Generalized Prompt Tuning: Adapting Frozen Univariate Time Series Foundation Models for Multivariate Healthcare Time Series

Carnegie Mellon University

Mingzhu Liu [1] Angela H. Chen [1] George H. Chen [2]

[1] CMU School of Computer Science, [2] CMU Heinz College of Information Systems and Public Policy

Introduction

Q: How do you adapt univariate time series foundation models to your healthcare data?

A: Just **fine-tune it like LLMs,** for example:

- Full fine-tuning
- LoRA
- Linear probing
- Prompt tuning (P-Tuning v2)

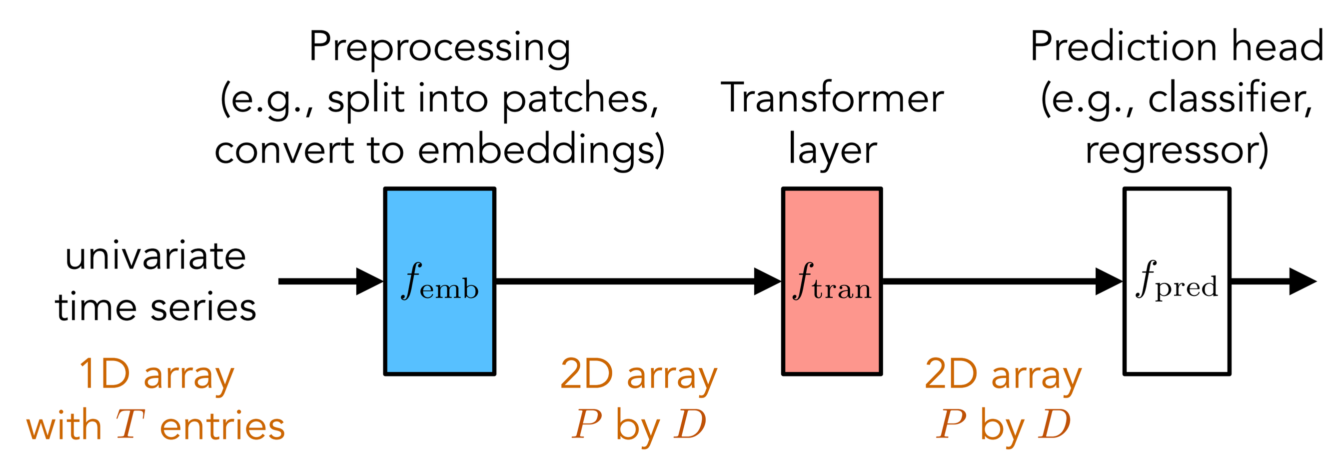
Or try our fine-tuning strategy for **multivariate time series** that we call **Generalized Prompt Tuning (Gen-P-Tuning)**

Contributions

- First benchmark on fine-tuning univariate time series foundation models on multivariate time series tasks
  - **Classify** in MIMIC-III:
    - in-hospital mortality
    - phenotype
  - **Forecast** influenza-like illness
- We propose Gen-P-Tuning
  - Parameter-efficient
  - Performs well in practice
  - Special cases: linear probing, standard prompt tuning

Background

Pre-trained univariate foundation model:

