The Capacity and Robustness Trade-off: Revisiting the Channel Independent Strategy for Multivariate Time Series Forecasting

Lu Han, Han-Jia Ye, De-Chuan Zhan
State Key Laboratory for Novel Software Technology, Nanjing University
{hanlu,yehj}@lamda.nju.edu.cn,zhandc@nju.edu.cn

ABSTRACT

Multivariate time series data comprises various channels of variables. The multivariate forecasting models need to capture the relationship between the channels to accurately predict future values. However, recently, there has been an emergence of methods that employ the Channel Independent (CI) strategy. These methods view multivariate time series data as separate univariate time series and disregard the correlation between channels. Surprisingly, our empirical results have shown that models trained with the CI strategy outperform those trained with the Channel Dependent (CD) strategy, usually by a significant margin. Nevertheless, the reasons behind this phenomenon have not yet been thoroughly explored in the literature. This paper provides comprehensive empirical and theoretical analyses of the characteristics of multivariate time series datasets and the CI/CD strategy. Our results conclude that the CD approach has higher capacity but often lacks robustness to accurately predict distributionally drifted time series. In contrast, the CI approach trades capacity for robust prediction. Practical measures inspired by these analyses are proposed to address the capacity and robustness dilemma, including a modified CD method called Predict Residuals with Regularization (PRReg) that can surpass the CI strategy. We hope our findings can raise awareness among researchers about the characteristics of multivariate time series and inspire the construction of better forecasting models.

PVLDB Reference Format:
Lu Han, Han-Jia Ye, De-Chuan Zhan. The Capacity and Robustness Trade-off: Revisiting the Channel Independent Strategy for Multivariate Time Series Forecasting. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX W **PVLDB Artifact Availability:**

The source code, data, and/or other artifacts have been made available at https://github.com/hanlu-nju/channel_independent_MTSF.

1 INTRODUCTION

Time series forecasting is a critical area of research that finds applications in both industry and academia. Multivariate time series are common and comprise multiple channels of variates that are usually correlated, with examples ranging from stock market prices and traffic flows to solar power plant outputs and temperatures across various cities [20]. With the powerful representation capability of deep models, channel correlation can be implicitly learned or explicitly modeled by performing forecasting tasks [6, 7, 22, 39, 40]. Two widely used methods for time series forecasting are recurrent neural networks (RNNs) and convolutional neural networks (CNNs). RNNs model successive time points based on the Markov assumption [5, 16, 32], while CNNs extract variation information along the temporal dimension using techniques such as temporal convolutional networks (TCNs) [2, 12]. However, due to the Markov assumption in RNN and the local reception property in TCN, both of the two models are unable to capture the long-term dependencies in sequential data. Recently, Transformers with attention mechanisms have gained increasing popularity in other fields like natural language processing [8], speech recognition [9], and even computer vision [10]. Researchers have also explored the potential of Transformer models in long-term multivariate time series forecasting (MTSF) tasks [24, 38, 43, 44].

Despite the significant progress made by Transformer-based methods in forecasting long-term future values, a recent paper questions the effectiveness of Transformer [42]. The authors have demonstrated that a simple linear model can outperform all state-of-the-art Transformer-based methods. However, it is important to note that the linear model used by the authors employs a channel-independent training strategy that is different from previous works. Instead of considering all the channels as a whole, the authors train a univariate forecast model that is shared across all the channels. This training strategy is closely related to the global [33] or cross-learning [34] approach when there is a set of related univariate time series. Global methods assume that all the time series in the set come from the same process and fit a single univariate forecasting function [31]. Despite the heterogeneity of real-world time series, global methods have demonstrated unexpectedly good performance [13, 21, 29]. [27] attribute the improvement to the relief of overfit by larger number of training examples. Multivariate time series can be viewed as a collection of multiple interdependent series that are synchronized in time. However, it is necessary to consider all channels of the variables in order to fully capture the characteristics of the object at each time step. Moreover, disregarding the correlation between channels can result in incomplete modeling. Therefore, the effectiveness of a channel-dependent strategy in improving the modeling of multivariate time series remains to be thoroughly investigated, along with the underlying reasons for its success.

This paper conducts a comprehensive investigation of the two training strategies that have emerged in recent works on multivariate time series forecasting. The first strategy is the Channel-Dependent (CD) approach, which predicts future values by taking into account the historical data of all the channels [24, 38, 43, 44]. The second strategy is the Channel-Independent (CI) approach, which treats multivariate time series as separate univariate time

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

series and constructs a multivariate forecaster using univariate forecasting functions [28, 42]. With this strategy, the predicted value of a particular channel depends solely on its own historical values, while the other channels are ignored. Intuitively, since an object cannot be fully described by considering only one of its features, the CI is supposed to perform poorly. We test the two strategies with various kinds of machine learning algorithms on 9 commonly used long-term forecast datasets. Interestingly, our results indicate that, regardless of the algorithm used, the CI strategy consistently outperforms the CD strategy, often by a substantial margin.

To explore the reasons behind this, we examined the linear model as an illustrative example, both theoretically and empirically. First, we observed the distribution drift in the real-world multivariate time series. Specifically, we found that the autocorrelation functions (ACFs) of each channel, which are relevant to the linear model, exhibit substantial differences between training and testing phases. Next, we demonstrated that the linear model using CI strategy relies solely on the mean of ACFs across all channels, while CD strategy relies on each ACF separately. Given that the mean ACF drifts less than most of the channel ACFs, this leads to CI strategy achieving superior performance. Our analysis led us to the conclusion that CI and CD exhibit different trade-offs in terms of capacity and robustness. Specifically, CI has lower capacity but better robustness, whereas CD is the opposite.

Through our analyses, we give some practice to improve the performance of existing algorithms. First, we propose an new objective called Predict Residuals with Regularization (PRReg). This objective is designed to address the non-robustness of the CD strategy and has demonstrated superior performance compared to both the original CD and CI strategies in the majority of cases. Furthermore, we have identified several factors that may influence algorithm performance. By taking these factors into consideration and implementing the PRReg objective, it may be possible to further enhance algorithm performance.

We conclude our contribution as follows:

- We present the Channel Dependent (CD) and Channel Independent (CI) training strategies for multivariate time series forecasting, and find that CI outperformed CD by a significant margin, despite ignoring channel correlation.
- Through theoretical and empirical analysis on linear model, we identified that CI has high capacity and low robustness, while CD has low capacity and high robustness. In real-world non-stationary time series forecasting, robustness is more important, which explains CI's superior performance in most cases.
- We presented practical strategies for improving forecasting model performance, including the use of the Predict Residuals with Regularization (PRReg) objective and other factors that can influence CD and CI performance.

2 PRELIMINARIES

In this section, we introduce the concepts of Multivariate Time Series Forecasting (MTSF), Channel Dependent (CD) Strategy, and Channel Independent (CI) Strategy.

2.1 Multivariate Time Series Forecasting

MTSF deals with time series data that contain multiple variables, or channels, at each time step. Given historical values $\boldsymbol{X} \in \mathbb{R}^{L \times C}$ where L represents the length of the look-back window, and C is the number of channels. the goal of MTSF is to predict the future values $\boldsymbol{Y} \in \mathbb{R}^{H \times C}$, where $H > 0$ is the forecast horizon.

2.2 Channel Dependent (CD) Strategy

The CD strategy involves building a model that forecasts the future values of each channel by considering all the history of all the channels. Most of the multivariate forecaster employ this strategy [33, 38, 43, 44]. To be specified, the objective of CD model is the minimize the forecasting risk $\mathcal{R}$:

$$\min_f \mathcal{R}(f) = \min_f \mathbb{E}_{(\boldsymbol{X},\boldsymbol{Y})} \ell(f(\boldsymbol{X}), \boldsymbol{Y}). \tag{1}$$

ℓ is the regression loss. We apply the commonly used L-2 (MSE) loss unless specified otherwise [11, 38, 42–44]. To minimize the expectation objective (eq. (1)), the model f is trained by the empirical loss on the training set $\{(\boldsymbol{X}^{(i)}, \boldsymbol{Y}^{(i)})\}_{i=1}^{N}$. This is referred to as the *Channel Dependent (CD) loss*:

$$\min_f \frac{1}{N} \sum_{i=1}^{N} \ell(f(\boldsymbol{X}^{(i)}), \boldsymbol{Y}^{(i)}). \tag{2}$$

Here N is the number of time series used for training.

