assume the entire globe (denoted as D) is gridded into $m \times n$ cells. Given the data from a set of observed or partially observed grid cells D_O (i.e., where we have sensory measurements), we aim to generate data for the remaining unobserved grid cells $D_U = \{D \setminus D_O\}$.

The ESC problem faces unique challenges and opportunities:

- *Complete lack of temporal dimension information.* In ESC task, we have no prior data or knowledge for unobserved grid cells. Therefore, traditional (spatio-)temporal imputation models, which assume data are partially missing in the temporal domain [165, 28, 159, 311, 292, 168], cannot solve the ESC problem. Spatial imputation methods can be applied to each time snapshot separately [104, 38, 80, 225]. They, however, miss the temporal trends from observed grid cells. Spatio-temporal interpolation methods [138, 139, 233, 8, 274], on the

other hand, do not sufficiently exploit the spatio-temporal properties (e.g. global context and multi-scale structure) for imputation.

- *Existence of global context.* Sensors at a distance might exhibit similar readings due to similar geographical contexts (e.g., similar landform) [292]. This complements the first law of geography ("...near things are more related than distant things") [239] and inspires us to explore global contexts. Solutions to image and video inpainting [147, 296, 32, 258] typically consider only local context (e.g., patterns in nearby pixels), and thus cannot capture global patterns. It is necessary to combine global and local context views of the spatio-temporal data.
- *Multi-scale structure.* Spatio-temporal data often exhibit multi-scale structures. Specifically, while fine-grained data of a particular cell can reveal accurate and detailed local patterns, coarse-grained data distributed across a large area present a "macro" view, which is less sensitive to local missing information. Therefore, we need to jointly consider coarse-level information for completeness and fine-level information for accurate modeling.

We propose a novel framework ESC-GAN to address all these challenges. Figure 3.2 illustrates our proposed model architecture. ESC-GAN comprises a generator and a discriminator following the Generative Adversarial Network (GAN) framework [82], as GAN has been widely used to model the complex distribution in spatio-temporal data [165, 166, 159, 141, 238, 322]. For the generator, we leverage local 3D partial convolutions together with global attention modules to focus on both local (e.g., patterns in nearby locations) and global (e.g., phenomena that exist regardless of location) contextual information. We also design a multi-branch encoder for aggregating information of different granularity. The discriminator is composed of 3D convolutional layers. Extensive experiments on three geo-sensor datasets have verified the effectiveness of ESC-GAN under different missing data scenarios. Moreover, as our model does not impose any additional constraints or assume any domain knowledge, it can also be applied to other spatio-temporal problems like urban environment monitoring, traffic estimation, etc.

In summary, we explore a challenging task of extending spatial coverage, where we attempt to generate data at locations with no historical observations. We have analyzed the

unique challenges and opportunities, which guide our model design. Our main contributions are summarized as follows:

- We propose a novel ESC-GAN framework that can address all identified challenges.
- We leverage 3D partial convolutions to learn the local correlations in both spatial and temporal dimensions, global attention modules to capture global contextual information, and a multi-branch encoder to exploit information of different granularity.
- Extensive experiments on three real-world datasets have demonstrated the superiority of ESC-GAN over all the compared methods, including state-of-the-art spatio-temporal imputation methods, image and video inpainting methods.

3.2 Related Work

Spatio-Temporal Data Imputation. For traditional spatio-temporal imputation, existing approaches mainly include statistical models and deep generative models. Statistical models include filling with zero, mean of existing values, filling with last observation, regression-based models [7], MICE [24], Matrix Factorization [182], k-nearest neighbours [104], tensor factorization [44], multi-view learning method [292]. Deep generative models have so far shown promising results with different sequential neural network based models [34, 28, 159, 168] and generative adversarial network (GAN) [82] based models [165, 166, 238, 141]. These methods typically assume partial missing in the temporal domain.

For spatial missing data imputation, existing approaches are mainly statistical, e.g. inverse distance weighting [38], matrix factorization [174, 80], variogram modeling [225]. These methods miss modeling the temporal trend from available locations.

Meanwhile, existing spatio-temporal interpolation methods do not sufficiently exploit properties of spatio-temporal data (e.g. global context and multi-scale structure) [138, 139, 233, 8, 274]. Miao et al. propose a pyramid dilated spatial-temporal network for learning crowd flow representations, but the model is designed for forecasting task and only learns temporal

attention [175]. Tang et al. infer traffic volume of observed regions through joint modeling of dense and incomplete trajectories [237]. However, their method is uniquely focused on trajectory data and is not directly applicable to other domains. We propose to jointly model spatial and temporal dependencies, learning from both local and global patterns. Our model is applicable to a wide range of spatio-temporal problems.

Image and Video Inpainting. Image or video inpainting aims to reconstruct missing regions in an image or video frame. Unlike conventional convolution neural network that treats all the pixels equally as valid pixels, Partial Convolution and Gated Convolution based methods assign different weights to different input pixels to reduce color discrepancy and blurriness [147, 296, 32]. To synthesize different image components for image inpainting, Wang et al. propose a multi-column network to extract features at different levels [258]. However, these approaches, which are specifically designed for image and video inpainting, cannot achieve satisfactory results on our ESC task, as spatio-temporal data contain more complex correlations and stochastic properties compared with images. These methods also only exploit local features, while our model jointly models local and global patterns exhibited in spatio-temporal data.

Global Attention. Convolution neural network has been successful in image or video processing, but convolution by nature is a local operation. Attention mechanism [15] has enabled a model to gain a global view of the input data. In particular, self-attention [248] draws global dependencies between input and output based solely on the attention mechanism. Self-attention calculates response at one position based on weighted sum of values from all the other positions, and has shown impressive results in sequential task like machine translation and sequence generation. Non-local model is further introduced to bridge the gap of applying self-attention to image and video tasks [256, 306, 29, 293]. We follow the attention module in [293] to capture global context in spatio-temporal data, and further combine it with multi-scale structure.

3.3 Preliminary and Problem Formulation

In the extending spatial coverage (ESC) problem, we aim to learn to generate data at locations where data are completely missing at all timestamps $t = 1, \ldots, T$. Formally, we denote a grid cell as S_{ij}, where $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Let $\mathbf{X} \in \mathbb{R}^{t \times m \times n}$ denote the data, and $\mathbf{M} \in \{0,1\}^{t \times m \times n}$ denote the corresponding masking matrix. If $M_{t,i,j} = 1$, it means $S_{i,j}$ is valid at time t (i.e. we have data in grid cell $S_{i,j}$ at time t); otherwise, $S_{i,j}$ is invalid at time t (i.e. data is missing in cell $S_{i,j}$ at time t). Let Φ denote the set of missing timestamps. In the ESC problem, we do not have any data available in certain grid cells. Therefore, assuming $S_{i,j}$ is a grid cell that is unobserved, we have $M_{t,i,j} = 0$ for all timestamps $t \in \Phi = \{1, \ldots, T\}$. Given data observed at other locations $\{X_{t,i,j} | M_{t,i,j} = 1\}$ for $t = 1, \ldots, T$, our goal is to generate data for all the unobserved grid cells and output a complete set of grid cells $\{\tilde{X}_{t,i,j}\}$ at all timestamps $t = 1, \ldots, T$. This is in contrast to the traditional spatio-temporal imputation problem, where a given missing grid cell S_{ij} is only *partially unobserved*, i.e., $M_{t,i,j} = 0$ for $t \in \Phi \subseteq \{1, \ldots, T\}$.

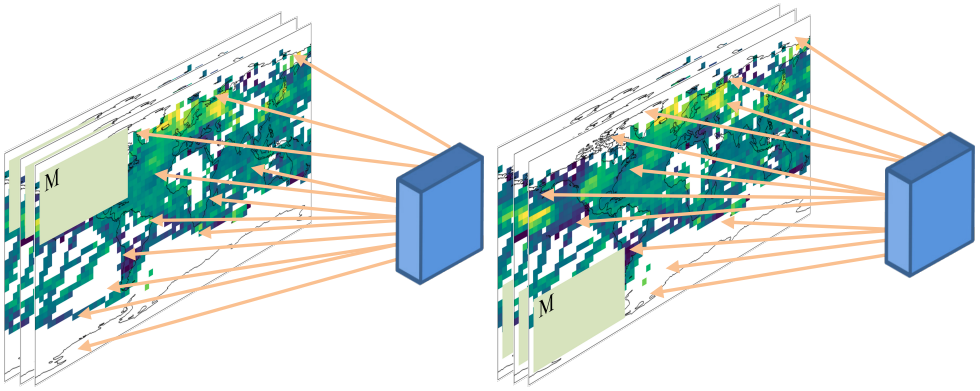
3.4 Methodology

As illustrated in Figure 3.2, ESC-GAN comprises a generator of UNet-like structure [210] and a discriminator consisting of multiple 3D convolutional layers. The generator combines local partial 3D convolution with global attention module in multiple branches. We next detail our ESC-GAN framework.

3.4.1 Randomly Masked 3D Partial Convolution

Spatio-Temporal data display local similarity, so we first apply convolutions to model the local patterns. Vanilla convolutions on grid map would treat each grid cell equally as valid. To obtain the output $Y_{t,y,x}$, we calculate (for simplicity we omit the bias term in the formula):

$$Y_{t,y,x} = \sum_{k=-k'_t}^{k'_t} \sum_{i=-k'_h}^{k'_h} \sum_{j=-k'_w}^{k'_w} W_{k'_t+k,k'_h+i,k'_w+j} \cdot X_{t+k,y+i,x+j}, \tag{3.1}$$



(a) iteration i **(b)** iteration j

Figure 3.3. Randomly Masked 3D Partial Convolution. We randomly mask out parts of the grid maps on different iterations (shown as **M**). Blue rectangle denotes convolution filter, and orange arrows show that convolution operation is only performed on unmasked grid cells.

where $X_{t,y,x}$ and $Y_{t,y,x}$ represent the input and output of a specific convolution layer, respectively; **W** represents convolution filter weights; and $k'_t = \frac{k_t-1}{2}, k'_h = \frac{k_h-1}{2}, k'_w = \frac{k_w-1}{2}$ with k_t, k_h, k_w representing kernel size along the time, height, and width dimensions, respectively. From the equation we can see that $Y_{t,y,x}$ is based on all the grid cells within the receptive field, regardless of whether the corresponding cells contain valid values. These invalid values involved introduce bias into the training process.

One way to address this problem is to apply partial convolutions [147] only on valid grid cells, i.e., locations with data in our context. For each iteration, we apply a training mask **M** removing a random subset of locations, as shown in Figure 3.3. For visualization purpose, **M** is shown as rectangle in Figure 3.3, but in practice **M** is randomly scattered across the map. **M** randomly masks out locations over all the cells. If the original cell has no data, then it does not change after being masked. We defer more complex masking strategy that coordinates missing distributions as future work. If $M_{ij} = 0$ then we mask out the corresponding grid cell S_{ij} at all timestamps; otherwise, we keep the data at this grid cell. Such masking during training helps the model learn how to recover data over time in the masked-out cell. Moreover, since the training mask is randomly generated during each iteration, it is able to cover the entire map and introduces minimum bias of region difference.

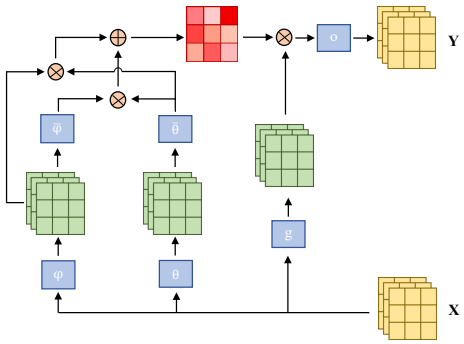


Figure 3.4. Global attention module architecture. The input $\mathbf{X}$ is transformed through three linear embeddings θ, ϕ, g. We calculate pairwise and unary scores, and multiply the scores with embeddings to obtain output $\mathbf{Y}$.

Formally, to generate data at missing locations, we extend partial convolutions to *randomly masked 3D partial convolutions*:

$$\mathbf{Y} = \begin{cases} \mathbf{W}^T(\mathbf{X} \odot \mathbf{M}) \frac{|\mathbf{1}|_1}{|\mathbf{M}|_1} + b, & \text{if } |\mathbf{M}|_1 > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

Here, we keep the same notation such that $\mathbf{X}, \mathbf{Y}, \mathbf{W}$ represent the input, output, and convolution filter weights, respectively, in a specific convolution layer, and b is the bias term. $\mathbf{1}$ has the same size as $\mathbf{M}$ with all values being 1. Therefore, $\frac{|\mathbf{1}|_1}{|\mathbf{M}|_1}$ serves as a scaling factor to compensate for the number of valid grid cells. We also apply mask updating after each layer following prior study [147]: if the output is able to condition on at least one valid input, then the corresponding mask after updating would be 1.

3.4.2 Global Attention Module

In the real world, sensor values are not necessarily only correlated within a small window, in terms of both space and time. Two sensors afar can still have similar values if they share similar geographical contexts (e.g. landform) [292]. Moreover, sensor readings in different years might be similar too when they share similar temporal contexts (e.g., seasonal pattern).

The 3D partial convolution with random mask we introduce in the previous subsection

only operates locally, since it mainly sums up product of kernel weights and input within local sliding windows. To take the global view into account, we incorporate global attention module as illustrated in Figure 3.4.

The global attention module aims to take both neighboring and distant grid cells into account for calculation. Similar to [256, 248, 306], we first apply linear embedding $\theta(\mathbf{X}) = \mathbf{W}_\theta \mathbf{X}, \phi(\mathbf{X}) = \mathbf{W}_\phi \mathbf{X}$ to embed the input. As found in [29, 293], vanilla non-local block often degenerates to purely unary term in some image recognition tasks. Therefore, we follow [293] to split the attention computation into a pairwise and a unary term, for better modeling both global pairwise similarity and global unary effect. We compute the attention scores of every two input regions $\mathbf{X}, \mathbf{X}'$ in the embedding space as

$$f(\mathbf{X}, \mathbf{X}') = \frac{e^{(\theta(\mathbf{X})-\mu_\theta)^T(\phi(\mathbf{X}')-\mu_\phi)+\mu_\theta^T\phi(\mathbf{X}')}}{\sum_{\mathbf{X}'} e^{(\theta(\mathbf{X})-\mu_\theta)^T(\phi(\mathbf{X}')-\mu_\phi)+\mu_\theta^T\phi(\mathbf{X}')}}, \tag{3.3}$$

where μ_θ, μ_ϕ are average embedding values over all the regions from θ, ϕ. The first term in the numerator $(\theta(\mathbf{X}) - \mu_\theta)^T(\phi(\mathbf{X}') - \mu_\phi)$ captures pairwise long-range dependency, and the second term $\mu_\theta^T \phi(\mathbf{X}')$ captures the global unary effect. Intuitively, $f(\mathbf{X}, \mathbf{X}')$ indicates the global context similarity between $\mathbf{X}'$ and $\mathbf{X}$. Then, to calculate the output value $O_{t,y,x}$, we calculate the weighted sum of attention scores from all input values $X_{t',y',x'}$:

$$O_{t,y,x} = \sum_{\forall t',y',x'} f(X_{t,y,x}, X_{t',y',x'}) g(X_{t',y',x'}), \tag{3.4}$$

where $g(\mathbf{X}) = \mathbf{W}_g \mathbf{X}$ is also a linear embedding layer. Finally, we apply linear embedding $\mathbf{W}_o$ with residual link [92] to compute the output feature $\mathbf{Y}$ as

$$\mathbf{Y} = \mathbf{W}_o \mathbf{O} + \mathbf{X}, \tag{3.5}$$

The above linear embedding layers $\mathbf{W}_\theta, \mathbf{W}_\phi, \mathbf{W}_g, \mathbf{W}_o$ are implemented as $1 \times 1 \times 1$ convolutions.

Following [256], a batchnorm layer with scale parameter initialized to zero follows $\mathbf{W}_o$ so that the module starts from an identity mapping that relies on local information, then gradually learns the long range dependency. We follow the sub-sampling trick in [256] and apply max pooling after ϕ and g for computational efficiency. We leave a more efficient design for global attention module as future work.

3.4.3 Multi-Scale Structure Learning

Spatio-temporal data often demonstrate multi-resolution structures [159]. In our case, fine-grained data in a specific grid cell reflect accurate measurement of that cell, while coarse-grained data covering multiple grid cells are less sensitive to missing values. The one-branch encoder-decoder U-Net architecture is not able to effectively extract representations at different levels [258]. To better capture the dependencies across different scales, we adopt a multi-scale learning procedure. More specifically, we apply n_b parallel branches of U-Net structure in the generator. These n_b branches apply convolution filters of different receptive fields to extract multi-resolution features. Assume $\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{n_b}$ are hidden features computed after the last layer of decoders from different branches, then we concatenate these features and feed the concatenated feature into a convolution layer shared across branches to obtain the aggregated features $\mathbf{h}' \in \mathbb{R}^{c \times t \times m \times n}$:

$$\mathbf{h}' = \text{Conv}([\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{n_b}]), \tag{3.6}$$

where c, t, m, n are the channel, time, height, and width dimension size, respectively. The number of branches n_b can be decided by the input granularity, i.e., more branches when input data is fine-grained. The parallel U-Net structure overcomes limitation of the coarse-to-fine architecture where the fine stage is dependent on the coarse stage, thus being susceptible to upstream errors.

3.4.4 ESC-GAN Generator

As demonstrated in Figure 3.2, the overall architecture of ESC-GAN contains a generator G and a discriminator D. The generator contains 3D partial convolutions for learning local

patterns and global attention module for learning global trends. The generator also adopts multiple branches to aggregate multi-level features. We combine grid reconstruction loss ℓ_{rec}, variation loss ℓ_{var}, and adversarial loss ℓ_{adv} to optimize the generator.

To ensure grid reconstruction accuracy, we calculate Mean Square Error (MSE) between the ground-truth and generated grid maps. More precisely, let $\mathbf{X}$ and $\mathbf{Z}$ denote the ground truth and generated grid map, $\mathbf{M}$ denote the random training mask, we calculate MSE for these randomly masked out regions as

$$\ell_{\text{rec}} = \frac{1}{N_{\text{masked}}} \sum_t \sum_y \sum_x (1 - M_{t,y,x})(Z_{t,y,x} - X_{t,y,x})^2, \tag{3.7}$$

where N_{masked} denotes the number of masked out grid cells.

Apart from reconstruction loss, we also compute variation between the masked out region and valid region in the generated maps. The goal is to ensure smooth transition between masked out and valid portions. We first calculate the composite grid map $\tilde{\mathbf{X}}$, where valid regions keep the same value as the original grid map, and both randomly masked out regions and originally invalid regions are filled with generated values:

$$\tilde{\mathbf{X}} = \mathbf{X} \odot \mathbf{M} + \mathbf{Z} \odot (1 - \mathbf{M}). \tag{3.8}$$

Then, we compute the variation over the composite grid map as

$$\begin{aligned} \ell_{\text{var}} = \frac{1}{m \times n} (& \sum_{(y,x) \in R, (y+1,x) \in R} ||\tilde{X}_{t,y+1,x} - \tilde{X}_{t,y,x}||_1 \\ & + \sum_{(y,x) \in R, (y,x+1) \in R} ||\tilde{X}_{t,y,x+1} - \tilde{X}_{t,y,x}||_1). \end{aligned} \tag{3.9}$$

In the above equation, $m \times n$ is the number of grid cells in the map, and R is the 1-cell dilation of the masked region, similar to [147]. To compute the variation loss, we shift one grid cell in two spatial dimensions within R and penalize the shifted difference.

3.4.5 ESC-GAN Discriminator

Spatio-temporal data follow complex distributions and demonstrate high variations across time and space. They are influenced by a number of external factors (e.g. adverse weather), thus exhibiting irregular and stochastic forms. A model trained with only ℓ_{rec} and ℓ_{var} is not adequate to model these correlations, as it tends to output an average over different data [214]. Thus, we further train a discriminator with the generator using an adversarial strategy.

The discriminator $D(\cdot)$ is composed of 3D convolution layers. The generator $G(\cdot)$ generates grid maps $\mathbf{z} \sim P_{\mathbf{Z}}(\mathbf{z})$ that are indistinguishable from ground-truth grid maps $\mathbf{x} \sim P_{\mathbf{X}}(\mathbf{x})$, and the discriminator learns to classify feature maps as real or fake:

$$\ell_D = \mathbb{E}_{\mathbf{x}\sim P_{\mathbf{X}}(\mathbf{x})}[RELU(1-D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim P_{\mathbf{Z}}(\mathbf{z})}[RELU(1+D(\mathbf{z})], \tag{3.10}$$

$$\ell_{\text{adv}} = \ell_G = -\mathbb{E}_{\mathbf{z}\sim P_{\mathbf{Z}}(\mathbf{z})}[D(\mathbf{z})]. \tag{3.11}$$

The overall training objective of ESC-GAN is a weighted sum of the reconstruction loss, variation loss, and adversarial loss, namely,

$$\ell = \ell_{\text{rec}} + \lambda_{\text{var}}\ell_{\text{var}} + \lambda_{\text{adv}}\ell_{\text{adv}}, \tag{3.12}$$

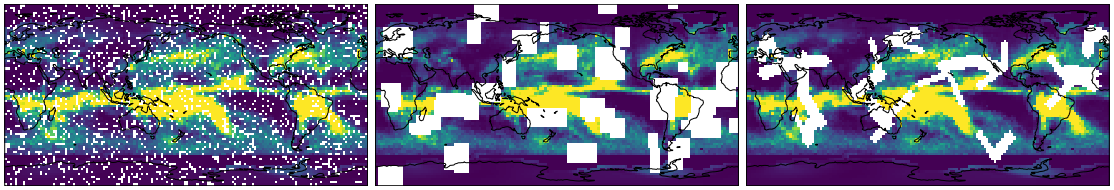
where λ_{var} and λ_{adv} are hyper-parameters balancing between reconstruction loss, variation loss, and adversarial loss.

3.5 Evaluation

We evaluate our ESC-GAN in this section. In particular, we investigate the following perspectives: (1) How does the proposed model perform compared with other baselines in the ESC task? (2) How effective are the different components in the proposed model? (3) How robust is the proposed model with respect to various missing region distributions and missing

Table 3.1. Dataset Statistics

Dataset	Lat	Lon	Time	Granularity	#Grid Cells
HadCRUT	36	72	2004	$5^\circ \times 5^\circ$	5,194,368
CMAP	72	144	503	$2.5^\circ \times 2.5^\circ$	5,215,104
KDD CUP 2018	6	8	8736	$0.0167^\circ \times 0.0175^\circ$	96096



(a) Scatter **(b)** Regular Cluster **(c)** Irregular Cluster

Figure 3.5. Three missing data distributions

ratios? Finally, (4) How does the model perform when extended to traditional spatio-temporal imputation task with random missing data in the temporal domain? We provide both qualitative and quantitative analyses to verify the effectiveness and reliability of our model.

3.5.1 Datasets

We use three publicly available geo-sensory datasets to validate our proposed model. The third dataset is used to evaluate ESC-GAN for imputing random missing values in the temporal domain, which is a well-established task. Dataset statistics are summarized in Table 3.1.

HadCRUT[1] is a global temperature dataset, providing gridded temperature anomalies (measured by annual temperature shift) across the world [180, 181]. Temporally, the data contain monthly mean spanning from 1850 to 2020; spatially, the data covers grids of 5° latitude by 5° longitude globally (72×36).

CMAP[2] consists of monthly averaged precipitation level values (mm/day) [276] ranging approximately from 0 to 70mm/day. Values are obtained from 5 kinds of satellite estimates (GPI,OPI,SSM/I scattering, SSM/I emission, and MSU) and rainfall gauge. The data span from

[1] available on Climate Research Unit website: https://crudata.uea.ac.uk/

[2] CMAP Precipitation data is provided by the NOAA/OAR/ESRL PSL, Boulder, Colorado, USA, from their website at https://psl.noaa.gov/

1979 to 2020, and spatially, cover a 2.5° latitude by 2.5° longitude global grid (144 × 72).

For HadCRUT and CMAP, we first normalize the dataset using z-normalization (i.e., subtracting the population mean from the individual raw stream and then dividing the difference by the population standard deviation). The whole grid map data are split into sequences of length 12, and each sequence corresponds to one-year worth of data. To simulate different real-world missing distribution, we study three types of common missing scenarios: (1) scattered locations, (2) regular clustered locations, and (3) irregular clustered locations, as visualized in Figure 3.5. These missing areas are left out for test set and are kept being masked out during training. The remaining area is further randomly split into 80% training set and 20% validation set. We tune the hyper-parameters for both our model and baseline models on the validation set.

We also study how ESC-GAN performs on traditional spatio-temporal imputation task using the benchmark KDD CUP dataset. The **KDD CUP** 2018 dataset[3] measures hourly air quality and meterological data at city-scale. We follow previous study using this dataset [165, 168] to select 11 common locations in Beijing that measure both air quality and meterological values. Same as previous studies, we use 12 variables including PM2.5, PM10, temperature, weather, etc. To adapt the input to our model, we map the data from 11 stations into a 6 × 8 grid according to their geographical locations provided on the KDD CUP 2018 website. Then, prediction in the mapped grid cell is regarded as the prediction for the corresponding location.

We will discuss more details about random spatio-temporal missing in Section 3.5.7.

3.5.2 Compared Methods and Metrics

We compare ESC-GAN with four types of methods.

(1) Classical Imputation. We apply the methods for each timestamp separately.

- **Mean/Zero**: Mean/Zero filling extends the missing station values by filling mean value of the current timestamp/zero (which is the mean of all timestamps after normalization).
- **sKNN**: Spatial K Nearest Neighbour, which extends the readings with the average values of

[3]KDD CUP Challenge 2018 dataset, available at: http://www.kdd.org/kdd2018/

the missing station's k nearest available spatial neighbors [104].

- **IDW**: Inverse Distance Weighting, a global spatial learning method that interpolates with weighted average of available data points as a function of inverse distance [38].
- **Kriging**: A geo-statistical interpolation method which assumes that the distance between points reflects spatial correlation that can be used to explain surface variation [225]. We implement Kriging method using PyKrige library[4].
- **MF**: Matrix Factorization, which iteratively replaces missing elements with those obtained from soft-thresholded SVD [174]. We implement MF method using fancyimpute library [212].

(2) State-of-the-art Spatio-temporal Imputation. We implement the state-of-the-art spatio-temporal imputation methods based on public code of the respective paper.

- **BTTF**: Bayesian temporal factorization framework for modeling spatio-temporal data with missing values. The method integrates low-rank matrix/tensor factorization and vector autoregressive (VAR) process into a single probabilistic graphical model [44].
- **ST-MVL**: A multi-view learning imputation method combining empirical statistical models for global view, with data-driven algorithms for local view [292]. In the original implementation, weights for combining four views are optimized for each location. However, in ESC task we do not have any available data to train on these unobserved locations. Therefore, we use optimized weights from their neighbors to predict the missing values.
- **NAOMI**: A non-autoregressive deep generative spatio-temporal imputation method. It also exploits the multi-resolution structure by decoding recursively from coarse to fine-grained resolutions [159]. The original NAOMI implementation mainly relies on temporal information for imputation. As in ESC task we have no temporal information for unobserved locations, we concatenate observed locations with unobserved locations channelwise in order to better generalize from observed locations.
- **IGKNN**: Deep spatio-temporal kriging method based on inductive graph neural network. The method learns spatial message passing mechanism through generating random subgraph and

[4]https://github.com/GeoStat-Framework/PyKrige

reconstructing subgraph signals [274].

(3) State-of-the-art Image and Video Inpainting. We implement the state-of-the-art image and video inpainting methods based on public code of the respective paper.

- **PConv**: Partial Convolution [147, 111] for imputing 2D data. The convolution is conditioned only on valid cells.
- **3DGated**: 3D video inpainting method, which uses 3D gated convolution as generator and proposes a Temporal Patch-GAN loss to enhance temporal consistency [32].

(4) Ablations of ESC-GAN.: We also implement three ablations of ESC-GAN.

- **ESC-GAN-vanilla**: Single branch with only local convolution.
- **ESC-GAN-local**: Multiple branches with only local convolution.
- **ESC-GAN-single**: Single branch with global attention.

Following previous spatio-temporal imputation research [165, 292], we evaluate the performance of our model and baselines using $MSE(x,y) = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2$ (Mean Square Error) and $MAE(x,y) = \frac{1}{N}\sum_{i=1}^{N}|x_i - y_i|$ (Mean Absolute Error).

3.5.3 Experimental Setup

Our generator is a UNet-like architecture containing a four-layer encoder and a four-layer decoder with skip connections. We use partial convolutional layers instead of convolutional layers. We set the number of branches $n_b = 2$ for aggregating information from different granularity, based on hyper-parameter tuning on the validation set. The first two layers in the coarse branch use larger filter sizes ($3 \times 7 \times 7$ and $3 \times 5 \times 5$), the other layers use $3 \times 3 \times 3$ filters. Both branches contain global attention modules after the last layer of decoder. Our discriminator contains five convolutional layers, with filter size $3 \times 4 \times 4$. We optimize the model using the Adam optimizer [121] with a learning rate of 0.005. Batch size is set to 16 for the CMAP and KDD CUP datasets, and 4 for the HadCRUT dataset. We set $\lambda_{\text{var}} = 0.1, \lambda_{\text{adv}} = 0.001$ based on hyper-parameter tuning on the validation set.

Table 3.2. Experimental results of ESC-GAN and compared methods for different missing data patterns on HadCRUT and CMAP. We mark the best results (in bold) and the second best.

Method	HadCRUT						CMAP					
	Scatter		Reg Cluster		Irr Cluster		Scatter		Reg Cluster		Irr Cluster	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Zero	1.0396	0.6638	0.8551	0.5983	0.7446	0.5879	1.0290	0.6830	1.2861	0.7620	1.2869	0.7390
Mean	0.9697	0.6375	0.7985	0.5744	0.6971	0.5677	1.0272	0.6804	1.3294	0.7548	1.2969	0.7364
sKNN	0.3756	0.3991	0.4645	0.4649	0.3791	0.4210	0.1120	0.1785	0.7159	0.4863	0.4888	0.3960
IDW	0.3524	0.3868	0.4440	0.4535	0.3596	0.4087	0.1042	0.1719	0.7036	0.4792	0.4658	0.3839
Kriging	0.9517	0.6308	0.7995	0.5767	0.6906	0.5703	0.8838	0.5863	0.9709	0.6279	1.1257	0.6545
MF	0.6181	0.5216	0.7669	0.5782	0.6111	0.5390	0.1721	0.2395	0.8583	0.5753	0.5942	0.4974
BTTF	0.5867	0.5225	0.6798	0.5553	0.5764	0.5332	0.2474	0.3137	0.9423	0.6237	0.5723	0.5154
ST-MVL	0.3648	0.3964	0.4710	0.4655	0.3581	0.4084	0.1162	0.1832	0.7202	0.4919	0.5039	0.4177
IGKNN	0.7214	0.5492	0.7212	0.5501	0.6405	0.5491	0.8474	0.6132	1.1710	0.7285	1.1254	0.7170
NAOMI	1.0391	0.6637	0.8550	0.5983	0.7442	0.5877	1.0288	0.6833	1.2859	0.7620	1.2863	0.7396
PConv	0.3908	0.4211	0.4759	0.4784	0.4122	0.4494	0.1008	0.1704	0.6469	0.4492	0.2969	0.3024
3DGated	0.3610	0.3907	0.4265	0.4454	0.3581	0.4066	0.1400	0.2071	0.5532	0.4381	0.2952	0.3033
ESC-GAN	**0.3354**	**0.3800**	**0.4097**	**0.4295**	**0.3418**	**0.3971**	**0.0802**	**0.1531**	**0.5441**	**0.4308**	**0.2739**	**0.3017**

3.5.4 Results and Analysis

Quantitative Results. We evaluate our model on HadCRUT and CMAP and report on the results in Table 3.2. On both datasets, ESC-GAN consistently outperforms all the baselines.

Classical zero filling and mean filling have high errors by both metrics, as they do not consider any neighborhood information. sKNN and IDW take into account local information based purely on distance weighting, and their performances degenerate under clustered missing patterns, where neighboring locations are simultaneously missing. Kriging and MF put strong assumptions on the input distribution, which may not be observed in real-world dataset.

For the state-of-the-art spatio-temporal imputation methods, BTTF assumes Gaussian spatial factor and does not sufficiently model spatial dependencies. ST-MVL is not fully data-driven and does not fully capture the underlying spatio-temporal correlations. IGKNN mainly leverages neighboring nodes for imputation without incorporating the global and multi-scale structure. NAOMI exploits the multi-resolution structure but heavily relies on temporal domain information. Although during training, locations that are randomly masked out in the temporal domain can well recover the missing values, it basically generates mean value of the dataset

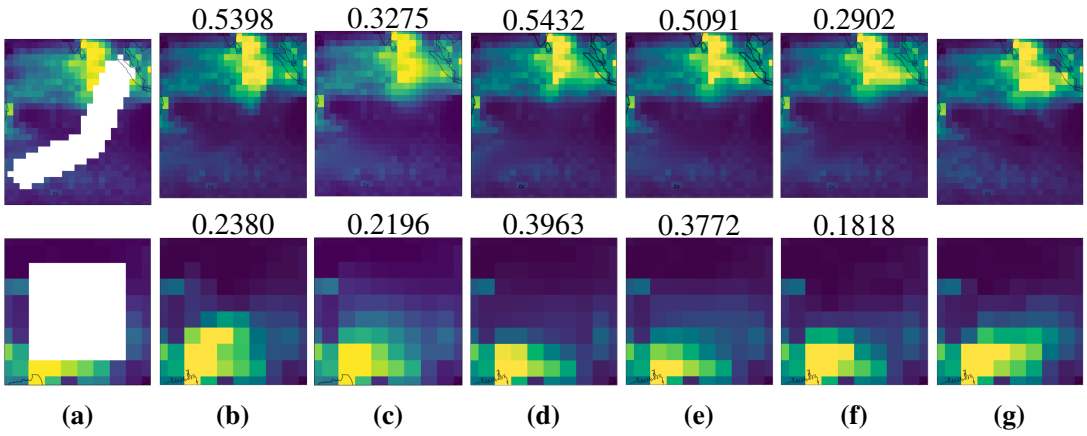


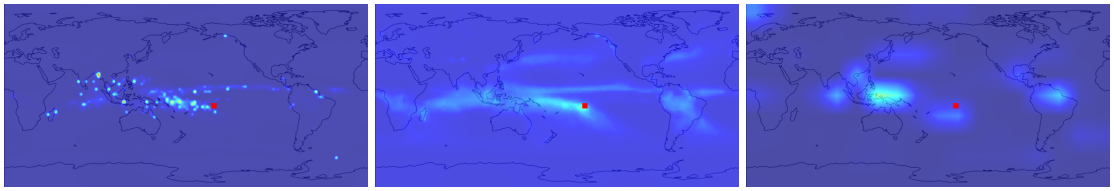
Figure 3.6. Case Study: Example grid maps for qualitative comparison. (a) Missing Area, (b) IDW, (c) ST-MVL, (d) PConv, (e) 3DGated, (f) ESC-GAN, and (g) GT. Numbers above the figures are the MSE for the corresponding generated grid maps compared with the GT grid map.

when extended to unobserved locations.

For image and video inpainting methods, Partial Convolution only uses local convolution operators to learn to recover missing information in two-dimensional space but does not leverage temporal information. 3D gated convolution combines temporal and spatial information in a three-dimensional space but does not model the global context information and the multi-scale structure of spatio-temporal data. By contrast, our ESC-GAN learns both local and global similarity and aggregates features at multiple scales in the three-dimensional spatio-temporal space. Therefore, ESC-GAN achieves the lowest errors compared to both classical and state-of-the-art imputation or inpainting methods.

Qualitative Results. In addition to quantitative improvements, in Figure 3.6 we present example grid maps generated by ESC-GAN for qualitative comparison. We zoom in on the area with missing data for better view. The numbers above the figures are the MSE for the corresponding grid map. The first row of results in Figure 3.6 illustrates imputation for irregular clustered locations, and the second row is for regular clustered locations. Compared with all the baselines, ESC-GAN generates values for the missing regions closest to the ground truth and shows smoother transition to the observed regions, qualitatively demonstrating its effectiveness.

3.5.5 Ablation Studies



(a) Dot Product GT **(b)** Cosine GT **(c)** ESC-GAN

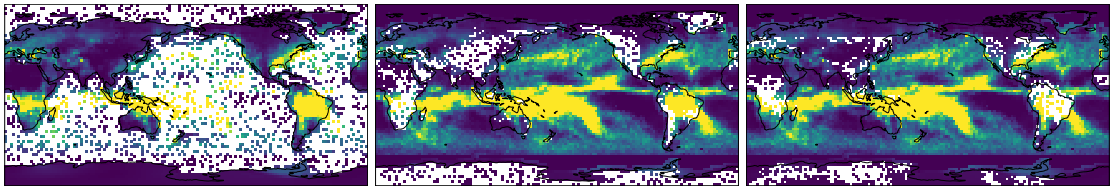
Figure 3.7. Ground-truth (GT) and generated attention maps.

Quantitative Analysis. We also conducted ablation study to separately examine the effect of our global attention module and multi-scale structure. We report MSE and MAE after removing global attention module (ESC-GAN-local), removing multi-scale structure (ESC-GAN-single), and removing both of these modules (ESC-GAN-vanilla). As shown in Table 3.3, the performance degenerates after removing either one or both of these components, which validates the necessity of the proposed structure. The vanilla model ESC-GAN-vanilla is based only on local operators of randomly masked 3D partial convolution, and it does not model the global context similarity or leverage information at multiple scales. The local model ESC-GAN-local aggregates features from multiple granularity, but ignores the underlying global patterns exhibited in spatio-temporal data. The single model ESC-GAN-single considers the global trends in spatio-temporal data, but it learns such features at one single scale using one branch in the generator. Our proposed ESC-GAN jointly learns global and local dependencies, and aggregates multi-level features, thus producing more accurate estimations compared with different ESC-GAN ablations.