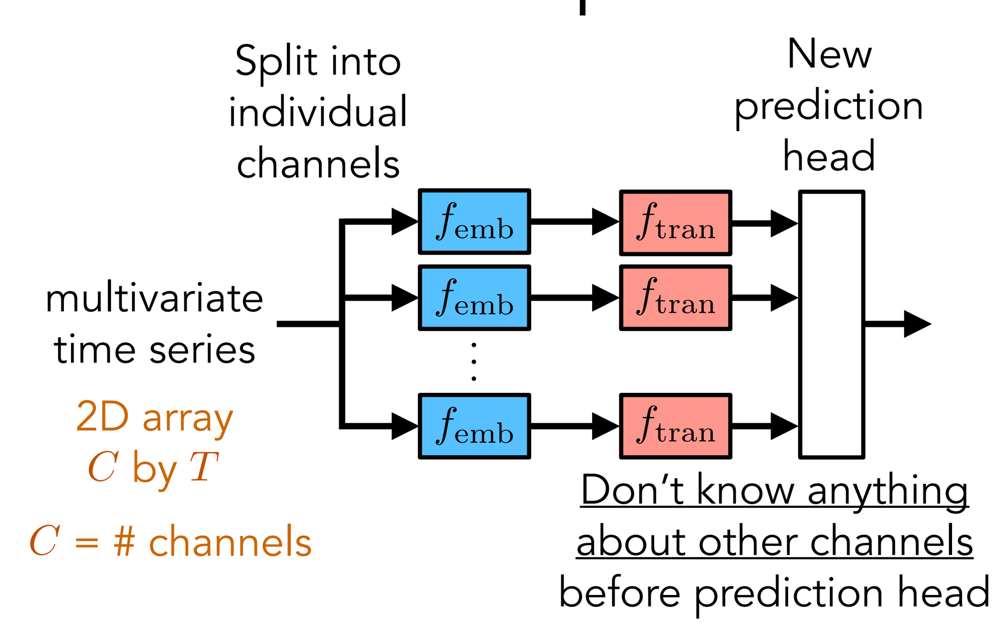


Can use multiple transformer layers (for simplicity, in this poster we use 1 transformer layer)

T = # time steps
P = # patches
D = embedding dimension

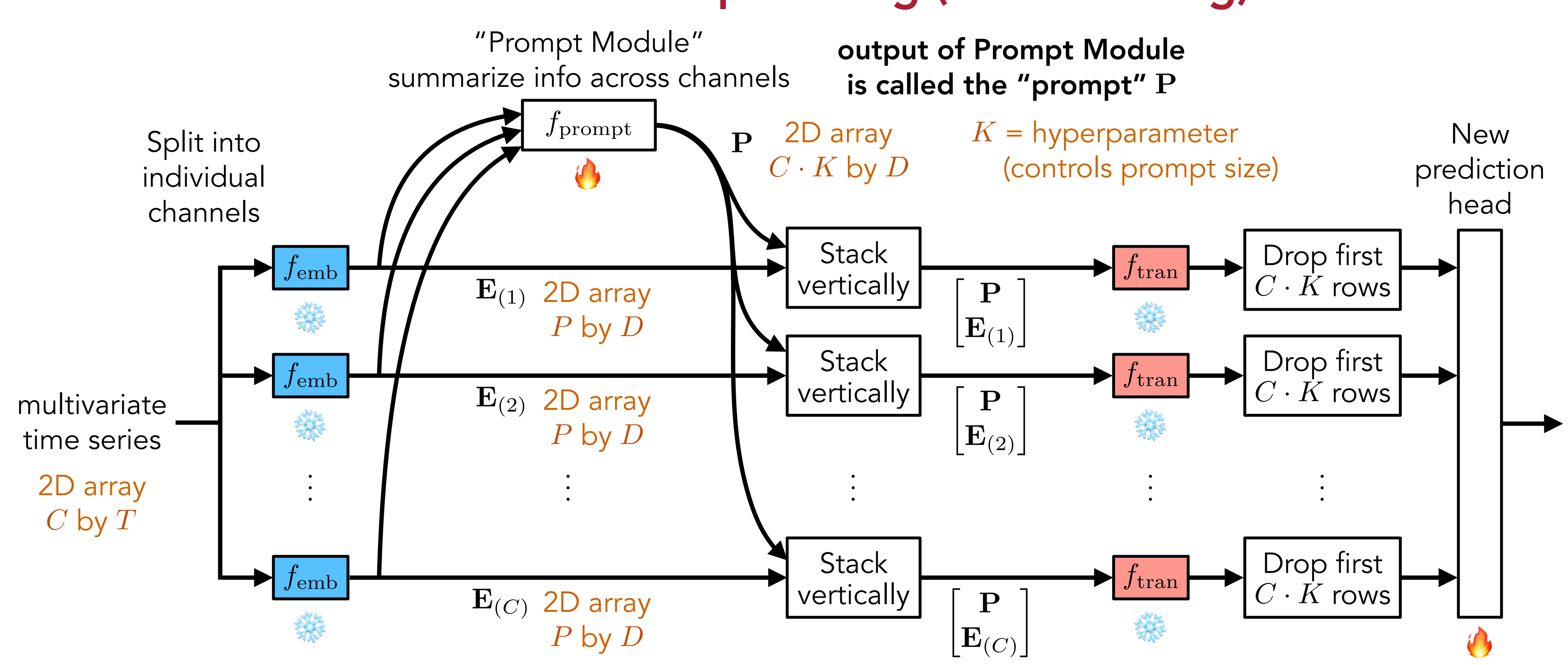
Example models: MOMENT, GPT4TS, …

Handle multivariate time series with "channel independence":