2.3 Channel Independent (CI) Strategy

Alternatively, multivariate time series can also be viewed as a set of multiple time series, *i.e.*, the given look-back window $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_C]$ and the target $Y = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_C]$, where $\boldsymbol{x}_c \in R^L, \boldsymbol{y}_c \in R^H, 1 \le c \le C$ is history and future values of the univariate time seires for c-th channel. In this case, a forecast model can be learned by the following Channel Independent (CI) loss:

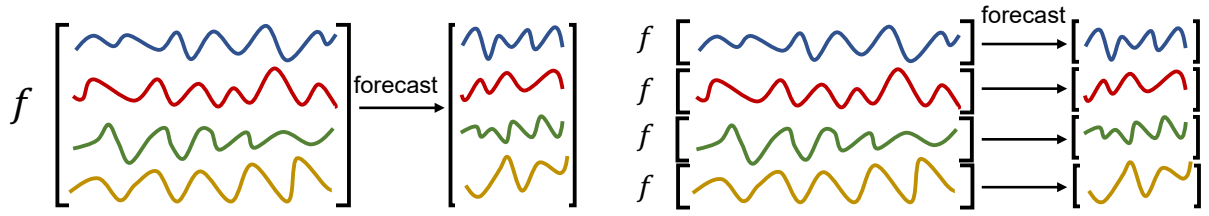
$$\min_f \frac{1}{NC} \sum_{i=1}^{N} \sum_{c=1}^{C} \ell(f(\boldsymbol{x}_c^{(i)}), \boldsymbol{y}_c^{(i)}). \tag{3}$$

The CI loss is the mean of the losses of all channels, with each channel's loss being minimized independently.

In fig. 1, we illustrate the difference between the Channel Dependent (CD) and Channel Independent (CI) strategies for multivariate time series forecasting. CD takes all the channels of a time series as input and aims to capture the relationships between them, while CI handles each channel independently. It is natural to assume that CD would outperform CI, but in the next section, we demonstrate that the opposite is true across different benchmarks and algorithms, including both non-deep and deep methods.

3 EMPIRICAL COMPARISON OF CD AND CI

The previous section introduced two strategies – CD and CI – for solving multivariate time series forecasting tasks. While CD considers all channels, one might assume that models trained with CD would outperform CI by a significant margin. Surprisingly, the opposite is true: **CI outperforms CD in most cases**. In this section, we present empirical comparisons of CD and CI across diverse datasets on various methods, including recent Transformer-based



(a) Channel Dependent (CD) Strategy (b) Channel Independent (CI) Strategy

Figure 1: Comparison of two training strategies for Multivariate Time Series Forecasting (MTSF) tasks. The left shows the Channel Dependent (CD) strategy where all the channels are taken as input and forecasted future values depend on the history of all the channels. The right shows the Channel Independent (CI) strategy, which treats the multivariate series as multiple univariate series and trains a unified model on these series. The prediction of each channel depends solely on its own historical values, and the relationship between different channels is ignored.

methods. Additionally, we provide both theoretical and empirical analyses to explain the reasons behind these results.

3.1 Experiment Setup

Datasets. We conduct extensive experiments on nine widely-used, real-world datasets that cover five mainstream time series forecasting applications, namely energy, traffic, economics, weather, and disease. The datasets include:

- **ETT (Electricity Transformer Temperature)** [43][1] comprises two hourly-level datasets (ETTh) and two 15-minute-level datasets (ETTm). Each dataset contains seven oil and load features of electricity transformers from July 2016 to July 2018.
- **Traffic**[2] describes the road occupancy rates. It contains the hourly data recorded by the sensors of San Francisco freeways from 2015 to 2016.
- **Electricity**[3] collects the hourly electricity consumption of 321 clients from 2012 to 2014.
- **Exchange-Rate** [20][4] collects the daily exchange rates of 8 countries from 1990 to 2016.
- **Weather**[5] includes 21 indicators of weather, such as air temperature, and humidity. Its data is recorded every 10 min for 2020 in Germany.
- **ILI**[6] describes the ratio of patients seen with influenza-like illness and the number of patients. It includes weekly data from the Centers for Disease Control and Prevention of the United States from 2002 to 2021.

We also summarize the datasets in table 1.

Evaluation metrics. In line with previous research [38, 42–44], we compare the performance of different methods using two primary evaluation metrics: Mean Squared Error (MSE) and Mean Absolute Error (MAE).

Compared methods. Our analysis includes a range of methods, including non-deep models, Transformer-based models, and other deep learning models. Specifically:

[1]https://github.com/zhouhaoyi/ETDataset
[2]http://pems.dot.ca.gov
[3]https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014
[4]https://github.com/laiguokun/multivariate-time-series-data
[5]https://www.bgc-jena.mpg.de/wetter/
[6]https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

Table 1: Statistics of nine multivariate time series datasets.

Dataset(s)	channels	Timesteps	Granularity
ETTh1&ETTh2	7	17,420	1hour
ETTm1&ETTm2	7	69,680	5min
Traffic	862	17,544	1hour
Electricity	321	26,304	1hour
Exchange-Rate	8	7,588	1day
Weather	21	52,696	10min
ILI	7	966	1week

- **Non-deep methods.** We select two popular non-deep models in recent time – **Linear** [42] and **GBRT** [11]. Following the practice in [42], we use the auto-gradient framework [30] to implement the Linear model and optimize it using gradient descent, even though it is non-deep. The results are reproduced by their codes in the public repository [7]. For GBRT, we integrate the XGBoost [4] implementation in the repository [8]. **Linear** is a representative linear model and **GBRT** is a non-linear model.
- **Deep methods.** Transformers are popular and enjoy rapid development in long-term multivariate forecast tasks. We include two recent Transformer-based methods: **Informer** [43] and traditional **Transformer** [36]. Codes are taken from the Informer repository [9]. For generality, we also include the CNN-based method **TCN** [2], RNN-based method **DeepAR** [33] and a simple two-layer **MLP** model with ReLU activation.

Other details. For each experiment, we set the length of the look-back window to 36 for ILI and 96 for other datasets. These values follow the setup in [43] and differ from the values in [42]. When using the CD strategy for Linear and GBRT, we flatten the input as the feature for these models. Specifically, for a look-back window with L time steps and C channels, the input feature has a dimensionality of LC. However, this approach may result in high input dimensionality when dealing with datasets with many channels, such as the Traffic dataset, which has 862 channels. This can lead to computational and storage issues for dense methods like Linear. Therefore, we report only those results that are feasible for one RTX 3090 GPU.

[7]https://github.com/cure-lab/LTSF-Linear
[8]https://github.com/Daniela-Shereen/GBRT-for-TSF
[9]https://github.com/zhouhaoyi/Informer2020

3.2 Main Results

The study's results are presented in tables 2 and 3 where the algorithms' performance is measured by MAE and MSE, respectively. A violin plot in fig. 2 illustrates the performance distribution across the seven algorithms. The study reveals several noteworthy results.

CI outperforms CD in the majority of cases. (1) CI significantly enhances the performance of almost all algorithms, with an average improvement of at least 20%. OOn complex and dense algorithms like MLP, Transformer, and Informer, the improvement exceeds 30%. Simple methods like linear experience less improvement. In most cases, replacing the CD strategy with CI yields significant improvement (>10%). On all algorithms, the improvement is observed in more than half of the cases. Only **3** cases show a significant drop (<-10%) in MAE and **9** in MSE, while the number of significant improvements is **92** in MAE and 95 in MSE. (2) On most benchmarks, CI improves performance consistently. This is apparent in the left seven datasets, where the improvement is consistent. On ETTh2, CI improves performance by at least 30%, while Weather and ILI show less improvement. Nonetheless, CI remains superior, as evidenced by the performance distribution in fig. 2.

CI strategy narrows the performance difference. Figure 2 reveals that the CI strategy has not only a lower error mean but also a smaller variance than the CD strategy. This indicates that when using the CI strategy, the model performance does not differ significantly. With the exception of Weather, methods with the CI strategy achieve the best results on the other datasets. But the best methods vary. For instance, on Electricity, GBRT and Transformer yield the best results. MLP achieves the best outcome on the 96 horizon of ETTh2. While most state-of-the-art (SOTA) results are achieved using linear, other methods are not too far behind.

Conclusion. This section demonstrates that changing the CD strategy to the CI strategy can significantly enhance the performance of multivariate forecasting methods. Hence, the superiority of some recent methods is mainly due to the training strategy rather than the algorithm's design [42]. For a fair comparison, the training strategy and algorithm should be decoupled. The subsequent section explores why the CI strategy outperforms and elucidates the trade-off between capacity and robustness.