Attention Visualization. To provide more interpretable results, we randomly select query region and visualize the softmax attention score between query region and all the other regions on CMAP, as shown in Figure 3.7. We mark the query regions with red rectangles. Attention scores in three figures are calculated based on the average data from all timestamps. The left two figures are ground-truth attention scores measured by dot product and cosine similarity, both followed by softmax normalization. As CMAP dataset measures monthly precipitation, the query region

Table 3.3. Ablation Study of our global attention and multi-scale structure on CMAP, measured by MSE and MAE.

Method	Scatter		Reg Cluster		Irr Cluster	
	MSE	MAE	MSE	MAE	MSE	MAE
ESC-GAN-vanilla	0.0825	0.1557	0.5874	0.4433	0.2900	0.3099
ESC-GAN-local	0.0818	0.1545	0.5848	0.4362	0.2785	0.3032
ESC-GAN-single	0.0842	0.1588	0.5814	0.4388	0.2880	0.3107
ESC-GAN	**0.0802**	**0.1531**	**0.5441**	**0.4308**	**0.2739**	**0.3017**



(a) Ocean **(b)** High-altitude **(c)** High-vegetation

Figure 3.8. Patterned missing data distributions

exhibits patterns similar to regions near the equator and regions in the Pacific Ocean. Comparing ESC-GAN generated attention map with the ground truth, we can observe that ESC-GAN is able to accurately capture the global patterns through the attention mechanism.

3.5.6 Robustness Studies

Robustness to Non-Random Missing Shapes. Apart from the missing distributions in Table 3.2, extending sensor spatial coverage in real-world also encounters non-random realistic missing distributions, i.e. missing distribution follows a specific pattern as a result of land, elevation, vegetation, etc. For example, it is more difficult to deploy sensors on mountains than plains, so locations of higher elevation are expected to have a lower coverage of sensory data. Similar comparison also resides in ocean vs land, forests vs locations with lower vegetation cover rate. In light of this, we study the effectiveness of ESC-GAN with respect to three non-random missing data distributions (as shown in Figure 3.8), namely, missing data in the ocean, high-altitude area, and area with high vegetation cover. This also evaluates the model's transferability, as there exists a distribution gap between training regions and testing regions. We report on the results

Table 3.4. Evaluations for patterned missing distributions.

Method	Ocean		High-altitude		High-vegetation	
	MSE	MAE	MSE	MAE	MSE	MAE
sKNN	0.3162	0.3160	0.0865	0.1428	0.1857	0.2178
IDW	0.2738	0.2929	0.0786	0.1335	0.1687	0.2060
ST-MVL	0.2855	0.2972	0.0784	0.1347	0.1710	0.2061
PConv	0.2174	0.2517	0.0743	0.1303	0.1588	0.2004
3DGated	0.2989	0.3093	0.1231	0.1878	0.2254	0.2644
ESC-GAN	**0.1929**	**0.2512**	**0.0663**	**0.1234**	**0.1399**	**0.1911**

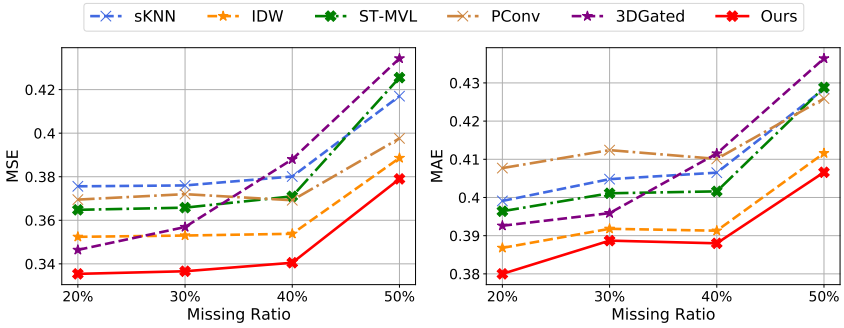


Figure 3.9. MSE and MSE for different missing ratio $|D_U|/|D|$ on HadCRUT.

in Table 3.4. ESC-GAN achieves the best performance under different real-world non-random realistic missing distributions.

Robustness to Amount of Missing Data. We also study model robustness with respect to missing area size. Following previous notation, we use D to represent the whole map and D_U to represent the set of unobserved grid cells. For scattered missing distribution in Section 3.5.4, the missing ratio $|D_U|/|D|$ is 20%. We increase the missing ratio $|D_U|/|D|$ from 20% to 50%, and calculate the corresponding MSE and MAE of the best performing baselines in Figure 3.9. As shown in the figure, for different models, both MSE and MAE generally grow as the missing ratio increases. Moreover, under settings of all varying missing ratios, ESC-GAN is able to outperform all the baselines for both MSE and MAE, demonstrating the proposed model's robustness to varying missing area size.

Table 3.5. Results of spatio-temporal imputation for random missing values on KDD 2018 Dataset, measured by MSE[5].

%Missing	20%	30%	40%	50%	60%	70%	80%	90%
Last	1.073	0.894	0.901	0.990	1.040	1.236	1.689	2.870
Mean	0.916	0.907	0.914	0.923	0.973	0.935	0.937	1.002
KNN	0.892	0.803	0.776	0.798	0.856	0.852	0.873	1.243
MF	0.850	0.785	0.787	0.772	0.834	0.805	0.860	1.196
MTSI	0.844	0.780	0.753	0.743	0.803	0.780	0.837	1.018
BRITS	0.455	0.421	0.372	0.409	0.440	0.482	0.648	0.725
DCRNN	0.579	0.565	0.449	0.506	0.589	0.622	0.720	0.861
CDSA	0.373	0.393	0.287	0.291	0.387	0.495	0.521	0.631
ESC-GAN	**0.207**	**0.229**	**0.232**	**0.231**	**0.274**	**0.299**	**0.326**	**0.434**

3.5.7 Generalization to Traditional Spatio-Temporal Imputation

In addition to our proposed extension of the spatial coverage task, we also apply our model to the traditional spatio-temporal imputation task for random missing values, to evaluate its generalizability. For this, we conduct experiments on the KDD CUP 2018 dataset. We normalize the data using z-normalization and split the sequences into chunks of length 48, following previous studies [165, 168]. We compare the results with a list of methods for doing traditional spatio-temporal imputation, including both statistical imputation methods (filling with last available observation (Last) or mean value (Mean), k-Nearest Neighbors (KNN), Matrix Factorization (MF)) and deep learning based models (MTSI [165], BRITS [28], DCRNN [142], CDSA [168]) following previous studies [165, 168].

In Table 3.5, our model outperforms all the other baselines at various missing data ratios from 20% to 90%. Compared with other methods, our model can jointly learn temporal and spatial dependencies at different scales. Without modification of the model structure, ESC-GAN is directly applicable to spatio-temporal imputation tasks. This demonstrates the generalizability of ESC-GAN, indicating its potential in a broader range of applications.

[5]We directly adopt the numbers for the compared methods from prior work [165, 168].

3.6 Summary

In this chapter, we present how to impute missing data in our robust framework by leveraging spatio-temporal correlations. We address the challenge of extending the spatial coverage (ESC) of sensory data to locations without any historical values. Traditional spatio-temporal imputation methods do not work well as they rely on partial data availability for a location. This ESC task has far-reaching applications in geographical discovery, physical modeling, weather forecasting, urban planning, etc. It faces challenges related to collaborative use of spatial and temporal domains, local and global context and structure across multiple scales. To address these challenges, we devise a model to recover data for "new" locations leveraging both spatial and temporal dependencies. In view of the non-linear, multi-resolution and stochastic nature of spatio-temporal data, our method generates data considering both the global and local perspectives. We optimize the model with multi-scale and adversarial training to better capture the underlying patterns. We evaluate ESC-GAN on real-world geo-sensory datasets where our model outperforms all the baselines under different missing scenarios.

There are limitations that we plan to address in future studies. As geometric distance on a sphere is not strictly preserved after being mapped to a 2D gridded map, we plan to incorporate spherical convolutions to better model spatial dependencies. We will also explore approaches to extend the model to irregular super-resolution task for generating data at finer spatial granularity.

Chapter 3 incorporates material from the publication "ESC-GAN: Extending Spatial Coverage of Physical Sensors", by Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, and Dezhi Hong, published in Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM 2022). The dissertation author was primary investigator and the lead author of this paper.

Chapter 4

Physical Context of Time and Frequency Dependencies

Apart from noise and missing values, another problem with sensor time series is that we do not have sufficient data. In this chapter, we present how to augment sensor time series given insufficient data by leveraging physical context of time and frequency dependencies, which is the third step in our robust sensor time series analysis framework.

Time series data augmentation mitigates the issue of insufficient training data for deep learning models. Yet, existing augmentation methods are mainly designed for classification, where class labels can be preserved even if augmentation alters the temporal dynamics. We note that augmentation designed for forecasting requires *diversity* as well as *coherence* with the original temporal dynamics. As time series data generated by real-life physical processes exhibit characteristics in both the time and frequency domains, we propose to combine Spectral and Time Augmentation (STAug) for generating more diverse and coherent samples. Specifically, in the frequency domain, we use the Empirical Mode Decomposition to decompose a time series and reassemble the subcomponents with random weights. This way, we generate diverse samples while being coherent with the original temporal relationships as they contain the same set of base components. In the time domain, we adapt a mix-up strategy that generates diverse as well as linearly in-between coherent samples. Experiments on five real-world time series datasets demonstrate that STAug outperforms the base models without data augmentation as well as

state-of-the-art augmentation methods.

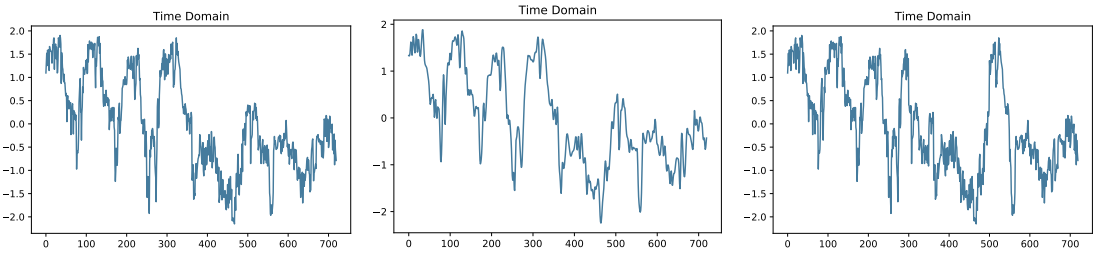
4.1 Introduction to Augmentation for Sensor Time Series Forecasting

Deep learning has been successful in various time series applications given enormous amount of data to train. However, time series data collected through real-world sensors is often marked by irregular samples with missing values due to collection difficulties. Such data *scarcity* commonly observed in time series data can significantly degrade the performance of deep learning methods that would otherwise perform well.

A rich line of research tries to address this problem through data augmentation, that is to generate synthetic data points to augment the original dataset [244, 126, 19, 71, 65, 33, 226, 143, 294, 185, 101, 113]. However, existing augmentation methods are mainly designed for classification, where augmented samples remain effective as long as they preserve the class labels. We note that augmentation designed for forecasting requires both *diversity* and *coherence* with the original temporal dynamics. Yet, existing augmentation methods generate samples that often miss one of the criteria (Figure 4.1). For example, filtering-based methods are deterministic processes that produce a fixed set of synthetic samples by removing noises. Permutation-based methods change the temporal order of the original series, worsening forecasting performance.

Moreover, time series data generated by real-life physical processes exhibit characteristics both in the time and frequency domains that are not available in other data modalities like image and text. Therefore, temporal dynamics can be best captured through a joint consideration of time domain that carries changes over time and frequency domain that conveys periodic patterns. By contrast, existing augmentation methods mostly generate data in one domain, ignoring the complementary strengths of both domains.

We propose STAug (Figure 4.2), by combining Spectral and Time Augmentation for time series forecasting task. In the frequency domain, we first apply the Empirical Mode Decomposi-



(a) Original **(b)** Filtering **(c)** Permutation

Figure 4.1. Visualization of original and augmented time series from ETTm2. Augmentation methods often (b) miss diversity or (c) miss coherence with the original temporal dynamics.

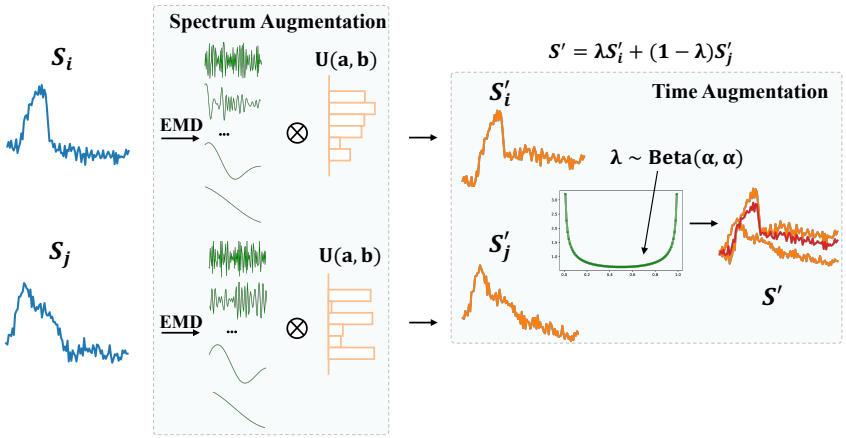


Figure 4.2. Overview of STAug. For frequency-domain augmentation, we decompose two random series $\mathbf{S_i}, \mathbf{S_j}$ with EMD, and then reassemble the subcomponents with random weights to obtain $\mathbf{S'_i}, \mathbf{S'_j}$. In the time domain, we further linearly mix $\mathbf{S'_i}, \mathbf{S'_j}$ to obtain the augmented series $\mathbf{S'}$.

tion (EMD) [102] to decompose time series into multiple subcomponents, each representing a certain pattern embedded in the data. We then reassemble these subcomponents with random weights to generate new synthetic series. This offers a *principled* way of augmentation as it generates diverse samples while maintaining the same basic set of subcomponents. We adopt EMD for the frequency information as it better captures patterns for non-stationary time series compared with Fourier transform. In the time domain, we adapt a mix-up strategy [307] to learn linearly in-between randomly sampled pairs of training series, which produces varied and coherent samples. We evaluate STAug on five real-world time series datasets, and the method demonstrates state-of-the-art performance compared with existing augmentation methods.

4.2 Related Work

Existing time series data augmentation techniques mainly fall into four categories [266, 106] as follows.

Basic Random Operations. This include time-domain, frequency-domain and time-frequency domain transformations. Time-domain transformation contains scaling, rotation, jittering [244], slicing, warping [126, 170], etc. For frequency-domain transformation, Robust-TAD [75] in the frequency domain makes perturbations in both magnitude and phase spectra. For time-frequency domain transformation, time-frequency features are generated from Short Fourier Transform (STFT), and then local averaging together with feature vector shuffling are applied for augmentation [226]. SpecAugment [193] proposes augmentation in Mel-Frequency by combining warping, masking frequency channels and masking timestep blocks together.

Decomposition-Based Augmentation. This leverages the Seasonal-Trend Decomposition (STL) [52] or Empirical Mode Decomposition (EMD) [102] to extract patterns for generating synthetic samples. Bagging Exponential Smoothing method [19, 117, 16] uses Box–Cox transformation followed by STL decomposition, and bootstraps the reminder to assemble new series. STL decomposition components can also be adjusted and combined with a stochastic component generated by statistical models [117, 16]. Nam et al. [185] decompose series using EMD into components from high frequency to low frequency, and adds the residue each time one IMF occurs. This can be viewed as a special case of STAug with weights equal to one for low-frequency components and zero for high-frequency components, essentially only filtering out high-frequency noise. Moreover, it does not benefit from time-domain augmentation.

Pattern Mixing Method. DBA calculates weighted average of multiple time series under DTW as new samples [71, 16]. Mix-up [307] constructs new examples through linear interpolation of features and labels, but such interpolation methods mainly focus on classification.

Generative Method. This models underlying distribution of the dataset for generation, including both statistical generative models [27, 113] and deep generative adversarial network

(GAN) [82] based models [65, 33, 205, 143, 294, 101].

In this chapter, we combine decomposition-based augmentation method in the frequency domain and pattern mixing augmentation method in the time domain to find diverse samples that preserve the original data characteristics.

4.3 Methodology

4.3.1 Overview

We focus on multivariate time series forecasting task. A multivariate time series sequence of length T and feature number c is denoted as $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_t, \ldots, \mathbf{s}_T] \in \mathbb{R}^{c \times T}$, where $\mathbf{s}_t = [s_1, \cdots, s_c]^T \in \mathbb{R}^c$. In a forecasting task, we only observe history values $\mathbf{H}$ up to timestamp $d < T$: $\mathbf{H} = [\mathbf{s}_1, \ldots, \mathbf{s}_d] \in \mathbb{R}^{c \times d}$, and the goal is to forecast future values $\mathbf{F}$ at timestamp $d+1, \ldots, T$: $\mathbf{F} = [\mathbf{s}_{d+1}, \ldots, \mathbf{s}_T] \in \mathbb{R}^{c \times (T-d)}$, where $\mathbf{S} = [\mathbf{H}, \mathbf{F}]$. During training, we have full access to both $\mathbf{H}$ and $\mathbf{F}$, and the training objective is to learn a model g that forecasts $\mathbf{F}$ given $\mathbf{H}$ for each $\mathbf{S}$ in the training set. During testing, we have access to *only* $\mathbf{H}$ and input $\mathbf{H}$ to model g to obtain predictions for the future part.

Our method STAug comprises augmentation in both the frequency domain and time domain. In the frequency domain, we first apply empirical mode decomposition to obtain a set of components. Then during each iteration, these components are re-combined with random weights to construct a new synthetic series. Then, we adapt mix-up as a time-domain augmentation. We linearly interpolate two randomly re-combined series to obtain the final augmented series. The augmented series are fed into the forecasting model for updating gradient. The EMD components of different series can be pre-computed, and STAug only requires randomly re-combining components or series during training, which introduces minimal computational overhead.

4.3.2 Frequency-Domain Augmentation

The Empirical Mode Decomposition (EMD) [102] method was originally designed to analyze nonlinear and non-stationary data, whose constituent components are not necessarily cyclic. EMD preserves temporal information in contrast with Fourier transform, and is data-driven compared with the linear wavelet transform. It decomposes a signal into a finite number of Intrinsic Mode Functions (IMF). The first several IMFs usually carry components of higher frequency (e.g., noise), while the last several IMFs represent the low-frequency trend information embedded in the sequence. Therefore, EMD provides a principled way to decompose a signal into multiple components, and each of these components represents certain patterns embedded in the original signal.

After EMD, the original sequence can be written as

$$\mathbf{S} = \sum_{i=1}^{n} \mathrm{IMF}_i + \mathbf{R}. \tag{4.1}$$

With a list of n decomposed IMFs $\{\mathrm{IMF}_1, \ldots, \mathrm{IMF}_n\}$ and residual $\mathbf{R}$, we apply a random vector $\mathbf{w} = [w_1, \ldots, w_n]^T$ as weights to re-combine these IMFs as $\mathbf{S}'$:

$$\mathbf{S}' = \sum_{i=1}^{n} w_i \cdot \mathrm{IMF}_i \tag{4.2}$$

where w_i is sampled from uniform distribution $\mathscr{U}(0,2)$. This way, the augmented samples are diverse by emphasizing different frequency components via random weights, and at the same time coherent with original distributions as they contain the same basic sets of components.

4.3.3 Time-Domain Augmentation

Complementing the frequency-domain information, time domain also provides useful patterns. Therefore, we propose to mix up sequences in the time domain, inspired by Mix-up augmentation [307]. Mix-up was originally designed for classification, and we adapt it to time series forecasting by mixing up values at both past timestamps $1, \ldots, d$ and future timestamps

$d+1,\ldots,T$. Assume $\mathbf{S_i} = [\mathbf{H_i}, \mathbf{F_i}]$ and $\mathbf{S_j} = [\mathbf{H_j}, \mathbf{F_j}]$ are two randomly sampled sequences after spectral augmentation, where $\mathbf{H_i} = [\mathbf{s_i^1}, \ldots, \mathbf{s_i^d}], \mathbf{F_i} = [\mathbf{s_i^{d+1}}, \ldots, \mathbf{s_i^T}]$ respectively represents past and future data, similarly for $\mathbf{S_j}$. We construct new sequence as $\mathbf{S'} = [\mathbf{H'}, \mathbf{F'}]$, where

$$\mathbf{H'} = \lambda \mathbf{H_i} + (1-\lambda)\mathbf{H_j}, \tag{4.3}$$

$$\mathbf{F'} = \lambda \mathbf{F_i} + (1-\lambda)\mathbf{F_j}, \tag{4.4}$$

where λ is sampled from a Beta distribution, i.e., $\lambda \sim \text{Beta}(\alpha, \alpha)$, and α is the hyper-parameter that controls how similar the newly constructed sequence is compared with the original sequences $\mathbf{S_i}$ and $\mathbf{S_j}$. Mix-up augments patterns in the time domain meanwhile by its interpolation nature generates only linearly in-between coherent samples.

4.3.4 Time Series Forecasting

For each training iteration, we apply both frequency-domain and time-domain augmentation to obtain an augmented series $\mathbf{S'} = [\mathbf{H'}, \mathbf{F'}]$. During training, we feed the augmented series history $\mathbf{H'}$ as input, and optimize the forecasting model by reconstructing future part of the series $\mathbf{F'}$. In our experiments, we adopt the state-of-the-art forecasting model Informer [327] (AAAI 2021 best paper) as the base forecaster. To minimize the reconstruction loss ℓ, we calculate Mean Square Error (MSE) between the forecasting model output $\mathbf{Y}$ and the ground-truth future part $\mathbf{F'}$:

$$\ell = \frac{1}{N}\sum_{i=1}^{N} ||\mathbf{Y_i} - \mathbf{F'_i}||_2^2, \tag{4.5}$$

where N is the number of augmented series in training set.

Table 4.1. MSE and MAE on benchmark datasets with input context length 96 and forecasting horizon $\{96, 192, 336, 720\}$. We **bold** the best performing results, underline the second best, and mark with dashline the best baseline.

Methods		None		WW		RobustTAD		STL		EMD-R		GAN		DBA		STAug-noTime		STAug-noFreq		STAug	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.947	0.760	0.894	0.730	1.011	0.789	0.910	0.738	0.838	0.686	0.904	0.734	0.943	0.759	0.658	0.573	0.841	0.693	**0.645**	**0.572**
	192	0.977	0.767	0.999	0.775	0.946	0.744	0.942	0.740	0.942	0.729	0.985	0.757	0.977	0.767	**0.754**	**0.616**	0.962	0.755	0.772	0.628
	336	1.112	0.833	1.058	0.808	1.094	0.816	1.086	0.814	1.056	0.804	1.070	0.816	1.111	0.832	**0.860**	**0.681**	1.103	0.829	0.889	0.705
	720	1.182	0.864	1.201	0.884	1.146	0.841	1.187	0.863	1.187	0.866	1.221	0.892	1.184	0.866	0.994	0.752	1.162	0.857	**0.972**	**0.746**
ETTh2	96	3.084	1.383	3.399	1.463	3.028	1.380	3.042	1.374	3.440	1.451	2.906	1.377	3.005	1.370	**1.802**	**1.012**	2.984	1.371	1.869	1.046
	192	5.966	2.023	6.432	2.098	6.203	2.083	5.894	2.023	6.326	2.089	5.467	1.939	5.636	1.939	4.023	1.569	5.229	1.886	**3.467**	**1.483**
	336	4.775	1.829	4.965	1.852	5.287	1.883	4.774	1.833	5.436	1.948	4.906	1.868	4.765	1.829	3.391	**1.482**	4.374	1.736	**3.365**	1.513
	720	3.991	1.694	3.828	1.677	3.896	1.659	3.563	1.579	4.082	1.717	4.060	1.744	4.009	1.698	2.700	1.377	3.571	1.619	**2.621**	**1.340**
ETTm1	96	0.633	0.570	0.629	0.570	0.593	0.546	0.549	0.534	0.640	0.564	0.626	0.576	0.634	0.573	0.428	0.439	0.550	0.525	**0.403**	**0.424**
	192	0.797	0.673	0.792	0.665	0.802	0.663	0.738	0.646	0.682	0.590	0.842	0.689	0.803	0.675	0.538	0.505	0.668	0.591	**0.489**	**0.482**
	336	1.149	0.839	0.997	0.774	0.983	0.759	0.992	0.773	0.869	0.695	0.902	0.725	1.146	0.838	0.648	0.567	0.900	0.715	**0.627**	**0.559**
	720	1.128	0.805	1.077	0.786	1.063	0.789	1.184	0.820	1.008	0.748	1.225	0.862	1.131	0.806	0.828	0.657	0.934	0.720	**0.764**	**0.628**
ETTm2	96	0.415	0.506	0.385	0.476	0.386	0.476	0.388	0.489	0.398	0.478	0.334	0.430	0.415	0.506	0.297	0.394	0.328	0.423	**0.292**	**0.388**
	192	0.776	0.675	0.784	0.680	0.875	0.724	0.712	0.650	0.542	0.551	0.625	0.604	0.767	0.673	**0.420**	**0.490**	0.592	0.585	0.429	0.497
	336	1.515	0.946	1.425	0.923	1.390	0.909	1.402	0.908	1.287	0.875	1.453	0.929	1.568	0.962	0.943	0.735	1.287	0.876	**0.811**	**0.692**
	720	3.462	1.423	4.370	1.637	4.792	1.654	3.271	1.378	3.247	1.353	3.129	1.355	3.475	1.427	**2.645**	**1.201**	3.765	1.509	2.790	1.268
Exchange	96	0.959	0.784	0.799	0.687	0.962	0.788	0.860	0.746	0.923	0.761	0.942	0.772	0.958	0.784	0.285	0.402	0.809	0.708	**0.271**	**0.395**
	192	1.113	0.837	1.084	0.787	1.180	0.873	1.143	0.852	1.182	0.853	1.115	0.855	1.109	0.836	0.627	0.615	1.125	0.822	**0.606**	**0.605**
	336	1.605	1.009	1.697	0.995	1.562	1.006	1.548	0.987	1.606	1.001	1.465	0.972	1.595	1.007	1.007	0.807	1.438	0.944	**0.935**	**0.779**
	720	2.847	1.390	3.198	1.481	2.816	1.380	2.484	1.297	2.914	1.413	2.571	1.309	2.847	1.390	**1.409**	**0.928**	2.056	1.153	1.771	1.052

4.4 Evaluation

4.4.1 Datasets, Baselines and Experimental Setup

We evaluate our augmentation method STAug on five time series datasets: ETTh1, ETTh2, ETTm1, ETTm2, Exchange [327, 271]. We follow previous studies [327, 271] for 96 context length and $96, 192, 336, 720$ forecasting horizon. All the datasets in our experiments are multivariate, and we apply EMD and mix-up separately for each original variable, and concatenate them to obtain the augmented multivariate time series. We follow previous works for $7:2:1$ train, validation and test set split, and evaluate the performance with Mean Square Error (MSE) and Mean Absolute Error (MAE).

We use Informer [327] as the base forecasting model. α of the Beta distribution equals 0.5, which is chosen from a grid search of $\{0.25, 0.5, 0.75, 1.0\}$. a, b of the Uniform distribution $\mathscr{U}(a,b)$ are set to 0 and 2, respectively. We use Adam optimizer with a decaying learning rate starting from 0.0001. We compare STAug with base model without any augmentation (None), as well as state-of-the-art time series augmentation methods: WW [126], DBA [71, 68, 16], EMD-R [185], GAN [65], STL [19, 16], RobustTAD [75]. We conduct careful grid search for hyper-parameter tuning for each baseline. We repeat all the experiments for 3 runs and record both average performance and standard deviation.

4.4.2 Main Results

We evaluate STAug and baselines, and report the average performance in Table 4.1. For easier comparison, we also scale MSE (average and standard deviation) with respect to the average MSE of the base model without augmentation. Similarly, we scale MAE with respect to the average MAE of the base model without augmentation. We bold the best results, underline the second best, and mark with dash line the best baseline. STAug consistently outperforms baselines on different datasets by jointly leveraging time-domain and frequency-domain information, with an average reduction of 28.2% for MSE and 18.0% for MAE, compared with the best baseline

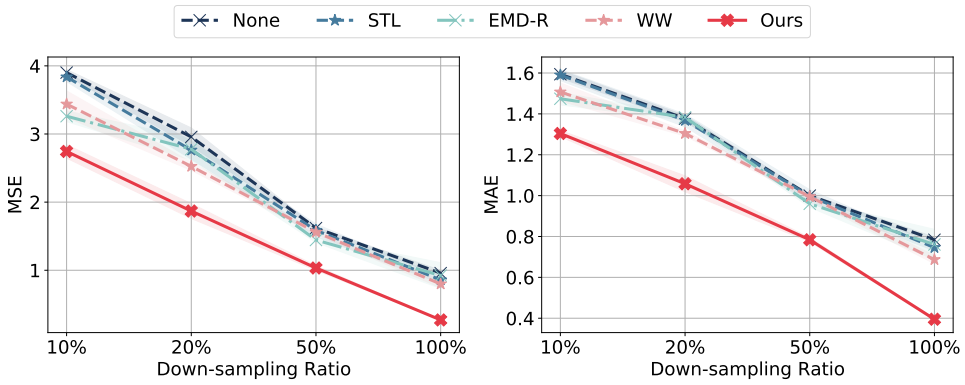
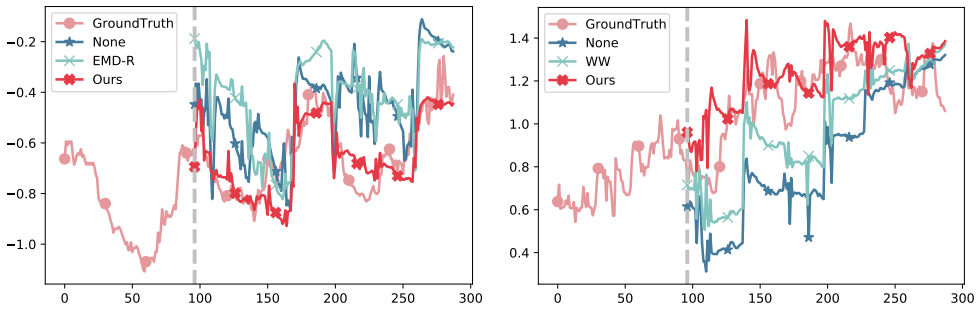


Figure 4.3. MSE and MAE for different down-sampling ratios.



(a) ETTm1 **(b)** Exchange

Figure 4.4. Predictions on ETTm1 and Exchange with 192 horizon. STAug predicts data that best align with the ground truth.

for each dataset. To evaluate the statistical significance, we also run the Friedman test and the Wilcoxon-signed rank test with Holm's α (5%) following previous work [134]. The Friedman test shows statistical significance $p = 0.00$ (much smaller than $\alpha = 0.05$), so there exists a significant difference among different methods. The Wilcoxon-signed rank test indicates the statistical significance of STAug compared with all the baselines with $p = 0.00$ far below 0.05. We also present qualitative comparisons in Figure 4.4. Compared with the predictions by the base model and the best-performing baseline for the corresponding setting, STAug forecasts values that better align with the original time series.

4.4.3 Ablation Study

To examine the effect of augmentation in different domains, we also conduct ablation study by augmenting data only in the frequency or time domain in Table 4.1. STAug-noFreq stands for the model that *removes* frequency-domain augmentation, and STAug-noTime stands for the model that *removes* time-domain augmentation. The performance degrades after removing augmentation in either domain, validating the need of combining both domains.

4.4.4 Robustness Study

We sub-sample 10%, 20%, 50% of the Exchange dataset, and compare STAug with the base model and the best-performing baseline methods for the corresponding setting (horizon 96). We report on MSE and MAE in Figure 4.3. The performances of different methods increase as we have more data. When the sample size is small, the performance gap between augmentation methods and base model without augmentation becomes larger, which shows that augmentation is especially helpful when the original dataset is small. STAug leverages information from both time and frequency domains, and performs consistently better with respect to the original sample size. Moreover, STAug shows more significant improvement over base model and the best-performing baselines with fewer available samples in the original dataset, which demonstrates its robustness with respect to the number of data samples.

4.5 Summary

This chapter presents using time and frequency dependencies for sensor time series augmentation. We propose a generic yet effective time series data augmentation method STAug to combine patterns in both the time domain and frequency domain. In the frequency domain, the re-combined subcompacts of the original time series are both diverse and preserve the original basic components. In the time domain, we adapt the mix-up strategy to generate diverse and in-between coherent samples by linearly interpolating both past and future part of a time series.

Experiments on five real-world time series datasets show that STAug best reduces the forecasting errors of base model compared with existing augmentation methods. We also perform robustness analysis and observe that STAug stays robust with respect to sampling size. Our ongoing work in the area involves systematically studying the effectiveness of various time domain and frequency domain augmentation methods, and designing data-dependent selection procedure to choose the most suitable augmentation method for different datasets.

Chapter 4 incorporates material from the publication "Towards Diverse and Coherent Augmentation for Time-Series Forecasting", by Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, Dezhi Hong, published in ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2023). The dissertation author was primary investigator and the lead author of this paper.

Chapter 5

Semantic Context of Large Language Models

The previous chapters focus on the first stage of our robust framework by incorporating physical context to denoise, impute and augment sensor time series and therefore refine the raw data quality. Additionally, the physical contextual knowledge can be applied to other tasks, such as anomaly detection and data calibration, and physical models can be replaced or augmented by numerical simulation models, which we leave as future work.

However, physics principles may not always be known, especially for human-centered sensing applications. Therefore, we need to go beyond physics and incorporate semantic contextual knowledge from languages or our social behaviors. At this point in this dissertation, we turn from physics to sociological contextual data. By adopting semantic context, we can further improve robustness and generalization of sensor time series analysis methods. In order to enhance understanding of the literature, in this chapter, we first present a comprehensive taxonomy on how to incorporate the semantic context from the recently emerging Large Language Models (LLMs) to improve sensor time series analysis.

Large Language Models have seen significant use in domains such as natural language processing and computer vision. Going beyond text, image and graphics, LLMs present a significant potential for analysis of time series data, benefiting domains such as climate, IoT, healthcare, traffic, audio and finance. This chapter of survey provides an in-depth exploration

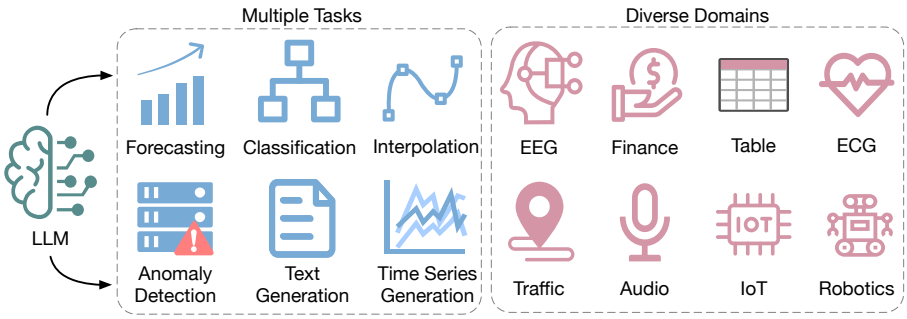


Figure 5.1. Large language models have recently been applied for various time series tasks in diverse application domains.

and a detailed taxonomy of the various methodologies employed to harness the power of LLMs for time series analysis. We address the inherent challenge of bridging the gap between LLMs' original text data training and the numerical nature of time series data, and explore strategies for transferring and distilling knowledge from LLMs to numerical time series analysis. We detail various methodologies, including (1) direct prompting of LLMs, (2) time series quantization, (3) aligning techniques, (4) utilization of the vision modality as a bridging mechanism, and (5) the combination of LLMs with tools. Additionally, this chapter of survey offers a comprehensive overview of the existing multimodal time series and text datasets and delves into the challenges and future research opportunities of this emerging field. We maintain an up-to-date Github repository[1] which includes all the papers and datasets discussed in the survey.

5.1 Introduction to Large Language Models for Time Series

In recent years, Large Language Models (LLMs) have gained substantial attention particularly in the fields of Natural Language Processing (NLP) and Computer Vision (CV). Prominent models such as GPT-4 [190] have transformed the landscape of text processing by offering unprecedented accuracy in tasks such as text generation, translation, sentiment analysis, question answering and summarization. In the CV domain, Large Multimodal Models (LMMs) have also facilitated advancements in image recognition, object detection, and generative tasks,

[1] https://github.com/xiyuanzh/awesome-llm-time-series

leading to more intelligent and capable visual systems [223]. Inspired by these successes, researchers are now exploring the potential of LLMs in the realm of time series analysis, expecting further breakthroughs, as shown in Figure 5.1. While several surveys offer a broad perspective on large models for time series in general [110, 169], these do not specifically focus on LLMs or the key challenge of bridging modality gap, which stems from LLMs being originally trained on discrete textual data, in contrast to the continuous numerical nature of time series.

We start with a survey to build a taxonomy of sensory data analysis using semantic contextual knowledge. Our survey uniquely contributes to the existing literature by emphasizing how to bridge such modality gap and transfer knowledge from LLMs for time series analysis. Our survey also covers more diverse application domains, ranging from climate, Internet of Things (IoT), to healthcare, traffic management, and finance. Moreover, certain intrinsic properties of time series, like continuity, auto-regressiveness, and dependency on the sampling rate, are also shared by audio, speech, and music data. Therefore, we also present representative LLM-based works from these domains to explore how we can use LLMs for other types of time series. We present a comprehensive taxonomy by categorizing these methodologies into five distinct groups, as shown in Figure 5.2. If we outline typical LLM-driven NLP pipelines in five stages - input text, tokenization, embedding, LLM, output - then each category of our taxonomy targets one specific stage in this pipeline. Specifically, (i) *Prompting* (input stage) treats time series data as raw text and directly prompts LLMs with time series; (ii) *Time Series Quantization* (tokenization stage) discretizes time series as special tokens for LLMs to process; (iii) *Aligning* (embedding stage) designs time series encoder to align time series embeddings with language space; (iv) *Vision as Bridge* (LLM stage) connects time series with Vision-Language Models (VLM) by employing visual representations as a bridge; (v) *Tool Integration* (output stage) adopts language models to output tools to benefit time series analysis. Beyond this taxonomy, our survey also compiles an extensive list of existing multimodal datasets that incorporate both time series and text. We conclude this chapter by discussing future directions in this promising field.