We call this **"linear probing"**

Generalized Prompt Tuning (Gen-P-Tuning)



Behaves like adding $C \cdot K$ initial fictitious patches as a prefix

- The prompt $\mathbf{P}$ is a **summary across channels** that we attach as a prefix to each channel's embedding representation
- The prompt is learned through **Prompt Module** f_{prompt}
  - Can be any neural net for sequential data (eg., transformer, RNN, …)
  - If Prompt Module does not depend on inputs: get standard Prompt Tuning
  - If prompt size hyperparameter $K = 0$: get standard linear probing
- Only the Prompt Module and prediction head need to be trained 🔥

Experimental Results

For more detailed results, see the paper

MIMIC Mortality Classification

Model	Fine-Tuning Method	AUROC	AUPRC
MOMENT	Full	0.687 ± 0.020	0.255 ± 0.038
	LoRA	0.720 ± 0.019	0.272 ± 0.025
	Linear Probing	<u>0.730</u> ± 0.035	0.260 ± 0.018
	Prompt Tuning	0.724 ± 0.035	0.274 ± 0.020
	Gen-P-Tuning	**0.754** ± 0.021	**0.292** ± 0.026
GPT4TS	Full	**0.743** ± 0.018	**0.309** ± 0.023
	LoRA	0.708 ± 0.056	0.254 ± 0.024
	Linear Probing	<u>0.737</u> ± 0.033	<u>0.265</u> ± 0.037
	Prompt Tuning	0.689 ± 0.062	0.236 ± 0.022
	Gen-P-Tuning	0.708 ± 0.025	0.255 ± 0.038
STraTS	N/A	0.601 ± 0.039	0.159 ± 0.039

MIMIC Phenotype Classification

Model	Fine-Tuning Method	AUROC	AUPRC
MOMENT	Full	<u>0.643</u> ± 0.019	<u>0.276</u> ± 0.021
	LoRA	0.640 ± 0.025	0.273 ± 0.027
	Linear Probing	0.631 ± 0.026	0.264 ± 0.022
	Prompt Tuning	0.634 ± 0.012	0.268 ± 0.015
	Gen-P-Tuning	**0.666** ± 0.015	**0.294** ± 0.012
GPT4TS	Full	0.593 ± 0.014	<u>0.234</u> ± 0.014
	LoRA	<u>0.596</u> ± 0.023	**0.241** ± 0.015
	Linear Probing	0.555 ± 0.016	0.213 ± 0.015
	Prompt Tuning	0.581 ± 0.012	0.227 ± 0.014
	Gen-P-Tuning	**0.599** ± 0.010	0.231 ± 0.010
STraTS	N/A	0.573 ± 0.020	0.217 ± 0.016

Influenza-Like Illness Forecasting

Model	Fine-Tuning Method	MSE
MOMENT	Full	3.199 ± 0.102
	LoRA	3.109 ± 0.021
	Linear Probing	**2.622** ± 0.036
	Prompt Tuning	<u>2.918</u> ± 0.047
	Gen-P-Tuning	3.083 ± 0.080
GPT4TS	Full	3.219 ± 0.093
	LoRA	3.247 ± 0.337
	Linear Probing	3.202 ± 0.303
	Prompt Tuning	<u>3.105</u> ± 0.429
	Gen-P-Tuning	**2.939** ± 0.378

In these tables, Gen-P-Tuning uses a nontrivial Prompt Module

MIMIC Mortality # Trainable Parameters

Model (# params)	Full	LoRA	Linear Probing	Prompt Tuning	Gen-P-Tuning
	Fine-Tuning Method				
MOMENT (342k)	100%	0.2%	0.1%	0.2%	0.2%
GPT4TS (61.0k)	100%	1.9%	0.2%	2.1%	2.7%

Findings

- Gen-P-Tuning often performs the best
- Linear probing sometimes perform well, suggesting that channel-independence is sometimes sufficient
- Keep in mind that Gen-P-Tuning includes linear probing and Prompt Tuning as special cases: can choose between special cases by tuning on a validation set

This work was supported in part by NSF CAREER award #2047981