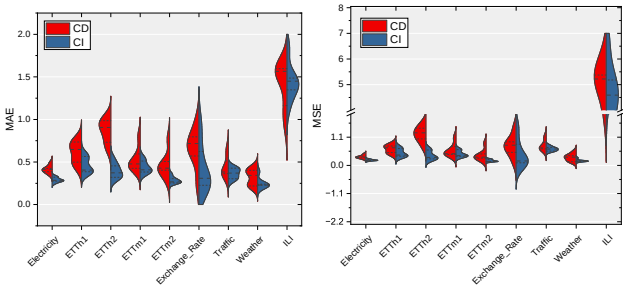


Figure 2: The performance distribution of 7 models utilizing the CI and CD strategy. Values come from table 2 and table 3. The prediction length is 24 for ILI dataset and 48 for the others. In most cases, CI has a lower error mean and a smaller variance than CD strategy. It means that CI performs better than CD. Also, when using CI strategy, the model performance does not differ very much.

4 ANALYSIS

This section aims to provide an in-depth analysis of why CI is superior for multivariate forecasting tasks in most cases, using the Linear [42] model as an example. It is closely related to the AutoRegression (AR) in statistics [3]. We begin by examining the presence of distribution shift in real-world datasets. Subsequently, we evaluate the Linear model with CI and CD strategies, demonstrating how the drifted statistics impact its performance. Our analysis highlights the fact that CI reduces the statistics gap between the training and test data. Finally, we decompose the risk to demonstrate that CI trades capacity for robustness, which translates to improved performance on many real-world non-stationary time series.

4.1 Distribution Drift

Real-world datasets are characterized by time series with changing values over time, often accompanied by changes in the underlying distribution, referred to as non-stationarity [1, 18]. In this section, we investigate the AutoCorrelation Function (ACF), which is commonly used in time series analysis:

Definition 4.1 (AutoCorrelation Function (ACF) [25])**.** The autocorrelation function of a stochastic process, $\{X(t)\}$, is defined as:

$$\rho\left(t_1, t_2\right) = \frac{\gamma\left(t_1, t_2\right)}{\sqrt{\sigma^2\left(t_1\right)\sigma^2\left(t_2\right)}}$$

where $\gamma\left(t_1, t_2\right) = \mathrm{Cov}\left[X(t_1), X(t_2)\right]$ is the covariance function, and $\sigma^2(t) = \gamma(t, t)$ is the variance at time t.

If the process is stationary, then ACF is only a function of the time difference $\tau = t_2 - t_1$, *i.e.*:

$$\rho(\tau) = \frac{\gamma(\tau)}{\gamma(0)}, \tag{4}$$

where $\gamma(\tau) = \mathrm{Cov}\left[X(t), X(t+\tau)\right]$.

In this paper, we temporarily assume that the stochastic process of each channel is stationary. But we will show that our analysis results still holds in real-world data. To estimate the AutoCorrelation Function (ACF) of a given time series $\boldsymbol{x} \in \mathbb{R}^T$, we employ the method from [19], which is a commonly used practice. Specifically, we use the following equation to estimate the ACF:

$$\hat{\rho}(\tau) = \frac{\hat{\gamma}(\tau)}{\hat{\gamma}(0)} \tag{5}$$

where $\hat{\gamma}(\tau) = \frac{1}{T-k}\sum_{t=1}^{T-\tau}\left(\boldsymbol{x}_t - \bar{\boldsymbol{x}}\right)\left(\boldsymbol{x}_{t+\tau} - \bar{\boldsymbol{x}}\right)$ is the estimated covariance function, $\bar{x} = \frac{1}{T}\sum_{i=1}^{T}\boldsymbol{x}_t$ is the mean value of $\boldsymbol{x}$.

We display the ACF of the train series and test series on each dataset in fig. 3, following the train-test split used in previous works [38, 43, 44]. From fig. 3, it is evident that distribution drift is present in each dataset, owing to various reasons. For instance, in (MCIL, ETTh2) and (146, Electricity), the anomaly in the training series leads to distribution drift. In (OT, ILI) and (wv (m/s), Weather), variation in the trend is the main cause. The rest of two figures can not be concluded by simple reasons. Nevertheless, distribution drift is a prevalent phenomenon in real-world time series datasets.

The observed distribution drift has a profound impact on the performance of machine learning models, as the fundamental assumption of these models is that the training and test data are drawn

Table 2: MAE on nine multivariate time series datasets across various forecasting models. CD means taking the Channel Dependent strategy where the algorithm takes all the channels in the look-back window as input. CI means the algorithm takes each channel as an individual univariate series and trains a shared model. For each benchmark, we mark the best model results in bold. We also display the improvement percentage by CI relative to CD. The significant improvement (> 10%) and significant drop (< −10%) are marked by bold red and bold green respectively. The last column displays the number of significant improvement/total cases, significant drop/total cases and average improvement (%) respectively.

Dataset	Electricity		ETTh1		ETTh2		ETTm1		ETTm2		Exchange_Rate		Traffic		Weather		ILI		Mean
Horizon	48	96	48	96	48	96	48	96	48	96	48	96	48	96	48	96	24	36	
Linear (CD)	0.488	0.493	0.426	0.497	0.645	0.961	0.427	0.441	0.277	0.372	0.246	0.366	-	-	0.219	0.247	**0.955**	**0.945**	11/16
Linear (CI)	0.275	0.279	**0.374**	**0.398**	**0.302**	0.373	0.382	**0.377**	**0.251**	**0.285**	**0.164**	**0.218**	0.428	0.397	0.229	0.261	1.163	1.135	2/16
Improve (%)	+43.57	+43.39	+12.26	+19.87	+53.28	+61.15	+10.49	+14.35	+9.37	+23.29	+33.29	+40.52	-	-	-4.81	-5.36	-21.75	-20.09	+19.55
GBRT (CD)	-	-	0.499	0.560	0.732	0.936	0.424	0.477	0.404	0.517	0.719	0.912	-	-	0.236	0.268	1.597	1.554	12/14
GBRT (CI)	**0.249**	0.256	0.385	0.415	0.454	0.614	**0.360**	0.380	0.297	0.355	0.336	0.401	**0.282**	0.286	**0.190**	**0.232**	1.459	1.501	0/14
Improve (%)	-	-	+22.87	+25.84	+37.92	+34.38	+15.00	+20.41	+26.49	+31.49	+53.19	+56.05	-	-	+19.49	+13.61	+8.62	+3.44	+26.34
MLP (CD)	0.385	0.398	0.523	0.625	1.028	1.543	0.480	0.511	0.439	0.410	0.617	0.676	26.834	26.054	0.218	0.251	1.161	1.254	12/18
MLP (CI)	0.287	0.289	0.395	0.422	0.319	**0.365**	0.453	0.483	0.266	0.294	0.265	0.255	0.406	0.388	0.230	0.261	1.358	1.369	1/18
Improve (%)	+25.43	+27.43	+24.45	+32.52	+68.93	+76.36	+5.66	+5.38	+39.39	+28.09	+57.07	+62.33	+98.49	+98.51	-5.67	-4.00	-16.89	-9.14	+34.13
DeepAR (CD)	0.401	0.378	0.668	0.763	0.938	1.042	0.594	0.612	0.505	0.662	0.795	0.874	0.361	0.386	0.395	0.456	1.593	1.570	13/18
DeepAR (CI)	0.330	0.342	0.587	0.594	0.541	0.597	0.511	0.520	0.304	0.354	0.623	0.660	0.370	0.410	0.240	0.287	1.449	1.454	0/18
Improve (%)	+17.72	+9.51	+12.14	+22.17	+42.28	+42.66	+14.03	+15.09	+39.76	+46.54	+21.65	+24.55	-2.46	-6.08	+39.29	+37.09	+9.00	+7.38	+21.80
TCN (CD)	0.423	0.440	0.647	0.746	0.985	0.985	0.803	0.712	0.769	0.841	0.971	0.955	0.627	0.637	0.427	0.399	1.600	1.482	12/18
TCN (CI)	0.322	0.349	0.405	0.471	0.441	0.585	0.555	0.502	0.358	0.386	0.929	0.971	0.441	0.469	0.388	0.411	1.837	1.593	1/18
Improve (%)	+23.75	+20.69	+37.42	+36.92	+55.20	+40.65	+30.83	+29.45	+53.44	+54.16	+4.29	-1.69	+29.63	+26.37	+9.26	-2.89	-14.81	-7.49	+23.62
Informer (CD)	0.424	0.424	0.766	0.959	0.906	1.386	0.477	0.568	0.428	0.478	0.717	0.769	0.403	0.416	0.402	0.371	1.565	1.590	15/18
Informer (CI)	0.285	0.285	0.509	0.655	0.372	0.427	0.408	0.447	0.264	0.350	0.308	0.312	0.337	0.297	0.228	0.343	1.486	1.552	0/18
Improve (%)	+32.90	+32.90	+33.56	+31.77	+58.89	+69.23	+14.54	+21.29	+38.41	+26.74	+56.99	+59.48	+16.31	+28.58	+43.27	+7.31	+5.08	+2.40	+32.20
Transformer (CD)	0.352	0.357	0.734	0.774	0.829	1.111	0.458	0.533	0.404	0.547	0.571	0.769	0.364	0.359	0.343	0.452	1.508	1.555	17/18
Transformer (CI)	0.281	**0.255**	0.565	0.501	0.347	0.461	0.407	0.466	0.254	0.321	0.227	0.312	0.303	**0.273**	0.232	0.287	1.348	1.525	0/18
Improve (%)	+20.06	+28.66	+23.03	+35.36	+58.18	+58.51	+11.30	+12.62	+37.15	+41.27	+60.25	+59.48	+16.70	+23.78	+32.57	+36.49	+10.62	+1.88	+31.55

Table 3: MSE on nine multivariate time series datasets across various forecasting models.