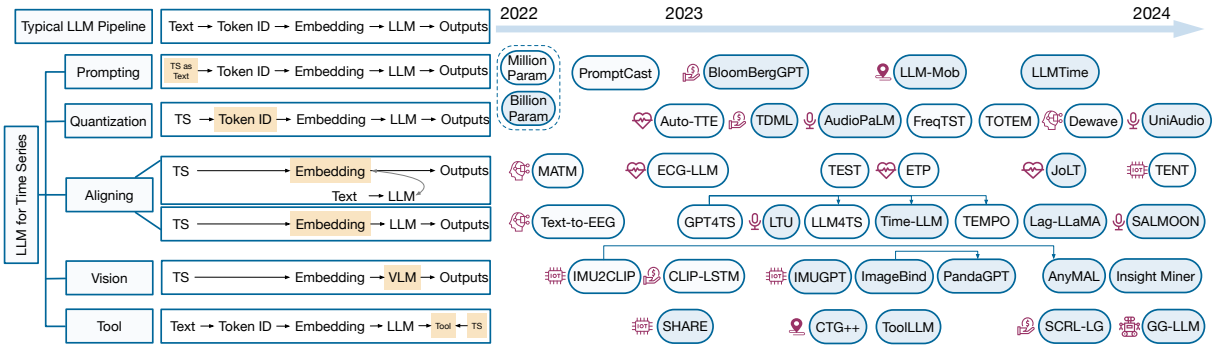


Figure 5.2. Left: Taxonomy of LLMs for time series analysis. For each category, key distinctions are drawn compared to the standard LLM pipeline at the top. Right: Representative works for each category, sorted by their publication dates. The use of arrows indicates that later works build upon earlier studies.

5.2 Preliminary

Large language models are characterized by their vast number of parameters and extensive training data. They excel in understanding, generating, and interpreting human language and recently represent a significant advancement in artificial intelligence. The inception of LLMs can be traced back to models like GPT-2 [201], BERT [60], BART [131], and T5 [203], which laid the foundational architecture. Over time, the evolution of these models has been marked by increasing complexity and capabilities, such as LLAMA-2 [242], PaLM [48], and GPT-4. More recently, researchers have developed multimodal large language models to integrate and interpret multiple forms of data, such as text, images, and time series, to achieve a more comprehensive understanding of information.

This chapter of survey focuses on how LLMs can benefit time series analysis. We first define the mathematical formulation for the input and output, which may contain time series or (and) text depending on the downstream tasks, as well as the models.

Input: denoted as $\mathbf{x}$, composed of time series $\mathbf{x}_s \in \mathbb{R}^{T \times c}$ and optional text data $\mathbf{x}_t$ represented as strings, where T, c represent the sequence length and the number of features.

Output: denoted as $\mathbf{y}$ and may represent time series, text or numbers depending on the specific downstream task. For time series generation or forecasting task, $\mathbf{y}$ represents generated

Table 5.1. Examples of representative direct prompting methods.

Method	Example
PromptCast [282]	"From $\{t_1\}$ to $\{t_{\text{obs}}\}$, the average temperature of region $\{U_m\}$ was $\{x_t^m\}$ degree on each day. What is the temperature going to be on $\{t_{\text{obs}}\}$?"
[156]	"Classify the following accelerometer data in meters per second squared as either walking or running: 0.052,0.052,0.052,0.051,0.052,0.055,0.051,0.056,0.06,0.064"
TabLLM [94]	"The person is 42 years old and has a Master's degree. She gained $594. Does this person earn more than 50000 dollars? Yes or no? Answer:"
LLMTime [86]	"0.123, 1.23, 12.3, 123.0" → "1 2 , 1 2 3 , 1 2 3 0 , 1 2 3 0 0"

time series $\mathbf{y}_s$ or predicted k-step future time series $\mathbf{y}_s^{T+1:T+k}$. For text generation task, such as report generation, $\mathbf{y}$ represents text data $\mathbf{y}_t$. For time series classification or regression task, $\mathbf{y}$ represents numbers indicating the predicted classes or numerical values.

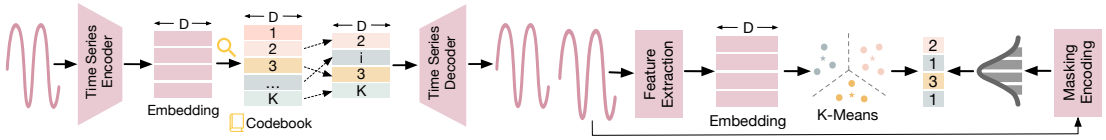
Model: We use f_θ parameterized by θ, g_ϕ parameterized by ϕ, and h_ψ parameterized by ψ to represent language, time series and vision models, where f_θ is typically initialized from pre-trained LLMs. We optimize parameters θ, ϕ and ψ through loss function ℓ.

5.3 Proposed Taxonomy for Incorporating Semantics in LLMs

In this section, we detail our taxonomy of applying LLMs for time series analysis, categorized by five groups. We summarize the representative works, mathematical formulation, advantages and limitations of each category in Table 5.2.

5.3.1 Prompting

Number-Agnostic Tokenization. The method treats numerical time series as raw textual data and directly prompts existing LLMs. For example, PromptCast [282] proposes prompt-based time series forecasting by converting numerical time series into text prompts and forecasting time series in a sentence-to-sentence manner. The input prompts are composed of context and questions following pre-defined templates. An illustrative prompt template for temperature



(a) VQ-VAE based quantization method. **(b)** K-Means based quantization method.

Figure 5.3. Two types of index-based quantization methods.

forecasting, along with examples from other representative works, are showcased in Table 5.1. Similar prompting methods have been applied to forecast Place-of-Interest (POI) customer flows (AuxMobLCast [284]), energy load [283], and user's next location (LLM-Mob [257]). [156] prompt PaLM-24B for health-related tasks such as activity recognition and daily stress estimate. TabLLM [94] prompts large language models with a serialization of the tabular data to a natural-language string for few-shot and zero-shot tabular data classification. [324] prompt large language models to detect anomalous behaviors from mobility data. [277] extract historical price features such as open, close, high, and low prices to prompt ChatGPT in a zero-shot fashion.

Number-Specific Tokenization. More recently, LLMTime [86] pointed out that Byte Pair Encoding (BPE) tokenization has the limitation of breaking a single number into tokens that don't align with the digits, leading to inconsistent tokenization across different floating point numbers and complicating arithmetic operations [224]. Therefore, following LLMs such as LLaMA and PaLM, they propose to insert spaces between digits to ensure distinct tokenization of each digit and use a comma (",") to separate each timestep in a time series. They also scale time series to optimize token usage and keep fixed precision (e.g., two digits of precision) to efficiently manage context length. Meanwhile, BloomberGPT [272] trains on financial data with text and numerical data and places each digit in its own chunk to better handle numbers. Using similar space-prefixed tokenization, [176] show that LLMs are general pattern machines capable of sequence transformation, completion and improvement.

5.3.2 Quantization

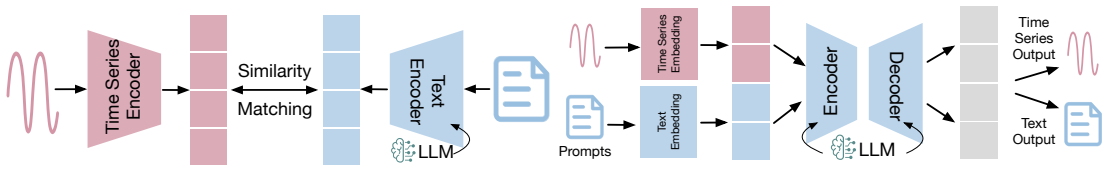
Quantization-based method [199] converts numerical data into discrete representations as input to LLMs. This approach can be further divided into two main categories based on the discretization technique employed.

Discrete Indices from VQ-VAE. The first type of quantization method transforms continuous time series into discrete indices as tokens. Among them one of the most popular methods is training a Vector Quantized-Variational AutoEncoder (VQ-VAE) [247], which learns a codebook $\mathscr{C} = \{\mathbf{c}_i\}_{i=1}^{K}$ of K D-dimensional codewords $\mathbf{c}_i \in \mathbb{R}^D$ to capture the latent representations, as illustrated in Figure 5.3a. The method identifies the nearest neighbor k_i of each step i of the encoded time series representation $g_\phi(\mathbf{x}_s) \in \mathbb{R}^{\frac{T}{S} \times D}$ in the codebook (S denotes the cumulative stride of VQ-VAE encoder), and uses the corresponding indices $\mathbf{k}$ as the quantized input to language models:

$$\mathbf{q}_i = \mathbf{c}_{k_i}, k_i = \arg\min_j \|g_\phi(\mathbf{x}_s)_i - \mathbf{c}_j\|_2, \mathbf{k} = [k_i]_{i=1}^{\frac{T}{S}}. \tag{5.1}$$

Based on VQ-VAE, Auto-TTE [51] quantizes ECGs and generates 12-lead ECG signals conditioned on text reports. DeWave [63] adapts VQ-VAE to derive discrete codex encoding and aligns it with pre-trained BART for open-vocabulary EEG-to-text translation tasks. TOTEM [234] also quantizes time series through VQ-VAE as input to Transformers for multiple downstream applications such as forecasting, classification, and translation. In the audio domain, UniAudio [288] tokenizes different types of target audio using Residual Vector Quantization (RVQ) [300] (a hierarchy of multiple vector quantizers) and supports 11 audio generation tasks. VioLA [254] unifies various crossmodal tasks involving speech and text by converting speech utterances to discrete tokens through RVQ. AudioGen [123] learns discrete audio representations using vector quantization layers and generates audio samples conditioned on text inputs.

Discrete Indices from K-Means. Apart from employing VQ-VAE, researchers have also explored K-Means clustering for index-based tokenization, which uses the centroid indices as discretized tokens [100], as shown in Figure 5.3b. Such methods are mostly applied in the audio



(a) Aligning by similarity matching. **(b)** Aligning with LLMs as backbones.

Figure 5.4. Type one (a) and Type two (b) of aligning-based methods. For Type two (b) aligning, the output can be time series (e.g., forecasting) or text (e.g., EEG-to-text) depending on the downstream tasks.

domain. For example, SpeechGPT [305] shows capability to perceive and generate multi-modal contents using K-Means based discrete unit extractor. AudioLM [22] discretizes codes produced by a neural audio codec using K-means clustering to achieve high-quality synthesis. It also combines discretized activations of language models pre-trained on audio using RVQ to capture long-term structure. Following the same quantization, AudioPaLM [211] fuses PaLM-2 [6] and AudioLM with a joint vocabulary that can represent speech and text with discrete tokens.

Discrete Indices from Other Techniques. Apart from the aforementioned time-domain quantization, FreqTST [136] utilizes frequency spectrum as a common dictionary to discretize time series into frequency units with weights for downstream forecasting task.

Text Categories. The second type of quantization converts numerical data into pre-defined text categories, which is primarily adopted in financial domain. As an example, TDML [297] categorizes the weekly price fluctuations into 12 bins represented as "Di" or "Ui", where "D" indicates a decrease in price and "U" means an increase, and $i = 1, 2, 3, 4, 5, 5+$ represents the level of price change.

5.3.3 Aligning

The third type of works trains a separate encoder for time series, and aligns the encoded time series to the semantic space of language models. These works can be further categorized into two groups based on their specific aligning strategies, as illustrated in Figure 5.4.

Similarity Matching through Contrastive Loss. The first type of method aligns the

time series embeddings with text embeddings through similarity matching, such as minimizing the contrastive loss:

$$\ell = -\frac{1}{B}\sum_{i=1}^{B}\log\frac{\exp(\text{sim}(g_\phi(\mathbf{x}_{si}), f_\theta(\mathbf{x}_{ti})))^{\frac{1}{\gamma}}}{\sum_{k=1}^{B}\exp(\text{sim}(g_\phi(\mathbf{x}_{si}), f_\theta(\mathbf{x}_{tk})))^{\frac{1}{\gamma}}}, \tag{5.2}$$

where B, γ represent batch size and temperature parameter that controls distribution concentrations, and sim represents similarity score, typically computed as inner product:

$$\text{sim}(g_\phi(\mathbf{x}_{si}), f_\theta(\mathbf{x}_{ti})) = \langle g_\phi(\mathbf{x}_{si}), f_\theta(\mathbf{x}_{ti})\rangle. \tag{5.3}$$

For instance, ETP [145, 135] integrates contrastive learning based pre-training to align electrocardiography (ECG) signals with textual reports. [120] use similar contrastive framework to align 17 clinical measurements collected in Intensive Care Unit (ICU) to their corresponding clinical notes. TEST [228] uses contrastive learning to generate instance-wise, feature-wise, and text-prototype-aligned time series embeddings to align with text embeddings. TENT [329] aligns text embeddings with IoT sensor signals through a unified semantic feature space using contrastive learning. JoLT [25] utilizes Querying Transformer (Q-Former) [137] optimized with contrastive loss to align the time series and text representations.

Similarity Matching through Other Losses. Apart from contrastive loss, other loss functions are also employed to optimize similarity matching between time series embeddings and text embeddings. ECG-LLM [198] aligns the distribution between ECG and language embedding from ECG statements with an Optimal Transport based loss function to train an ECG report generation model. MTAM [90] uses various aligning techniques, such as Canonical Correlation Analysis and Wasserstein Distance, as loss functions to align electroencephalography (EEG) features with their corresponding language descriptions.

LLMs as Backbones. The second type of aligning method directly uses large language models as backbones following time series embedding layers. EEG-to-Text [259] feeds EEG

embeddings to pre-trained BART for open vocabulary EEG-To-Text decoding and EEG-based sentiment classification. GPT4TS [328] uses patching embeddings [186] as input to frozen pre-trained GPT-2 where the positional embedding layers and self-attention blocks are retained during time series fine-tuning. The method provides a unified framework for seven time series tasks, including few-shot or zero-shot learning. Following GPT4TS, researchers further incorporated seasonal-trend decomposition (TEMPO [26]), two-stage fine-tuning (LLM4TS [31]), domain descriptions (UniTime [157]), graph attention mechanism (GATGPT [45]), and spatial-temporal embedding module (ST-LLM [146]). Time-LLM [109] reprograms time series data into text prototypes as input to LLaMA-7B. It also provides natural language prompts such as domain expert knowledge and task instructions to augment input context. Lag-Llama [206] builds univariate probabilistic time series forecasting model based on LLaMA architecture. In the audio, speech and music domains, researchers have also designed dedicated encoders to embed speech (WavPrompt [74], Speech LLaMA [125])), music (MU-LLaMA [152]), and general audio inputs (LTU [81], SALMONN [236]), and feed the embeddings to large language models.

5.3.4 Vision as Bridge

Time series data can be effectively interpreted or associated with visual representations, which align closer with textual data and have demonstrated successful integrations with large language models. Therefore, researchers have also leveraged vision modality as a bridge to connect time series with LLMs.

Paired Data. ImageBind [78] uses image-paired data to bind six modalities (images, text, audio, depth, thermal, and Inertial Measurement Unit (IMU) time series) and learn a joint embedding space, enabling new emergent capabilities. PandaGPT [227] further combines the multimodal encoders from ImageBind and LLMs to enable visual and auditory instruction-following capabilities. IMU2CLIP [177] aligns IMU time series with video and text, by projecting them into the joint representation space of Contrastive Language-Image Pre-training (CLIP) [200]. AnyMAL [178] builds upon IMU2CLIP by training a lightweight adapter to

project the IMU embeddings into the text token embedding space of LLaMA-2-70B. It is also capable of transforming data from other modalities, such as images, videos, audio, into the same text embedding space.

Physics Relationships. IMUGPT [130] generates IMU data from ChatGPT-augmented text descriptions. It first generates 3D human motion from text using pre-trained motion synthesis model T2M-GPT [308]. Then it derives IMU data from 3D motion based on physics relationships of motion kinetics.

Time Series Plots as Images. CLIP-LSTM [269] transforms stock market data into sequences of texts and images of price charts, and leverages pre-trained CLIP vision-language model to generate features for downstream forecasting. Insight Miner [323] converts time series windows into images using lineplot, and feeds images into vision language model LLaVA [148] to generate time series trend descriptions.

5.3.5 Using LLMs as Indirect Tools

This type of method does not directly use large language models to process time series. Instead, it applies large language models to generate indirect tools $z(\cdot)$, such as code and API calls, to benefit time series related tasks.

Code. CTG++ [325] applies GPT-4 to generate differentiable loss functions in a code format from text descriptions to guide the diffusion model to generate traffic trajectories. With this two-step translation, the LLM and diffusion model efficiently bridge the gap between user intent and traffic simulation.

API Call. ToolLLM [197] introduces a general tool-use framework composed of data construction, model training, and evaluation. This framework includes API calls for time series tasks such as weather and stock forecasting.

Text Domain Knowledge. SHARE [319] exploits the shared structures in human activity label names and proposes a sequence-to-sequence structure to generate label names as token sequences to preserve the shared label structures. It applies GPT-4 to augment semantics of

Table 5.2. Summary of five major categories of applying LLMs for time series analysis, including their respective subcategories, representative works, mathematical formulations, advantages and limitations. q and $\mathbf{x}_v$ represent text-based quantization process and image data.

Method	Subcategory	Representative Works	Equations	Advantages	Limitations
Prompting	Number-Agnostic	PromptCast [282]	$\mathbf{y} = f_\theta(\mathbf{x}_s, \mathbf{x}_t)$	easy to implement; zero-shot capability	lose semantics; not efficient
	Number-Specific	LLMTime [86]			
Quantization	VQ-VAE	DeWave [63]	$k_i = \arg\min_j \|g_\phi(\mathbf{x}_s)_i - \mathbf{c}_j\|_2$	flexibility of index and time series conversion	may require two-stage training
	K-Means	AudioLM [22]	$\mathbf{k} = [k_i]_{i=1}^{\frac{T}{S}}, \mathbf{y} = f_\theta(\mathbf{k}, \mathbf{x}_t)$		
	Text Categories	TDML [297]	$\mathbf{y} = f_\theta(q(\mathbf{x}_s), \mathbf{x}_t)$		
Aligning	Similarity Match	ETP [145]	$\mathbf{y} = g_\phi(\mathbf{x}_s)$	align semantics of different modalities; end-to-end training	complicated design and fine-tuning
		MATM [90]	$\ell = \text{sim}(g_\phi(\mathbf{x}_s), f_\theta(\mathbf{x}_t))$		
	LLM Backbone	GPT4TS [328]	$\mathbf{y} = f_\theta(g_\phi(\mathbf{x}_s), \mathbf{x}_t)$		
Vision as Bridge	Paired Data	ImageBind [78]	$\ell = \text{sim}(g_\phi(\mathbf{x}_s), h_\psi(\mathbf{x}_v))$	additional visual knowledge	not hold for all data
	TS Plots as Images	[269]	$\mathbf{y} = h_\psi(\mathbf{x}_s)$		
Tool	Code	CTG++ [325]	$z = f_\theta(\mathbf{x}_t)$	empower LLM with more abilities	optimization not end-to-end
	API	ToolLLM [197]	$\mathbf{y} = z(\mathbf{x}_s)$		

label names. GG-LLM [83] leverages LLaMA-2 to encode world knowledge of common human behavioral patterns to predict human actions without further training. SCRL-LG [61] leverages LLaMA-7B as stock feature selectors to extract meaningful representations from news headlines, which are subsequently employed in reinforcement learning for precise feature aligning.

5.4 Comparison within the Taxonomy

We compare the five categories of our taxonomy and provide general guidelines for which category to choose based on considerations of data, model, efficiency and optimization.

Data. When no training data is available and the objective is to apply LLM for time series in an zero-shot fashion, it is preferable to use prompting-based methods. This is because direct prompting enables the utilization of pre-trained language models' inherent capabilities without fine-tuning. However, representing numbers as strings can diminish the semantic value intrinsically tied to numerical data. Therefore, with adequate training data, quantization or aligning-based methods become more advantageous. As shown in Figure 5.2, these two

categories are the most extensively studied ones in existing literature. Furthermore, if time series data can be interpreted or associated with visual representations, these representations can be incorporated to utilize the intrinsic knowledge embedded in the vision modality or pre-trained vision-language models.

Model. Prompting and tool integration methods tend to apply billion-parameter models as they often apply off-the-self LLMs without architectural modifications. By contrast, aligning and quantization methods vary from million to billion-parameter models, depending on the specific application requirements and available computational resources.

Efficiency. Prompting-based methods are not efficient for numerical data with high precision, as well as multivariate time series as it requires transforming each dimension into separate univariate time series, resulting in extremely long input. They are also less efficient for long-term predictions due to the computational demands of generating long sequences. These methods are more effective when dealing with simple numerical data that is richly interwoven with textual information, such as opening and closing stock prices in financial news articles. By contrast, quantization and aligning methods are more efficient to handle long sequences, as time series are typically down-sampled or segmented into patches before feeding into LLMs.

Optimization. Depending on the specific discretization technique, quantization-based method may require a two-stage training process (such as first training the VQ-VAE model), which may result in sub-optimal performance compared with that achieved through end-to-end training in aligning methods. Using large language models as indirect tools empowers LLMs with more capabilities to manage numerical data, but also raises the level of complexity to optimize both LLMs and other components in an end-to-end fashion. Therefore, existing works of tool integration typically employ off-the-shelf LLMs without further fine-tuning.

Table 5.3. Summary of representative time series and text multimodal datasets.

Domain	Dataset	Size	Major Modalities	Task
Internet of Things	Ego4D[2] [84]	3,670h data, 3.85M narrations	text, IMU, video, audio, 3D	classification, forecasting
	DeepSQA[3] [279]	25h data, 91K questions	text, imu	classification, question answering
Finance	PIXIU[4] [278]	136K instruction data	text, tables	5 NLP tasks, forecasting
	MoAT[5] [127]	6 datasets, 2K timesteps in total	text, time series	forecasting
Healthcare	Zuco 2.0[6] [97]	739 sentences	text, eye-tracking, EEG	classification, text generation
	PTB-XL[7] [250]	60h data, 71 unique statements	text, ECG	classification
	ECG-QA[8] [188]	70 question templates	text, ECG	classification, question answering
Audio	OpenAQA-5M[9] [81]	5.6M (audio, question, answer) tuples	text, audio	tagging, classification
Music	MusicCaps[10] [1]	5.5K music clips	text, music	captioning, generation
Speech	CommonVoice[11] [9]	7,335 speech hours in 60 languages	text, speech	ASR, translation

5.5 Multimodal Datasets

Applying LLMs for time series benefits from the availability of multimodal time series and text data. In this section, we introduce representative multimodal time series and text datasets organized by their respective domains (Table 5.3).

Internet of Things (IoT). Human activity recognition is an important task in IoT domain, which identifies human activities given time series collected with IoT devices (such as IMU sensors). The corresponding text data are the labels or text descriptions of these activities. Ego4D [84] presents 3,670 hours of daily-life activity data across hundreds of scenarios, including household, outdoor, workplace, and leisure. The dataset is rich in modalities, including the IMU time series measurement, and dense temporally-aligned textual descriptions of the activities and object interactions, totaling 3.85 million sentences. Ego-Exo4D [85] further offers three kinds of paired natural language datasets including expert commentary, narrate-and-act descriptions

[2]https://ego4d-data.org/
[3]https://github.com/nesl/DeepSQA
[4]https://github.com/chancefocus/PIXIU
[5]https://openreview.net/pdf?id=uRXxnoqDHH
[6]https://osf.io/2urht/
[7]https://physionet.org/content/ptb-xl/1.0.3/
[8]https://github.com/Jwoo5/ecg-qa
[9]https://github.com/YuanGongND/ltu
[10]https://www.kaggle.com/datasets/googleai/musiccaps
[11]https://commonvoice.mozilla.org/en/datasets

provided by the participants, and atomic action descriptions similar as Ego4D. DeepSQA [279] presents a generalized Sensory Question Answering (SQA) framework to facilitate querying raw sensory data related to human activities using natural language.

Finance. PIXIU [278] presents multi-task and multi-modal instruction tuning data in the financial domain with 136K data samples. It contains both financial natural language understanding and prediction tasks, and covers 9 datasets of multiple modalities such as text and time series. MoAT [127] constructs multimodal datasets with textual information paired with time series for each timestep, such as news articles extracted with relevant keywords, mostly covering finance related domains such as fuel, metal, stock and bitcoin.

Healthcare. Zuco 1.0 [96] and Zuco 2.0 [97] datasets contain simultaneous eye-tracking and EEG during natural reading and during annotation. PTB-XL [250] offers comprehensive metadata regarding ECG annotated by expert cardiologists, covering information such as ECG reports, diagnostic statements, diagnosis likelihoods, and signal-specific properties. Based on PTB-XL, ECG-QA [188] introduces the first Question Answering (QA) dataset for ECG analysis, containing 70 question templates that cover a wide range of clinically relevant ECG topics.

Audio/Music/Speech. AudioSet [77] is a collection of 2 million 10-second audio clips from YouTube videos and labeled with the sounds that the clip contains from a set of 527 labels. OpenAQA-5M [81] consists of 1.9 million closed-ended and 3.7 million open-ended (audio, question, answer) tuples. MusicCaps [1] is a high-quality music caption dataset, including 5.5K music clips. MTG-Jamendo [21] is a dataset with 55,000 audio songs in various languages. Libri-Light [112] is an English dataset encompassing 60,000 hours of speech data. CommonVoice [9] is a multilingual speech dataset consisting of 7,335 validated hours in 60 languages.

These datasets offer valuable benchmarks for multimodal time series and text analysis. These contain both time series focused tasks, including classification, which is evaluated using accuracy and macro-F1 scores, and forecasting, which utilizes metrics such as MSE, MAE, RMSE, and MAPE, as well as NLP focused tasks such as captioning, question answering, and translation, assessed through BLEU, ROUGE, METEOR, and EM scores, among others.

5.6 Challenges and Future Directions

In this section, we introduce the challenges and promising future directions of applying LLMs for time series analysis.

5.6.1 Theoretical Understanding

Existing works empirically show the benefits of applying LLMs for time series analysis. For example, LIFT [62] empirically shows that language model fine-tuning can work for non-language tasks without changing the architecture or loss function; [88] empirically show that LLMs learn linear representations of space and time across multiple scales that are robust to prompting variations. Despite these empirical findings, there remains a gap in theoretical understanding of how models, primarily trained on textual data, can effectively interpret numerical time series. As a preliminary theoretical analysis, [299] prove that Transformer models can universally approximate arbitrary continuous sequence-to-sequence functions on a compact domain. Additionally, GPT4TS [328] theoretically shows that such generic capability of large language models can be related to Principal Component Analysis (PCA), as minimizing the gradient with respect to the self-attention layer shares similarities with PCA. Further investigations on the generalizability of LLMs on numerical data is essential to establish solid understanding of the synergy between LLMs and time series analysis.

5.6.2 Multimodal and Multitask Analysis

Existing papers that apply LLMs for time series analysis mostly focus on single modality and single task at a time, such as forecasting, classification, text generation, and do not support simultaneous multimodal and multitask analysis. In computer vision and audio domains, models such as Unified-IO [163] and UniAudio [288] have unified multiple input modalities into a sequence of discrete vocabulary tokens to support multiple tasks within a single transformer-based architecture. More research into leveraging LLMs for multimodal and multitask analysis

would lead to more powerful time series foundation models.

5.6.3 Efficient Algorithms

Time series, especially those that are multivariate or possess long history information may increase the computational complexity for existing large language models. Patching [186] has been a widely adopted strategy to improve performance as well as reduce complexity, but large patches may obscure the semantic information of time series and negatively impact the performance. Therefore, developing more efficient algorithms is especially crucial for facilitating large-scale time series analysis and enhancing interactions with end users.

5.6.4 Combining Domain Knowledge

Combining existing statistical domain knowledge with LLMs may further boost the model's capability for time series analysis. For example, TEMPO [26] applies time series seasonal-trend decomposition and treats decomposed components as different semantic inductive biases as input to the pre-trained transformer. FreqTST [136] leverages insights from the frequency domain by tokenizing single time series into frequency units with weights for downstream forecasting. Further incorporating domain knowledge, such as wavelet decomposition, auto-correlation analysis, and empirical mode decomposition may augment LLMs' capabilities in analyzing time series data.

5.6.5 Customization and Privacy

Existing works on LLMs and time series analysis typically train a global model for all end users. Training customized models for different users based on the global model may bring further benefits and flexibility. Another important consideration is privacy, especially as many time series data are collected in private settings for clinical purposes or smart home applications. As an initial attempt, FedAlign [309] leverages federated learning frameworks and uses the expressive natural language class names as a common ground to align the latent spaces across

different clients. Advancing research into model customization and user privacy preservation would broaden the scope and utility of LLM-empowered time series analysis.

5.7 Summary

This chapter presents a taxonomy of incorporating semantic context from large language models in our framework of robust time series analysis. We present the first survey that systematically analyzes the categorization of transferring knowledge from large language models for numerical time series analysis: direct prompting, time series quantization, aligning, the use of the vision modality to connect text and time series, and the integration of large language models with other analytical tools. For each category, we introduce their mathematical formulation, representative works, and compare their advantages and limitations. We also introduce representative multimodal text and time series datasets in various domains such as healthcare, IoT, finance, and audio. Concluding the chapter, we outline the challenges and emerging directions for potential future research of LLM-empowered time series analysis.

Chapter 5 incorporates material from the publication "Large Language Models for Time Series: A Survey", by Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh Gupta, Jingbo Shang, published in the survey track of the 33rd International Joint Conference on Artificial Intelligence (IJCAI 2024). The dissertation author was primary investigator and the lead author of this paper.

Chapter 6

Semantic Context of Label Textual Names

After introducing the taxonomy, in this chapter, we present a specific application of applying semantic contextual knowledge from large language models for sensor time series analysis. Specifically, we focus on an important sensing task of Human Activity Recognition (HAR) and applies LLMs to augment the semantics of activity label names. This application is categorized as using LLMs as indirect tools within our presented taxonomy in Chapter 5.

Current HAR techniques regard activity labels as integer class IDs without explicitly modeling the semantics of class labels. We observe that different activity names often have shared structures. For example, "open door" and "open fridge" both have "open" as the action; "kicking soccer ball" and "playing tennis ball" both have "ball" as the object. Such shared structures in label names can be translated to the similarity in sensory data and modeling common structures would help uncover knowledge across different activities, especially for activities with limited samples. In this chapter, we propose SHARE, a HAR framework that takes into account shared structures of label names for different activities. To exploit the shared structures, SHARE comprises an encoder for extracting features from input sensory time series and a decoder for generating label names as a token sequence. We also propose three label augmentation techniques to help the model more effectively capture semantic structures across activities, including a basic token-level augmentation, and two enhanced embedding-level and sequence-level augmentations utilizing the capabilities of pre-trained models. SHARE outperforms state-of-the-art HAR

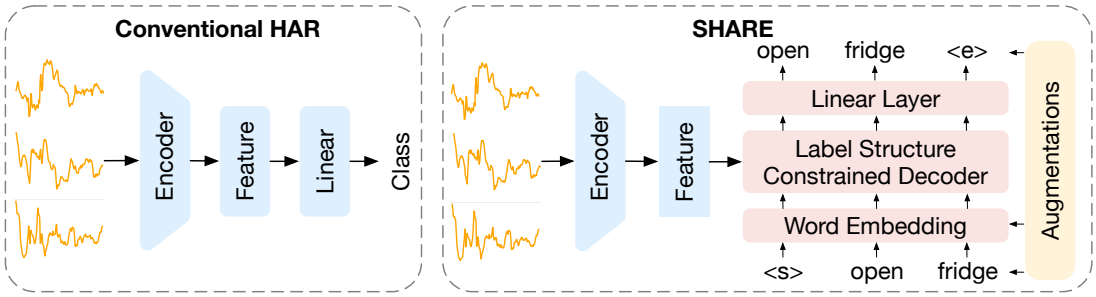


Figure 6.1. Existing HAR framework vs SHARE. SHARE exploits shared label structures and generates activity name sequences as prediction, rather than predicting integer class IDs. We also design three label augmentations at different levels to better capture shared structures.

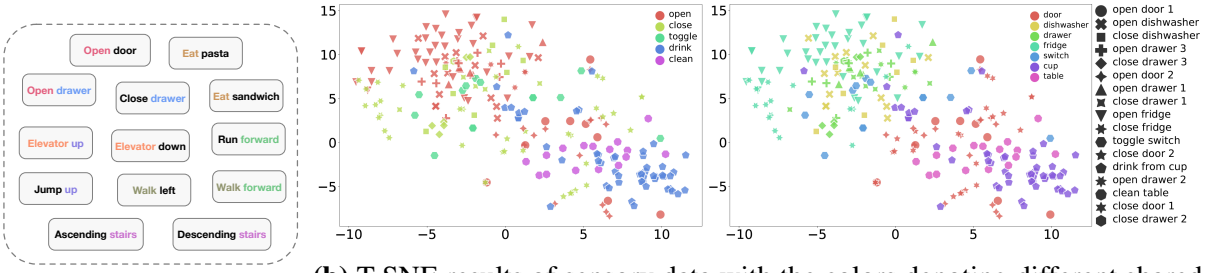
models in extensive experiments on seven benchmark datasets. We also evaluate in few-shot learning and label imbalance settings and observe even more significant performance gap.

6.1 Introduction to Leveraging Shared Label Structures for HAR

Sensor-based human activity recognition (HAR) identifies human activities using sensor readings from wearable devices. HAR has a variety of applications including healthcare, motion tracking, smart home automation, human-computer interaction [99, 47, 39, 171, 133, 290]. For example, acceleration sensors attached to legs record subjects walking around and performing daily activities for gait analysis for Parkinson's disease patients [13]; accelerometer and gyroscope can monitor user postures to detect falls for elderly people [267].

While tremendously valuable, HAR data remain difficult to collect due to security or privacy concerns, as human subjects involved in the collection process may not consent to data sharing or data transmission over the network. This often leads to local training at the edge using limited samples from just a few human subjects. Additionally, certain types of human activities happen less frequently by nature, further complicating data collection.

We note that existing HAR methods treat labels simply as integer class IDs and learn their semantics purely from annotated sensor data. This is less effective especially when labeled data are limited. To achieve better recognition performance, prior research mostly is concentrated on



(a) Label

(b) T-SNE results of sensory data with the colors denoting different shared label structures.

Figure 6.2. (a) Labels in HAR datasets typically share common structures. (b) T-SNE visualization of sensory data in Opportunity dataset [209]. Activities with the same actions or objects (marked by the same colors) are closer. The same color represents common actions (left) or common objects (right).

designing better feature extraction modules [191, 69, 202, 132] while largely overlooking the advantages of modeling label structures. Since sensory readings measuring human activities are time series data, existing time series classification models are also applicable to HAR. These methods, however, are also primarily focused on enhancing feature extraction [58, 302, 49]. It is noteworthy that both HAR and time series classification methods in the literature miss the modeling of label name structures.

We argue that a more effective approach to learning activity semantics is through label name modeling, as activity names in HAR datasets often share structures that reflect the similarity between different activities. For example, both "open door" and "open fridge" (sharing the action "open") involve pulling a (fridge) door around a hinge (while "open door" first rotates the knob to release the lock and "open fridge" directly pulls the handle); both "stairs up" and "stairs down" (sharing the object "stairs") need to bend the knees and extend the legs. Figure 6.2a illustrates more examples of activity label names in typical HAR datasets (e.g., "eat pasta" and "eat sandwich", "elevator up" and "elevator down"). The common actions or objects in these examples translate to similarities in the IMU data space. As shown in Figure 6.2b, we apply t-SNE visualization on sensor readings from the Opportunity dataset [209]. We color different activities by common actions or objects. Activities of the same color (sharing the same action or object in label names) appear closer in the embedding space, indicating stronger similarity in the

original sensory measurements. Such mapping between input features and label names motivates us to design a more effective learning framework that extracts knowledge from label structures.

To this end, we propose SHARE, shown in Figure 6.1, which models both input sensory data features and label name structures. SHARE comprises an encoder for extracting features from sensory input and a decoder for predicting label names. Unlike existing HAR models that output integer class IDs as prediction results, SHARE outputs *label name sequences*, thus preserving structures among various activities and providing a global view of activity relationships. During training, we optimize the model by minimizing the differences between predicted label names and ground-truth label names. During inference, we exploit a constrained decoding method to produce only valid labels.

We also design three label augmentation methods at different levels to better capture shared structures across activities. The basic token-level augmentation randomly replaces the original label sequences by their meaningful tokens (e.g., all actions of "eat X" are treated as a class of "eat"). This happens only during training and helps the model consolidate semantics of shared structures across different activities. We further develop two embedding- and sequence-level augmentations leveraging pre-trained models. At the embedding level, we integrate pre-trained word embeddings to capture shared semantic meanings not obvious in label names (e.g., the similarity between "walk" and "run"). At the label sequence level, for HAR datasets that do not have shared structures in their original labels, we offer an automated label generation method to generate new labels with shared tokens while preserving the same semantic meanings, leveraging large language models. Specifically, we use OpenAI's GPT-4 [190] to extend atomic, non-overlapping label names into sequences of meaningful tokens. To the best of our knowledge, SHARE is the first solution to HAR classification via decoding label sequences. We evaluate SHARE on seven HAR benchmark datasets and observe the new state-of-the-art performance. We summarize our main contributions as follows:

- We find shared structures in label names map to similarity in the input data, leading to a more effective HAR framework, SHARE, by modeling label structures. SHARE captures knowledge

across activities by uncovering information from label structures.

- We propose three label augmentation methods, each targeting at a different level, to more effectively identify shared structures across activities. These include a basic token-level augmentation and two pre-trained model-enhanced augmentations at the embedding level and at the label sequence level.
- We evaluate SHARE on seven HAR benchmark datasets and observe the new state-of-the-art performance. We also conduct experiments under few-shot settings and label imbalance settings and observe even more significant performance improvement.