Dataset	Electricity		ETTh1		ETTh2		ETTm1		ETTm2		Exchange_Rate		Traffic		Weather		ILI		Mean
Horizon	48	96	48	96	48	96	48	96	48	96	48	96	48	96	48	96	24	36	
Linear (CD)	0.442	0.444	0.402	0.514	0.711	1.520	0.404	0.433	0.161	0.269	0.119	0.274	-	-	0.142	0.165	**2.343**	**2.436**	11/16
Linear (CI)	0.195	0.196	**0.345**	**0.386**	**0.226**	**0.319**	0.354	**0.351**	**0.147**	**0.189**	**0.051**	**0.088**	0.703	0.651	0.169	0.202	2.847	2.857	4/16
Improve (%)	+55.88	+55.91	+14.17	+24.94	+68.16	+79.04	+12.20	+18.85	+8.45	+29.73	+56.95	+67.84	-	-	-19.15	-22.16	-21.52	-17.29	+25.75
GBRT (CD)	-	-	0.497	0.592	1.039	1.633	0.428	0.500	0.370	0.606	0.919	1.387	-	-	0.539	0.475	5.128	4.845	12/14
GBRT (CI)	**0.165**	0.171	0.365	0.414	0.636	1.167	**0.341**	0.367	0.236	0.318	0.270	0.335	**0.532**	0.550	**0.146**	**0.185**	5.186	4.983	0/14
Improve (%)	-	-	+26.63	+29.99	+38.77	+28.57	+20.43	+26.63	+36.13	+47.50	+70.58	+75.87	-	-	+72.96	+61.01	-1.13	-2.85	+37.93
MLP (CD)	0.293	0.305	0.517	0.695	1.664	3.651	0.453	0.507	0.323	0.303	0.590	0.802	1257.104	1118.137	0.140	0.167	2.959	3.494	12/18
MLP (CI)	0.199	0.199	0.360	0.408	0.254	0.321	0.457	0.513	0.157	0.197	0.172	0.118	0.666	0.639	0.169	0.202	3.618	3.840	3/18
Improve (%)	+32.31	+34.57	+30.40	+41.36	+84.74	+91.22	-0.73	-1.14	+51.33	+34.85	+70.87	+85.28	+99.95	+99.94	-21.35	-21.37	-22.28	-9.88	+37.78
DeepAR (CD)	0.316	0.293	0.755	0.918	1.326	1.609	0.736	0.735	0.444	0.747	0.912	1.093	0.644	0.691	0.380	0.473	5.593	5.418	14/18
DeepAR (CI)	0.231	0.247	0.723	0.724	0.601	0.714	0.616	0.566	0.200	0.268	0.824	0.878	0.641	0.708	0.173	0.221	4.590	4.501	0/18
Improve (%)	+26.73	+15.65	+4.30	+21.08	+54.71	+55.63	+16.26	+22.92	+54.91	+64.16	+9.67	+19.65	+0.55	-2.48	+54.38	+53.22	+17.94	+16.92	+28.12
TCN (CD)	0.359	0.383	0.735	0.890	1.453	1.539	1.095	0.834	0.858	1.114	1.453	1.334	1.088	1.095	0.377	0.348	5.224	4.775	13/18
TCN (CI)	0.258	0.290	0.401	0.507	0.404	0.663	0.614	0.534	0.251	0.313	1.488	1.562	0.784	0.835	0.290	0.339	6.671	5.142	2/18
Improve (%)	+28.28	+24.47	+45.40	+42.99	+72.17	+56.94	+43.93	+35.90	+70.77	+71.87	-2.41	-17.11	+27.98	+23.73	+23.05	+2.56	-27.70	-7.68	+28.62
Informer (CD)	0.326	0.349	0.689	0.959	1.270	3.137	0.517	0.632	0.310	0.370	0.790	0.894	0.715	0.736	0.322	0.301	5.377	5.288	16/18
Informer (CI)	0.208	0.183	0.560	0.532	0.311	0.382	0.366	0.426	0.156	0.262	0.169	0.190	0.601	0.549	0.162	0.260	4.980	5.254	0/18
Improve (%)	+36.07	+47.47	+18.67	+44.58	+75.49	+87.83	+29.29	+32.61	+49.70	+29.10	+78.64	+78.69	+15.95	+25.43	+49.74	+13.41	+7.38	+0.65	+40.04
Transformer (CD)	0.250	0.257	0.861	0.966	1.031	1.868	0.458	0.554	0.281	0.520	0.511	0.659	0.645	0.650	0.251	0.423	5.309	5.406	17/18
Transformer (CI)	0.185	**0.163**	0.655	0.533	0.274	0.466	0.379	0.496	0.148	0.237	0.101	0.137	0.558	**0.526**	0.168	0.225	4.307	5.033	0/18
Improve (%)	+26.10	+36.59	+23.85	+44.84	+73.43	+75.07	+17.32	+10.43	+47.27	+54.40	+80.27	+79.30	+13.43	+19.13	+33.14	+46.87	+18.88	+6.89	+39.29

from identical and independent distributions (i.i.d.) [26]. This discrepancy undermines the accuracy of these models in predicting unseen data. For instance, we extend the autoregressive (AR) model to long-term forecasting tasks and demonstrate the adverse effects of distribution drift on the model's performance:

Proposition 4.2 (Yule-Walker equation [35, 37] extended)**.** *Assuming a long-term AR model on time series $\boldsymbol{x}$ with look-back window (order) L and horizon H is defined as:*

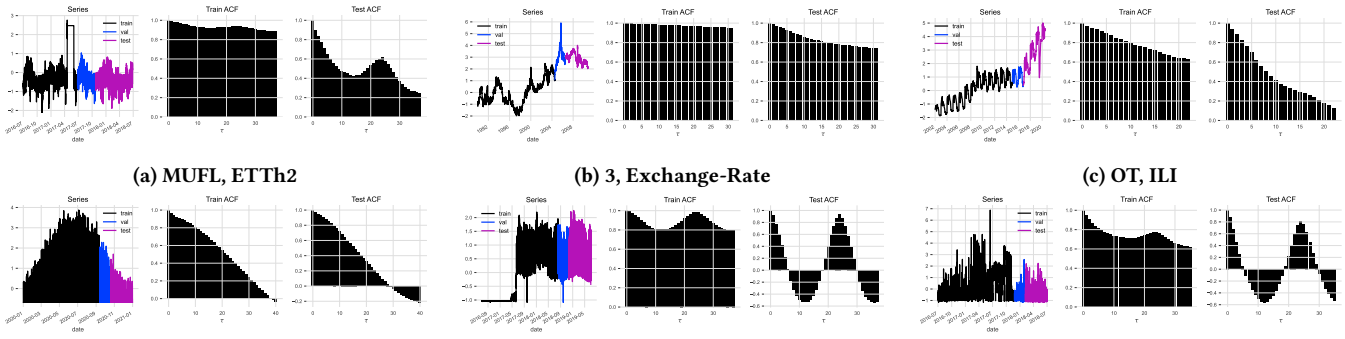
$$(\boldsymbol{x}_{t+H-1}, \ldots, \boldsymbol{x}_t)^T = \boldsymbol{W}(\boldsymbol{x}_{t-1}, \ldots, \boldsymbol{x}_{t-L})^\top \tag{6}$$

where $\boldsymbol{W} \in \mathbb{R}^{H\times L}$ is the coefficients of the model. Then the best estimation $\boldsymbol{W}^*$ can be computed by extended version of Yule-Walker equation [35, 37]:

$$\begin{bmatrix} \rho(1) & \rho(2) & \cdots & \rho(H) \\ \rho(2) & \rho(3) & \cdots & \rho(H+1) \\ \vdots & \vdots & \ddots & \vdots \\ \rho(L) & \rho(L+1) & \cdots & \rho(H+L-1) \end{bmatrix} = \begin{bmatrix} \rho(0) & \rho(-1) & \cdots & \rho(-L+1) \\ \rho(1) & \rho(0) & \cdots & \rho(-L+2) \\ \vdots & \vdots & \ddots & \vdots \\ \rho(L-1) & \rho(L-2) & \cdots & \rho(0) \end{bmatrix} \boldsymbol{W}^* \tag{7}$$

where $\rho(\tau) = \rho(-\tau)$ is the autocorrelation of time delay τ.

Proposition 4.2 establishes that the performance of an autoregressive (AR) model is closely linked to the autocorrelation function



(a) MUFL, ETTh2 (b) 3, Exchange-Rate (c) OT, ILI

(d) wv (m/s), Weather (e) 146, Electricity (f) 607, Traffic

Figure 3: The ACF of train series and test series. Captions of each subfigure represent the tuple (channel, dataset). For each subfigure, the leftmost plot displays the series split, with the training series in black, validation in blue, and test in purple. The middle and right display the ACF of train and test series respectively. The middle and right plots show the ACF of the training and test series, respectively. The results reveal a significant discrepancy in the statistics between the training and test series.

(ACF). Specifically, when there is a significant disparity between the ACF of the training and testing data, the resulting difference in the estimated values of $\boldsymbol{W}^*$ on the two datasets can be substantial. This can result in high error rates when applying the trained model to the test data.

Regrettably, real-world multivariate datasets often exhibit large disparities in ACF across channels. These distribution drifts in certain channels can significantly impact the performance of the trained model. However, we demonstrate in the next section that while the AR model using the CD strategy is sensitive to such drifts, the model using the CI strategy is more robust to them.

4.2 CI Alleviates Distribution Drift

In the previous section, we highlighted the presence of distribution drift (as measured by ACF) in real-world datasets. We also presented theoretical insights into how such drift can impact the performance of linear autoregressive (AR) models in univariate scenarios. n this section, we extend our analysis to multivariate tasks and demonstrate that the *CI strategy can alleviate distribution drift in each channel, while the CD strategy is vulnerable to such drift.*

Coefficients of CI and CD. To facilitate our analysis, we reshape the set of series data. Specifically, given a set of data $\{(\boldsymbol{X}^{(i)}, \boldsymbol{Y}^{(i)})\}_{i=1}^{N}$, we rearrange the series of each channel to its unique matrix $\boldsymbol{A}^{(c)} \in \mathbb{R}^{N\times L}$ and $\boldsymbol{B}^{(c)} \in \mathbb{R}^{N\times H}$, *i.e.*, $\boldsymbol{A}^{(c)}_{i,l} = \boldsymbol{X}^{(i)}_{l,c}$, $\boldsymbol{B}^{(c)}_{i,h} = \boldsymbol{Y}^{(i)}_{h,c}$. Basic on this representation, we can express the objectives of Linear (CD) and Linear (CI) as follows:

Definition 4.3 (Objective of Linear (CD) and Linear (CI))**.** Assuming the series of each channel is centered, the ordinary least square objective of **Linear (CD)** can be defined as:

$$\mathcal{L}_{cd} = \|\boldsymbol{A}_{cd}\boldsymbol{W}_{cd} - \boldsymbol{B}_{cd}\|_F^2 \tag{8}$$

where $\boldsymbol{A}_{cd} = \left[\boldsymbol{A}^{(1)}\boldsymbol{A}^{(2)}\dots\boldsymbol{A}^{(C)}\right] \in \mathbb{R}^{N\times LC}$ is the vertical concatenation of $\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}, \dots, \boldsymbol{A}^{(C)}$, $\boldsymbol{B}_{cd}$ the same. $\boldsymbol{W}_{cd} \in \mathbb{R}^{LC\times HC}$ is the coefficient.

Simiarly, the objective of **Linear (CI)** can be defined as:

$$\mathcal{L}_{ci} = \|\boldsymbol{A}_{ci}\boldsymbol{W}_{ci} - \boldsymbol{B}_{ci}\|_F^2 \tag{9}$$

where $\boldsymbol{A}_{ci} = \begin{bmatrix}\boldsymbol{A}^{(1)}\\ \boldsymbol{A}^{(2)}\\ \vdots\\ \boldsymbol{A}^{(C)}\end{bmatrix} \in \mathbb{R}^{NC\times L}$ is the horizontal concatenation of $\boldsymbol{A}^{(1)}, \boldsymbol{A}^{(2)}, \dots, \boldsymbol{A}^{(C)}$, $\boldsymbol{B}_{ci}$ the same. $\boldsymbol{W}_{cd} \in \mathbb{R}^{L\times H}$ is the coefficient.

From definition 4.3, we can see that the primary distinction between CD and CI strategy on Linear model is the way data are stacked. By solving the two objectives, the CD and CI coefficient of can be estimated according to the following proposition:

Proposition 4.4 (Yule-Walker equation of Linear (CD) and Linear (CI))**.** *Define the (auto-/cross-)correlation matrix:*

$$\boldsymbol{R}_{c_1,c_2} = \begin{bmatrix}\rho_{c_1,c_2}(0) & \rho_{c_1,c_2}(-1) & \cdots & \rho_{c_1,c_2}(-L+1)\\ \rho_{c_1,c_2}(1) & \rho_{c_1,c_2}(0) & \cdots & \rho_{c_1,c_2}(-L+2)\\ \vdots & \vdots & \ddots & \vdots\\ \rho_{c_1,c_2}(L-1) & \rho_{c_1,c_2}(L-2) & \cdots & \rho_{c_1,c_2}(0)\end{bmatrix} \in \mathbb{R}^{L\times L}.$$

$$\boldsymbol{R}'_{c_1,c_2} = \begin{bmatrix}\rho(1) & \rho(2) & \cdots & \rho(H)\\ \rho(2) & \rho(3) & \cdots & \rho(H+1)\\ \vdots & \vdots & \ddots & \vdots\\ \rho(L) & \rho(L+1) & \cdots & \rho(H+L-1)\end{bmatrix} \in \mathbb{R}^{L\times H}$$

where $\rho_{c_1,c_2}(\tau)$ is the auto-/cross-correlation at time delay τ when $c_1 = c_2$ / $c_1 \neq c_2$.

Assuming the series of each channel has the same variance, then the Yule-Walker equation of Linear (CD) is:

$$\begin{bmatrix}R'_{1,1} & R'_{1,2} & \dots & R'_{1,C}\\ R'_{2,1} & R'_{2,2} & \dots & R'_{2,C}\\ \vdots & \vdots & \ddots & \vdots\\ R'_{C,1} & R'_{C,2} & \dots & R'_{C,C}\end{bmatrix} = \begin{bmatrix}R_{1,1} & R_{1,2} & \dots & R_{1,C}\\ R_{2,1} & R_{2,2} & \dots & R_{2,C}\\ \vdots & \vdots & \ddots & \vdots\\ R_{C,1} & R_{C,2} & \dots & R_{C,C}\end{bmatrix} \boldsymbol{W}^*_{cd} \tag{10}$$

and the Yule-Walker equation of Linear (CI) is:

$$\sum_{c=1}^{C} R'_{c,c} = (\sum_{c=1}^{C} R_{c,c})\boldsymbol{W}^*_{ci} \tag{11}$$

Proof. Taking the derivative of eq. (8), we get the ordinary least square equation:

$$\boldsymbol{A}_{cd}^{\top}\boldsymbol{A}_{cd}\boldsymbol{W}_{cd} = \boldsymbol{A}_{cd}^{\top}\boldsymbol{B}_{cd} \tag{12}$$

$$\begin{aligned}\boldsymbol{A}_{cd}^{\top}\boldsymbol{A}_{cd} &= \left[\boldsymbol{A}^{(1)}\boldsymbol{A}^{(2)}\dots\boldsymbol{A}^{(C)}\right]^{\top}\left[\boldsymbol{A}^{(1)}\boldsymbol{A}^{(2)}\dots\boldsymbol{A}^{(C)}\right] \\ &= \begin{bmatrix} (\boldsymbol{A}^{(1)})^{\top}(\boldsymbol{A}^{(1)}) & (\boldsymbol{A}^{(1)})^{\top}(\boldsymbol{A}^{(2)}) & \dots & (\boldsymbol{A}^{(1)})^{\top}(\boldsymbol{A}^{(C)}) \\ (\boldsymbol{A}^{(2)})^{\top}(\boldsymbol{A}^{(2)}) & (\boldsymbol{A}^{(2)})^{\top}(\boldsymbol{A}^{(2)}) & \dots & (\boldsymbol{A}^{(2)})^{\top}(\boldsymbol{A}^{(C)}) \\ \vdots & \vdots & \ddots & \vdots \\ (\boldsymbol{A}^{(C)})^{\top}(\boldsymbol{A}^{(C)}) & (\boldsymbol{A}^{(C)})^{\top}(\boldsymbol{A}^{(C)}) & \dots & (\boldsymbol{A}^{(1)})^{\top}(\boldsymbol{A}^{(C)}) \end{bmatrix}\end{aligned} \tag{13}$$

is in a form of outer product. Each $(\boldsymbol{A}^{(c_1)})^{\top}(\boldsymbol{A}^{(c_2)})$ is a estimation of co-variance matrix [25]. Since the variances of each channel series are assumed to be the same. So by dividing on both side of eq. (12) by the variance, we can get eq. (10).