6.2 Related Work

Human Activity Recognition. Existing HAR approaches can be categorized into statistical methods and deep learning based methods [40, 313]. Traditional methods are based on data dimensionality reduction, spectral feature transformation (e.g., Fourier transformation), kernel embeddings [196], first-order logic [301] or handcrafted statistical feature extraction (e.g., mean, variance, maximum, minimum) [70]. These features are then used as input to shallow machine learning methods like SVM, and Random Forest. In recent years, deep learning methods have advanced automatic feature extraction and have begun to substitute hand-crafted feature engineering in HAR [89, 69, 290] , including convolutional neural networks, recurrent neural network, attention mechanism, and their combinations. DeepConvLSTM [191] is composed of convolutional layers for feature extractors and recurrent layers for capturing temporal dynamics of the feature representations. MA-CNN [202] designs modality-specific architecture to first learn sensor-specific information and then unify different representations for activity recognition. SenseHAR [107] proposes a sensor fusion model that maps raw sensory readings to a virtual activity sensor, which is a shared low-dimensional latent space. AttnSense [167] further integrates attention mechanism to convolutional neural network and gated recurrent units network. THAT [132] proposes a two-stream convolution augmented

Transformer model for capturing range-based patterns. We shall note that these models focus on designing more effective feature extractors for better performance but neglect the semantic information in label names, which is the focus of this work.

Time Series Classification. HAR data are time-stamped sensor series, enabling the use of time series classification methods. Existing time series classification models fall into two categories: statistical and deep learning methods. Statistical methods are based on nearest neighbor [14, 222], dictionary classifier [215], ensemble classifier [144, 219], etc. These statistical methods are more robust to data scarcity but do not scale well when the feature numbers in high-dimensional space become huge. On the other hand, deep learning methods can extract features from high-dimensional data but require abundant data points to train an effective model.

Convolutional Neural Networks (FCN and ResNet) [260, 105] and Recurrent Neural Networks [114, 115] show better performance compared with statistical methods. TapNet [321] is an attentional prototype network that calculates the distance to class prototypes to learn feature representations. ShapeNet [134] performs shapelet selection by embedding shapelet candidates into a unified space and trains the network with cluster-wise triplet loss. SimTSC [303] formulates time series classification as a graph node classification problem and uses a graph neural network to model similarity information. Recently, Rocket [58] applies plenty of random convolution kernels for data transformation and attains state-of-the-art accuracy. MiniRocket [59] maintains the accuracy and improves the processing time of Rocket. TST [302] and TARNet [50] incorporate unsupervised representation learning which offers benefits over fully supervised methods on the downstream classification tasks. Similar to existing HAR methods, time series classification models focus on designing more advanced feature extraction or unsupervised representation learning methods without taking into account the label semantics, whereas SHARE models the shared structures in the label set for more effective representation learning.

Label Semantics Modeling. Given label name semantics as prior knowledge, classification tasks can benefit from modeling such semantics through knowledge graph [249] or textual information [12, 200, 326, 309]. Tong et al. [241] exploit knowledge from video action

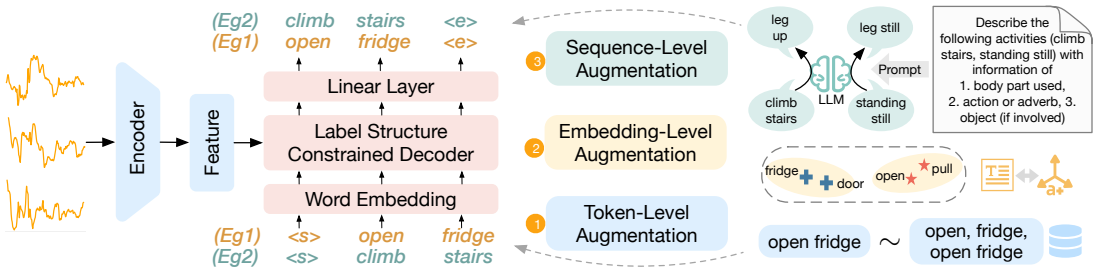


Figure 6.3. Framework of SHARE. We encode the time series features and decode the label sequences as predictions. We further design three augmentation methods at different levels to better capture the shared semantic structures.

recognition models to construct an informative semantic space that relates seen and unseen activity classes. Recent works designed specifically for zero-shot learning in human activity recognition also combine semantic embeddings [173, 273, 255]. However, these works mostly calculate the mean embeddings for labels with multiple words, which misses label structures and is suboptimal. Unlike these works, SHARE preserves label structures and enables knowledge sharing through decoding label names for generic HAR.

6.3 Preliminary

We focus on human activity recognition such as walking and sitting, captured by the sensory time series data in a given time period. We formulate HAR settings of conventional methods and SHARE.

Conventional HAR. We denote HAR data in conventional methods as $\mathscr{D}' = \{(\mathbf{x}_i, c_i)\}_{i=0}^{N}$, $\mathbf{x}_i \sim \mathscr{X}, c_i \sim \mathscr{C}$, where $\mathscr{X}$ and $\mathscr{C}$ denote the input space and the label space. Each sample of time series input is denoted as $\mathbf{x}_i \in \mathbb{R}^{T_i \times v}$, where T_i is the length of the time series, and v is the number of measured variables. The label space $\mathscr{C}$ contains C classes, and each label c is an integer from $\{1, 2, \cdots, C\}$.

SHARE. We denote dataset in SHARE as $\mathscr{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=0}^{N}, \mathbf{x}_i \sim \mathscr{X}, \mathbf{y}_i \sim \mathscr{Y}$, where data space $\mathscr{X}$ is the same as conventional HAR methods, and $\mathscr{Y}$ denotes the label space in SHARE. We denote $\mathbf{y}_i = [y_{i1}, y_{i2}, \cdots, y_{ik_i}]$ as a sample human activity label sequence, where

Algorithm 1: SHARE Framework

Input : Training set $\mathscr{D}_{\text{tr}} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=0}^{N_{\text{tr}}}$, test set $\mathscr{D}_{\text{te}} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=0}^{N_{\text{te}}}$.
Model : Encoder f_θ, Decoder g_ϕ.
Output : Predicted label sequences on test set $\{\hat{\mathbf{y}}_i\}_{i=0}^{N_{\text{te}}}$.

if *no shared tokens in* $\mathscr{Y}$ **then**
  Sequence-Level augmentation to $\mathscr{Y}$; `// Sec 6.4.4`
while *not converge* **do**
  Sample $(\mathbf{x}_i, \mathbf{y}_i) \sim \mathscr{D}_{\text{tr}}$;
  Token-Level augmentation $\mathbf{y}_i' \leftarrow \mathbf{y}_i$; `// Sec 6.4.3`
  Encoder feature extraction $\mathbf{z}_i = f(\mathbf{x}_i; \theta)$; `// Sec 6.4.1`
  Embedding-Level augmentation and label sequence decoding $\hat{\mathbf{y}}_i = g(\mathbf{z}_i; \phi)$; `// Sec 6.4.2,6.4.4`
  Optimize θ and ϕ through Equation 6.4;
for $(\mathbf{x}_i, \mathbf{y}_i) \in \mathscr{D}_{\text{te}}$ **do**
  Encoder feature extraction $\mathbf{z}_i = f(\mathbf{x}_i; \theta)$;
  Embedding-Level augmentation and label sequence constrained decoding $\hat{\mathbf{y}}_i = \text{argmax}_{\mathbf{y}_i \in \mathscr{Y}} P_\phi(\mathbf{y}_i | \mathbf{z}_i)$;
return predicted label sequences $\{\hat{\mathbf{y}}_i\}_{i=0}^{N_{\text{te}}}$.

k_i is the length of the label sequence $\mathbf{y}_i$. For example, the label "walk upstairs" contains a word sequence of length two, ["walk", "upstairs"] respectively. The label space $\mathscr{Y}$ contains C classes and M tokens. Instead of presenting labels as independent integer IDs, there exist shared structures across different labels in the label space $\mathscr{Y}$. For example, "walk upstairs" and "walk downstairs" both have "walk" in label names. Formally, there exist labels $\mathbf{y}_i, \mathbf{y}_j$, $i \neq j$ that have the same word $y_{im} = y_{jl}$, where $1 \leq m \leq k_i, 1 \leq l \leq k_j$ are positions in $\mathbf{y}_i$ and $\mathbf{y}_j$.

6.4 Methodology

We design a label structure decoding architecture for HAR, called SHARE, that exploits label structures and promotes knowledge sharing across activities. SHARE consists of two modules: Time Series Encoder and Label Structure-Constrained Decoder. We pass multivariate sensory readings as input to the encoder and use the extracted feature vector to initialize the hidden states of the decoder. The decoder generates an activity name sequence (e.g., "climb

stairs") as the prediction label. By binding sensory features with label structures, the structures in label names help the model better learn the similarity in the sensory data. We further propose three augmentation methods, including one basic token-level augmentation (randomly selecting from "climb", "stairs", "climb stairs") and two pre-trained model-enhanced augmentations at embedding (using pre-trained embeddings to initialize "climb" and "stairs" word embeddings) and label sequence levels (rephrasing "climb stairs" as "leg up" which share more tokens with other label names), to better capture shared structures across different activities. We summarize the pipeline of SHARE in Figure 6.3 and Algorithm 1.

6.4.1 Time Series Encoder

We use $f_\theta : \mathscr{X} \to \mathscr{Z} \subset \mathbb{R}^d$ parameterized by θ to denote the time series encoder. This part appears in both conventional HAR and SHARE. The encoder maps data from the input space $\mathscr{X}$ to the d-dimensional hidden space $\mathscr{Z}$. For conventional HAR, the final predictions are obtained from the hidden representations after a fully connected layer fc. Denote $\hat{c}_i = \text{fc}(f(\mathbf{x}_i;\theta)) \in \mathbb{R}^C$ as the distribution of the predicted label. Optimization is based on the cross-entropy loss between prediction $\hat{c}_i$ and ground truth c_i:

$$\arg\min \mathbb{E}_{(\mathbf{x}_i,c_i)\sim\mathscr{D}'} \text{CE}(c_i,\hat{c}_i). \tag{6.1}$$

In SHARE, the encoded representations $\mathbf{z}_i = f(\mathbf{x}_i;\theta)$ are used to initialize hidden states of the decoder, instead of being directly used for classification. This transfers learned representations from the encoder to inform the structured decoding process. To instantiate the time series encoder, we keep both efficacy and efficiency in mind, given that HAR models usually run on edge devices with limited compute. Therefore, we use one-dimensional Convolutional Neural Networks (CNN), as they are relatively lightweight with superior capability in extracting time series features [260, 56, 202, 317].

6.4.2 Label Structure-Constrained Decoder

We use $g_\phi : \mathscr{Z} \rightarrow \mathscr{Y}$ parameterized by ϕ to denote the label structure-constrained decoder in SHARE. The decoder generates word sequences in the label space $\mathscr{Y}$ given the encoded representations as initialization of the decoder hidden states. Following our notation in Section 6.3 (Problem Setting), we further require that each label name sequence starts from a start token $\langle s \rangle$ and ends at an ending token $\langle e \rangle$. Specifically, $\mathbf{y}_i = [y_{i0}, y_{i1}, y_{i2}, \cdots, y_{ik_i}, y_{ik_i+1}]$, where $y_{i0} = \langle s \rangle$, $y_{ik+1} = \langle e \rangle$. Decoding the token $\langle e \rangle$ means that we reach the end of the label sequence. At each decoding step, we estimate the conditional probability P_ϕ of decoding label $\mathbf{y}_i$ from $\mathbf{x}_i$, given the encoded representations $\mathbf{z}_i$ from the encoder as:

$$P_\phi(y_{i1}, y_{i2}, \cdots, y_{ik_i+1} | \mathbf{z}_i) = \prod_{t=1}^{k_i+1} P_\phi(y_{it} | \mathbf{z}_i, y_{i0}, y_{i1}, \cdots, y_{it-1}). \tag{6.2}$$

Training. During the training of SHARE, we adopt the teacher forcing strategy [268] where the ground truth label token y_{it} at each decoding step t is used as input to be conditioned on for predictions at decoding step $t+1$. Teacher forcing improves convergence speed and stability during training. We optimize SHARE based on cross-entropy loss between the predicted label sequence $\hat{\mathbf{y}}_i$ and the ground truth label sequence $\mathbf{y}_i$:

$$\hat{\mathbf{y}}_i = g(f(\mathbf{x}_i; \theta); \phi), \tag{6.3}$$

$$\arg\min \mathbb{E}_{(\mathbf{x}_i, \mathbf{y}_i) \sim \mathscr{D}} \frac{1}{k_i} \sum_{j=1}^{k_i} \mathrm{CE}(y_{ij}, \hat{y}_{ij}), \tag{6.4}$$

where $\hat{y}_{ij} \in \mathbb{R}^M$ indicates distribution of jth predicted token of $\hat{\mathbf{y}}_i$.

Inference with Constrained Decoding. During inference decoding, predicted label token $\hat{y}_{it}$ from the current decoding step t is used as input to be conditioned on for predicting tokens at step $t+1$. In typical natural language processing tasks, e.g., machine translation, it is common to decode the sequence using beam search during inference. However, beam search

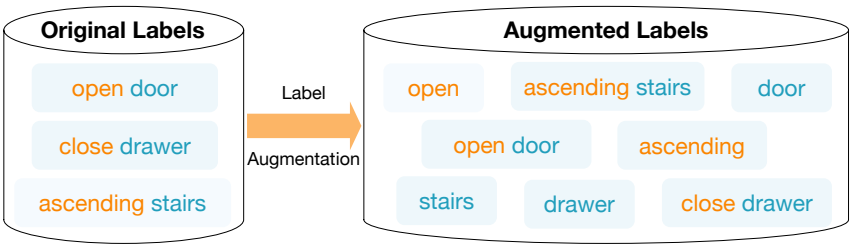


Figure 6.4. Illustration of basic token-level augmentation. We augment the original label name sequence by randomly choosing its meaningful tokens or the sequence itself .

would not work properly as it only tracks a pre-defined number of best partial solutions as candidates in decoding, and the final predictions may not belong to our label space. To guarantee that all generated labels are valid, we adopt a *constrained decoding* method. We start from the start token and iterate over all valid label sequences in the label set. We then calculate the probability of decoding each valid label sequence and choose the one with the highest probability as the final predicted label. The decoding is constrained as we only keep track of the valid partial sequences during decoding. In HAR datasets, the size of the label set is relatively small, and constrained decoding consumes only a small constant of memory (the size of the label set). At step t, we calculate the probability for all the valid partial sequences of length t and pass them into the decoder for generating tokens at step $t+1$. The final inference prediction is the sequence that maximizes the overall sequence probability:

$$\hat{\mathbf{y}}_i = \arg\max_{\mathbf{y}_i \in \mathscr{Y}} P_\phi(\mathbf{y}_i|\mathbf{z}_i). \tag{6.5}$$

We use Long Short-Term Memory (LSTM) as an example for our label structure-constrained decoder, given its effectiveness in modeling sequential dependencies [191]. We transform the CNN-extracted features $\mathbf{z}_i$ through two separate linear layers to initialize the hidden state and cell state of LSTM.

6.4.3 Basic Token-Level Label Augmentation

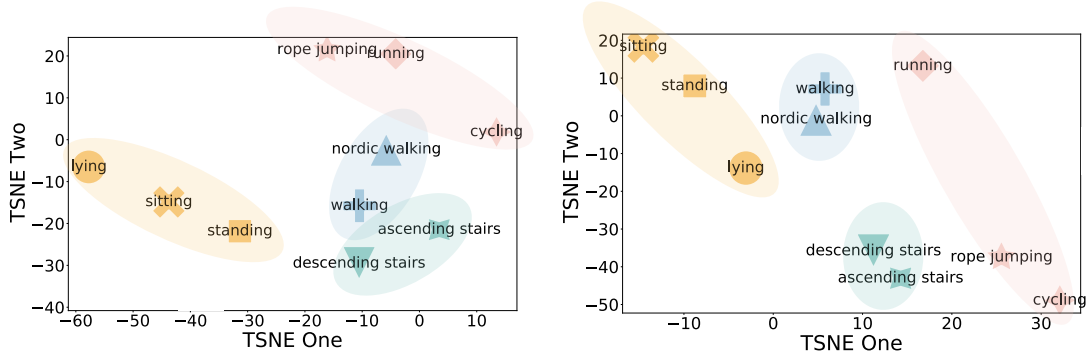
To better learn the semantics of each token in the label sequence, we apply a token-level label augmentation strategy as illustrated in Figure 6.4. During training, with pre-defined probability, we randomly choose meaningful single words from the original label sequence as the new labels. For example, an original label sequence "ascending stairs" contains single words "ascending" and "stairs", so we randomly select from "ascending", "stairs", and "ascending stairs" as the new labels during training. Following notation in Section 6.3 (Problem Setting), the original label $\mathbf{y}_i = [y_{i1}, y_{i2}, \cdots, y_{ik_i}]$ is augmented as a set of new labels $\{\mathbf{y}_i, y_{i1}, y_{i2}, \cdots, y_{ik_i}\}$ containing the label sequence $\mathbf{y}_i$ and its meaningful tokens. For each iteration, with a pre-defined probability we randomly select the new label $\mathbf{y}'_i$ from the new label set as the actual label. Optimization with token-level label augmentation can be formulated as:

$$\arg\min \mathbb{E}_{(\mathbf{x}_i,\mathbf{y}_i)\sim D} \mathbb{E}_{\mathbf{y}'_i \sim \{\mathbf{y}_i, y_{i1}, y_{i2}, \cdots, y_{ik_i}\}} \frac{1}{k'_i} \sum_{j=1}^{k'_i} \mathrm{CE}(y'_{ij}, \hat{y}_{ij}), \tag{6.6}$$

where k'_i is the length of the new label $\mathbf{y}'_i$, y'_{ij} is the jth token of $\mathbf{y}'_i$, and $\hat{y}_{ij}$ is the distribution of the predicted jth token. Since the goal of label augmentation is to help the model better capture the semantics of different activities, we only choose meaningful single tokens in the original label sequences (e.g., actions and objects) as new labels. Other single tokens like stop words or numbers (e.g., "1" in "open door 1") will not count as new labels. Note that the token-level augmentation is only applied during training. During evaluation, the ground truth label stays the same as the original label. Because we adopt a constrained decoding method during inference, it is guaranteed that all the generated label sequences are valid sequences in the original label sets.

6.4.4 Enhanced Embedding-Level and Sequence-Level Augmentations

Apart from the basic token-level augmentation, we also develop two enhanced augmentation techniques to better capture label structures from embedding and sequence levels by leveraging the power of pre-trained models.



(a) Data **(b)** ImageBind Embeddings

Figure 6.5. T-SNE visualizations show analogous clusters between input data and ImageBind word embeddings. Activities in the same color represent clusters of similar activities.

Embedding-Level Augmentation. Our label structure decoding architecture can capture label structures explicitly presented as shared label names. Yet, apart from these explicit shared label names, there may also exist semantic structures that implicitly span across different activities. For example, "walk" and "run" are similar activities involving the movement of legs, but they don't directly share label names. We have observed that such semantic structures can be captured by word embeddings from pre-trained models. We thus propose to use word embeddings from pre-trained models to initialize our decoder's word embedding layer, replacing the original random initialization. Specifically, we utilize word embeddings from ImageBind, a multimodal pre-trained model that learns a joint embedding space across six modalities. As shown in Figure 6.5, we apply t-SNE visualization to both the ImageBind word embeddings and the input sensor readings from some example activities in PAMAP dataset [207]. For activity names comprising multiple tokens, we calculate the average embedding of the aggregated tokens. T-SNE visualizations show similar clusters between ImageBind word embeddings and original data embeddings. As a result, incorporating pre-trained word embeddings helps SHARE better capture semantic structures.

Sequence-Level Augmentation. Most HAR datasets have sufficient overlapping structures in label names. However, there also exist datasets that do not have or rarely have shared

Table 6.1. Dataset statistics and an example subset of shared label names.

Dataset	Train	Test	Window Size	Channel	Class Num	An Example Subset of Shared Label Names
Opportunity [209]	2891	235	150	45	17	open door, open drawer, close drawer, open fridge, open dishwasher
PAMAP2 [207]	14438	2380	512	27	12	ascending stairs, descending stairs, walking, nordic walking
UCI-HAR [5]	7352	2947	128	9	6	walk, walk upstairs, walk downstairs
USC-HAD [312]	17576	9769	100	6	12	run forward, walk forward, elevator up, elevator down, jump up
WISDM [265]	12406	3045	200	6	18	eating soup, eating pasta, kicking soccer ball, playing tennis ball
Harth [160]	14166	3588	300	6	12	sitting, standing, cycling sitting, cycling standing, cycling sitting inactive

tokens in their original label names. For these datasets, we can use large-scale language models to automatically generate label names with shared tokens. Specifically, we employ GPT-4 with the following prompt:

Describe the following activities one by one with information of 1. body part used, 2. action or adverb, 3. object (if involved). Please maximize the number of shared tokens across different activities and make the description as short as possible.

As human activities naturally have shared actions and objects, the prompt helps find common tokens across activities. With the aid of pre-trained language model, such a process is performed with minimal human expert effort. Based on the structured information provided by the pre-trained model, we can summarize the label names with shared tokens. We apply sequence-level augmentation mostly for datasets without original shared tokens. If the target HAR dataset already has sufficient overlapping tokens, we will directly use the original label names provided by human experts.

6.5 Evaluation

6.5.1 Datasets, Baselines, and Metrics

We use six HAR benchmark datasets for evaluation, summarized in Table 6.1 with examples of shared label names. We split data and choose window size following previous works [107, 50]. The training and testing split is based on different participating subjects.

Opportunity[1] [209] collects readings from 4 users with 6 runs per user. Sensors include body-worn, object, and ambient sensors. The full dataset includes annotations on multiple levels, and we use mid-level gesture annotations which contain shared label structures.

PAMAP2[2] [207] comprises readings collected from 9 subjects wearing 3 IMUs sampled at 100 Hz and a heart rate monitor sampled at 9Hz. Three IMUs are positioned over the wrist on the dominant arm, on the chest, and on the dominant side's ankle, respectively.

UCI-HAR[3] [5] is collected from a group of 30 volunteers. A Samsung Galaxy S II smartphone was attached on their waist. Feature vectors were further extracted from each sliding window of the collected data in the time and frequency domain.

USC-HAD[4] [312] involves 14 subjects performing 12 low-level activities. They use MotionNode (6-DOF IMU for human motion sensing applications) to collect the datasets.

WISDM[5] [265] is collected from accelerometer and gyroscope sensors in smartphone and smartwatch at a rate of 20Hz. 51 subjects perform 18 activities for 3 minutes respectively.

Harth[6] [160] involves 12 activities and 22 subjects using two three-axial accelerometers attached to the thigh and lower back, and a chest-mounted camera (for data annotation).

We compare SHARE with a list of human activity recognition (DeepConvLSTM [191], MA-CNN [202], HHAR-net [69], THAT [132]) and time series classification baselines (XGBoost [42], Rocket [58], TST [302], TARNet [50]), including both statistical approaches and state-of-the-art deep learning based models.

We evaluate the performance of SHARE and baselines using accuracy and macro-F1. Macro-F1 is defined as macro-F1$= \frac{1}{C}\sum_{i=1}^{C} 2 \times \frac{\text{Prec}_\text{i} \times \text{Rec}_\text{i}}{\text{Prec}_\text{i} + \text{Rec}_\text{i}}$, where $\text{Prec}_\text{i}, \text{Rec}_\text{i}$ represent the precision and recall for each category i, and C is the total number of categories.

[1] https://archive.ics.uci.edu/ml/datasets/opportunity+activity+recognition

[2] http://archive.ics.uci.edu/ml/datasets/pamap2+physical+activity+monitoring

[3] http://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphones

[4] https://sipi.usc.edu/had/

[5] https://archive.ics.uci.edu/ml/datasets/WISDM+Smartphone+and+Smartwatch+Activity+and+Biometrics+Dataset+

[6] https://github.com/ntnu-ai-lab/harth-ml-experiments

Table 6.2. Accuracy and Macro-F1 for SHARE and baselines. We **bold** the best score and underline the second best.

Datasets	Metrics	DeepConvLSTM [191]	XGBoost [42]	MA-CNN [202]	HHAR-net [69]	TST [302]	TARNet [50]	Rocket [58]	THAT [132]	SHARE
Opp	Accuracy	0.746±0.049	0.688±0.017	0.549±0.029	0.753±0.027	0.784±0.018	0.789±0.024	0.811±0.008	0.803±0.012	**0.849±0.015**
	Macro-F1	0.634±0.036	0.547±0.011	0.416±0.036	0.620±0.021	0.668±0.023	0.669±0.034	0.670±0.016	0.691±0.015	**0.766±0.013**
PAMAP2	Accuracy	0.891±0.012	0.939±0.003	0.926±0.011	0.885±0.031	0.922±0.037	0.931±0.011	0.928±0.008	0.943±0.005	**0.960±0.002**
	Macro-F1	0.884±0.018	0.939±0.007	0.925±0.012	0.893±0.031	0.925±0.039	0.935±0.010	0.934±0.008	0.949±0.005	**0.965±0.002**
UCI-HAR	Accuracy	0.900±0.016	0.907±0.003	0.921±0.025	0.926±0.005	0.926±0.005	0.904±0.011	0.939±0.002	0.906±0.007	**0.960±0.002**
	Macro-F1	0.899±0.016	0.906±0.003	0.921±0.024	0.926±0.005	0.925±0.006	0.904±0.011	0.942±0.002	0.909±0.006	**0.959±0.002**
USC-HAD	Accuracy	0.574±0.016	0.571±0.007	0.543±0.044	0.524±0.011	0.641±0.028	0.564±0.037	0.580±0.005	0.643±0.015	**0.674±0.041**
	Macro-F1	0.557±0.015	0.573±0.006	0.520±0.047	0.523±0.009	0.594±0.023	0.533±0.021	0.601±0.007	0.619±0.012	**0.627±0.027**
WISDM	Accuracy	0.689±0.014	0.668±0.005	0.634±0.059	0.566±0.016	0.715±0.003	0.733±0.011	0.643±0.007	0.774±0.005	**0.794±0.003**
	Macro-F1	0.685±0.013	0.662±0.006	0.631±0.060	0.538±0.012	0.710±0.004	0.737±0.010	0.767±0.004	0.634±0.005	**0.790±0.004**
Harth	Accuracy	0.979±0.006	0.977±0.001	0.973±0.016	0.981±0.001	0.974±0.005	0.962±0.009	0.897±0.003	0.960±0.016	**0.983±0.007**
	Macro-F1	0.578±0.032	0.522±0.003	0.538±0.025	0.515±0.049	0.501±0.031	0.481±0.031	0.472±0.019	0.485±0.025	**0.593±0.020**

6.5.2 Experimental Setup

We use a two-layer convolutional neural network as the encoder for extracting features. The kernel sizes for both layers are set to 3 and each layer is followed by batch normalization. We adopt LSTM with a hidden dimension of 128 as the decoder, based on a grid search of $\{64, 128, 256\}$. We use Adam optimizer with learning rate 0.0001 based on a grid search of $\{0.00001, 0.0001, 0.001, 0.01\}$ and batch size 16. For all datasets, we further randomly split the training set into 80% for training and 20% for validation. We conduct the experiments in PYTORCH with NVIDIA RTX A6000 (with 48GB memory), AMD EPYC 7452 32-Core Processor, and Ubuntu 18.04.5 LTS. We tune the hyper-parameters on the validation set and then combine training and validation set to re-train the models after hyper-parameter tuning.

6.5.3 Results

We repeat 5 runs and report the average accuracy, macro-F1 score, and standard deviations of SHARE and baselines in Table 6.2. We see that SHARE consistently outperforms both statistical and deep learning based human activity recognition and time series classification approaches, in terms of both accuracy and macro-F1 score. SHARE reduces the error rate (i.e., 1 - accuracy) on six datasets by approximately 20%, 30%, 34%, 9%, 9%, 11% compared with each dataset's best-performing baseline. Compared with the hierarchical baseline HHAR-net which models activities in a simple 2-layer hierarchical model, SHARE can model much more complex dependencies not necessarily in a hierarchical structure (e.g., "open door", "open drawer", "close drawer" with pairwise overlap, forming a graph rather than tree structure), without the cost of manual labeling from experts. TST and TARNet leverage unsupervised representation learning to boost classification performance. However, they do not explicitly take account of label structures to model relations across different activities. Other top-performing HAR or time series classification methods, such as Rocket and THAT, propose better feature extractors to improve recognition performance, but they also neglect the label name structures. SHARE

is capable of leveraging the inherent shared structures in label names, leading to the highest accuracy and macro-F1 score.

To assess the statistical significance of the performance differences between SHARE and the baselines, we apply the Wilcoxon-signed rank test with Holm's α (5%) following the procedures described in ShapeNet [134, 98]. The Wilcoxon-signed rank test indicates that the improvement of SHARE compared with all the baselines is statistically significant with p far below 0.05 (e.g., $p = 5 \times 10^{-4}$ for the best-performing baseline THAT).

6.5.4 Model Variants

We also compare SHARE with some of its variants to examine the source of the performance gain. For all variants, we use the same encoder for feature extraction as SHARE.

- **VanillaHAR**: We use the same encoder as SHARE to extract features embedded in the data, and directly append a linear layer for classification without label name modeling.
- **VanillaHAR + ImageBind embeddings**: We also try directly incorporating ImageBind embeddings into VanillaHAR. This variant has two separate linear branches at the end. One branch is for classifying the labels, and the other branch predicts embeddings for the label names. During training, apart from the classification cross-entropy loss, we maximize the cosine similarity between the predicted embeddings and the pre-trained ImageBind embeddings. If the label names have multiple words, we use the average ImageBind embedding of each word as the embedding for the entire label name sequence.
- **multi-label classification**: We also try two separate classifiers subsequent to the encoder. The first classifier predicts the original labels, and the second operates as a multi-label classifier that estimates individual tokens within the label sequences. For example, to predict the class "walk forward", the second classifier labels "walk" and "forward" as positive and other tokens as negative. Classification of shared tokens helps learn dependencies across activities, and during testing, we only compare scores from the first classifier for original activity classes.
- **no aug**: We keep the model architecture but remove all three label augmentations.

Table 6.3. Accuracy and Macro-F1 for different model variants. We **bold** the best score and underline the second best.

Datasets	Metrics	VanillaHAR no label modeling	VanillaHAR+ ImageBind	multi label	no aug	no token aug	no embed aug	best baseline	SHARE
Opp	Accuracy	0.745±0.015	0.759±0.010	0.755±0.030	0.819±0.005	0.823±0.022	0.847±0.015	0.811±0.008	**0.849±0.015**
	Macro-F1	0.618±0.023	0.632±0.009	0.624±0.034	0.732±0.014	0.737±0.019	0.741±0.029	0.691±0.015	**0.766±0.013**
PAMAP2	Accuracy	0.921±0.033	0.933±0.011	0.926±0.011	0.951±0.006	0.952±0.005	0.956±0.008	0.943±0.005	**0.960±0.002**
	Macro-F1	0.924±0.024	0.938±0.014	0.928±0.010	0.960±0.006	0.958±0.005	0.964±0.009	0.949±0.005	**0.965±0.002**
UCI-HAR	Accuracy	0.921±0.005	0.928±0.005	0.926±0.005	0.954±0.006	0.957±0.004	0.958±0.004	0.939±0.002	**0.960±0.002**
	Macro-F1	0.921±0.005	0.928±0.005	0.926±0.005	0.953±0.006	0.957±0.004	0.958±0.004	0.942±0.002	**0.959±0.002**
USC-HAD	Accuracy	0.543±0.028	0.566±0.013	0.550±0.023	0.622±0.050	0.635±0.034	0.663±0.012	0.643±0.015	**0.674±0.041**
	Macro-F1	0.538±0.024	0.558±0.012	0.546±0.016	0.600±0.029	0.615±0.023	0.623±0.011	0.619±0.012	**0.627±0.027**
WISDM	Accuracy	0.644±0.006	0.655±0.004	0.649±0.010	0.788±0.006	0.786±0.008	0.793±0.008	0.774±0.005	**0.794±0.003**
	Macro-F1	0.639±0.006	0.648±0.003	0.645±0.012	0.783±0.008	0.784±0.008	0.786±0.009	0.767±0.004	**0.790±0.004**
Harth	Accuracy	0.977±0.005	0.980±0.002	0.979±0.007	0.981±0.006	0.975±0.008	0.982±0.003	0.981±0.001	**0.983±0.007**
	Macro-F1	0.481±0.004	0.487±0.014	0.482±0.005	0.566±0.034	0.591±0.043	0.583±0.023	0.578±0.032	**0.593±0.020**

Table 6.4. Original/generated label names for Mhealth data.

Original Label Names	Generated Label Names
standing still	leg still
sitting and relaxing	buttocks still
lying down	back down
walking	leg walk
climbing stairs	leg up
waist bends forward	back forward
frontals elevation of arms	arm up
knees bending (crouching)	leg forward
cycling	leg cycle
jogging	leg jog
running	leg jog fast
jump front and back	leg jump

Table 6.5. Model variants on Mhealth. We **bold** the best score and underline the second best.

Datasets	Metrics	no token aug	no embed aug	no seq aug	SHARE
Mhealth	Accuracy	0.968±0.027	0.949±0.019	0.908±0.008	**0.975±0.014**
	Macro-F1	0.969±0.027	0.949±0.021	0.905±0.012	**0.974±0.013**

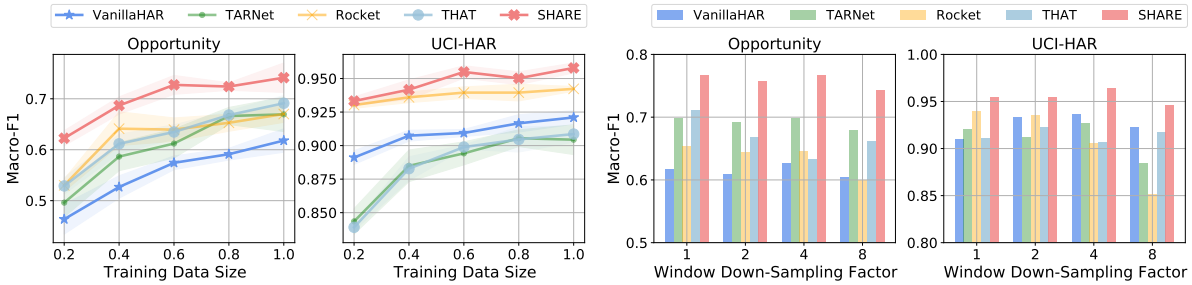
- **no token aug**: We stay with the label structure decoding architecture but remove token-level augmentation during training.
- **no embed aug**: We randomly initialize the decoder word embedding layer instead of using ImageBind word embeddings.
- **no seq aug**: The Mhealth dataset [17] (available at UCI Machine Learning Repository[7]) rarely has shared tokens in its original label names. We compare the performance of SHARE on its original non-overlapping label names and pre-trained model-augmented shared label names.

As shown in Table 6.3, we observe significant improvement from only applying a feature

[7] http://archive.ics.uci.edu/ml/datasets/mhealth+dataset

Table 6.6. Accuracy and Macro-F1 on Mhealth dataset. We **bold** the best score and underline the second best.

Datasets	Metrics	DeepConvLSTM [191]	XGBoost [42]	MA-CNN [202]	HHAR-net [69]	TST [302]	TARNet [50]	Rocket [58]	THAT [132]	SHARE
Mhealth	Accuracy	0.868±0.023	0.809±0.011	0.839±0.010	0.854±0.020	0.863±0.007	0.895±0.042	0.902±0.006	0.907±0.019	**0.975±0.014**
	Macro-F1	0.871±0.023	0.775±0.023	0.834±0.009	0.811±0.022	0.863±0.007	0.892±0.039	0.908±0.007	0.910±0.017	**0.974±0.013**

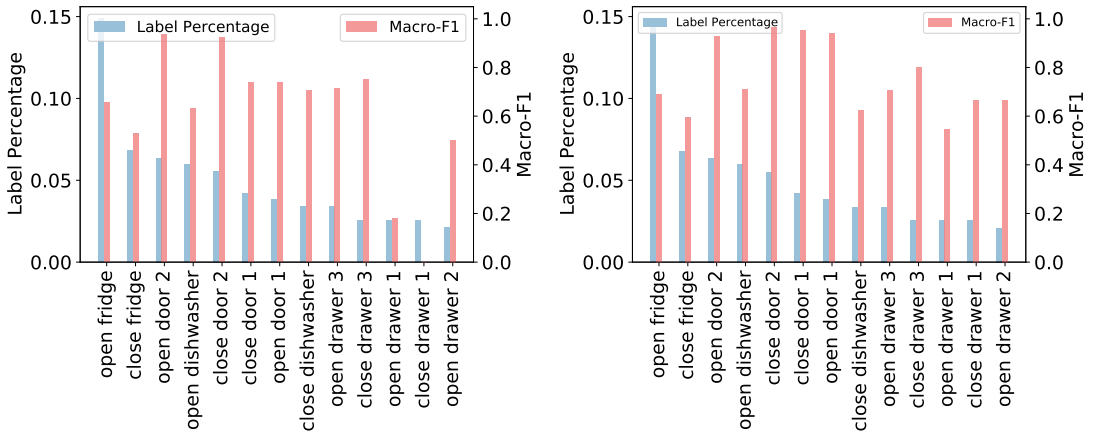


(a) Reduced Training Sample **(b)** Reduced Window Size

Figure 6.6. Macro-F1 of SHARE, VanillaHAR and best-performing baselines with reduced training samples and window size.

encoder to the proposed label structure architecture that decodes label names. Regressing label name embeddings by optimizing a cosine similarity loss with ImageBind embeddings only slightly improves the performance. This demonstrates that directly incorporating word embeddings does not explicitly take into account the shared label name structures and loses information when aggregating multiple words into a single label embedding. By contrast, SHARE generates label sequences which preserves the label structures and encourages knowledge sharing across activities. Compared with multi-label classification, our label structure decoding approach can preserve the word order (especially for multi-gram) and word correlation in label sequence.