Simiarly, $\boldsymbol{A}_{cd}^{\top}\boldsymbol{A}_{cd} = \left[\boldsymbol{A}^{(1)}\boldsymbol{A}^{(2)}\dots\boldsymbol{A}^{(C)}\right]\left[\boldsymbol{A}^{(1)}\boldsymbol{A}^{(2)}\dots\boldsymbol{A}^{(C)}\right]^{\top}$ is in the form of inner product. By the same process, we can get eq. (11). □

By analyzing the difference between eq. (10) and eq. (11), we can draw an important conclusion– **coefficients of CD is determined by the (auto-/cross-)correlation function of each channel, while the coefficients of the CI strategy are determined solely by the summation (or mean) of the ACF of all channels.**

> Takeaways
>
> The optimal coefficients of the Linear model using the CD strategy are determined by the ACF of all channels, while the optimal coefficients of the model using the CI strategy are only determined by the **sum** of the ACF across all channels.

CI strategy leads to less distribution drift. It is noteworthy that the summation operation used in the CI strategy mitigates the distribution gap between the training and test series. To demonstrate this, we examine the differences in the values of the ACF between the training and test portions. Specifically, we denote the ACF of the training portion in channel c as $\rho_c^{(tr)}$ and the corresponding test ACF as $\rho_c^{(te)}$. Then, we calculate the ACF difference in channel c as follows:

$$\text{Diff}_c = \sum_{t=0}^{T}(\rho_c^{(tr)}(t) - \rho_c^{(te)}(t))^2.$$

When employing the CD strategy, the linear model is susceptible to the distribution drift of each channel. However, the CI strategy ensures that the linear model is solely determined by the sum of ACF over all channels. Therefore, we only need to evaluate the changes in the sum of ACF when using the CI strategy. Hence, we calculate the difference in the ACF summation between the training and test sets using the following equation, referred to as the **sum diff**:

$$\text{Diff}_{\text{sum}} = \sum_{t=0}^{T}(\frac{1}{C}\sum_{c=1}^{C}\rho_c^{(tr)}(t) - \frac{1}{C}\sum_{c=1}^{C}\rho_c^{(te)}(t))^2.$$

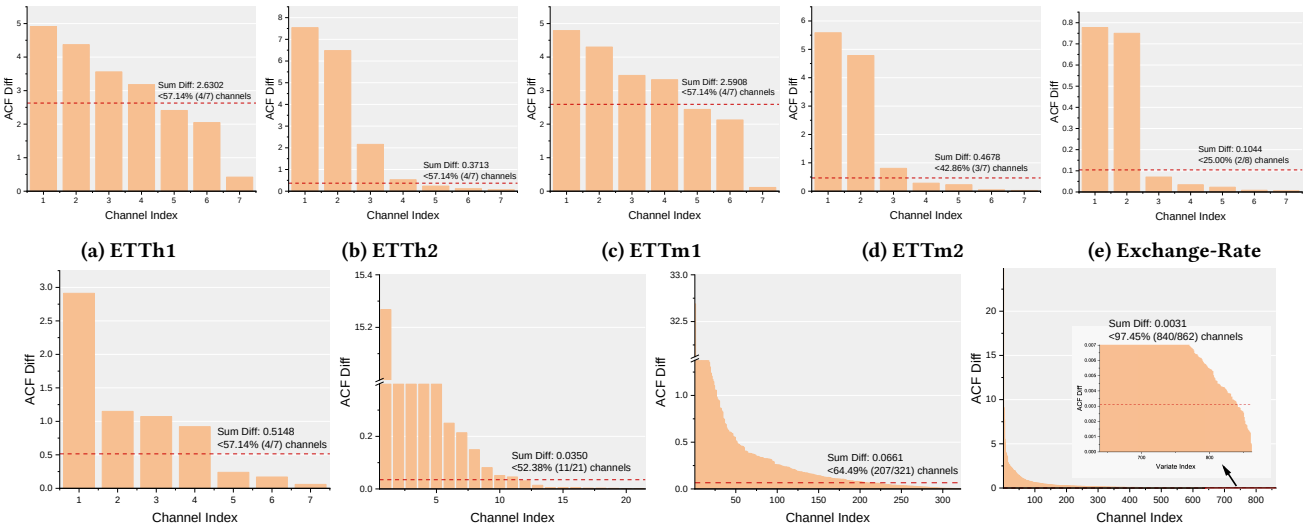
Considering the scale, we compute the mean instead of the sum. But we still name it sum diff. Sum diff can be regarded as the ACF difference when using CI strategy.

We present the ACF difference (Diff c) using bar plots, sorted in descending order, and indicate the **sum diff** (Diff sum) with a horizontal line. Our results are summarized in fig. 4.Several observations can be made from the figure: (1) **Most real-world datasets exhibit channels with significant ACF differences between the training and test data, indicating severe distribution drift in the time series of these channels.** In ETT datasets (a)-(d), for instance, the largest ACF differences range between 5 and 8, which is substantial given that ACF values typically fall within $[0, 1]$. ETTh1 and ETTm1 show relatively uniform ACF differences, while ETTh2 and ETTm2 feature two channels with particularly large ACF differences compared to the rest. Other datasets exhibit similar patterns, with Exchange (e) displaying two channels with ACF differences that greatly exceed those of other channels, and ILI featuring a largest difference that is more than twice that of the rest. In datasets with many channels, such as weather (g), electricity (h), and traffic (i), the largest difference can be up to 15, 32, and 24, respectively, which is much greater than that of most other channels. Furthermore, we observe a rapid decay of the ACF difference as the channel index increases. (2) **The sum diff is typically smaller than the ACF difference of most channels, suggesting that the distribution drift with CI strategy is less severe than with CD strategy.** Across the 9 benchmarks, 7 datasets have a sum diff that is smaller than that of more than 50% of the channels, indicating that the distribution drift, as measured by the ACF difference, is smaller than that of most channels when using CI. Even in the two exceptions, ETTm2 (d) and Exchange-Rate (e), the sum diff is still much smaller than the head channels. On ETTm2 (d), for example, the sum diff is 0.4678, significantly smaller than the head values, which are nearly 5. The sum diff is also not much larger than channels 4-7. Similar observations hold for Exchange-Rate (e), where the sum diff is only lower than two channels, but its value of 0.1044 is much smaller than 0.75. On the remaining 7 datasets, not only is the sum diff smaller than that of most channels, but it is also much smaller in value. For instance, on ETTh2 (b), the sum diff of 0.3713 is 20 times smaller than that of channel 1, while on Electricity (h) and Traffic (i), it is 500 and 8000 times smaller, respectively.

> Takeaways
>
> The sum of ACF differences between training and test data exhibits less variation than the ACF differences of most individual channels. This means employing the CI strategy results in reduced distribution drift.

4.3 Capacity and Robustness

Although the CI strategy can reduce the distribution gap between training and test data, we cannot conclude that it leads to better generalization performance. This is primarily due to the fact that the hypothesis spaces under CI and CD strategies are not the same, where $\boldsymbol{W}_{ci} \in \mathbb{R}^{L\times H}$ and $\boldsymbol{W}_{cd} \in \mathbb{R}^{LC\times HC}$. To analyze the risks associated with these strategies, we follow the risk analysis framework



(a) ETTh1 (b) ETTh2 (c) ETTm1 (d) ETTm2 (e) Exchange-Rate

(f) ILI (g) Weather (h) Electricity (i) Traffic

Figure 4: The difference of ACF between training data and test data. The ACF difference for each channel is depicted in bar charts, arranged in descending order. The sum diff, which represents the overall ACF difference under the CI strategy, is shown as a horizontal line. the sum diff is smaller than the ACF difference of each channel, indicating that the CI strategy can effectively mitigate distribution drift.

in machine learning proposed by Mohri et al. [26] and decompose the risk according to the following equation:

$$\mathcal{R}(\hat{\boldsymbol{W}}) = \underbrace{\left(\mathcal{R}(\hat{\boldsymbol{W}}) - \inf_{\boldsymbol{W}\in\mathcal{W}} \mathcal{R}(\boldsymbol{W})\right)}_{\text{non-robustness}} + \underbrace{\inf_{\boldsymbol{W}\in\mathcal{W}} \mathcal{R}(\boldsymbol{W})}_{\text{incapacity}}. \tag{14}$$

where $\mathcal{R}(\cdot)$ is the risk defined by eq. (1) and $\hat{\boldsymbol{W}}$ denotes the model obtained by minimizing the empirical CD loss as in eq. (2), or the empirical CI loss as in eq. (3). We use the notation $\mathcal{W}$ to refer to the hypothesis space. In this paper, we consider two types of hypothesis space: the hypothesis space of CI, denoted by $\mathcal{W}ci = \mathbb{R}^{L\times H}$, and the hypothesis space of CD, denoted by $\mathcal{W}cd = \mathbb{R}^{LC\times HC}$.

Unlike the traditional terminology [26], we interpret the first term as the **(non-)robustness** of a model, which is the risk gap between the model trained on the training set and the optimum model on the test data distribution. It measures the ability of the model to handle unseen data and achieve nearly optimal performance. A lower value of this term indicates a more robust model. The **(in)capacity** measures how well the optimum model fits the data, with a lower value indicating a better ability to fit the data. Simple algorithms like linear regression usually have low capacity (high incapacity), while complex algorithms like neural networks have high capacity. In addition to the choice of algorithm, different training strategies such as CI and CD also affect the robustness and capacity of the obtained model. In the following sections, we provide empirical results to further illustrate this concept.

It is not possible to calculate the risk $\mathcal{R}$ directly as access to the underlying data distribution is unavailable. In this paper, we opt to approximate it using the empirical risk on the test data. For ease of reference, we represent the training and test set utilizing the CI strategy as $(A^{(tr)}ci, B^{(tr)}ci)$ and $(A^{(te)}ci, B^{(te)}ci)$, while the training and test set using the CD strategy is denoted as $(A^{(tr)}cd, B^{(tr)}cd)$ and $(A^{(te)}cd, B^{(te)}cd)$. We compute the subsequent statistics to demonstrate the performance of CI and CD on the benchmarks:

(1) **Train Error (Incapacity).** The training error $\mathcal{L}_i^{(tr)}$ is computed as the following:

$$\mathcal{L}_i^{(tr)} = \|\boldsymbol{A}_i^{(tr)}\boldsymbol{W}_i^{(tr)} - \boldsymbol{B}_i^{(tr)}\|_F^2, \quad i \in \{ci, cd\} \tag{15}$$

where

$$\boldsymbol{W}_i^{(tr)} = \arg\min_{\boldsymbol{W}} \|\boldsymbol{A}_i^{(tr)}\boldsymbol{W} - \boldsymbol{B}_i^{(tr)}\|_F^2$$

is the optimum parameter for the training data. **Train Error** is also a measure of capacity but empirically computed on the training set.

(2) **Test Error (Incapacity).** The test error $\mathcal{L}_i^{(te)}$ is computed as the following:

$$\mathcal{L}_i^{(te)} = \|\boldsymbol{A}_i^{(te)}\boldsymbol{W}_i^{(te)} - \boldsymbol{B}_i^{(te)}\|_F^2, \quad i \in \{ci, cd\} \tag{16}$$

where:

$$\boldsymbol{W}_i^{(te)} = \arg\min_{\boldsymbol{W}} \|\boldsymbol{A}_i^{(te)}\boldsymbol{W} - \boldsymbol{B}_i^{(te)}\|_F^2$$

is the optimum parameter for the test data. Test loss describes the best error a linear model can achieve on the test data. It is an approximation of $\inf_{\boldsymbol{W}\in\mathcal{W}} \mathcal{R}(\boldsymbol{W})$ in eq. (14).

(3) **Gen Error ($\mathcal{R}(\hat{\boldsymbol{W}})$).** The generalization error $\mathcal{L}_i^{(gen)}$ is computed as:

$$\mathcal{L}_i^{(gen)} = \|\boldsymbol{A}_i^{(te)}\boldsymbol{W}_i^{(tr)} - \boldsymbol{B}_i^{(te)}\|_F^2. \quad i \in \{ci, cd\}$$

It is the performance measure on the benchmarks.

(4) **W Diff (Non-Robustness).** It is an approximation of non-robustness in eq. (14). Its value is computed as:

$$\text{Diff}_{W_i} = \|\boldsymbol{A}_i^{(te)}(\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)})\|_F^2. \tag{17}$$

eq. (17) is inspired by ordinal least square in fixed design settings [26], where the estimation error is computed as the Mahalanobis distance between $\boldsymbol{W}_i^{(tr)}$ and $\boldsymbol{W}_i^{(te)}$. eq. (17) is an extension of Mahalanobis distance, since:

$$\begin{aligned}\text{Diff}_{W_i} &= \|\boldsymbol{A}_i^{(te)}(\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)})\|_F^2 \\ &= \text{tr}((\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)})^\top (\boldsymbol{A}_i^{(te)})^\top \boldsymbol{A}_i^{(te)}(\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)})). \\ &= \text{tr}((\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)})^\top \hat{\Sigma}_i^{(te)}(\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)}))\end{aligned} \tag{18}$$

tr is the trace operation for a matrix and $\hat{\Sigma}_i^{(te)} = (\boldsymbol{A}_i^{(te)})^\top \boldsymbol{A}_i^{(te)}$ is the unnormalized sample covariance matrix. When $\boldsymbol{W}_i^{(tr)}$ and $\boldsymbol{W}_i^{(te)}$ become vectors, eq. (18) falls back to the Mahalanobis distance parameterized by Mahalanobis matrix $\hat{\Sigma}_i^{(te)}$. In this sense, the W diff can also be considered as a measure of distribution drift, since it is a distance measure between $\boldsymbol{W}_i^{(tr)}$ and $\boldsymbol{W}_i^{(te)}$, and $\boldsymbol{W}_i^{(tr)}$ and $\boldsymbol{W}_i^{(te)}$ are derived by the ACF of train and test data.

Another interpretation of Diff_{W_i} takes it as a lower bound for the estimation error:

$$\begin{aligned}\text{Diff}_{W_i} &= \|\boldsymbol{A}_i^{(te)}(\boldsymbol{W}_i^{(tr)} - \boldsymbol{W}_i^{(te)})\|_F^2 \\ &\leq \|\boldsymbol{A}_i^{(te)}\boldsymbol{W}_i^{(te)} - \boldsymbol{B}_i^{(te)}\|_F^2 - \|\boldsymbol{A}_i^{(te)}\boldsymbol{W}_i^{(tr)} - \boldsymbol{B}_i^{(te)}\|_F^2 \\ &= \mathcal{L}_i^{(gen)} - \mathcal{L}_i^{(te)}, \quad i \in \{ci, cd\}\end{aligned}$$

Similar to the risk decomposition (eq. (14)), we can also decompose the gen loss by the following equation:

$$\mathcal{L}_i^{(gen)} = \underbrace{(\mathcal{L}_i^{(gen)} - \mathcal{L}_i^{(te)})}_{\approx \text{Diff}_{W_i}} + \mathcal{L}_i^{(te)}. \quad i \in \{ci, cd\} \tag{19}$$