Moreover, the performances degrade after removing either token-level or embedding-level augmentation (or removing both), which validates their importance in capturing shared word semantics. For sequence-level augmentation, we summarize the original and generated label names from pre-trained model (GPT-4) in Table 6.4. We compare SHARE using generated label names against both baselines (Table 6.6) and our model variants (Table 6.5) on the Mhealth dataset. With the help of the automated label generation method, SHARE demonstrates state-of-the-art performance for HAR datasets without original shared label names. Moreover, we observe that sequence-level augmentation and embedding-level augmentation serve as complementary strategies that synergistically enhance performance.



(a) VanillaHAR **(b)** SHARE

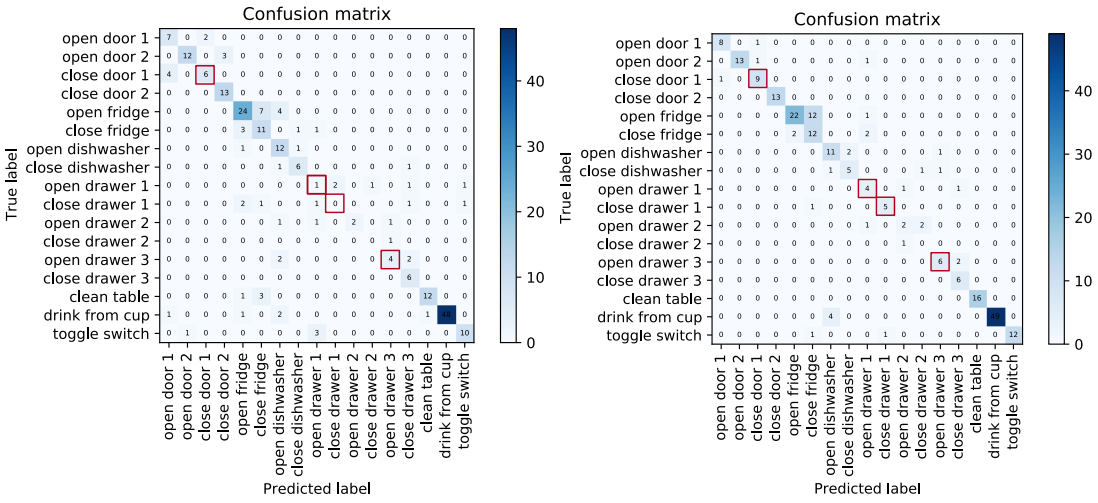
Figure 6.7. Macro-F1 of example activities with shared label names for SHARE and VanillaHAR on Opportunity dataset with long-tail label distribution.

6.5.5 Few-Shot Settings

We further evaluate SHARE under various few-shot settings.

Reduced Training Samples. We randomly reduce the number of samples in the training set from two HAR datasets (Opportunity and UCI-HAR) to 20%, 40%, 60%, and 80%, and evaluate the macro-F1 on the same original test set. We conduct the experiments for 5 runs and report both average Macro-F1 as well as standard deviation. Figure 6.6a illustrates the performance trend of SHARE, VanillaHAR as well as the best-performing baselines when we vary the size of the training set. As we can observe from the figure, the macro-F1 generally increases as the number of available training samples increases. On top of that, the performance gap between SHARE and other methods becomes larger when there are fewer training data available, showing that decoding label names helps learn the common structures that are shared across different classes.

Label Imbalance. The above experiment reduces training samples for all the classes. Many HAR datasets also naturally have a long-tail distribution where some activities have fewer samples as being more difficult to collect. We also experiment under such label imbalance scenarios as shown in Figure 6.7. We compare SHARE and the vanilla classification model



(a) VanillaHAR **(b)** SHARE

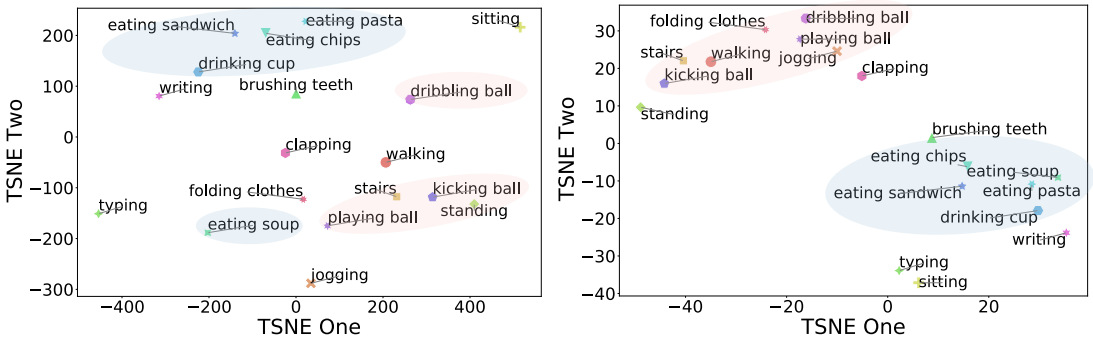
Figure 6.8. Confusion matrix of VanillaHAR and SHARE on Opportunity dataset. SHARE better discriminates different activities, exemplified by classes with red squares.

VanillaHAR by visualizing example activities with shared tokens. The activity names are sorted in decreasing order by the label percentage in the dataset. The performance grows significantly when adopting the label structure decoding architecture, as decoding label names helps transfer the shared word semantics to those classes with fewer available samples. For example, for the tail classes "open drawer 1", "close drawer 1", "open drawer 2", VanillaHAR shows a low F1 score (even zero for "close drawer 1"), while SHARE substantially improves the performance on these classes, as SHARE is able to leverage label structures to learn from other classes.

Reduced Window Size. We also reduce the sampling frequency (window size) on both training and test sets by a factor of 2,4,8 and report the performance of SHARE, VanillaHAR as well as the best-performing baselines in Figure 6.6b. SHARE also stays robust with respect to down-sampling factors, as it encourages knowledge transfer via modeling label name structures. We observe that our proposed SHARE consistently outperforms VanillaHAR and baselines under different down-sampling factors.

6.5.6 Case Study

We further explore the benefits of modeling label structures through some case studies.



(a) VanillaHAR **(b)** SHARE

Figure 6.9. T-SNE visualization on feature space. SHARE better preserves the semantics.

Confusion Matrix. We use Opportunity dataset as an example and show through the confusion matrix in Figure 6.8 that SHARE better discriminates activities compared with VanillaHAR, especially for activities with fewer samples. In Figure 6.8, values at the i-th row and j-th column represent the number of instances that have ground truth label i and are predicted as label j. "open drawer 1" instances mispredicted as "close drawer 1" are reduced from 2 to 0, and the correctly predicted instances increase from 1 to 4.

Feature Embedding. We apply t-SNE visualization to the feature space of VanillaHAR and SHARE on the WISDM dataset. We visualize the average feature of each activity, as illustrated in Figure 6.9. VanillaHAR loses the semantic information in the feature space. For example, "eating soup" is positioned at a large distance from other "eating"-related activities. By contrast, SHARE preserves the label structures in the feature space, indicating a more coherent and precise mapping of related activities.

6.5.7 Complexity Analysis

We compare the model complexity of SHARE and the best-performing deep models TST, TARNet and THAT on PAMAP2 data. We compute the number of parameters, the model size (number of bytes required to store the parameters), and the average running time for a batch of 16 samples (averaged over 10000 runs). We conduct the complexity analysis on a single NVIDIA

Table 6.7. Model complexity analysis.

Model	# of Params	Model Size	Avg Running Time Per Batch
TST	1.195M	4.786MB	0.014s
TARNet	0.310M	2.465MB	0.016s
THAT	3.207M	12.828MB	0.018s
SHARE	**0.219M**	**0.878MB**	**0.003s**

RTX A6000 48G GPU. For TST, we only compare the complexity for the supervised fine-tuning phase. As shown in Table 6.7, SHARE has the smallest number of parameters, model size, and average running time, while outperforming more complex deep models.

6.6 Summary

In this chapter, we study a specific application of incorporating semantic contextual knowledge to improve human activity recognition. We propose a novel HAR approach, SHARE, that explicitly models the semantic structure of class labels and classifies the activities by decoding label sequence. SHARE enables knowledge sharing across different activity types via label name modeling and alleviates the challenges of annotated data shortage in HAR, compared with conventional methods that treat labels simply as integer IDs. We also design three label augmentation techniques, at token, embedding and sequence levels, to help the model better capture semantic structures across activities. We evaluated SHARE on seven HAR benchmark datasets, and the results demonstrate that our model outperforms state-of-the-art methods.

There are a few remaining challenges that we plan to address in the future. We plan to adapt our design to more complex backbone models, as well as image-based or video-based human activity recognition. We also plan to experiment on other types of datasets that also have shared label name structures , e.g., medical datasets with shared disease names. Also, in this chapter, we assume that the shared label name structures very likely imply similarity in activity types. However, the assumption may not hold when we extend the problem scope to

simultaneously handling multiple datasets where the same label names may correspond to slightly different data collection settings. We believe further investigation to lift such an assumption will offer meaningful insights.

Chapter 6 incorporates material from the publication “Unleashing the Power of Shared Label Structures for Human Activity Recognition”, by Xiyuan Zhang, Ranak Roy Chowdhury, Jiayun Zhang, Dezhi Hong, Rajesh Gupta, Jingbo Shang, published in Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM 2023). The dissertation author was primary investigator and the lead author of this paper.

Chapter 7

Combining Physical and Semantic Context

The model we introduced in the previous chapter enables knowledge transfer across activities by incorporating semantic contextual knowledge. Despite its effectiveness, the model's ability to generalize is restricted to a limited set of activities, and it also fails to generalize to new datasets featuring varied device placements and mounting orientations. In this chapter, we combine physical and semantic contextual knowledge to develop a more unified pre-trained model for human activity recognition or motion time series classification, which generalizes to diverse device latent factors and activities. We integrate physical context by simulating motion time series from motion skeletons based on the motion equations. Additionally, we incorporate semantic context by similarity matching between paired motion time series and motion textual descriptions, which is categorized as aligning within our presented taxonomy in Chapter 5.

Motion time series collected from mobile and wearable devices such as smartphones and smartwatches offer significant insights into human behavioral patterns, with wide applications in healthcare, automation, IoT, and AR/XR due to their low-power, always-on nature. However, given security and privacy concerns, building large-scale motion time series datasets remains difficult, preventing the development of pre-trained models for human activity analysis. Typically, existing models are trained and tested on the same dataset, leading to poor generalizability across variations in device location, device mounting orientation and human activity type. In this chapter, we introduce UniMTS, the first unified pre-training procedure for motion time series

that generalizes across diverse device latent factors and activities. Specifically, we employ a contrastive learning framework that aligns motion time series with text descriptions enriched by large language models. This helps the model learn the semantics of time series to generalize across activities. Given the absence of large-scale motion time series, we derive and synthesize time series from existing motion skeleton data with all-joint coverage. Spatio-temporal graph networks are utilized to capture the relationships across joints for generalization across different device locations. We further design rotation-invariant augmentation to make the model agnostic to changes in device mounting orientations. Our model shows exceptional generalizability across 18 motion time series classification benchmark datasets, outperforming the best baselines by **342.3%** in the zero-shot setting, **16.3%** in the few-shot setting, and **9.2%** in the full-shot setting.

7.1 Introduction to Unified Pre-training for Motion Time Series

Recognition of human motion using time series from mobile and wearable devices, such as accelerations and angular velocities, is widely adopted as a key context information for various applications from health condition monitoring [17], sports activity analysis [3] to user habit studies [221]. Compared with vision-based approaches, methods based on motion sensor time series offer more energy-efficient and cost-effective solutions with enhanced privacy protection [235], making them preferable.

Despite being valuable, collecting motion time series data at large scale remains challenging due to security or privacy concerns. Labeling motion time series proves even more difficult as such data cannot be easily interpreted by humans for post annotation. This results in data insufficiency that negatively affects the performance of existing supervised learning methods. In other fields such as natural language processing [190, 242] and computer vision [200, 149], pre-trained foundation models have shown remarkable performance in such settings with insufficient data. However, in the motion time series domain, lack of comprehensive datasets and

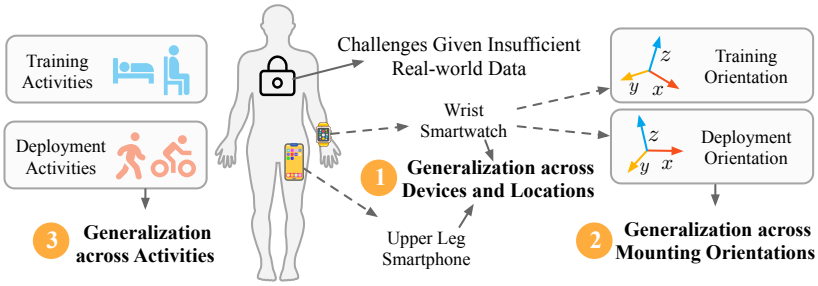


Figure 7.1. Our framework addresses all three generalization challenges (variation in device location, orientation and activity) where existing methods fall short.

an effective pre-training task makes it difficult to develop similar pre-trained models that can operate with limited data. Typically, existing models perform training and testing on the same dataset, and they struggle to generalize across different datasets given the following three unique challenges within the motion time series problem domain.

We summarize these three unique generalization challenges in Figure 7.1. First of all, the variation in device placement during deployment poses a significant issue; for instance, data from a smartwatch on the wrist vary considerably from data gathered from a smartphone near the upper leg. Therefore, models trained on data from one body location are hard to generalize to others during the testing phase. Secondly, devices can experience arbitrary orientations during data collection, making it difficult for models trained on specific device orientations to adapt to new ones during deployment. Thirdly, different motion time series datasets can be focused on different types of human activities. For example, some datasets aim to identify stationary activities such as lying or sitting, while others concentrate on dynamic movements such as walking or cycling. Models trained on specific types of activities typically struggle to generalize to new activities introduced by other datasets.

We introduce UniMTS, the first Unified pre-trained model for Motion Time Series, which addresses all the above three generalization issues and shows state-of-the-art zero-shot and fine-tuning performance. UniMTS follows a contrastive learning framework that aligns motion time series with LLM-enriched textual descriptions to learn the time series semantics for *activity* generalization. To prepare large-scale motion time series for pre-training, we synthesize these

time series based on existing extensive motion skeleton data [87] with comprehensive coverage of different body locations. These synthesized time series are modeled using graph networks to capture the spatio-temporal relationships across devices for *location* generalization. We further implement rotation-invariant augmentation to ensure the model's robustness to any device *orientation* during testing.

We summarize our primary contributions as follows:

- We introduce a unified pre-training procedure for motion time series, UniMTS, which successfully generalizes to various device locations, device orientations and activities.
- We design a contrastive learning framework to align motion time series with corresponding semantic meanings for activity generalization. For device location generalization, we propose to synthesize motion time series covering various body locations and model their spatio-temporal correlations using graph convolutional neural networks. We also design rotation-invariant augmentation to make the model agnostic to different device orientations.
- Our pre-trained model demonstrates state-of-the-art performance across 18 real-world motion time series benchmark datasets, notably with performance improvement of **342.3%** in the zero-shot setting, **16.3%** in the few-shot setting, and **9.2%** in the full-shot setting, compared with the respective best-performing baselines.

7.2 Related Work

Conventional motion time series classification approaches train a dedicated classifier for each dataset, and can be categorized into statistical feature extraction methods [70] and deep learning methods, including convolutional neural networks (MA-CNN [202], SenseHAR [107], Rocket [58]), recurrent neural network (DeepConvLSTM [191]), and the attention mechanism based models (AttnSense [167], THAT [132]). Recently, IMUGPT [130, 129] generates motion sequences given activity textual descriptions and trains conventional classification models such as DeepConvLSTM [191]. TimesNet [270], GPT4TS [328] and TEST [228] propose task-general

time series models including classification. SHARE [319] presents a seq2seq framework that leverages shared structures of label names. However, these models require that training and testing are performed on the same dataset, and cannot generalize across datasets.

Self-supervised motion time series representation learning methods first learn time series representations based on mask reconstruction (TST [302], TARNet [50]) or contrastive learning (TNC [240], TS-TCC [64], TS2Vec [298], TF-C [314], FOCAL [153]). Subsequently, they fine-tune classifier heads for specific downstream tasks. However, the representation learning and fine-tuning phases of these methods generally occur on the same or highly similar datasets, which continues to pose challenges for generalization across diverse datasets.

Pre-trained models for motion time series are inspired by the recent success of large language or multimodal models. ImageBind [78] and IMU2CLIP [177] leverage recent large vision-language models [200] to learn a joint embedding across multiple modalities including motion time series and text. However, both ImageBind and IMU2CLIP are trained on motion time series collected from head-mounted devices [84], limiting their generalizability across different device locations and orientations. Furthermore, several studies have explored directly applying LLMs for motion time series classification. For example, HARGPT [108] processes raw motion time series through LLMs and incorporates role-play and chain-of-thought strategies for prompting. ContextGPT [11] designs prompt engineering approaches leveraging context information. However, since LLMs are not directly trained on raw motion time series, such methods require extensive context information that is not usually available, and struggle with accurately recognizing complex activities.

7.3 Methodology

UniMTS follows the contrastive learning framework which aligns paired motion time series with text descriptions to enable activity generalization, as shown in Figure 7.2. We simulate motion time series from motion skeleton data (Section 7.3.1) and augment them for orientation

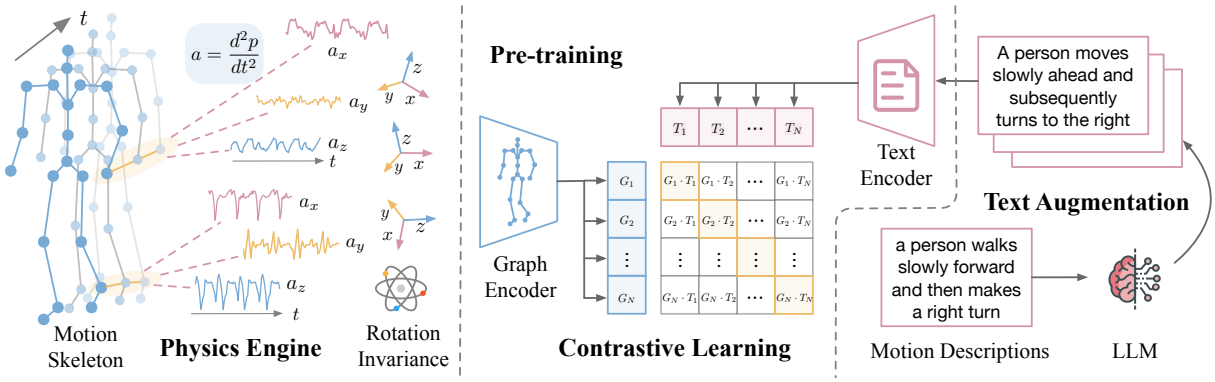


Figure 7.2. UniMTS pre-training framework: The physics engine computes motion time series based on motion skeleton data and enhances time series through rotation-invariant augmentation. Pre-training follows contrastive learning framework to align time series with text descriptions.

generalization (Section 7.3.2). We use graph encoder to model the simulated motion time series, capturing correlations among joints to generalize across different device locations (Section 7.3.3). To enhance semantics learning, text descriptions are also augmented by LLMs (Section 7.3.3).

7.3.1 Physics Engine for Motion Time Series Simulation

Motion skeleton data [87] describe the movements of human skeleton joints over time, containing positions and orientations for each joint. On the other hand, motion time series captured by physical sensors typically measure higher-order data such as accelerations and angular velocities. Consequently, we apply motion equations [295] to synthesize these time series of accelerations and angular velocities from motion skeleton data. More specifically, for each skeleton joint J_i, we input both positions $\mathbf{p}_{J_i,\mathscr{G}}$ (mapped from time domain $\mathscr{T}$ to $\mathbb{R}^3$, defined in global frame $\mathscr{G}$), and orientation quaternions $\mathbf{q}_{J_i,\mathscr{G}\mathscr{L}}$ (mapped from time domain $\mathscr{T}$ to the Special Orthogonal Group SO(3), defined in Hamilton convention with subscript $\mathscr{G}\mathscr{L}$ representing a frame rotation from local frame $\mathscr{L}$ to global frame $\mathscr{G}$). We drop the subscript $\mathscr{G}$ and $\mathscr{G}\mathscr{L}$ from here on for simplicity of notation. Based on motion equations [295], we calculate velocities $\mathbf{v}_{J_i}$ and accelerations $\mathbf{a}_{J_i}$ by taking the first and second order derivatives of positions $\mathbf{p}_{J_i}$. These derivatives are then transformed from global frames to local frames using the corresponding orientation sequences $\mathbf{q}_{J_i}$. Similarly, angular velocities $\boldsymbol{\omega}_{J_i}$ are computed by

taking the first order derivatives of orientation quaternions $\mathbf{q}_{J_i}$. Mathematically,

$$\mathbf{v}_{J_i}(t) = \mathbf{q}^*_{J_i}(t) \otimes \mathbf{p}'_{J_i}(t) \otimes \mathbf{q}_{J_i}(t), \tag{7.1}$$

$$\mathbf{a}_{J_i}(t) = \mathbf{q}^*_{J_i}(t) \otimes \mathbf{p}''_{J_i}(t) \otimes \mathbf{q}_{J_i}(t), \tag{7.2}$$

$$\omega_{J_i}(t) = 2\mathbf{q}^*_{J_i}(t) \otimes \mathbf{q}'_{J_i}(t), \tag{7.3}$$

where $\otimes$ and * represent the quaternion multiplication operator and the quaternion conjugate, respectively.

Recognizing the inherent presence of noise carried by sensors in practice, the physics engine incorporates Gaussian noise with a zero mean into the simulated data. Representing the above motion time series as $\mathbf{x}_{J_i}(t)$, which can denote either $\mathbf{a}_{J_i}(t)$ (accelerations) or $\omega_{J_i}(t)$ (angular velocities), the noisy time series $\tilde{\mathbf{x}}_{J_i}(t)$ are formulated as

$$\tilde{\mathbf{x}}_{J_i}(t) = \mathbf{x}_{J_i}(t) + \mathbf{n}_{J_i}(t), \mathbf{n}_{J_i}(t) \sim \mathscr{N}(\mathbf{0}, \sigma). \tag{7.4}$$

7.3.2 Rotation-Invariant Augmentation

A common limitation we have identified from prior studies that leads to their poor generalization is that they fail to consider the impact of latent device orientation factors on the motion time series. For example, end users can potentially wear devices in various orientations, such as with a phone facing towards or against the body in a pocket. Additionally, the software driver API for axis definition can be arbitrarily configured by the developers. For example, the iOS system defines acceleration in an opposite direction compared to the Android system. With the listed risk factors considered, we apply a data augmentation technique to simulate random orientations during pre-training, so that our learned model achieves rotation-invariance during deployment. Specifically, during pre-training, for each iteration we sample a random rotation

matrix for each joint J_i,

$$\mathbf{R}_{J_i}^{\delta} \sim \text{Uniform}(\text{SO}(3)), \tag{7.5}$$

and compute the augmented time series $\hat{\mathbf{x}}_{J_i}^t$ at timesteps $t = 1, 2, \cdots, T$ as

$$\hat{\mathbf{x}}_{J_i}^t = \mathbf{R}_{J_i}^{\delta} \tilde{\mathbf{x}}_{J_i}^t. \tag{7.6}$$

During one iteration, the same $\mathbf{R}_{J_i}^{\delta}$ is consistently applied to J_i for every timestep $t = 1, 2, \cdots, T$ within one motion time series. The rotation-invariant augmentation ensures that the simulated time series are adaptable to any downstream orientation, thereby enhancing the generalization capabilities of the pre-trained model.

7.3.3 Contrastive Learning

The physics engine generates sufficient motion time series data, which are subsequently encoded by graph convolutional neural networks and aligned with their corresponding text embeddings through contrastive learning.

Graph Encoder

To capture the spatio-temporal correlations among different joints over time, we adopt spatio-temporal graph convolutional network [286] as our motion time series encoder. We denote the initial input graph representation as follows,

$$\mathscr{G} = (\mathscr{V} = \{\hat{\mathbf{x}}_{J_i}\}_{i=1}^{V}, \mathscr{E}_s = \{(\hat{\mathbf{x}}_{J_i}, \hat{\mathbf{x}}_{J_l}) | (J_i, J_l) \in \mathscr{H}\}, \mathscr{E}_t = \{(\hat{\mathbf{x}}_{J_i}^{t-1}, \hat{\mathbf{x}}_{J_i}^{t})\}_{i=1,t=2}^{V,T}). \tag{7.7}$$

Nodes $\mathscr{V}$ contain skeleton joints with features $\mathbf{X} \in \mathbb{R}^{C \times T \times V}$, where C, T, V represent the number of signal channels, temporal steps and joint nodes. Spatial edges $\mathscr{E}_s$ connect adjacent nodes defined by the skeleton structure $\mathscr{H}$ and temporal edges $\mathscr{E}_t$ connect temporally adjacent frames.

In practice, devices may not cover the complete joints but are rather positioned at arbitrary

subsets of the complete joints. To simulate this, during each pre-training iteration, we randomly select a subset of joints and mask data from the remaining joints with zeros. We denote the mask at one iteration as $\mathbf{M} \in \mathbb{R}^{C\times T\times V}$, where $\mathbf{M}_i \in \mathbb{R}^{C\times T}$ is $\mathbf{1}$ if joint J_i is selected, and $\mathbf{M}_i = \mathbf{0}$ if joint J_i is masked:

$$\tilde{\mathbf{X}} = \mathbf{X} \odot \mathbf{M}, \tag{7.8}$$

The graph convolution network g_ϕ first computes the spatial output features as

$$\mathbf{X}_{\text{out}} = \Sigma_k^{K_s} \Phi_k (\tilde{\mathbf{X}} (\mathbf{D}_k^{-\frac{1}{2}} \mathbf{A}_k \mathbf{D}_k^{-\frac{1}{2}})), \tag{7.9}$$

where K_s denotes the spatial kernel size, $\mathbf{A}_k^{il}$ represents whether node $\mathbf{x}_{J_l}$ belongs to the spatial convolution sampling subset $\mathscr{S}_{J_i}^k$ of node $\mathbf{x}_{J_i}$, and $\mathbf{D}_k^{ii} = \Sigma_l(\mathbf{A}_k^{il}) + \alpha$ represents the normalized diagonal matrix, with α set to 0.001 to prevent empty rows [286, 218]. $\Phi_k \in \mathbb{R}^{C'\times C\times 1\times 1}$ represents weights of the 1×1 convolution operation with C' denoting output channel dimension. Following spatial convolution, we further perform $K_t \times 1$ temporal convolution on the spatial output features $\mathbf{X}_{\text{out}}$, similar to classical convolution operations, where K_t represents the temporal kernel size. The final graph representation $g_\phi(\mathbf{X})$ is derived by averaging features across both spatial and temporal dimensions with a graph average pooling layer at the end.

Text Encoder

To increase the diversity of paired text descriptions in the pre-training motion corpus [87], we apply large language models to augment original motion text descriptions with the following prompt template: *The following one or multiple descriptions are describing the same human activities: <motion descriptions >. Generate k paraphrases to describe the same activities.*

We denote the original text descriptions combined with the LLM-augmented text descriptions as $\mathbf{Y}$. We encode them using the same text encoder f_θ as CLIP [200], utilizing its pre-trained weights for initialization.

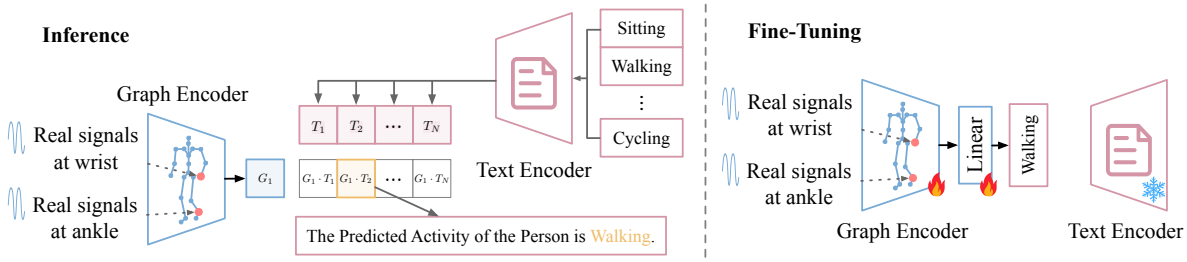


Figure 7.3. Inference (left) and fine-tuning (right) phases of UniMTS. During inference, we predict the label candidate with the highest score with the time series embedding. During fine-tuning, we freeze the text encoder and update weights of the graph encoder and linear layer.

Training and Inference

During pre-training, we maximize the similarities of paired simulated motion time series and text descriptions through contrastive learning:

$$\ell_{\text{ctr}} = -\frac{1}{B}\sum_{i=1}^{B}\log\frac{\exp(\text{sim}(g_\phi(\mathbf{X}_i), f_\theta(\mathbf{Y}_i)))^{\frac{1}{\gamma}}}{\sum_{k=1}^{B}\exp(\text{sim}(g_\phi(\mathbf{X}_i), f_\theta(\mathbf{Y}_k)))^{\frac{1}{\gamma}}}, \tag{7.10}$$

where B, γ represent batch size and temperature parameter that controls distribution concentrations, and sim represents similarity score computed as inner product:

$$\text{sim}(g_\phi(\mathbf{X}_i), f_\theta(\mathbf{Y}_i)) = \langle g_\phi(\mathbf{X}_i), f_\theta(\mathbf{Y}_i)\rangle. \tag{7.11}$$

We pre-train the graph and text encoders using simulated motion time series and augmented text descriptions. During inference, we evaluate the model on real-world motion time series, as illustrated in the left part of Figure 7.3. For the text encoder, we input all label candidates. For the graph encoder, we assign real motion time series to the nearest joint in the skeleton graph and assign zeros to the remaining joints. The random mask $\mathbf{M}$ during pre-training emulates the zero-masking process. We compute the similarity score of the graph embedding with text embedding from each label candidate, and choose the label with the highest similarity score as the predicted activity.

We can further fine-tune the pre-trained model on downstream real-world data, as depicted

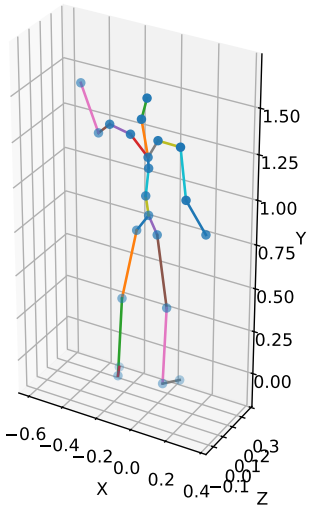


Figure 7.4. Skeleton of "waving hands".

in the right part of Figure 7.3. Specifically, we freeze the text encoder f_θ and update weights of the graph encoder g_ϕ followed by a linear classifier h_ψ. Following the same process as inference, we assign the real motion time series to the nearest joint in the skeleton graph and assign zeros to the remaining joints to construct the graph input representation $\mathbf{X}$. We fine-tune the model using $\mathbf{X}$ and one-hot encoded labels $\mathbf{z}$ with D classes based on cross-entropy loss, where $\sigma(\cdot)$ represents the softmax operation:

$$\ell_{\text{ce}} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{j=1}^{D}\mathbf{z}_{ij}\log(\sigma(h_\psi(g_\phi(\mathbf{X}_i)))_j). \tag{7.12}$$

We report both zero-shot and fine-tuning performance in the experiment section.

7.4 Evaluation

7.4.1 Datasets and Experimental Settings

We simulate motion time series from existing motion skeleton dataset HumanML3D [87]. HumanML3D [87] is a large-scale motion skeleton data consisting of 14,616 3D human motion skeletons spanning 28.59 hours. The average motion skeleton sequence length is 7.1 seconds. Paired with each motion skeleton sequence there is an average of 3 textual descriptions, resulting in a total of 44,970 textual descriptions with a vocabulary size of 5,371. The average and median

Table 7.1. Zero-Shot performance. We bold the **best** and underline the second best. UniMTS performs the best compared with both baselines and our model ablations. The last column shows the average performance across 18 datasets with standard deviation.

Dataset	Metrics	*Opportunity*	*UCI-HAR*	*MotionSense*	*w-HAR*	*Shoaib*	*HAR70+*	*RealWorld*	*TNDA-HAR*	*PAMAP2*	*USC-HAD*	*Mhealth*	*Harth*	*UT-Complex*	*Wharf*	*WISDM*	*DSADS*	*UTD-MHAD*	*MMAct*	*Average*
Number of Classes		4	6	6	7	7	7	8	8	12	12	12	12	13	14	18	19	27	35	
Level		Easy								Medium								Hard		Avg
ImageBind [78]	Acc	50.2	14.1	18.1	13.1	19.4	0.0	18.9	19.3	14.6	11.8	7.9	9.8	9.7	3.2	7.1	2.1	3.3	2.9	$12.5_{(11.0)}$
	F1	30.0	5.0	16.4	8.6	15.5	0.0	10.1	13.7	8.1	7.3	1.7	6.1	6.3	1.9	4.5	1.2	1.6	1.5	$7.8_{(7.2)}$
	R@2	**83.6**	21.4	42.8	54.1	35.5	0.2	23.9	32.8	15.6	25.0	21.3	22.5	17.8	6.8	13.7	6.9	5.1	3.7	$24.0_{(20.0)}$
IMU2CLIP [177]	Acc	25.9	17.8	15.5	6.6	16.7	5.6	6.1	8.5	1.9	12.8	7.9	2.5	7.3	1.4	4.3	4.6	3.7	6.4	$8.6_{(6.4)}$
	F1	10.3	11.4	8.6	3.3	9.2	1.8	4.5	4.2	1.1	9.3	1.3	1.1	3.4	0.7	2.7	1.1	2.1	1.6	$4.3_{(3.6)}$
	R@2	65.5	35.1	39.1	19.7	34.6	22.2	19.7	22.4	9.8	19.8	13.4	4.8	16.3	5.5	9.6	8.3	6.5	10.1	$20.1_{(14.9)}$
IMUGPT [130]	Acc	10.1	1.1	11.8	**67.2**	12.4	0.0	16.9	14.3	8.9	6.0	9.8	4.8	11.6	2.7	8.3	7.5	3.7	2.0	$11.1_{(14.4)}$
	F1	10.4	0.3	3.5	38.8	6.2	0.0	4.0	6.1	1.5	6.9	2.5	1.9	8.3	1.8	6.6	2.0	0.3	0.7	$5.7_{(8.6)}$
	R@2	33.7	18.2	40.4	**67.2**	32.1	0.0	33.7	28.5	19.3	31.8	19.5	27.8	17.4	17.7	13.9	14.6	8.8	5.1	$23.9_{(14.9)}$
HARGPT [108]	Acc	28.8	15.0	11.6	4.9	21.0	34.3	12.7	13.7	11.1	9.5	10.4	28.8	7.5	5.9	5.5	5.8	3.3	2.3	$12.9_{(9.2)}$
	F1	17.3	12.7	5.6	3.1	12.4	10.6	5.3	5.4	2.1	3.6	7.4	7.5	4.4	1.4	3.5	3.4	1.5	1.1	$6.0_{(4.4)}$
	R@2	47.0	31.4	35.9	11.5	38.6	51.9	31.7	25.2	23.0	17.6	27.4	48.6	16.5	11.4	11.8	12.1	9.3	5.0	$25.3_{(14.2)}$
LLaVA [149]	Acc	40.1	16.3	22.8	0.0	16.7	10.3	16.8	12.9	10.3	11.1	18.9	16.3	2.1	3.6	5.6	5.3	3.7	4.0	$12.0_{(9.4)}$
	F1	14.3	6.5	6.2	0.0	4.8	3.7	3.7	2.9	1.6	2.6	7.4	5.0	0.6	0.7	0.6	0.5	0.3	0.2	$3.4_{(3.5)}$
	R@2	67.6	34.6	34.4	0.0	33.3	43.4	28.4	25.2	18.7	19.6	33.5	16.4	9.0	3.6	10.6	10.5	7.4	7.3	$22.4_{(16.5)}$
UniMTS	Acc	45.9	35.2	**45.2**	59.0	**63.6**	**68.2**	43.6	**59.1**	**47.2**	**30.5**	**70.7**	**68.9**	**34.8**	18.2	27.8	**31.5**	**22.8**	10.2	**43.5**$_{(17.9)}$
	F1	**42.2**	22.0	33.7	**42.9**	**57.2**	**34.8**	36.7	**53.7**	**43.6**	**27.8**	**61.8**	**41.1**	**29.2**	13.7	25.5	**27.3**	**18.5**	10.0	**34.5**$_{(14.0)}$
	R@2	80.0	53.1	57.2	60.7	**82.1**	**86.6**	64.0	**77.5**	**63.2**	45.4	**78.7**	**85.0**	**44.2**	38.6	**47.1**	**46.0**	**32.6**	18.7	**58.9**$_{(19.3)}$
w/o rot aug	Acc	37.7	18.6	25.1	36.1	19.4	53.4	**55.5**	35.6	32.5	20.7	27.4	59.4	9.9	3.6	13.5	21.5	13.5	4.3	$27.1_{(16.3)}$
	F1	30.4	8.2	11.2	26.4	13.5	27.7	**41.1**	29.6	28.3	10.5	24.0	20.5	4.9	4.8	10.6	13.4	7.2	4.7	$17.6_{(10.7)}$
	R@2	74.3	40.1	**64.3**	52.5	40.1	76.5	**76.3**	54.4	48.8	29.7	49.4	61.2	28.5	6.8	23.3	34.0	32.1	7.6	$44.4_{(20.8)}$
w/o text aug	Acc	**52.7**	36.3	43.4	57.4	55.6	61.0	40.7	40.9	38.4	29.6	59.2	62.8	30.3	**28.2**	**31.3**	29.3	6.5	**12.6**	$39.8_{(15.8)}$
	F1	39.4	21.3	**34.7**	41.4	49.5	32.8	29.7	32.7	33.9	21.6	49.0	26.7	21.2	**19.6**	**27.2**	22.2	4.7	**10.3**	$28.8_{(11.6)}$
	R@2	70.9	39.6	63.7	57.4	78.7	77.4	60.4	63.5	48.8	**49.1**	63.4	77.9	41.4	**43.2**	45.1	41.7	10.7	**23.2**	$53.1_{(18.1)}$
w/o graph	Acc	41.4	**37.6**	18.9	23.0	50.9	18.8	42.7	33.8	30.0	22.4	28.7	34.8	23.4	0.5	12.4	19.8	1.9	11.2	$25.1_{(13.4)}$
	F1	20.5	**28.5**	21.6	17.9	38.6	9.1	23.7	28.9	23.6	23.0	19.3	15.3	11.5	1.1	8.5	16.0	2.7	7.7	$17.6_{(9.5)}$
	R@2	63.7	**66.7**	54.2	34.4	66.1	35.3	51.1	62.1	41.8	41.7	43.9	51.6	35.6	8.2	23.7	29.1	4.7	18.0	$40.7_{(18.4)}$

lengths of these descriptions are 12 and 10 words. We further augment the textual descriptions using large language models as described in Section 7.3.3. All motion skeletons follow the skeleton structure of SMPL [161] with 22 joint nodes. Figure 7.4 provides an example skeleton of "a person waves his hands".

We also try to incorporate additional motion skeleton datasets into pre-training, such as KIT-ML [195] and NTU RGB+D 120 [150]. However, these data are less diverse in terms of both motion skeletons and text descriptions. We do not observe performance improvement from adding them, and therefore use HumanML3D as our primary pre-training corpus.

We evaluate on the most extensive motion time series classification benchmark to date, comprising 18 real-world datasets that cover diverse activities. These datasets are collected from various body locations such as head, chest, back, arm, wrist, waist, hip, leg, knee and ankle. We categorize these datasets into three difficulty levels: (1) easy level (with fewer than 10 activities): Opportunity [209], UCI-HAR [5], MotionSense [172], w-HAR [20], Shoaib [220], HAR70+ [246], RealWorld [231], TNDA-HAR [287]; (2) medium level (with 10 to 20 activities): PAMAP2 [207], USC-HAD [312], Mhealth [17], Harth [160], UT-Complex [221], Wharf [23], WISDM [264], DSADS [3]; (3) hard level (with more than 20 activities): UTD-MHAD [37], MMAct [122]. We provide the specific number of activities for each dataset in Table 7.1 and Table 7.2, and detail the collection settings for each dataset as follows.

Opportunity [209] contains data from back, upper and lower arms, and features multiple sets of activities. We aim to predict the modes of locomotion such as standing and walking.

UCI-HAR [5] collects motion data from a smartphone located on the subject's waist. The subject performs daily activities such as walking upstairs and walking downstairs.

MotionSense [172] collects data from a smartphone in the participant's front pocket, featuring daily activities such as sitting and jogging.

w-HAR [20] contains motion time series data collected from the ankle. It captures daily physical activities such as jumping and lying down.

Shoaib [220] contains daily activities such as biking. Each participant is equipped with

five smartphones on right jean's pocket, left jean's pocket, belt, right upper arm and right wrist.

HAR70+ [246] tracks activities such as shuffling for older adult subjects. The motion time series are collected from the right front thigh and the lower back.

RealWorld [231] records daily activities such as climbing stairs from multiple body positions including chest, forearm, head, shin, thigh, upper arm, and waist.

TNDA-HAR [287] collects static as well as periodic daily activities such as cycling, from devices located at multiple body positions such as wrist, ankle and back.

PAMAP2 [207] monitors physical activities such as ironing, vacuum cleaning and rope jumping using devices located on the wrist, chest and ankle.

USC-HAD [312] records daily activities such as sleeping and taking the elevator with devices attached to the subject's front right hip.

Mhealth [17] comprises body motion for common activities such as waist bending forward, frontal elevation of arms and knees bending. Devices are placed on the user's chest, right wrist and left ankle.

Harth [160] records activities in a free-living setting with devices located at the right thigh and lower back.

UT-Complex [221] contains different smartphone sensor data such as typing, drinking coffee and giving a talk, with devices positioned at wrist and pocket positions.

Wharf [23] records activities from wrist-worn devices, such as combing hair.

WISDM [264] collects diverse activities such as brushing teeth, eating soup, playing balls, and folding clothes, using data from the smartphone in the pocket and smartwatch on hand.

DSADS [3] comprises daily and sports activities such as exercising and rowing. Multiple devices are positioned at the torso, right arm, left arm, right leg, and left leg.

UTD-MHAD [37] contains diverse activities such as swiping arms, hand clapping, throwing, arm crossing, drawing and squatting. The devices are worn on the subject's right wrist or the right thigh depending on whether the action is mostly an arm or a leg type of action.

MMAct [122] presents a large-scale activity dataset covering a wide range of daily life

activities such as carrying, talking on phone and falling. Devices recording motion time series include a smartwatch as well as a smartphone inside the pocket of the subject's pants.

We re-sample the real-world test data to the same sampling frequency as the simulation data (20 Hz). We pre-train UniMTS using Adam optimizer [121] with a learning rate of 0.0001 on a single NVIDIA A100 GPU. The pre-training process consumes approximately 13 GB of memory given a batch size of 64. For text augmentation, we prompt GPT-3.5 ("gpt-3.5-turbo") to generate $k = 3$ paraphrases. During each iteration, we randomly generate the mask $\mathbf{M}$ by selecting 1 to 5 joints and mask the remaining joints as zeros. We adopt learnable temperature parameter γ initialized from CLIP.

We detail settings for all the compared baselines as follows.

ImageBind [78]: We employ the pre-trained weights from "imagebind_huge" for zero-shot evaluation. During fine-tuning, we add a linear layer to map ImageBind embeddings to the number of activity classes. We fine-tune both ImageBind and the linear layer during fine-tuning, which performs better than simply tuning the linear layer.

IMU2CLIP [177]: The pre-trained weights of IMUCLIP are not released. Therefore, we first follow their pre-training implementation[1] to pre-train on Ego4D datasets [84]. During fine-tuning, we add a linear layer after IMU2CLIP embeddings and fine-tune both IMU2CLIP and the linear layer.

IMUGPT [130]: We choose DeepConvLSTM as the backbone model, which shows the best performance as reported in their original paper. We remove the supervised distribution calibration phase, which relies on labeled downstream data and conflicts with the zero-shot setting objectives.

HARGPT [108]: The method directly prompts LLMs to classify motion time series. We down-sample time series to 10 Hz as used in their paper and follow their prompt template.

LLaVA [149]: We visualize motion time series as 2D plots and use these visualizations as input for the pre-trained model of "llava-v1.5-7b".

[1] https://github.com/facebookresearch/imu2clip

TST [302]: This is a Transformer-based representation learning framework with several downstream tasks including multivariate time series classification. We follow the framework to first pre-train the Transformer model in an unsupervised fashion and then fine-tune the pre-trained model on the downstream classification task.

TARNet [50]: The model proposes task-aware representation learning that reconstructs important timestamps guided by self-attention score distribution from end-task training. We jointly train the reconstruction task and the classification task.

TS2Vec [298]: The method performs contrastive learning to learn contextual representations of time series. We follow their implementation to first apply contrastive learning and then train a linear regression model for each dataset.

DeepConvLSTM [191] (shown as "DeepCL" in Table 7.3 due to space limit): The model applies convolutional layers to automatically learn feature representations and further applies LSTM to capture the temporal dependencies between their activations.

MA-CNN [202]: The model first extracts preliminary features for each motion time series modality through its own dedicated convolutional layers, then the extracted intra-modality features are combined through fully-connected layers for motion time series classification.

XGBoost [42]: This is a scalable end-to-end machine learning system for tree boosting, which has been widely recognized in machine learning and time series analysis.

THAT [132]: The model proposes a two-stream convolution augmented transformer which captures both time-over-channel and channel-over-time features in a two-stream structure.

TimesNet [270]: This is a task-general backbone for time series analysis including classification by modeling the multi-periodicity and extracting temporal variations.

GPT4TS [328]: This is also a task-general framework including time series classification. The model is based on a frozen pre-trained language model, so we also adopt it as a few-shot fine-tuning baseline. However, the method is not suitable for zero-shot evaluation, as it requires training a separate classifier head for each dataset and does not generalize across activities.

SHARE [319]: This is a sequence-to-sequence model that contains an encoder to extract

motion time series features, as well as a decoder to generate label name sequences to capture label semantics.

We evaluate UniMTSand the baselines using three metrics: accuracy, macro-F1 and the top-2 retrieval performance R@2.

7.4.2 Zero-Shot Results

We pre-train UniMTS exclusively on simulated data and evaluate on 18 real-world motion time series classification benchmark datasets. We compare UniMTS against classification models with zero-shot capabilities: ImageBind [78], IMU2CLIP [177], IMUGPT [130] and HARGPT [108]. We also input the 2D visualizations of motion time series to pre-trained vision-language model LLaVA [149] for comparison. As shown in Table 7.1, UniMTS significantly outperforms all baselines in the zero-shot setting. We also apply the Wilcoxon-signed rank test with Holm's α (5%) following previous works [98, 319]. The Wilcoxon-signed rank test indicates that the improvement of UniMTS compared with all the baselines is statistically significant, with p-values significantly lower than 0.05 (e.g., p-value = 8×10^{-6} for ImageBind, which has the highest F1 score among the baselines).

Compared with UniMTS, ImageBind and IMU2CLIP are trained on data from single location (head-mounted devices), limiting their generalization to data collected from other locations. IMUGPT struggles to generalize across datasets featuring different activities and requires individual training for each downstream dataset. Both HARGPT and LLaVA focus on simple and easily distinguishable activities as these language or vision models are not originally trained on motion time series, and they also require careful prompt designs. Another limitation for all the above models is that they do not generalize across device orientations. In contrast, UniMTS shows remarkable generalizability to various downstream device locations, orientations and activities, achieving state-of-the-art performance. We also compare with a few ablations of UniMTS as illustrated in the Ablation Study section.

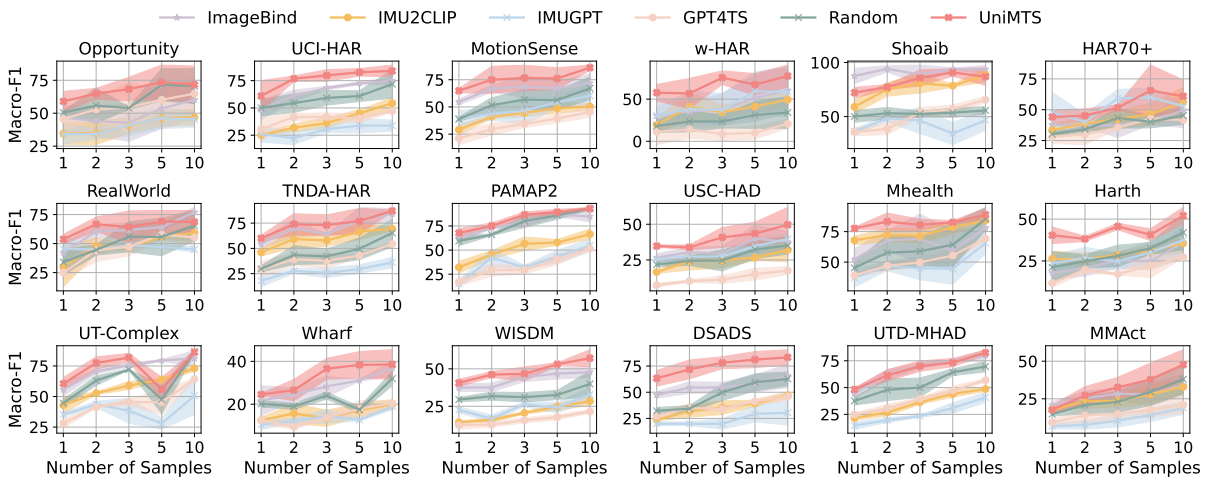


Figure 7.5. Few-shot fine-tuning results. UniMTS consistently outperforms both baselines and our model ablation. We repeat 3 runs and report both mean and standard deviation.

7.4.3 Few-Shot Fine-tuning Results

Apart from the zero-shot setting, we provide a few real samples for each activity and fine-tune the models. More specifically, we provide 1, 2, 3, 5, 10 samples for each activity and compare UniMTS against ImageBind [78], IMU2CLIP [177], IMUGPT [130], GPT4TS [328], and a randomly initialized model with the same model architecture as UniMTS (referred to as Random). We report both the mean and the standard deviation in Figure 7.5. UniMTS also demonstrates state-of-the-art performance in the few-shot fine-tuning setting, showing the effectiveness of pre-training. Following the same Wilcoxon-signed rank test as in the zero-shot setting, we observe p-values far below 0.05 (e.g., p-value = 2×10^{-25} for the best-performing baseline ImageBind), indicating the statistical significance of our improvement. We fine-tune both the graph encoder g_ϕ and the linear classifier h_ψ of UniMTS. We observe similar performance between this full fine-tuning approach and linear probe (which freezes the graph encoder and only fine-tunes the linear layer) as shown in Figure 7.6.

Table 7.2. Full-Shot performance. We bold the **best** and underline the second best. UniMTS performs the best compared with both pre-trained, self-supervised and conventional models. The last column shows the average performance across 18 datasets with standard deviation.

Dataset	Metrics	*Opportunity*	*UCI-HAR*	*MotionSense*	*w-HAR*	*Shoaib*	*HAR70+*	*RealWorld*	*TNDA-HAR*	*PAMAP2*	*USC-HAD*	*Mhealth*	*Harth*	*UT-Complex*	*Wharf*	*WISDM*	*DSADS*	*UTD-MHAD*	*MMAct*	*Average*
Number of Classes		4	6	6	7	7	7	8	8	12	12	12	12	13	14	18	19	27	35	
Level		Easy								Medium								Hard		Avg
ImageBind [78]	Acc	82.8	90.9	94.0	95.1	98.5	91.8	83.8	95.0	94.3	52.1	82.9	93.0	97.0	71.4	77.2	90.0	74.0	65.8	$85.0_{(12.2)}$
	F1	84.6	90.7	93.1	61.9	98.5	67.7	**86.1**	94.9	94.6	52.2	83.2	62.7	96.9	43.3	76.8	89.3	71.6	58.3	$78.1_{(16.5)}$
IMU2CLIP [177]	Acc	71.9	87.1	79.3	88.5	98.2	87.8	79.9	97.1	90.7	55.7	94.5	93.9	90.6	52.3	62.7	88.7	62.8	66.9	$80.5_{(14.3)}$
	F1	70.0	87.0	77.2	75.1	98.1	64.4	80.5	97.1	91.2	52.0	95.0	65.7	90.2	35.6	62.8	88.2	59.6	60.3	$75.0_{(17.0)}$
TST [302]	Acc	**89.1**	91.6	91.4	75.4	73.5	95.8	68.9	94.1	88.9	66.2	86.0	98.8	92.9	59.1	70.2	80.0	61.4	54.5	$79.9_{(13.6)}$
	F1	**91.1**	91.4	89.6	79.3	72.3	62.9	67.2	94.1	89.8	60.9	83.8	72.7	92.8	34.5	69.1	76.1	59.6	48.4	$74.2_{(16.3)}$
TARNet [50]	Acc	82.6	91.1	72.0	78.7	67.9	97.2	60.9	92.5	92.4	57.6	82.9	94.7	94.4	63.2	58.5	30.7	78.1	56.7	$75.1_{(17.5)}$
	F1	83.4	91.2	70.0	41.0	66.1	76.9	54.4	92.4	90.6	48.7	77.5	47.8	94.4	38.1	58.5	23.0	75.5	55.9	$65.9_{(20.5)}$
TS2Vec [298]	Acc	80.7	92.1	94.2	**100.0**	87.4	**98.4**	74.6	95.9	91.8	56.1	93.3	98.2	98.5	73.2	69.5	84.3	80.0	53.4	$84.5_{(13.9)}$
	F1	83.3	92.1	93.9	**100.0**	87.2	**88.1**	68.8	95.9	92.9	53.6	93.8	58.3	98.5	47.9	68.8	82.3	76.7	53.4	$79.8_{(16.7)}$
THAT [132]	Acc	83.1	86.8	87.9	77.1	92.3	95.8	67.7	97.7	96.5	53.9	89.0	97.6	86.9	60.0	60.3	82.1	71.2	55.0	$80.1_{(14.7)}$
	F1	84.5	86.7	86.5	65.0	92.2	73.9	62.9	97.7	96.7	55.0	89.9	71.9	87.0	29.5	60.9	79.0	70.0	52.3	$74.5_{(17.5)}$
IMUGPT [130]	Acc	84.8	87.0	86.2	85.3	61.7	98.3	62.2	87.7	85.6	41.1	71.3	98.3	86.5	54.6	67.8	71.9	57.7	51.9	$74.4_{(16.3)}$
	F1	84.9	87.0	85.2	48.8	61.8	78.0	56.0	87.5	83.8	42.4	63.6	65.2	85.8	24.6	67.4	71.0	53.2	49.0	$66.4_{(17.7)}$
TimesNet [270]	Acc	80.0	88.5	90.5	88.5	82.4	97.5	65.4	93.2	88.0	58.3	81.7	97.7	92.7	41.4	67.4	77.3	62.3	50.9	$78.0_{(16.2)}$
	F1	82.4	88.6	88.3	79.5	82.6	77.2	62.2	93.1	88.0	48.7	79.1	61.4	92.8	30.2	67.1	78.5	61.0	45.7	$72.6_{(17.3)}$
GPT4TS [328]	Acc	83.6	88.4	92.3	70.5	73.5	97.8	66.5	95.7	92.6	61.7	87.8	97.2	95.3	51.8	63.8	79.9	66.1	52.6	$78.7_{(15.2)}$
	F1	86.1	88.5	92.1	54.8	74.1	76.6	56.6	95.7	93.4	58.9	85.2	58.2	95.2	34.4	64.5	78.0	63.7	48.8	$72.5_{(17.7)}$
SHARE [319]	Acc	89.0	**92.2**	99.6	96.7	77.5	97.7	66.0	95.9	94.1	**72.9**	**97.0**	**99.1**	98.7	63.2	77.0	92.0	73.0	62.1	$85.8_{(13.2)}$
	F1	90.2	92.1	99.6	85.5	78.0	77.5	57.1	95.8	94.8	**66.7**	97.1	74.0	98.8	40.6	76.5	91.9	69.3	56.5	$80.1_{(16.5)}$
UniMTS	Acc	88.2	92.1	**99.8**	98.4	**99.7**	96.9	**84.0**	**99.6**	**98.0**	58.3	**97.0**	98.0	**99.1**	**82.7**	**81.3**	**94.3**	**85.6**	**77.1**	$\mathbf{90.6}_{(10.6)}$
	F1	89.1	**92.2**	**99.8**	98.8	**99.7**	87.9	81.2	**99.6**	**98.1**	63.1	**97.3**	**86.7**	**99.2**	**51.7**	**80.9**	**94.2**	**83.2**	**71.7**	$\mathbf{87.5}_{(13.4)}$
Random	Acc	87.7	89.3	95.5	95.1	66.1	98.1	74.2	98.7	96.2	48.5	82.9	98.6	98.5	76.8	79.4	90.3	74.4	73.9	$84.7_{(13.5)}$
	F1	88.9	89.4	95.5	60.1	65.0	85.9	74.0	98.7	96.8	51.9	78.2	69.1	98.5	50.8	79.3	90.1	69.7	71.6	$78.5_{(15.1)}$

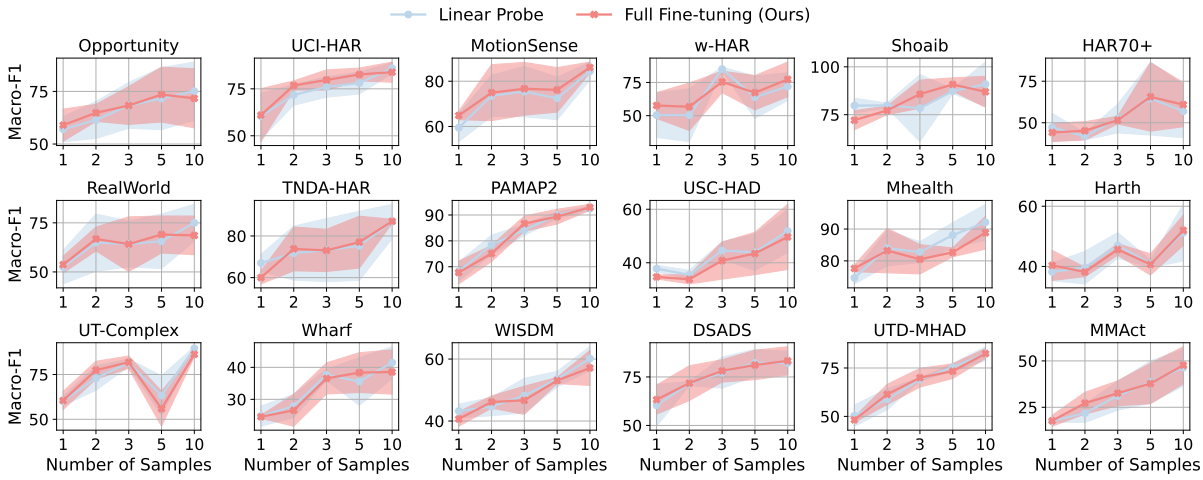


Figure 7.6. Only tuning the linear classifier (linear probe) performs similarly as full fine-tuning. We repeat 3 runs and report both mean and standard deviation.

7.4.4 Full-Shot Results

We also compare the full-shot performance where UniMTS and the baselines are fine-tuned or trained using all the available training samples of the downstream datasets. We compare UniMTS with pre-trained models (ImageBind [78], IMU2CLIP [177]), self-supervised models (TST [302], TARNet [50], TS2Vec [298]), as well as conventional models (DeepConvLSTM [191], MA-CNN [202], XGBoost [42], THAT [132], IMUGPT [130], TimesNet [270], GPT4TS [328], SHARE [319]). We also compare pre-trained UniMTS with a randomly initialized UniMTS (referred to as Random). As shown in Table 7.2, UniMTS also demonstrates state-of-the-art performance in the full-shot setting, outperforming pre-trained, self-supervised and conventional models. Due to space limit, we report baselines before 2021 in Table 7.3. Following the same Wilcoxon-signed rank test as previous zero-shot and few-shot settings, we observe p-values far below 0.05 (e.g., p-value = 7×10^{-4} for the best-performing baseline SHARE), indicating the statistical significance of our improvement.

Table 7.3. Full-Shot performance on additional baselines before 2021. We bold the **best** results. The last column shows the average performance across 18 datasets with standard deviation.

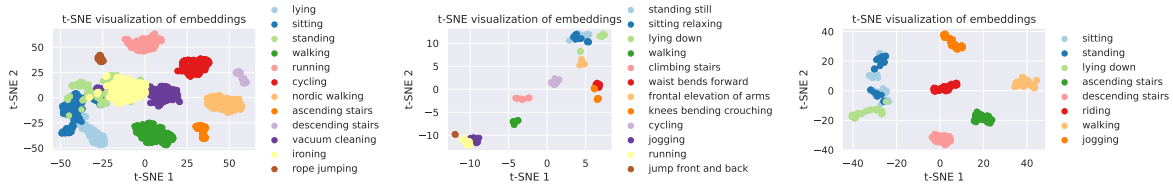
Dataset	Metrics	*Opportunity*	*UCI-HAR*	*MotionSense*	*w-HAR*	*Shoaib*	*HAR70+*	*RealWorld*	*TNDA-HAR*	*PAMAP2*	*USC-HAD*	*Mhealth*	*Harth*	*UT-Complex*	*Wharf*	*WISDM*	*DSADS*	*UTD-MHAD*	*MMAct*	*Average*
Number of Classes		4	6	6	7	7	7	8	8	12	12	12	12	13	14	18	19	27	35	
Level		Easy								Medium								Hard		Avg
DeepCL [191]	Acc	84.4	86.9	86.2	93.4	62.4	**97.7**	69.9	89.6	82.7	45.2	73.8	95.5	88.0	46.8	69.0	69.6	54.9	50.2	$74.8_{(16.7)}$
	F1	86.4	87.0	84.5	67.1	63.4	**92.2**	60.3	89.3	77.4	46.4	75.8	56.7	88.2	25.9	69.0	66.1	53.0	46.9	$68.6_{(17.8)}$
MA-CNN [202]	Acc	82.9	89.8	92.0	67.2	93.2	86.1	73.9	96.0	94.7	37.3	78.7	97.9	93.8	20.9	51.5	84.5	48.8	28.9	$73.2_{(24.2)}$
	F1	84.8	89.5	91.7	51.7	93.1	59.8	71.2	96.0	95.2	35.3	70.7	53.2	94.0	17.2	51.8	83.9	48.5	18.4	$67.0_{(25.5)}$
XGBoost [42]	Acc	83.1	90.2	92.9	68.9	94.8	**97.7**	78.2	93.2	93.9	47.8	79.9	97.2	96.6	52.3	66.6	80.5	61.9	53.0	$79.4_{(16.5)}$
	F1	85.1	90.1	91.6	56.1	94.7	77.5	77.6	93.2	93.9	47.7	74.1	64.4	96.7	30.2	66.0	79.4	60.3	51.4	$73.9_{(18.6)}$
UniMTS	Acc	**88.2**	**92.1**	**99.8**	**98.4**	**99.7**	96.9	**84.0**	**99.6**	**98.0**	**58.3**	**97.0**	**98.0**	**99.1**	**82.7**	**81.3**	**94.3**	**85.6**	**77.1**	$\mathbf{90.6}_{(10.6)}$
	F1	**89.1**	**92.2**	**99.8**	**98.8**	**99.7**	87.9	**81.2**	**99.6**	**98.1**	**63.1**	**97.3**	**86.7**	**99.2**	**51.7**	**80.9**	**94.2**	**83.2**	**71.7**	$\mathbf{87.5}_{(13.4)}$

7.4.5 Ablation Study

In the zero-shot setting, we compare UniMTS with a few ablations by removing rotation-invariant augmentation (w/o rot aug), removing text augmentation (w/o text aug) and by replacing the graph encoder with a CNN-based encoder that directly concatenates joints without modeling their spatial relationships (w/o graph). We can observe in Table 7.1 that the performance declines after removing each of the above components, verifying their respective importance in improving generalization across locations (graph encoder), orientations (rotation-invariant augmentation) and activities (text augmentation). We also compare the pre-trained UniMTS with randomly initialized UniMTS in both few-shot and full-shot settings. As shown in Figure 7.5 and Table 7.2, pre-trained UniMTS consistently outperforms randomly initialized UniMTS, highlighting the benefits of pre-training.

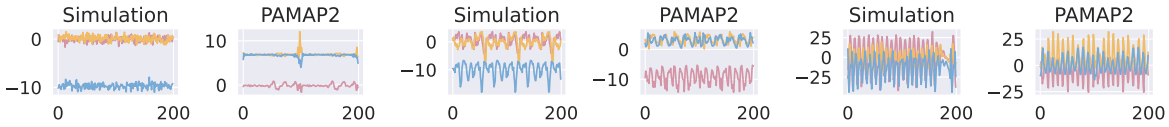
7.4.6 Case Study

UniMTS's time series embeddings align with corresponding semantic meanings. As shown in Figure 7.7, the t-SNE visualizations of UniMTS's time series embeddings form



(a) PAMAP2 **(b)** Mhealth **(c)** TNDA-HAR

Figure 7.7. T-SNE visualizations show that signal clusters align with their semantic meanings.



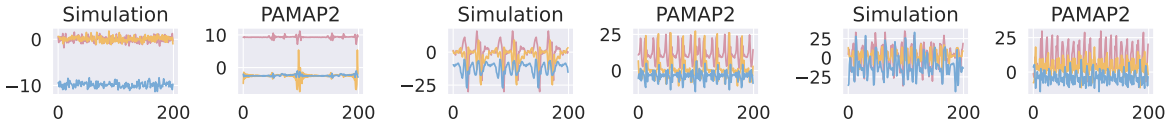
(a) Sitting **(b)** Walking **(c)** Rope Jumping

Figure 7.8. Simulated time series closely resemble patterns of the real PAMAP2 time series.

distinguishable clusters that align with their semantic meanings. Notably, UniMTS is only pre-trained on the simulated data but its embeddings for real-world data closely align with the semantic space, which again demonstrates our model's zero-shot generalization due to contrastive learning. For example, in Figure 7.7a, stationary activities such as lying and sitting group together; light-movement activities such as standing, ironing, and vacuum cleaning are close to each other; while high-intensity activities such as running and cycling cluster closer.

UniMTS generalizes well to new activities unseen in pre-training. To show UniMTS's capability to generalize to new activities not seen during pre-training, we visualize the *zero-shot* performance for some example new activities in Figure 7.10. Compared with the best-performing baseline ImageBind, UniMTS shows significant performance improvement on these previously unseen activities, verifying the effectiveness of semantic generalization via contrastive learning.

Simulated data from our physics engine closely resemble real signal patterns. In Figure 7.8, we compare some example simulated data alongside their real-world counterparts. We show three example activities of different intensity levels (sitting, walking, rope jumping), where both simulated and real-world data are near the wrist. The patterns in simulated data closely resemble those of real data, in terms of both magnitude and frequency. Although the



(a) Sitting **(b)** Walking **(c)** Rope Jumping

Figure 7.9. Simulated motion time series show similar patterns as real PAMAP2 time series.

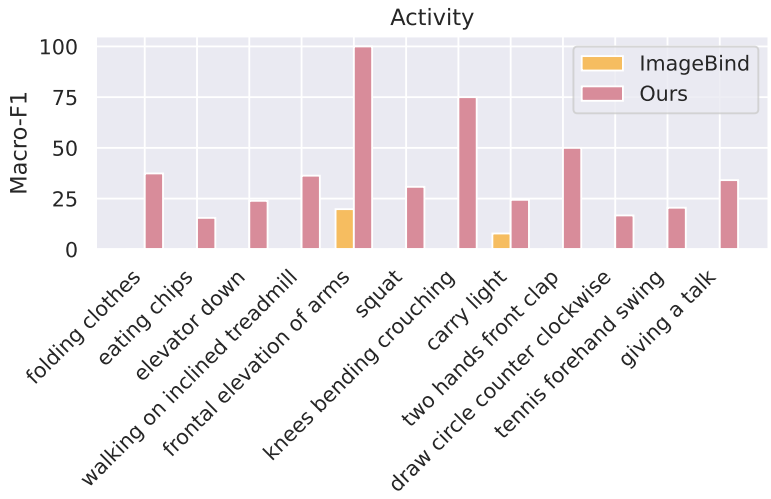


Figure 7.10. UniMTS shows significant performance improvement compared with the best baseline when evaluated on new activities not seen.

tri-axial distributions might differ, these variations are mostly due to orientation differences and are effectively managed by our rotation-invariant augmentation. We observe similar patterns between simulated and real data for other locations such as ankle, as shown in Figure 7.9.

7.5 Summary

In this chapter, we combine physical and semantic contextual knowledge to build the first unified pre-trained model, UniMTS, for motion time series classification. Our model is pre-trained only on physics-simulated data, yet demonstrates remarkable generalization across diverse real-world motion time series datasets featuring different device locations, orientations and activities. The simulated data with all-joint coverage are augmented for rotation invariance and modeled by a graph encoder, improving generalization across various device factors. During

pre-training, contrastive learning further aligns time series with their semantic meanings to improve generalization across activities. Extensive evaluation in zero-shot, few-shot and full-shot settings consistently demonstrate the state-of-the-art performance of our proposed UniMTS.

Limitation and Future Work. We acknowledge a few limitations which we leave as future works. (1) Simulated motion time series can only be approximations of the real signals, which are usually collected near – rather than directly on – the body joints. For example, sensors on smartwatches collect data near the wrist, not on the wrist joint itself. We plan to incorporate random offset vectors to better simulate real-world signal variations near joints. (2) While our framework effectively addresses the classification task, we intend to extend its applicability to other motion time series tasks such as inertial navigation. (3) Our current pre-training utilizes existing motion datasets, and we plan to enrich our pre-training corpus with additional motion data extracted from large-scale video-based pose estimation. (4) We also plan to integrate our model with efficient inference techniques such as quantization, pruning and distillation for deployment optimization on edge devices.

Broad Impact. UniMTS is the first pre-trained motion time series classification model that generalizes to diverse downstream datasets, irrespective of device locations, orientations and activities. The primary societal concern centers around privacy as motion time series might reveal personal information, so we ensure strict privacy controls at the earliest stages of model development by pre-training exclusively on synthetic data. With UniMTS's state-of-the-art performance in zero-shot, few-shot and full-shot settings, we believe it brings broadly positive impact to the community.

Chapter 7, in part is currently being prepared for submission for publication of the material. Zhang, Xiyuan; Teng, Diyan; Chowdhury, Ranak Roy; Li, Shuheng; Hong, Dezhi; Gupta, Rajesh K.; Shang, Jingbo. The dissertation author was the primary investigator and author of this material.

Chapter 8

Conclusion and Future Works

8.1 Conclusion

This dissertation starts with a thesis that sensory data can be improved for better and more robust inferencing in various applications by putting information about physics of the processes being observed as well as the semantic contextual information. Indeed, machine models alone are limited and their integration with world models is the new frontier of machine learning research [275]. For sensing systems, the analysis of time series data is behind the state of the art for textual, image or video data. This is because of four specific challenges of noise, sparsity, scarcity and variability that we outlined earlier.

We present framework that addresses all these four challenges. Our framework first leverages physical contextual knowledge such as explicit mathematical models, inherent spatio-temporal correlations, and spectral properties, to denoise, impute and augment sensor time series. This first stage significantly improves the quality of the raw data, thereby enhancing the robustness of subsequent models. In the second stage, we incorporate semantic contextual knowledge from large language models or domain-specific foundation models to further improve the robustness and generalizability of our models. Demonstrating broad applicability, our robust sensor time series analysis framework is effective in diverse sensing systems, ranging from small-scale individual healthcare monitoring, smart home automation, to large-scale smart building control, energy management, climate modeling and beyond.

8.2 Limitations and Open Challenges

The long-term objective of my research lies in designing robust frameworks that perform well under dynamic conditions of sensing systems, with real-world deployment and broad impact in various sensing applications such as healthcare, building and home automation, and robotics. My vision is to create a future where robust sensing devices seamlessly and intelligently interact with the physical world around us. Below are a few potential directions that I am interested to explore in the near future.

Multi-modal Framework. Our current robust framework targets only a single type of time series at a time. However, data collected from real-world sensing applications usually contain heterogeneous modalities. For example, a human activity recognition system might include multiple modalities such as IMU, EEG, microphone, and WiFi. We plan to collect multi-modal sensor data and build multi-modal robust learning frameworks that can simultaneously harness the rich contextual information carried by these diverse modalities.

Efficient Algorithms. Real-world sensor time series often feature high dimensions and extensive historical records. When paired with text metadata descriptions, the scale and complexity of the data further increase. Consequently, it is critical to design efficient processing algorithms, particularly for computations that occur on edge devices. Our current framework in the second stage incorporates semantic context from large language models, which, although powerful, are parameter-heavy and thus inefficient. Looking ahead, we plan to design specialized language models tailored for sensor time series and therefore improves the inference efficiency.

Real-Time Interactions. Our current robust analysis framework transforms raw sensor time series into valuable insights to support downstream decision making processes. However, in practice, users often provide feedback and pose additional questions upon receiving these

insights. For example, in the context of human activity recognition, users might provide real-time corrections for misclassified activities. This interaction provides an opportunity for the system to adjust and improve its future predictions. Moving forward, we plan to incorporate user feedback directly into the system, enhancing real-time interactions between users and the system.

Large-Scale Deployment. We prioritize not only the creation of innovative framework designs but also the actual deployment and implementation of these systems in real-world scenarios. Effectively running these robust learning methods requires both local and edge processing, as shipping all data to cloud for processing is not a viable strategy due to bandwidth limitations or privacy concerns. Effective edge processing requires not only partitioning of various functions but also restructuring basic inference algorithms that enable migration of computation and data appropriately. We plan to design FPGA co-processing methods for larger-scale deployment of our robust learning methods without compromising efficiency.

Appendix A

Open-Source Tools

We have released the tools related to this dissertation as follows.

- PILOT (Chapter 2): We develop a physics-informed denoising model that leverages the inherent relationships between different measurements to denoise sensor data[1]. We study three sensing applications: inertial navigation[2], CO2 monitoring, and HVAC control.
- ESC-GAN (Chapter 3): We extend the spatial coverage of analysis based on existing sensory data to generate data for locations where no historical data exists[3]. The proposed method is evaluated on CMAP Precipitation data[4], HadCRUT temperature data[5] and air quality data[6].
- ST-Aug (Chapter 4): We combine spectral and time augmentation to generate more diverse and coherent time series samples[7]. We evaluate on exchange rate data, available in the same repository, as well as electricity datasets (ETTh1, ETTh2, ETTm1, ETTm2)[8].
- Survey (Chapter 5): We keep an up-to-date Github repository to track papers that transfer semantic knowledge from LLMs for time series analysis[9]. The listed papers are grouped into five categories as detailed in our proposed taxonomy. We also compile a list of multimodal

[1] https://github.com/xiyuanzh/PILOT
[2] http://deepio.cs.ox.ac.uk/
[3] https://github.com/xiyuanzh/ESC-GAN
[4] https://psl.noaa.gov/
[5] https://crudata.uea.ac.uk/
[6] https://www.kdd.org/kdd2018/
[7] https://github.com/xiyuanzh/STAug
[8] https://github.com/zhouhaoyi/Informer2020
[9] https://github.com/xiyuanzh/awesome-llm-time-series

text and time series datasets in the same repository.

- SHARE (Chapter 6): We model label name semantics for human activity recognition[10]. We evaluate on 7 HAR datasets: Opportunity [209], PAMAP2 [207], UCI-HAR [5], USC-HAD [312], WISDM [264], Harth [160], MHealth [17].
- UniMTS(Chapter 7): We pre-train the motion time series model on synthetic series simulated from motion skeleton data[11] following motion equations[12]. We evaluate the zero-shot, few-shot and full-shot classification performance of our pre-trained model on 18 HAR datasets: Opportunity [209], UCI-HAR [5], MotionSense [172], w-HAR [20], Shoaib [220], HAR70+ [246], RealWorld [231], TNDA-HAR [287]; (2) medium level (with 10 to 20 activities): PAMAP2 [207], USC-HAD [312], Mhealth [17], Harth [160], UT-Complex [221], Wharf [23], WISDM [264], DSADS [3]; (3) hard level (with more than 20 activities): UTD-MHAD [37], MMAct [122].

[10]https://github.com/xiyuanzh/SHARE
[11]https://github.com/EricGuo5513/HumanML3D
[12]https://github.com/martinling/imusim

Appendix B

Notations

B.1 Chapter 2: Physical Context of Explicit Mathematical Models

N: Number of samples.

$\mathbf{X}_i \sim \mathscr{X}, \mathbf{X}_i \in \mathbb{R}^{c\times T}$: Clean sample, where $\mathscr{X}$ denotes the clean data space, T denotes the number of timesteps in the sensor data, and c denotes the number of sensor measurement channels. Each sample $\mathbf{X}_i = [\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{ic}]^T$, where $\mathbf{x}_{ij} \in \mathbb{R}^T$ represents the jth sensor measurement channel for the ith sample in the ground truth clean dataset.

$\mathscr{D}_X = \{\mathbf{X}_i\}_{i=0}^N$: Unobserved ground truth clean dataset.

$\mathbf{Y}_i \sim \mathscr{Y}, \mathbf{Y}_i \in \mathbb{R}^{c\times T}$: Noisy sample, where $\mathscr{Y}$ denotes the noisy data space. Each sample $\mathbf{Y}_i = [\mathbf{y}_{i1}, \mathbf{y}_{i2}, \cdots, \mathbf{y}_{ic}]^T$, where $\mathbf{y}_{ij} \in \mathbb{R}^T$ represents the jth sensor measurement channel for the ith sample in the noisy dataset.

$\mathscr{D}_Y = \{\mathbf{Y}_i\}_{i=0}^N$: Observed noisy sensor dataset.

$\boldsymbol{\varepsilon}_i \sim \mathscr{E}, \boldsymbol{\varepsilon}_i \in \mathbb{R}^{c\times T}$: Noise. Each noisy sample $\mathbf{Y}_i$ is corrupted from clean sample $\mathbf{X}_i$ by noise $\boldsymbol{\varepsilon}_i$, where $\mathscr{E}$ denotes the noise space. Specifically, $\mathbf{Y}_i = \mathbf{X}_i + \boldsymbol{\varepsilon}_i, i = 1, 2, \cdots, N$.

$g(\cdot)$: The relationships between different sensor measurement channels in the ground truth clean data, i.e., $g(\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{ic}) = 0, i = 1, 2, \cdots, N$.

$f(\cdot;\boldsymbol{\theta}) : \mathscr{Y} \to \mathscr{X} \subset \mathbb{R}^{c\times T}$: The denoising model, which is parameterized by $\boldsymbol{\theta}$ and maps noisy data to clean data by mapping $f(\cdot;\boldsymbol{\theta}) : \mathscr{Y} \to \mathscr{X} \subset \mathbb{R}^{c\times T}$. We incorporate the physics

equation $g(\cdot)$ as a constraint to optimize $f(\cdot;\theta)$ during training.

p: Location.

q: Orientation (in quaternion form).

w: Angular velocity.

a: Acceleration.

R_q: Rotation matrix.

g_0: Gravity constant.

$\otimes$: Quaternion multiplication.

c_t: The average CO_2 concentration in the room at time t.

c_0: Initial average CO_2 concentration in the room.

c_{in}^t: CO_2 concentration for the input airflow.

c_{out}^t: CO_2 concentration for the output airflow.

v: Airflow velocity.

V: Room volume.

s: CO_2 emission rate per individual.

n_t: The number of occupants in the room at time t.

ΔQ: The enthalpy difference across the heating and cooling coils.

m: The air's mass undergoing reconditioning.

c_h: The air's specific heat capacity.

T_{sa}: Temperature of the supply air.

T_{mix}: Temperature of the mix air.

$\mathbf{n}_i$: Gaussian noise sampled during the first phase of training.

$\mathbf{Z}_i$: Samples with a higher degree of noise during the first phase of training, i.e., $\mathbf{Z}_i = \mathbf{Y}_i + \mathbf{n}_i, \mathbf{n}_i \sim \mathcal{N}(0, \sigma^2)$, where σ^2 represents the variance.

ℓ_{rec}: Reconstruction loss.

ℓ_{phy}: Physics loss.

ℓ: General loss.

η: Non-zero mean of noise ε in Corollary 1.

$\sigma_{\mathbf{y}}^2$: Variance of $\mathbf{y}$ in Corollary 1.

$a_i = ||g(\mathbf{X}_i)||_2^2$: Physics alignment for the i-th sample.

$\lambda = l_{\text{rec}}/l_{\text{phy}}$: The ratio between reconstruction loss (l_{rec}) and physics-based loss (l_{phy}).

B.2 Chapter 3: Physical Context of Spatio-Temporal Correlations

D: The entire globe gridded into $m \times n$ cells.

D_O: Observed grid cells (i.e., where we have sensory measurements).

$D_U = \{D \setminus D_O\}$: The remaining unobserved grid cells.

T: Total number of timestamps.

S_{ij}: Grid cell, where $i = 1, \ldots, m$ and $j = 1, \ldots, n$.

$\mathbf{X} \in \mathbb{R}^{t \times m \times n}$: Input data.

$\mathbf{M} \in \{0,1\}^{t \times m \times n}$: Corresponding masking matrix. If $M_{t,i,j} = 1$, it means $S_{i,j}$ is valid at time t (i.e. we have data in grid cell $S_{i,j}$ at time t); otherwise, $S_{i,j}$ is invalid at time t (i.e. data is missing in cell $S_{i,j}$ at time t).

Φ: The set of missing timestamps.

$\tilde{\mathbf{X}}$: Complete set of grid cells.

$\mathbf{Y}$: Output of a specific layer.

$\mathbf{W}$: Convolution filter weights.

$k_t' = \frac{k_t - 1}{2}, k_h' = \frac{k_h - 1}{2}, k_w' = \frac{k_w - 1}{2}$: k_t, k_h, k_w represent kernel size along the time, height, and width dimensions.

$\theta(\mathbf{X}) = \mathbf{W}_\theta \mathbf{X}, \phi(\mathbf{X}) = \mathbf{W}_\phi \mathbf{X}, g(\mathbf{X}) = \mathbf{W}_g \mathbf{X}$: Input linear embedding layers of the global attention module.

μ_θ, μ_ϕ: Average embedding values over all the regions from θ, ϕ.

$\mathbf{O}$: Attention output.

$\mathbf{W}_o$: Weights for the output linear layer of the global attention module.

n_b: The number of branches.

$\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{n_b}$: Hidden features of the last layer of decoders from different branches.

$\mathbf{h}' \in \mathbb{R}^{c \times t \times m \times n}$: The aggregated features, where c, t, m, n are the channel, time, height, and width dimension size, respectively.

ℓ_{rec}: Grid reconstruction loss.

ℓ_{var}: Variation loss.

ℓ_{adv}: Adversarial loss.

ℓ_D: Loss for the discriminator.

ℓ_G: Loss for the generator, $\ell_G = \ell_{\text{adv}}$.

ℓ: Total loss.

$\mathbf{Z}$: Generated grid map.

N_{masked}: The number of masked out grid cells.

R: The 1-cell dilation of the masked region.

λ_{var} and λ_{adv}: Hyper-parameters balancing between reconstruction loss, variation loss, and adversarial loss.

B.3 Chapter 4: Physical Context of Time and Frequency Dependencies

$\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_t, \ldots, \mathbf{s}_T] \in \mathbb{R}^{c \times T}$: Multivariate time series sequence of length T and feature number c, where $\mathbf{s}_t = [s_1, \cdots, s_c]^T \in \mathbb{R}^c$.

$\mathbf{H} = [\mathbf{s}_1, \ldots, \mathbf{s}_d] \in \mathbb{R}^{c \times d}$: History values up to timestamp $d < T$.

$\mathbf{F} = [\mathbf{s}_{d+1}, \ldots, \mathbf{s}_T] \in \mathbb{R}^{c \times (T-d)}$: Future values at timestamp $d+1, \ldots, T$, where $\mathbf{S} = [\mathbf{H}, \mathbf{F}]$.

g: Forecasting model.

n: Number of IMFs.

$\mathbf{R}$: EMD Residual.

$\mathbf{w} = [w_1, \ldots, w_n]^T$: Random vector to re-combine IMFs, where w_i is sampled from uniform distribution $\mathscr{U}(0,2)$.

$\lambda \sim \text{Beta}(\alpha, \alpha)$: Mix-up ratio, where α is the hyper-parameter that controls how similar the newly constructed sequence is compared with the original sequences.

ℓ: Reconstruction loss.

$\mathbf{Y}$: Forecasting model output.

N: The number of augmented series in the training set.

B.4 Chapter 5: Semantic Context of Large Language Models

$\mathbf{x}$: Input, composed of time series $\mathbf{x}_s \in \mathbb{R}^{T \times c}$ and optional text data $\mathbf{x}_t$ represented as strings, where T, c represent the sequence length and the number of features.

$\mathbf{x}_v$: Image data.

$\mathbf{y}$: Output, which may represent time series, text or numbers depending on the specific downstream task. For time series generation or forecasting task, $\mathbf{y}$ represents generated time series $\mathbf{y}_s$ or predicted k-step future time series $\mathbf{y}_s^{T+1:T+k}$. For text generation task, such as report generation, $\mathbf{y}$ represents text data $\mathbf{y}_t$. For time series classification or regression task, $\mathbf{y}$ represents numbers indicating the predicted classes or numerical values.

f_θ: Language model.

g_ϕ: Time series model.

h_ψ: Vision model.

ℓ: Loss.

$\mathscr{C} = \{\mathbf{c}_i\}_{i=1}^K$: Codebook of K D-dimensional codewords $\mathbf{c}_i \in \mathbb{R}^D$ to capture the latent representations for VQ-VAE.

$\mathbf{k}$: Quantized indices in VQ-VAE.

q: Text-based quantization process.

B: Batch size in contrastive aligning.

γ: Temperature parameter for distribution concentrations in contrastive aligning.

$z(\cdot)$: Indirect tools generated by large language models.

B.5 Chapter 6: Semantic Context of Label Textual Names

$\mathscr{D}' = \{(\mathbf{x}_i, c_i)\}_{i=0}^{N}$: HAR data in conventional methods, $\mathbf{x}_i \sim \mathscr{X}, c_i \sim \mathscr{C}$, where $\mathscr{X}$ and $\mathscr{C}$ denote the input space and the label space. Each sample of time series input is denoted as $\mathbf{x}_i \in \mathbb{R}^{T_i \times v}$, where T_i is the length of the time series, and v is the number of measured variables. The label space $\mathscr{C}$ contains C classes, and each label c is an integer from $\{1, 2, \cdots, C\}$.

$\mathscr{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=0}^{N}, \mathbf{x}_i \sim \mathscr{X}, \mathbf{y}_i \sim \mathscr{Y}$: dataset in SHARE, where data space $\mathscr{X}$ is the same as conventional HAR methods, and $\mathscr{Y}$ denotes the label space in SHARE. We denote $\mathbf{y}_i = [y_{i1}, y_{i2}, \cdots, y_{ik_i}]$ as a sample human activity label sequence, where k_i is the length of the label sequence $\mathbf{y}_i$. The label space $\mathscr{Y}$ contains C classes and M tokens.

$f_\theta : \mathscr{X} \to \mathscr{Z} \subset \mathbb{R}^d$: Time series encoder.

$\mathscr{Z}$: Hidden space.

$\hat{c}_i = \text{fc}(f(\mathbf{x}_i; \theta)) \in \mathbb{R}^C$: The distribution of the predicted label after fully connected layer fc for conventional HAR model.

$\mathbf{z}_i = f(\mathbf{x}_i; \theta)$: Encoded representation.

$g_\phi : \mathscr{Z} \to \mathscr{Y}$: The label structure-constrained decoder in SHARE.

P_ϕ: Conditional probability of decoding label $\mathbf{y}_i$ from $\mathbf{x}_i$.

$\hat{\mathbf{y}}_i$: Predicted label sequence.

B.6 Chapter 7: Combining Physical and Semantic Context

J_i: Joint i.

$\mathbf{p}_{J_i,\mathscr{G}}$: Positions mapped from time domain $\mathscr{T}$ to $\mathbb{R}^3$, defined in global frame $\mathscr{G}$.

$\mathbf{q}_{J_i,\mathscr{G}\mathscr{L}}$: Orientation quaternions mapped from time domain $\mathscr{T}$ to the Special Orthogonal

Group SO(3), defined in Hamilton convention with subscript $\mathscr{G}\mathscr{L}$ representing a frame rotation from local frame $\mathscr{L}$ to global frame $\mathscr{G}$.

$\mathbf{v}_{J_i}$: Velocities.

$\mathbf{a}_{J_i}$: Accelerations.

ω_{J_i}: Angular velocities.

$\otimes$: Quaternion multiplication operator.

*: Quaternion conjugate.

$\mathbf{x}_{J_i}$: Motion time series, which can denote either $\mathbf{a}_{J_i}$ or ω_{J_i}.

$\mathbf{n}_{J_i}(t) \sim \mathscr{N}(\mathbf{0}$: Gaussian noise during simulation.

$\tilde{\mathbf{x}}_{J_i}$: Motion time series after adding Gaussian noise.

$\mathbf{R}_{J_i}^{\delta} \sim \text{Uniform}(\text{SO}(3))$: Random rotation matrix.

$\hat{\mathbf{x}}_{J_i}$: Augmented motion time series after applying random rotation matrix.

$\mathscr{G} = (\mathscr{V} = \{\hat{\mathbf{x}}_{J_i}\}_{i=1}^{V}, \mathscr{E}_s = \{(\hat{\mathbf{x}}_{J_i}, \hat{\mathbf{x}}_{J_l}) | (J_i, J_l) \in \mathscr{H}\}, \mathscr{E}_t = \{(\hat{\mathbf{x}}_{J_i}^{t-1}, \hat{\mathbf{x}}_{J_i}^{t})\}_{i=1,t=2}^{V,T})$: Input graph representation. Nodes $\mathscr{V}$ contain skeleton joints with features $\mathbf{X} \in \mathbb{R}^{C \times T \times V}$, where C, T, V represent the number of signal channels, temporal steps and joint nodes. Spatial edges $\mathscr{E}_s$ connect adjacent nodes defined by the skeleton structure $\mathscr{H}$ and temporal edges $\mathscr{E}_t$ connect temporally adjacent frames.

$\mathbf{M} \in \mathbb{R}^{C \times T \times V}$: Random mask during pre-training, where $\mathbf{M}_i \in \mathbb{R}^{C \times T}$ is $\mathbf{1}$ if joint J_i is selected, and $\mathbf{M}_i = \mathbf{0}$ if joint J_i is masked.

$\tilde{\mathbf{X}} = \mathbf{X} \odot \mathbf{M}$: Masked input node features.

$\mathbf{X}_{\text{out}}$: Spatial output features from graph network.

K_s: Spatial kernel size.

K_t: Temporal kernel size.

$\mathbf{A}_k^{il}$: Whether node $\mathbf{x}_{J_l}$ belongs to the spatial convolution sampling subset $\mathscr{S}_{J_i}^k$ of node $\mathbf{x}_{J_i}$.

$\mathbf{D}_k^{ii} = \Sigma_l(\mathbf{A}_k^{il}) + \alpha$: Normalized diagonal matrix. α is set to 0.001 to prevent empty rows.

$\Phi_k \in \mathbb{R}^{C' \times C \times 1 \times 1}$: Weights of the 1×1 convolution operation with C' denoting output channel dimension.

k: Number of generated paraphrases in text augmentation.

$\mathbf{Y}$: The original text descriptions combined with the LLM-augmented ones.

f_θ: Text encoder.

g_ϕ: Graph encoder.

h_ψ: Linear classifier during fine-tuning.

B: Batch size.

γ: Temperature parameter that controls distribution concentrations.

ℓ_{ctr}: Contrastive loss.

ℓ_{ce}: Cross-entropy loss.

$\mathbf{z}$: One-hot encoded labels.

D: Number of classes.

$\sigma(\cdot)$: The softmax operation.

Bibliography

[1] Andrea Agostinelli, Timo I. Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, Matt Sharifi, Neil Zeghidour, and Christian Frank. Musiclm: Generating music from text, 2023.

[2] Lorenzo Alfieri, Peter Burek, Emanuel Dutra, Blazej Krzeminski, David Muraro, Jutta Thielen, and Florian Pappenberger. Glofas–global ensemble streamflow forecasting and flood early warning. *Hydrology and Earth System Sciences*, 17(3):1161–1175, 2013.

[3] Kerem Altun, Billur Barshan, and Orkun Tunçel. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognition*, 43(10):3605–3620, 2010.

[4] ams OSRAM AG. Ccs811 datasheet. https://cdn.sparkfun.com/assets/learn_tutorials/1/4/3/CCS811_Datasheet-DS000459.pdf, 2016.

[5] D. Anguita, Alessandro Ghio, L. Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *The European Symposium on Artificial Neural Networks*, 2013.

[6] Rohan Anil, Andrew M. Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, Eric Chu, Jonathan H. Clark, Laurent El Shafey, Yanping Huang, Kathy Meier-Hellstern, Gaurav Mishra, Erica Moreira, Mark Omernick, Kevin Robinson, Sebastian Ruder, Yi Tay, Kefan Xiao, Yuanzhong Xu, Yujing Zhang, Gustavo Hernandez Abrego, Junwhan Ahn, Jacob Austin, Paul Barham, Jan Botha, James Bradbury, Siddhartha Brahma, Kevin Brooks, Michele Catasta, Yong Cheng, Colin Cherry, Christopher A. Choquette-Choo, Aakanksha Chowdhery, Clément Crepy, Shachi Dave, Mostafa Dehghani, Sunipa Dev, Jacob Devlin, Mark Díaz, Nan Du, Ethan Dyer, Vlad Feinberg, Fangxiaoyu Feng, Vlad Fienber, Markus Freitag, Xavier Garcia, Sebastian Gehrmann, Lucas Gonzalez, Guy Gur-Ari, Steven Hand, Hadi Hashemi, Le Hou, Joshua Howland, Andrea Hu, Jeffrey Hui, Jeremy Hurwitz, Michael Isard, Abe Ittycheriah, Matthew Jagielski, Wenhao Jia, Kathleen Kenealy, Maxim Krikun, Sneha Kudugunta, Chang Lan, Katherine Lee, Benjamin Lee, Eric Li, Music Li, Wei Li, YaGuang Li, Jian Li, Hyeontaek Lim, Hanzhao Lin, Zhongtao Liu, Frederick Liu, Marcello Maggioni, Aroma Mahendru, Joshua Maynez, Vedant Misra, Maysam Moussalem, Zachary Nado, John Nham, Eric Ni, Andrew Nystrom, Alicia Parrish, Marie Pellat, Martin Polacek, Alex Polozov, Reiner Pope, Siyuan Qiao, Emily Reif, Bryan

Richter, Parker Riley, Alex Castro Ros, Aurko Roy, Brennan Saeta, Rajkumar Samuel, Renee Shelby, Ambrose Slone, Daniel Smilkov, David R. So, Daniel Sohn, Simon Tokumine, Dasha Valter, Vijay Vasudevan, Kiran Vodrahalli, Xuezhi Wang, Pidong Wang, Zirui Wang, Tao Wang, John Wieting, Yuhuai Wu, Kelvin Xu, Yunhan Xu, Linting Xue, Pengcheng Yin, Jiahui Yu, Qiao Zhang, Steven Zheng, Ce Zheng, Weikang Zhou, Denny Zhou, Slav Petrov, and Yonghui Wu. Palm 2 technical report, 2023.

[7] Craig F Ansley and Robert Kohn. On the estimation of arima models with missing values. In *Time series analysis of irregularly observed data*. Springer, 1984.

[8] Gabriel Appleby, Linfeng Liu, and Li-Ping Liu. Kriging convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3187–3194, 2020.

[9] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670*, 2019.

[10] Irvan B Arief-Ang, Flora D Salim, and Margaret Hamilton. Da-hoc: semi-supervised domain adaptation for room occupancy prediction using co2 sensor data. In *Proceedings of the 4th ACM International Conference on Systems for Energy-Efficient Built Environments*, pages 1–10, 2017.

[11] Luca Arrotta, Claudio Bettini, Gabriele Civitarese, and Michele Fiori. Contextgpt: Infusing llms knowledge into neuro-symbolic activity recognition models. *arXiv preprint arXiv:2403.06586*, 2024.

[12] Jimmy Ba, Kevin Swersky, Sanja Fidler, and Ruslan Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions, 2015.

[13] Marc Bachlin, Meir Plotnik, Daniel Roggen, Inbal Maidan, Jeffrey M Hausdorff, Nir Giladi, and Gerhard Troster. Wearable assistant for parkinson's disease patients with the freezing of gait symptom. *IEEE Transactions on Information Technology in Biomedicine*, 14(2):436–446, 2009.

[14] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[15] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[16] Kasun Bandara, Hansika Hewamalage, Yuan-Hao Liu, Yanfei Kang, and Christoph Bergmeir. Improving the accuracy of global forecasting models using time series data augmentation. *Pattern Recognition*, 2021.

[17] Oresti Banos, Rafael Garcia, Juan A Holgado-Terriza, Miguel Damas, Hector Pomares, Ignacio Rojas, Alejandro Saez, and Claudia Villalonga. mhealthdroid: a novel framework for agile development of mobile health applications. In *International workshop on ambient assisted living*, pages 91–98. Springer, 2014.

[18] Alex Beltran and Alberto E Cerpa. Optimal hvac building control with occupancy prediction. In *Proceedings of the 1st ACM conference on embedded systems for energy-efficient buildings*, pages 168–171, 2014.

[19] Christoph Bergmeir, Rob J Hyndman, and José M Benítez. Bagging exponential smoothing methods using stl decomposition and box–cox transformation. *International journal of forecasting*, 2016.

[20] Ganapati Bhat, Nicholas Tran, Holly Shill, and Umit Y Ogras. w-har: An activity recognition dataset and framework using low-power wearable devices. *Sensors*, 20(18):5356, 2020.

[21] Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra. The mtg-jamendo dataset for automatic music tagging. ICML, 2019.

[22] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. Audiolm: a language modeling approach to audio generation, 2023.

[23] Barbara Bruno, Fulvio Mastrogiovanni, Antonio Sgorbissa, Tullio Vernazza, and Renato Zaccaria. Analysis of human behavior recognition algorithms based on acceleration data. In *2013 IEEE International Conference on Robotics and Automation*, pages 1602–1607. IEEE, 2013.

[24] S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68, 2010.

[25] Yifu Cai, Mononito Goswami, Arjun Choudhry, Arvind Srinivasan, and Artur Dubrawski. Jolt: Jointly learned representations of language and time-series. In *Deep Generative Models for Health Workshop NeurIPS 2023*, 2023.

[26] Defu Cao, Furong Jia, Sercan O Arik, Tomas Pfister, Yixiang Zheng, Wen Ye, and Yan Liu. Tempo: Prompt-based generative pre-trained transformer for time series forecasting. *arXiv preprint arXiv:2310.04948*, 2023.

[27] Hong Cao, Vincent YF Tan, and John ZF Pang. A parsimonious mixture of gaussian trees model for oversampling in imbalanced and multimodal time-series classification. *IEEE transactions on neural networks and learning systems*, 2014.

[28] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. *Advances in Neural Information Processing Systems*, 31:6775–6785, 2018.

[29] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[30] Zhiwei Cao, Ruihang Wang, Xin Zhou, and Yonggang Wen. Reducio: model reduction for data center predictive digital twins via physics-guided machine learning. In *Proceedings of the 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 1–10, 2022.

[31] Ching Chang, Wen-Chih Peng, and Tien-Fu Chen. Llm4ts: Two-stage fine-tuning for time-series forecasting with pre-trained llms. *arXiv preprint arXiv:2308.08469*, 2023.

[32] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Free-form video inpainting with 3d gated convolution and temporal patchgan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9066–9075, 2019.

[33] Zhengping Che, Yu Cheng, Shuangfei Zhai, Zhaonan Sun, and Yan Liu. Boosting deep learning risk prediction with generative adversarial networks for electronic health records. In *ICDM*, 2017.

[34] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):1–12, 2018.

[35] Changhao Chen, Xiaoxuan Lu, Andrew Markham, and Niki Trigoni. Ionet: Learning to cure the curse of drift in inertial odometry. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[36] Changhao Chen, Peijun Zhao, Chris Xiaoxuan Lu, Wei Wang, Andrew Markham, and Niki Trigoni. Oxiod: The dataset for deep inertial odometry. *arXiv preprint arXiv:1809.07491*, 2018.

[37] Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz. Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In *2015 IEEE International conference on image processing (ICIP)*, pages 168–172. IEEE, 2015.

[38] Feng-Wen Chen and Chen-Wuing Liu. Estimation of the spatial rainfall distribution using inverse distance weighting (idw) in the middle of taiwan. *Paddy and Water Environment*, 10(3):209–222, 2012.

[39] Kaixuan Chen, Lina Yao, Dalin Zhang, Xiaojun Chang, Guodong Long, and Sen Wang. Distributionally robust semi-supervised learning for people-centric sensing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 2019.

[40] Kaixuan Chen, Dalin Zhang, Lina Yao, Bin Guo, Zhiwen Yu, and Yunhao Liu. Deep learning for sensor-based human activity recognition: Overview, challenges, and opportunities. *ACM Computing Surveys (CSUR)*, 54(4), 2021.

[41] Mingkang Chen, Jingtao Sun, Kazushige Saga, Tomota Tanjo, and Kento Aida. An adaptive noise removal tool for iot image processing under influence of weather conditions. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pages 655–656, 2020.

[42] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[43] Xinlei Chen, Xiangxiang Xu, Xinyu Liu, Shijia Pan, Jiayou He, Hae Young Noh, Lin Zhang, and Pei Zhang. Pga: Physics guided and adaptive approach for mobile fine-grained air pollution estimation. In *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*, pages 1321–1330, 2018.

[44] Xinyu Chen and Lijun Sun. Bayesian temporal factorization for multidimensional time series prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[45] Yakun Chen, Xianzhi Wang, and Guandong Xu. Gatgpt: A pre-trained large language model with graph attention network for spatiotemporal imputation. *arXiv preprint arXiv:2311.14332*, 2023.

[46] Yuhua Cheng and Chenming Hu. *MOSFET modeling & BSIM3 user's guide*. Springer Science & Business Media, 1999.

[47] Belkacem Chikhaoui and Frank Gouineau. Towards automatic feature extraction for activity recognition from wearable sensors: a deep learning approach. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2017.

[48] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

[49] Ranak Roy Chowdhury, Jiacheng Li, Xiyuan Zhang, Dezhi Hong, Rajesh K Gupta, and Jingbo Shang. Primenet: Pre-training for irregular multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[50] Ranak Roy Chowdhury, Xiyuan Zhang, Jingbo Shang, Rajesh K Gupta, and Dezhi Hong. Tarnet: Task-aware reconstruction for time-series transformer. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA*, pages 14–18, 2022.

[51] Hyunseung Chung, Jiho Kim, Joon-myoung Kwon, Ki-Hyun Jeon, Min Sung Lee, and Edward Choi. Text-to-ecg: 12-lead electrocardiogram synthesis conditioned on clinical text reports. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[52] Robert B Cleveland, William S Cleveland, Jean E McRae, and Irma Terpenning. Stl: A seasonal-trend decomposition. *J. Off. Stat*, 1990.

[53] A. M. Collier-Oxandale, J. Thorson, H. Halliday, J. Milford, and M. Hannigan. Understanding the ability of low-cost mox sensors to quantify ambient vocs. *Atmospheric Measurement Techniques*, 12(3):1441–1460, 2019.

[54] A. M. Collier-Oxandale, J. Thorson, H. Halliday, J. Milford, and M. Hannigan. Understanding the ability of low-cost mox sensors to quantify ambient vocs. *Atmospheric Measurement Techniques*, 12(3):1441–1460, 2019.

[55] Kevin Cowtan and Robert G Way. Coverage bias in the hadcrut4 temperature series and its impact on recent temperature trends. *Quarterly Journal of the Royal Meteorological Society*, 140(683):1935–1944, 2014.

[56] Zhicheng Cui, Wenlin Chen, and Yixin Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

[57] Maria Luísa Lopes De Faria, Carlos Eduardo Cugnasca, and José Roberto Almeida Amazonas. Insights into iot data and an innovative dwt-based technique to denoise sensor signals. *IEEE Sensors Journal*, 18(1):237–247, 2017.

[58] Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

[59] Angus Dempster, Daniel F Schmidt, and Geoffrey I Webb. Minirocket: A very fast (almost) deterministic transform for time series classification. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 248–257, 2021.

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

[61] Yujie Ding, Shuai Jia, Tianyi Ma, Bingcheng Mao, Xiuze Zhou, Liuliu Li, and Dongming Han. Integrating stock features and global information via large language models for enhanced stock return prediction. *arXiv preprint arXiv:2310.05627*, 2023.

[62] Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. *Advances in Neural Information Processing Systems*, 35:11763–11784, 2022.

[63] Yiqun Duan, Charles Zhou, Zhen Wang, Yu-Kai Wang, and Chin-teng Lin. Dewave: Discrete encoding of eeg waves for eeg to text translation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[64] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. *arXiv preprint arXiv:2106.14112*, 2021.

[65] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[66] Joao Diogo Falcao, Prabh Simran S Baweja, Yi Wang, Akkarit Sangpetch, Hae Young Noh, Orathai Sangpetch, and Pei Zhang. Piwims: Physics informed warehouse inventory monitory via synthetic data generation. In *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*, pages 613–618, 2021.

[67] Matthew Faulkner, Annie H Liu, and Andreas Krause. A fresh perspective: Learning to sparsify for detection in massive noisy sensor networks. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 7–18, 2013.

[68] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Data augmentation using synthetic data for time series classification with deep residual networks. *arXiv preprint arXiv:1808.02455*, 2018.

[69] Mehrdad Fazli, Kamran Kowsari, Erfaneh Gharavi, Laura Barnes, and Afsaneh Doryab. Hhar-net: hierarchical human activity recognition using neural networks. In *International Conference on Intelligent Human Computer Interaction*, pages 48–58. Springer, 2021.

[70] Davide Figo, Pedro C Diniz, Diogo R Ferreira, and Joao MP Cardoso. Preprocessing techniques for context recognition from accelerometer data. *Personal and Ubiquitous Computing*, 14(7):645–662, 2010.

[71] Germain Forestier, François Petitjean, Hoang Anh Dau, Geoffrey I Webb, and Eamonn Keogh. Generating synthetic time series to augment sparse datasets. In *ICDM*, 2017.

[72] Xiaohan Fu, Jason Koh, Francesco Fraternali, Dezhi Hong, and Rajesh Gupta. Zonal air handling in commercial buildings. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '20, page 302–303, New York, NY, USA, 2020. Association for Computing Machinery.

[73] Han Gao, Luning Sun, and Jian-Xun Wang. Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels. *Physics of Fluids*, 33(7):073603, 2021.

[74] Heting Gao, Junrui Ni, Kaizhi Qian, Yang Zhang, Shiyu Chang, and Mark Hasegawa-Johnson. Wavprompt: Towards few-shot spoken language understanding with frozen language models. *arXiv preprint arXiv:2203.15863*, 2022.

[75] Jingkun Gao, Xiaomin Song, Qingsong Wen, Pichao Wang, Liang Sun, and Huan Xu. Robusttad: Robust time series anomaly detection via decomposition and convolutional neural networks. *arXiv preprint arXiv:2002.09545*, 2020.

[76] Victor Garcia Satorras, Zeynep Akata, and Max Welling. Combining generative and discriminative models for hybrid inference. *Advances in Neural Information Processing Systems*, 32, 2019.

[77] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 776–780. IEEE, 2017.

[78] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15190, 2023.

[79] JW Gjerloev. A global ground-based magnetometer initiative. *Eos, Transactions American Geophysical Union*, 90(27):230–231, 2009.

[80] Yongshun Gong, Zhibin Li, Jian Zhang, Wei Liu, Bei Chen, and Xiangjun Dong. A spatial missing value imputation method for multi-view urban statistical data. In *IJCAI*, pages 1310–1316, 2020.

[81] Yuan Gong, Hongyin Luo, Alexander H Liu, Leonid Karlinsky, and James Glass. Listen, think, and understand. *arXiv preprint arXiv:2305.10790*, 2023.

[82] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[83] Moritz A Graule and Volkan Isler. Gg-llm: Geometrically grounding large language models for zero-shot human activity forecasting in human-aware task planning. *arXiv preprint arXiv:2310.20034*, 2023.

[84] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant

Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4d: Around the world in 3,000 hours of egocentric video, 2022.

[85] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives, 2024.

[86] Nate Gruver, Marc Finzi, Shikai Qiu, and Andrew Gordon Wilson. Large language models are zero-shot time series forecasters. *arXiv preprint arXiv:2310.07820*, 2023.

[87] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5152–5161, June 2022.

[88] Wes Gurnee and Max Tegmark. Language models represent space and time. *arXiv preprint arXiv:2310.02207*, 2023.

[89] Nils Y Hammerla, Shane Halloran, and Thomas Plötz. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv preprint arXiv:1604.08880*, 2016.

[90] William Han, Jielin Qiu, Jiacheng Zhu, Mengdi Xu, Douglas Weber, Bo Li, and Ding Zhao. An empirical exploration of cross-domain alignment between language and electroencephalogram. *arXiv preprint arXiv:2208.06348*, 2022.

[91] Zhongkai Hao, Songming Liu, Yichi Zhang, Chengyang Ying, Yao Feng, Hang Su, and Jun Zhu. Physics-informed machine learning: A survey on problems, methods and applications. *arXiv preprint arXiv:2211.08064*, 2022.

[92] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[93] Lixing He, Carlos Ruiz, Mostafa Mirshekari, and Shijia Pan. Scsv2: physics-informed self-configuration sensing through vision and vibration context modeling. In *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, pages 532–537, 2020.

[94] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. Tabllm: Few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*, pages 5549–5581. PMLR, 2023.

[95] Sachini Herath, Hang Yan, and Yasutaka Furukawa. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3146–3152. IEEE, 2020.

[96] Nora Hollenstein, Jonathan Rotsztejn, Marius Troendle, Andreas Pedroni, Ce Zhang, and Nicolas Langer. Zuco, a simultaneous eeg and eye-tracking resource for natural sentence reading. *Scientific data*, 5(1):1–13, 2018.

[97] Nora Hollenstein, Marius Troendle, Ce Zhang, and Nicolas Langer. Zuco 2.0: A dataset of physiological recordings during natural reading and annotation. *arXiv preprint arXiv:1912.00903*, 2019.

[98] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70, 1979.

[99] HM Sajjad Hossain and Nirmalya Roy. Active deep learning for activity recognition with context aware annotator selection. In *KDD*, pages 1862–1870, 2019.

[100] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[101] Hailin Hu, MingJian Tang, and Chengcheng Bai. Datsing: Data augmented time series forecasting with adversarial domain adaptation. In *CIKM*, 2020.

[102] Norden E Huang, Zheng Shen, Steven R Long, Manli C Wu, Hsing H Shih, Quanan Zheng, Nai-Chyuan Yen, Chi Chao Tung, and Henry H Liu. The empirical mode decomposition and the hilbert spectrum for nonlinear and non-stationary time series analysis. *Proceedings of the Royal Society of London. Series A: mathematical, physical and engineering sciences*, 1998.

[103] Tao Huang, Songjiang Li, Xu Jia, Huchuan Lu, and Jianzhuang Liu. Neighbor2neighbor: Self-supervised denoising from single noisy images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14781–14790, 2021.

[104] Andrew T Hudak, Nicholas L Crookston, Jeffrey S Evans, David E Hall, and Michael J Falkowski. Nearest neighbor imputation of species-level, plot-scale forest structure attributes from lidar data. *Remote Sensing of Environment*, 112(5):2232–2245, 2008.

[105] Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.

[106] Brian Kenji Iwana and Seiichi Uchida. An empirical survey of data augmentation for time series classification with neural networks. *Plos one*, 2021.

[107] Jeya Vikranth Jeyakumar, Liangzhen Lai, Naveen Suda, and Mani Srivastava. Sensehar: a robust virtual activity sensor for smartphones and wearables. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, 2019.

[108] Sijie Ji, Xinzhe Zheng, and Chenshu Wu. Hargpt: Are llms zero-shot human activity recognizers? *arXiv preprint arXiv:2403.02727*, 2024.

[109] Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-llm: Time series forecasting by reprogramming large language models, 2024.

[110] Ming Jin, Qingsong Wen, Yuxuan Liang, Chaoli Zhang, Siqiao Xue, Xue Wang, James Zhang, Yi Wang, Haifeng Chen, Xiaoli Li, Shirui Pan, Vincent S. Tseng, Yu Zheng, Lei Chen, and Hui Xiong. Large models for time series and spatio-temporal data: A survey and outlook, 2023.

[111] Christopher Kadow, David Matthew Hall, and Uwe Ulbrich. Artificial intelligence reconstructs missing climate information. *Nature Geoscience*, 2020.

[112] J. Kahn, M. Riviere, W. Zheng, E. Kharitonov, Q. Xu, P.E. Mazare, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, May 2020.

[113] Yanfei Kang, Rob J Hyndman, and Feng Li. Gratis: Generating time series with diverse and controllable characteristics. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 2020.

[114] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. Lstm fully convolutional networks for time series classification. *IEEE access*, 6:1662–1669, 2017.

[115] Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate lstm-fcns for time series classification. *Neural Networks*, 116:237–245, 2019.

[116] GE Karniadakis, IG Kevrekidis, L Lu, P Perdikaris, S Wang, and L Yang. Physics-informed machine learning: Nature reviews physics. 2021.

[117] Lars Kegel, Martin Hahmann, and Wolfgang Lehner. Feature-based comparison and generation of time series. In *SSDBM*, 2018.

[118] Daniel Kelshaw and Luca Magri. Physics-informed convolutional neural networks for corruption removal on dynamical systems. *arXiv preprint arXiv:2210.16215*, 2022.

[119] Minsung Kim, Salvatore Mandrà, Davide Venturelli, and Kyle Jamieson. Physics-inspired heuristics for soft mimo detection in 5g new radio and beyond. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, pages 42–55, 2021.

[120] Ryan King, Tianbao Yang, and Bobak J Mortazavi. Multimodal pretraining of medical time series and notes. In *Machine Learning for Health (ML4H)*, pages 244–255. PMLR, 2023.

[121] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[122] Quan Kong, Ziming Wu, Ziwei Deng, Martin Klinkigt, Bin Tong, and Tomokazu Murakami. Mmact: A large-scale dataset for cross modal human action understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8658–8667, 2019.

[123] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*, 2022.

[124] Kuldeep Kurte, Kadir Amasyali, Jeffrey Munk, and Helia Zandi. Comparative analysis of model-free and model-based hvac control for residential demand response. In *Proceedings of the 8th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, pages 309–313, 2021.

[125] Egor Lakomkin, Chunyang Wu, Yassir Fathullah, Ozlem Kalinli, Michael L Seltzer, and Christian Fuegen. End-to-end speech recognition contextualization with large language models. *arXiv preprint arXiv:2309.10917*, 2023.

[126] Arthur Le Guennec, Simon Malinowski, and Romain Tavenard. Data augmentation for time series classification using convolutional neural networks. In *ECML/PKDD workshop*, 2016.

[127] Geon Lee, Wenchao Yu, Wei Cheng, and Haifeng Chen. Moat: Multi-modal augmented time series forecasting. 2023.

[128] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *arXiv preprint arXiv:1803.04189*, 2018.

[129] Zikang Leng, Amitrajit Bhattacharjee, Hrudhai Rajasekhar, Lizhe Zhang, Elizabeth Bruda, Hyeokhyen Kwon, and Thomas Plötz. Imugpt 2.0: Language-based cross modality transfer for sensor-based human activity recognition. *arXiv preprint arXiv:2402.01049*, 2024.

[130] Zikang Leng, Hyeokhyen Kwon, and Thomas Plötz. Generating virtual on-body accelerometer data from virtual textual descriptions for human activity recognition. *arXiv preprint arXiv:2305.03187*, 2023.

[131] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[132] Bing Li, Wei Cui, Wei Wang, Le Zhang, Zhenghua Chen, and Min Wu. Two-stream convolution augmented transformer for human activity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 286–293, 2021.

[133] Chenglin Li, Di Niu, Bei Jiang, Xiao Zuo, and Jianming Yang. Meta-har: Federated representation learning for human activity recognition. In *Proceedings of the Web Conference 2021*, pages 912–922, 2021.

[134] Guozhong Li, Byron Choi, Jianliang Xu, Sourav S Bhowmick, Kwok-Pan Chun, and Grace Lai-Hung Wong. Shapenet: A shapelet-neural network approach for multivariate time series classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8375–8383, 2021.

[135] Jun Li, Che Liu, Sibo Cheng, Rossella Arcucci, and Shenda Hong. Frozen language model helps ecg zero-shot learning. *arXiv preprint arXiv:2303.12311*, 2023.

[136] Junkai Li, Weizhi Ma, and Yang Liu. Modeling time series as text sequence a frequency-vectorization transformer for time series forecasting. 2023.

[137] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[138] Lixin Li and Peter Revesz. Interpolation methods for spatio-temporal geographic data. *Computers, Environment and Urban Systems*, 28(3):201–227, 2004.

[139] Lixin Li, Xingyou Zhang, James B Holt, Jie Tian, and Reinhard Piltner. Spatiotemporal interpolation methods for air pollution exposure. In *Ninth Symposium of Abstraction, Reformulation, and Approximation*, 2011.

[140] Shuheng Li, Jingbo Shang, Rajesh K Gupta, and Dezhi Hong. Sqee: A machine perception approach to sensing quality evaluation at the edge by uncertainty quantification. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, pages 277–290, 2022.

[141] Steven Cheng-Xian Li and Benjamin Marlin. Learning from irregularly-sampled time series: A missing data perspective. In *International Conference on Machine Learning*, pages 5937–5946. PMLR, 2020.

[142] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017.

[143] Swee Kiat Lim, Yi Loo, Ngoc-Trung Tran, Ngai-Man Cheung, Gemma Roig, and Yuval Elovici. Doping: Generative data augmentation for unsupervised anomaly detection with gan. In *ICDM*, 2018.

[144] Jason Lines, Sarah Taylor, and Anthony Bagnall. Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles. *ACM Transactions on Knowledge Discovery from Data*, 12(5), 2018.

[145] Che Liu, Zhongwei Wan, Sibo Cheng, Mi Zhang, and Rossella Arcucci. Etp: Learning transferable ecg representations via ecg-text pre-training. *arXiv preprint arXiv:2309.07145*, 2023.

[146] Chenxi Liu, Sun Yang, Qianxiong Xu, Zhishuai Li, Cheng Long, Ziyue Li, and Rui Zhao. Spatial-temporal large language model for traffic prediction. *arXiv preprint arXiv:2401.10134*, 2024.

[147] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018.

[148] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[149] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[150] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2684–2701, 2019.

[151] Miaomiao Liu, Sikai Yang, Wyssanie Chomsin, and Wan Du. Real-time tracking of smartwatch orientation and location by multitask learning. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, pages 120–133, 2022.

[152] Shansong Liu, Atin Sakkeer Hussain, Chenshuo Sun, and Ying Shan. Music understanding llama: Advancing text-to-music generation with question answering and captioning. *arXiv preprint arXiv:2308.11276*, 2023.

[153] Shengzhong Liu, Tomoyoshi Kimura, Dongxin Liu, Ruijie Wang, Jinyang Li, Suhas Diggavi, Mani Srivastava, and Tarek Abdelzaher. Focal: Contrastive learning for multimodal time-series sensing signals in factorized orthogonal latent space. *Advances in Neural Information Processing Systems*, 36, 2024.

[154] Tiantian Liu, Ming Gao, Feng Lin, Chao Wang, Zhongjie Ba, Jinsong Han, Wenyao Xu, and Kui Ren. Wavoice: A noise-resistant multi-modal speech recognition system fusing mmwave and audio signals. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, pages 97–110, 2021.

[155] Wenxin Liu, David Caruso, Eddy Ilg, Jing Dong, Anastasios I Mourikis, Kostas Daniilidis, Vijay Kumar, and Jakob Engel. Tlio: Tight learned inertial odometry. *IEEE Robotics and Automation Letters*, 5(4):5653–5660, 2020.

[156] Xin Liu, Daniel McDuff, Geza Kovacs, Isaac Galatzer-Levy, Jacob Sunshine, Jiening Zhan, Ming-Zher Poh, Shun Liao, Paolo Di Achille, and Shwetak Patel. Large language models are few-shot health learners. *arXiv preprint arXiv:2305.15525*, 2023.

[157] Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. Unitime: A language-empowered unified model for cross-domain time series forecasting. *arXiv preprint arXiv:2310.09751*, 2023.

[158] Yang Liu, Zhenjiang Li, Zhidan Liu, and Kaishun Wu. Real-time arm skeleton tracking and gesture inference tolerant to missing wearable sensors. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, pages 287–299, 2019.

[159] Yukai Liu, Rose Yu, Stephan Zheng, Eric Zhan, and Yisong Yue. Naomi: Non-autoregressive multiresolution sequence imputation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2019.

[160] Aleksej Logacjov, Kerstin Bach, Atle Kongsvold, Hilde Bremseth Bårdstu, and Paul Jarle Mork. Harth: A human activity recognition dataset for machine learning. *Sensors*, 21(23):7853, 2021.

[161] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 851–866. 2023.

[162] Chris Xiaoxuan Lu, Muhamad Risqi U Saputra, Peijun Zhao, Yasin Almalioglu, Pedro PB De Gusmao, Changhao Chen, Ke Sun, Niki Trigoni, and Andrew Markham. milliego: single-chip mmwave radar aided egomotion estimation via deep sensor fusion. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pages 109–122, 2020.

[163] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916*, 2022.

[164] Wenjie Luo, Zhenyu Yan, Qun Song, and Rui Tan. Phyaug: Physics-directed data augmentation for deep sensing model transfer in cyber-physical systems. In *Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2021)*, pages 31–46, 2021.

[165] Yonghong Luo, Xiangrui Cai, Y. Zhang, Jun Xu, and Xiaojie Yuan. Multivariate time series imputation with generative adversarial networks. In *Neural Information Processing Systems*, 2018.

[166] Yonghong Luo, Ying Zhang, Xiangrui Cai, and Xiaojie Yuan. E2gan: End-to-end generative adversarial network for multivariate time series imputation. In *AAAI Press*, pages 3094–3100, 2019.

[167] Haojie Ma, Wenzhong Li, Xiao Zhang, Songcheng Gao, and Sanglu Lu. Attnsense: Multi-level attention mechanism for multimodal human activity recognition. In *IJCAI*, pages 3109–3115, 2019.

[168] Jiawei Ma, Zheng Shou, Alireza Zareian, Hassan Mansour, Anthony Vetro, and Shih-Fu Chang. Cross-dimensional self-attention for multivariate, geo-tagged time series imputation. 2019.

[169] Qianli Ma, Zhen Liu, Zhenjing Zheng, Ziyang Huang, Siying Zhu, Zhongzhong Yu, and James T Kwok. A survey on time-series pre-trained models. *arXiv preprint arXiv:2305.10716*, 2023.

[170] Qianli Ma, Zhenjing Zheng, Jiawei Zheng, Sen Li, Wanqing Zhuang, and Garrison W Cottrell. Joint-label learning by dual augmentation for time series classification. In *AAAI*, 2021.

[171] Yuchao Ma and Hassan Ghasemzadeh. Labelforest: Non-parametric semi-supervised learning for activity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4520–4527, 2019.

[172] Mohammad Malekzadeh, Richard G Clegg, Andrea Cavallaro, and Hamed Haddadi. Mobile sensor data anonymization. In *Proceedings of the international conference on internet of things design and implementation*, pages 49–58, 2019.

[173] Moe Matsuki, Paula Lago, and Sozo Inoue. Characterizing word embeddings for zero-shot sensor-based human activity recognition. *Sensors*, 19(22):5043, 2019.

[174] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322, 2010.

[175] Congcong Miao, Jiajun Fu, Jilong Wang, Heng Yu, Botao Yao, Anqi Zhong, Jie Chen, and Zekun He. Predicting crowd flows via pyramid dilated deeper spatial-temporal network. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 806–814, 2021.

[176] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. Large language models as general pattern machines. *arXiv preprint arXiv:2307.04721*, 2023.

[177] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Alireza Dirafzoon, Aparajita Saraf, Amy Bearman, and Babak Damavandi. Imu2clip: Multimodal contrastive learning for imu motion sensors from egocentric videos and text. *arXiv preprint arXiv:2210.14395*, 2022.

[178] Seungwhan Moon, Andrea Madotto, Zhaojiang Lin, Tushar Nagarajan, Matt Smith, Shashank Jain, Chun-Fu Yeh, Prakash Murugesan, Peyman Heidari, Yue Liu, Kavya Srinet, Babak Damavandi, and Anuj Kumar. Anymal: An efficient and scalable any-modality augmented language model, 2023.

[179] Nick Moran, Dan Schmidt, Yu Zhong, and Patrick Coady. Noisier2noise: Learning to denoise from unpaired noisy data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12064–12072, 2020.

[180] Colin P Morice, John J Kennedy, Nick A Rayner, and Phil D Jones. Quantifying uncertainties in global and regional temperature change using an ensemble of observational estimates: The hadcrut4 data set. *Journal of Geophysical Research: Atmospheres*, 117(D8), 2012.

[181] Colin P Morice, John J Kennedy, Nick A Rayner, JP Winn, Emma Hogan, RE Killick, RJH Dunn, TJ Osborn, PD Jones, and IR Simpson. An updated assessment of near-surface temperature change from 1850: the hadcrut5 dataset. *Journal of Geophysical Research: Atmospheres*, page e2019JD032361, 2020.

[182] Morten Morup, Daniel M Dunlavy, Evrim Acar, and Tamara Gibson Kolda. Scalable tensor factorizations with missing data. Technical report, Sandia National Laboratories, 2010.

[183] Srinarayana Nagarathinam, Yashovardhan S Chati, Malini Pooni Venkat, and Arunchandar Vasan. Pacman: physics-aware control manager for hvac. In *Proceedings of the 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 11–20, 2022.

[184] Srinarayana Nagarathinam, Arunchandar Vasan, Venkata Ramakrishna P, Shiva R Iyer, Venkatesh Sarangan, and Anand Sivasubramaniam. Centralized management of hvac energy in large multi-ahu zones. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 157–166, 2015.

[185] Gue-Hwan Nam, Seok-Jun Bu, Na-Mu Park, Jae-Yong Seo, Hyeon-Cheol Jo, and Won-Tae Jeong. Data augmentation using empirical mode decomposition on neural networks to classify impact noise in vehicle. In *ICASSP*, 2020.

[186] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

[187] Wisconsin Department of Health Service. Carbon dioxide. https://www.dhs.wisconsin.gov/chemical/carbondioxide.htm, 2023.

[188] Jungwoo Oh, Seongsu Bae, Gyubok Lee, Joon-myoung Kwon, and Edward Choi. Ecg-qa: A comprehensive question answering dataset combined with electrocardiogram. *arXiv preprint arXiv:2306.15681*, 2023.

[189] Olufemi A Omitaomu, Vladimir A Protopopescu, and Auroop R Ganguly. Empirical mode decomposition technique with conditional mutual information for denoising operational sensor data. *IEEE sensors journal*, 11(10):2565–2575, 2011.

[190] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023.

[191] Francisco Javier Ordóñez and Daniel Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1):115, 2016.

[192] Stefan Palzer. Photoacoustic-based gas sensing: A review. *Sensors (Basel)*, 20(9):2745, May 2020.

[193] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

[194] Jaeyeon Park, Woojin Nam, Jaewon Choi, Taeyeong Kim, Dukyong Yoon, Sukhoon Lee, Jeongyeup Paek, and JeongGil Ko. Glasses for the third eye: Improving the quality of clinical data analysis with motion sensor-based data filtering. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, pages 1–14, 2017.

[195] Matthias Plappert, Christian Mandery, and Tamim Asfour. The kit motion-language dataset. *Big data*, 4(4):236–252, 2016.

[196] Hangwei Qian, Sinno Pan, and Chunyan Miao. Sensor-based activity recognition via learning from distributions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[197] Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis, 2023.

[198] Jielin Qiu, William Han, Jiacheng Zhu, Mengdi Xu, Michael Rosenberg, Emerson Liu, Douglas Weber, and Ding Zhao. Transfer knowledge from natural language to electrocardiography: Can we detect cardiovascular disease through language models? *arXiv preprint arXiv:2301.09017*, 2023.

[199] Stephan Rabanser, Tim Januschowski, Valentin Flunkert, David Salinas, and Jan Gasthaus. The effectiveness of discretization in forecasting: An empirical study on neural time series models. *arXiv preprint arXiv:2005.10111*, 2020.

[200] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[201] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[202] Valentin Radu, Catherine Tong, Sourav Bhattacharya, Nicholas D Lane, Cecilia Mascolo, Mahesh K Marina, and Fahim Kawsar. Multimodal deep learning for activity and context recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):1–27, 2018.

[203] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[204] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[205] Giorgia Ramponi, Pavlos Protopapas, Marco Brambilla, and Ryan Janssen. T-cgan: Conditional generative adversarial network for data augmentation in noisy time series with irregular sampling. *arXiv preprint arXiv:1811.08295*, 2018.

[206] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. Lag-llama: Towards foundation models for probabilistic time series forecasting, 2024.

[207] Attila Reiss and Didier Stricker. Introducing a new benchmarked dataset for activity monitoring. In *2012 16th international symposium on wearable computers*, pages 108–109. IEEE, 2012.

[208] Hongyu Ren, Russell Stewart, Jiaming Song, Volodymyr Kuleshov, and Stefano Ermon. Learning with weak supervision from physics and data-driven constraints. *AI Magazine*, 39(1):27–38, 2018.

[209] Daniel Roggen, Alberto Calatroni, Mirco Rossi, Thomas Holleczek, Kilian Förster, Gerhard Tröster, Paul Lukowicz, David Bannach, Gerald Pirkl, Alois Ferscha, Jakob Doppler, Clemens Holzmann, Marc Kurz, Gerald Holl, Ricardo Chavarriaga, Hesam Sagha, Hamidreza Bayati, Marco Creatura, and José del R. Millán. Collecting complex activity datasets in highly rich networked sensor environments. *2010 Seventh International Conference on Networked Sensing Systems (INSS)*, pages 233–240, 2010.

[210] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[211] Paul K. Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, Hannah Muckenhirn, Dirk Padfield, James Qin, Danny Rozenberg, Tara Sainath, Johan Schalkwyk, Matt Sharifi, Michelle Tadmor Ramanovich, Marco Tagliasacchi, Alexandru Tudor, Mihajlo Velimirović, Damien Vincent, Jiahui Yu, Yongqiang Wang, Vicky Zayats, Neil Zeghidour, Yu Zhang, Zhishuai Zhang, Lukas Zilka, and Christian Frank. Audiopalm: A large language model that can speak and listen, 2023.

[212] Alex Rubinsteyn and Sergey Feldman. fancyimpute: An imputation library for python.

[213] S N Rudnick and D K Milton. Risk of indoor airborne infection transmission estimated from carbon dioxide concentration. *Indoor Air*, 13(3):237–245, September 2003.

[214] Divya Saxena and Jiannong Cao. D-gan: Deep generative adversarial nets for spatio-temporal prediction. *arXiv preprint arXiv:1907.08556*, 2019.

[215] Patrick Schäfer and Ulf Leser. Multivariate time series classification with weasel+ muse. *arXiv preprint arXiv:1711.11343*, 2017.

[216] Sheng Shen, Mahanth Gowda, and Romit Roy Choudhury. Closing the gaps in inertial motion tracking. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, pages 429–444, 2018.

[217] Sheng Shen, He Wang, and Romit Roy Choudhury. I am a smartwatch and i can track my user's arm. In *Proceedings of the 14th annual international conference on Mobile systems, applications, and services*, pages 85–96, 2016.

[218] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12026–12035, 2019.

[219] Ahmed Shifaz, Charlotte Pelletier, François Petitjean, and Geoffrey I Webb. Ts-chief: a scalable and accurate forest algorithm for time series classification. *Data Mining and Knowledge Discovery*, 34(3):742–775, 2020.

[220] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. Fusion of smartphone motion sensors for physical activity recognition. *Sensors*, 14(6):10146–10176, 2014.

[221] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. Complex human activity recognition using smartphone and wrist-worn motion sensors. *Sensors*, 16(4):426, 2016.

[222] Mohammad Shokoohi-Yekta, Jun Wang, and Eamonn Keogh. On the non-trivial generalization of dynamic time warping to the multi-dimensional case. In *Proceedings of the 2015 SIAM international conference on data mining*, pages 289–297. SIAM, 2015.

[223] Shezheng Song, Xiaopeng Li, and Shasha Li. How to bridge the gap between modalities: A comprehensive survey on multimodal large language model. *arXiv preprint arXiv:2311.07594*, 2023.

[224] Dimitris Spathis and Fahim Kawsar. The first step is the hardest: Pitfalls of representing and tokenizing temporal data for large language models. *arXiv preprint arXiv:2309.06236*, 2023.

[225] Michael L Stein. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 2012.

[226] Odongo Steven Eyobu and Dong Seog Han. Feature representation and data augmentation for human activity classification based on wearable imu sensor data using a deep lstm neural network. *Sensors*, 2018.

[227] Yixuan Su, Tian Lan, Huayang Li, Jialu Xu, Yan Wang, and Deng Cai. Pandagpt: One model to instruction-follow them all. *arXiv preprint arXiv:2305.16355*, 2023.

[228] Chenxi Sun, Yaliang Li, Hongyan Li, and Shenda Hong. Test: Text prototype aligned embedding to activate llm's ability for time series. *arXiv preprint arXiv:2308.08241*, 2023.

[229] Scott Sun, Dennis Melamed, and Kris Kitani. Idol: Inertial deep orientation-estimation and localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6128–6137, 2021.

[230] Yifei Sun, Yuxuan Liu, Ziteng Wang, Xiaolei Qu, Dezhi Zheng, and Xinlei Chen. C-ridge: Indoor co2 data collection system for large venues based on prior knowledge. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, pages 1077–1082, 2022.

[231] Timo Sztyler and Heiner Stuckenschmidt. On-body localization of wearable devices: An investigation of position-aware activity recognition. In *2016 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–9. IEEE, 2016.

[232] Naoya Takeishi and Alexandros Kalousis. Physics-integrated variational autoencoders for robust and interpretable generative modeling. *Advances in Neural Information Processing Systems*, 34:14809–14821, 2021.

[233] Koh Takeuchi, Hisashi Kashima, and Naonori Ueda. Autoregressive tensor factorization for spatio-temporal predictions. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 1105–1110. IEEE, 2017.

[234] Sabera J Talukder and Georgia Gkioxari. Time series modeling at scale: A universal representation across tasks and domains. 2023.

[235] Shuhan Tan, Tushar Nagarajan, and Kristen Grauman. Egodistill: Egocentric head motion distillation for efficient video understanding. *Advances in Neural Information Processing Systems*, 36:33485–33498, 2023.

[236] Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. Salmonn: Towards generic hearing abilities for large language models. *arXiv preprint arXiv:2310.13289*, 2023.

[237] Xianfeng Tang, Boqing Gong, Yanwei Yu, Huaxiu Yao, Yandong Li, Haiyong Xie, and Xiaoyu Wang. Joint modeling of dense and incomplete trajectories for citywide traffic volume inference. In *The World Wide Web Conference*, pages 1806–1817, 2019.

[238] Xianfeng Tang, Huaxiu Yao, Yiwei Sun, Charu Aggarwal, Prasenjit Mitra, and Suhang Wang. Joint modeling of local and global temporal dynamics for multivariate time series forecasting with missing values. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5956–5963, 2020.

[239] Waldo R Tobler. A computer movie simulating urban growth in the detroit region. *Economic geography*, 46(sup1):234–240, 1970.

[240] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. *arXiv preprint arXiv:2106.00750*, 2021.

[241] Catherine Tong, Jinchen Ge, and Nicholas D Lane. Zero-shot learning for imu-based activity recognition using video embeddings. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 5(4):1–23, 2021.

[242] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[243] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.

[244] Terry T Um, Franz MJ Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kulić. Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. In *ICMI*, 2017.

[245] CO2Meter.com. K30 datasheet. https://cdn.shopify.com/s/files/1/0019/5952/files/DS-SenseAir-K30-CO2-Sensor-Revised-061722_96d4f229-643c-45b0-95a6-24efa252490d.pdf?v=1672160950, 2022.

[246] Astrid Ustad, Aleksej Logacjov, Stine Øverengen Trollebø, Pernille Thingstad, Beatrix Vereijken, Kerstin Bach, and Nina Skjæret Maroni. Validation of an activity type recognition model classifying daily physical behavior in older adults: the har70+ model. *Sensors*, 23(5):2368, 2023.

[247] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning, 2018.

[248] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[249] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Michal Walczak, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker.

Informed machine learning - a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, page 1–1, 2021.

[250] Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima I Lunze, Wojciech Samek, and Tobias Schaeffter. Ptb-xl, a large publicly available electrocardiography dataset. *Scientific data*, 7(1):154, 2020.

[251] Kai Wang, Bengbeng He, and Wei-Ping Zhu. Tstnn: Two-stage transformer based neural network for speech enhancement in the time domain. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7098–7102. IEEE, 2021.

[252] Rui Wang, Karthik Kashinath, Mustafa Mustafa, Adrian Albert, and Rose Yu. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1457–1466, 2020.

[253] Rui Wang and Rose Yu. Physics-guided deep learning for dynamical systems: A survey. *arXiv preprint arXiv:2107.01272*, 2021.

[254] Tianrui Wang, Long Zhou, Ziqiang Zhang, Yu Wu, Shujie Liu, Yashesh Gaur, Zhuo Chen, Jinyu Li, and Furu Wei. Viola: Unified codec language models for speech recognition, synthesis, and translation. *arXiv preprint arXiv:2305.16107*, 2023.

[255] Wei Wang, Chunyan Miao, and Shuji Hao. Zero-shot human activity recognition via nonlinear compatibility based method. In *Proceedings of the International Conference on Web Intelligence*, pages 322–330, 2017.

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

[257] Xinglei Wang, Meng Fang, Zichao Zeng, and Tao Cheng. Where would i go next? large language models as human mobility predictors. *arXiv preprint arXiv:2308.15197*, 2023.

[258] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018.

[259] Zhenhailong Wang and Heng Ji. Open vocabulary electroencephalography-to-text decoding and zero-shot sentiment classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 5350–5358, 2022.

[260] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[261] Ziqi Wang, Ankur Sarker, Jason Wu, Derek Hua, Gaofeng Dong, Akash Deep Singh, and Mani Srivastava. Capricorn: Towards real-time rich scene analysis using rf-vision sensor fusion. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, pages 334–348, 2022.

[262] Manuel Weber, Christoph Doblander, and Peter Mandl. Detecting building occupancy with synthetic environmental data. In *Proceedings of the 7th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 324–325, 2020.

[263] Kevin Weekly, Nikolaos Bekiaris-Liberis, Ming Jin, and Alexandre M Bayen. Modeling and estimation of the humans' effect on the co 2 dynamics inside a conference room. *IEEE Transactions on Control Systems Technology*, 23(5):1770–1781, 2015.

[264] Gary M Weiss. Wisdm smartphone and smartwatch activity and biometrics dataset. *UCI Machine Learning Repository: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set*, 7:133190–133202, 2019.

[265] Gary M Weiss, Kenichi Yoneda, and Thaier Hayajneh. Smartphone and smartwatch-based biometrics using activities of daily living. *IEEE Access*, 7:133190–133202, 2019.

[266] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*, 2020.

[267] Waskitho Wibisono, Dedy Nur Arifin, Baskoro Adi Pratomo, Tohari Ahmad, and Royyana M Ijtihadie. Falls detection and notification system using tri-axial accelerometer and gyroscope sensors of a smartphone. In *2013 Conference on Technologies and Applications of Artificial Intelligence*, pages 382–385. IEEE, 2013.

[268] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[269] Christopher Wimmer and Navid Rekabsaz. Leveraging vision-language models for granular market change prediction. *arXiv preprint arXiv:2301.10166*, 2023.

[270] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. In *The eleventh international conference on learning representations*, 2022.

[271] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *NeurIPS*, 2021.

[272] Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*, 2023.

[273] Tong Wu, Yiqiang Chen, Yang Gu, Jiwei Wang, Siyu Zhang, and Zhanghu Zhechen. Multi-layer cross loss model for zero-shot human activity recognition. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 210–221. Springer, 2020.

[274] Yuankai Wu, Dingyi Zhuang, Aurelie Labbe, and Lijun Sun. Inductive graph neural networks for spatiotemporal kriging. *arXiv preprint arXiv:2006.07527*, 2020.

[275] Jiannan Xiang, Guangyi Liu, Yi Gu, Qiyue Gao, Yuting Ning, Yuheng Zha, Zeyu Feng, Tianhua Tao, Shibo Hao, Yemin Shi, Zhengzhong Liu, Eric P. Xing, and Zhiting Hu. Pandora: Towards general world model with natural language actions and video states. 2024.

[276] Pingping Xie and Phillip A Arkin. Global precipitation: A 17-year monthly analysis based on gauge observations, satellite estimates, and numerical model outputs. *Bulletin of the American Meteorological Society*, 78(11):2539–2558, 1997.

[277] Qianqian Xie, Weiguang Han, Yanzhao Lai, Min Peng, and Jimin Huang. The wall street neophyte: A zero-shot analysis of chatgpt over multimodal stock movement prediction challenges. *arXiv preprint arXiv:2304.05351*, 2023.

[278] Qianqian Xie, Weiguang Han, Xiao Zhang, Yanzhao Lai, Min Peng, Alejandro Lopez-Lira, and Jimin Huang. Pixiu: A large language model, instruction data and evaluation benchmark for finance. *arXiv preprint arXiv:2306.05443*, 2023.

[279] Tianwei Xing, Luis Garcia, Federico Cerutti, Lance Kaplan, Alun Preece, and Mani Srivastava. Deepsqa: Understanding sensor data via question answering. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation*, pages 106–118, 2021.

[280] Jun Xu, Yuan Huang, Ming-Ming Cheng, Li Liu, Fan Zhu, Zhou Xu, and Ling Shao. Noisy-as-clean: Learning self-supervised denoising from corrupted image. *IEEE Transactions on Image Processing*, 29:9316–9329, 2020.

[281] Shichao Xu, Yangyang Fu, Yixuan Wang, Zhuoran Yang, Zheng O'Neill, Zhaoran Wang, and Qi Zhu. Accelerate online reinforcement learning for building hvac control with heterogeneous expert guidances. In *Proceedings of the 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 89–98, 2022.

[282] Hao Xue and Flora D Salim. Promptcast: A new prompt-based learning paradigm for time series forecasting. 2022.

[283] Hao Xue and Flora D Salim. Utilizing language models for energy load forecasting. In *Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, pages 224–227, 2023.

[284] Hao Xue, Bhanu Prakash Voutharoja, and Flora D Salim. Leveraging language foundation models for human mobility forecasting. In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems*, pages 1–9, 2022.

[285] Hang Yan, Qi Shan, and Yasutaka Furukawa. Ridi: Robust imu double integration. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 621–636, 2018.

[286] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[287] Yan Yan, Dali Chen, Yushi Liu, Jinjin Zhao, Bo Wang, Xuankun Wu, Xiaohao Jiao, Yuqian Chen, Huihui Li, and Xuchao Ren. Tnda-har, 2021.

[288] Dongchao Yang, Jinchuan Tian, Xu Tan, Rongjie Huang, Songxiang Liu, Xuankai Chang, Jiatong Shi, Sheng Zhao, Jiang Bian, Xixin Wu, Zhou Zhao, Shinji Watanabe, and Helen Meng. Uniaudio: An audio foundation model toward universal audio generation, 2023.

[289] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. Learning physics constrained dynamics using autoencoders. *Advances in Neural Information Processing Systems*, 35:17157–17172, 2022.

[290] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 351–360, 2017.

[291] Xinyu Yi, Yuxiao Zhou, Marc Habermann, Soshi Shimada, Vladislav Golyanik, Christian Theobalt, and Feng Xu. Physical inertial poser (pip): Physics-aware real-time human motion tracking from sparse inertial sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13167–13178, 2022.

[292] Xiuwen Yi, Yu Zheng, Junbo Zhang, and Tianrui Li. St-mvl: filling missing values in geo-sensory time series data. 2016.

[293] Minghao Yin, Zhuliang Yao, Yue Cao, Xiu Li, Zheng Zhang, Stephen Lin, and Han Hu. Disentangled non-local neural networks. In *European Conference on Computer Vision*, pages 191–207. Springer, 2020.

[294] Jinsung Yoon, Daniel Jarrett, and Mihaela Van der Schaar. Time-series generative adversarial networks. 2019.

[295] Alexander D Young, Martin J Ling, and Damal K Arvind. Imusim: A simulation environment for inertial sensing algorithm design and evaluation. In *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 199–210. IEEE, 2011.

[296] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *ICCV*, 2019.

[297] Xinli Yu, Zheng Chen, Yuan Ling, Shujing Dong, Zongyi Liu, and Yanbin Lu. Temporal data meets llm–explainable financial time series forecasting. *arXiv preprint arXiv:2306.11025*, 2023.

[298] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8980–8987, 2022.

[299] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? *arXiv preprint arXiv:1912.10077*, 2019.

[300] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

[301] Zhi Zeng and Qiang Ji. Knowledge based activity recognition with dynamic bayesian network. In *European conference on computer vision*, pages 532–546. Springer, 2010.

[302] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2114–2124, 2021.

[303] Daochen Zha, Kwei-Herng Lai, Kaixiong Zhou, and Xia Hu. Towards similarity-aware time-series classification. *arXiv preprint arXiv:2201.01413*, 2022.

[304] Chi Zhang, Sanmukh R Kuppannagari, Rajgopal Kannan, and Viktor K Prasanna. Building hvac scheduling using reinforcement learning via neural network based model approximation. In *Proceedings of the 6th ACM international conference on systems for energy-efficient buildings, cities, and transportation*, pages 287–296, 2019.

[305] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. *arXiv preprint arXiv:2305.11000*, 2023.

[306] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, pages 7354–7363, 2019.

[307] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[308] Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Shaoli Huang, Yong Zhang, Hongwei Zhao, Hongtao Lu, and Xi Shen. T2m-gpt: Generating human motion from textual descriptions with discrete representations. *arXiv preprint arXiv:2301.06052*, 2023.

[309] Jiayun Zhang, Xiyuan Zhang, Xinyang Zhang, Dezhi Hong, Rajesh K. Gupta, and Jingbo Shang. Navigating Alignment for Non-identical Client Class Sets: A Label Name-Anchored Federated Learning Framework. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, aug 2023.

[310] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.

[311] Lin Zhang, Alexander Gorovits, Wenyu Zhang, and Petko Bogdanov. Learning periods from incomplete multivariate time series. In *ICDM*, 2020.

[312] Mi Zhang and Alexander A Sawchuk. Usc-had: A daily activity dataset for ubiquitous activity recognition using wearable sensors. In *Proceedings of the 2012 ACM conference on ubiquitous computing*, pages 1036–1043, 2012.

[313] Shibo Zhang, Yaxuan Li, Shen Zhang, Farzad Shahabi, Stephen Xia, Yu Deng, and Nabil Alshurafa. Deep learning in human activity recognition with wearable sensors: A review on advances. *arXiv preprint arXiv:2111.00418*, 2021.

[314] Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in Neural Information Processing Systems*, 35:3988–4003, 2022.

[315] Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh K Gupta, and Jingbo Shang. Large language models for time series: A survey. *arXiv preprint arXiv:2402.01801*, 2024.

[316] Xiyuan Zhang, Ranak Roy Chowdhury, Dezhi Hong, Rajesh K Gupta, and Jingbo Shang. Modeling label semantics improves activity recognition. *arXiv preprint arXiv:2301.03462*, 2023.

[317] Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, and Dezhi Hong. Esc-gan: Extending spatial coverage of physical sensors. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 1347–1356, 2022.

[318] Xiyuan Zhang, Ranak Roy Chowdhury, Jingbo Shang, Rajesh Gupta, and Dezhi Hong. Towards diverse and coherent augmentation for time-series forecasting. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[319] Xiyuan Zhang, Ranak Roy Chowdhury, Jiayun Zhang, Dezhi Hong, Rajesh K. Gupta, and Jingbo Shang. Unleashing the power of shared label structures for human activity recognition. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, CIKM '23, page 3340–3350, 2023.

[320] Xiyuan Zhang, Xiaohan Fu, Diyan Teng, Chengyu Dong, Keerthivasan Vijayakumar, Jiayun Zhang, Ranak Roy Chowdhury, Junsheng Han, Dezhi Hong, Rashmi Kulkarni,

Jingbo Shang, and Rajesh K. Gupta. Physics-informed data denoising for real-life sensing systems. In *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, SenSys '23, page 83–96. Association for Computing Machinery, 2024.

[321] Xuchao Zhang, Yifeng Gao, Jessica Lin, and Chang-Tien Lu. Tapnet: Multivariate time series classification with attentional prototypical network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6845–6852, 2020.

[322] Yingxue Zhang, Yanhua Li, Xun Zhou, Xiangnan Kong, and Jun Luo. Curb-gan: Conditional urban traffic estimation through spatio-temporal generative adversarial networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 842–852, 2020.

[323] Yunkai Zhang, Yawen Zhang, Ming Zheng, Kezhen Chen, Chongyang Gao, Ruian Ge, Siyuan Teng, Amine Jelloul, Jinmeng Rao, Xiaoyuan Guo, Chiang-Wei Fang, Zeyu Zheng, and Jie Yang. Insight miner: A large-scale multimodal model for insight mining from time series. In *NeurIPS 2023 AI for Science Workshop*, 2023.

[324] Zheng Zhang, Hossein Amiri, Zhenke Liu, Andreas Züfle, and Liang Zhao. Large language models for spatial trajectory patterns mining. *arXiv preprint arXiv:2310.04942*, 2023.

[325] Ziyuan Zhong, Davis Rempe, Yuxiao Chen, Boris Ivanovic, Yulong Cao, Danfei Xu, Marco Pavone, and Baishakhi Ray. Language-guided traffic simulation via scene-level diffusion. *arXiv preprint arXiv:2306.06344*, 2023.

[326] Fengtao Zhou, Sheng Huang, and Yun Xing. Deep semantic dictionary learning for multi-label image classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3572–3580, 2021.

[327] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, 2021.

[328] Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. *arXiv preprint arXiv:2302.11939*, 2023.

[329] Yunjiao Zhou, Jianfei Yang, Han Zou, and Lihua Xie. Tent: Connect language models with iot sensors for zero-shot activity recognition. *arXiv preprint arXiv:2311.08245*, 2023.

[330] Jincao Zhu, Youngbin Im, Shivakant Mishra, and Sangtae Ha. Calibrating time-variant, device-specific phase noise for cots wifi devices. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, pages 1–12, 2017.

[331] Magauiya Zhussip, Shakarim Soltanayev, and Se Young Chun. Extending stein's unbiased risk estimator to train deep denoisers with correlated pairs of noisy images. *Advances in neural information processing systems*, 32, 2019.