We illustrate the above statistics on the 9 datasets in fig. 5 and draw the following conclusions. (1) **CD models exhibit lower train/test loss as compared to CI models, indicating that CD strategy trains a model with higher capacity.** On all of the 9 models, the train/test error is always lower than CI. When the number of channels is large, CD model may have 0 errors. This trend is consistent across all 9 models, with the CD model registering zero errors for larger channel numbers. This outcome is anticipated due to the fact that the hypothesis space of CI is a subset of CD, thereby enabling the construction of a CD linear model by replicating elements of a CI linear model. As per the definition, the best CD model will inevitably have a lower error rate than CI. (2) **CD models have a significantly larger W diff than CI models, indicating that CI is much more robust than CD.** This trend is apparent across all 9 datasets, with CD models having W diff values usually over 10 times the value of CI. The phenomenon is especially conspicuous in datasets with multiple channels such as Electricity (h) and Traffic (i). The distribution drift between train and test data is responsible for this trend. In the previous section, we have shown that the ACF coefficients of CD and CI models are determined by the ACF of all channels and the sum of ACF across all channels, with the latter exhibiting a lower difference than the former. This difference contributes to the gap between optimal models on train and test data being different for CD and CI strategies. CI strategy leads to a lighter distribution gap, resulting in a smaller W Diff value. (3) W Diff values are more significant than Test Error in most cases, leading to CI models having lower Gen Error than CD models. *i.e.*, **Robustness is more crucial than capacity.** For the 4 ETT benchmarks, there is not much difference in test error between CI and CD models, but W diff values are significantly distinct. Hence, CI models perform better than CD models in terms of gen loss. On datasets (e), (f), (h), (i), the test loss varies considerably, but the W diff values differ significantly more than the test loss, leading to CI models performing better than CD models. The only exception to this trend is the weather benchmark (g), where test error holds greater significance than W diff values. Consequently, CD strategy performs better in this case. Figure 6 summarizes these findings.

From analyses in this section, we draw an overall conclusion:

Section Conclusion

The Channel Dependent (CD) strategy has high capacity but low robustness. The Channel Dependent (CI) strategy has low capacity but high robustness. In numerous real-world non-stationary time series with distribution drifts, robustness is a more crucial factor than capacity in forecasting tasks. Consequently, CI strategy often delivers better performance.

To provide readers with a clear understanding, we demonstrate the differences in prediction outcomes between the CD and CI strategies using a visual representation in fig. 7. These examples were selected as they are indicative of the findings obtained across the experiments. From these four figures, We observe that the CD approach produces sharp predictions, while the CI approach generates smoother predictions. This discrepancy can be attributed to the sum over effect, which we have analysed in detail in the previous subsection. Unfortunately, the non-robust and sharp nature of CD predictions make them unsuitable for accurately predicting real-world non-stationary time-series. Figure 7.(a) displays a scenario where both strategies capture the correct trend and seasonal component, but the CI approach more closely aligns with the ground-truth compared to the CD approach. (b) shows that the CD approach may predict incorrect trends, while the CI approach is less prone to making such errors. When faced with anomalous time-series like (c), the CI approach is more robust and produces less oscillation. Nonetheless, there are instances where the CD approach outperforms the CI approach, as shown in fig. 7.(d), where the CD model's high capacity can be advantageous for capturing complex but predictable patterns. Intuitively, real-world time series that exhibit regular patterns and smooth changes are predictable. Conversely, drastically oscillating time series, such as the one depicted in fig. 7.(c), are anomalies and therefore unpredictable. In such cases, conservative and robust predictions are preferable. As a result, the CI strategy yields superior results on average compared to the CD strategy.

5 PRACTICAL GUIDES

The previous section's analyses have revealed that the CD strategy exhibits high capacity but low robustness, making it unsuitable for handling real-world non-stationary time series where distribution drifts are significant. Conversely, the CI approach displays higher robustness, resulting in superior performance compared to the CD

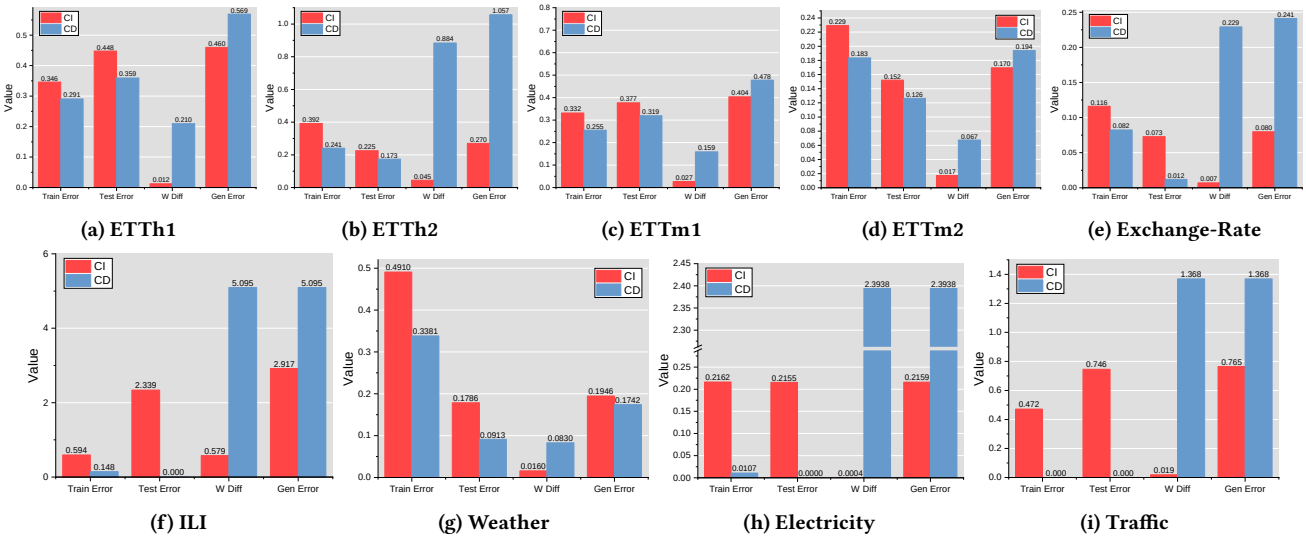


Figure 5: The train error, test error, W diff and gen error when using CI and CD strategy on the 9 datasets. Train/test error measures model capacity on train/test data. W diff measures the difference between the optimal model on train and test data. It reveals the robustness of a model. Gen error measures the risk of an algorithm. Although CD can achieve lower optimal error, it is much less robust to the distribution drift than CI. Consequently, in most cases, CI outperforms CD.

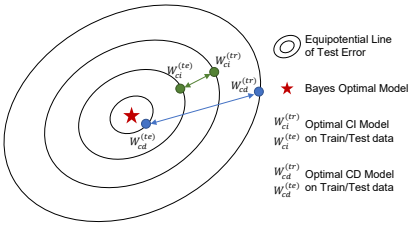


Figure 6: Illustration of the conclusions drawn from fig. 5. CD have better optimal model on test data but larger distance between the optimal train and test model. CI is the contrary. In most cases, model difference matters more than the optimal model. Thus CI often achieves better performance.

strategy. These findings provide valuable guidance for designing or improving existing multivariate forecasting models. Specifically, we recommend increasing the robustness of CD models and increasing the capacity of CI models.

In this section, we propose a simple modified CD objective that can help models surpass the CI strategy. Additionally, we discuss several factors that may influence the performance of CD or CI models. By considering these factors, we can further optimize and tailor the models for specific use cases.

5.1 Predict Residuals with Regularization

Our analysis of fig. 7 in the previous section has led us to conclude that the primary disadvantage of CD models is their tendency to generate "sharp" and non-robust predictions that often diverge from the actual trend. To address this issue, we propose a simple method to improve the performance of CD models called Predict Residuals with Regularization (PRReg), inspired by measures taken in N-BEATS [29] and NLinear [42]. The core idea of PRReg is to ensure that the prediction remains close to the nearest known history and that the forecasted series remains smooth. To achieve this objective, we reformulate the CD objective into the following form:

$$\min_f \frac{1}{N}\sum_{i=1}^{N} \ell(f(\boldsymbol{X}^{(i)} - N^{(i)}) + N^{(i)}, \boldsymbol{Y}^{(i)}) + \lambda\Omega(f). \tag{20}$$

where $N^{(i)} = \boldsymbol{X}^{(i)}_{:,L}$ is the last values of each channel in $\boldsymbol{X}^{(i)}$. With this objective, the goal of f is changed from accurately predicting future values to the variety from the nearest history. The regularization term Ω serves a dual purpose: to restrict the predictions within a reasonable distance from $N^{(i)}$ and to encourage smoothness in the predicted values. In our study, we adopted L_2 regularization, which was implemented as weight decay in PyTorch [30]. Our proposed objective is applicable to various forecasting models, and its effectiveness is illustrated in fig. 8 for Linear [42] and Transformer [36] models. Table 4 presents the results, where we compare the PRReg strategy with CD and CI. We observe that PRReg outperforms both CD and CI in most cases when the regularization strength λ is chosen appropriately. A too-small λ fails to provide the required robustness since PRReg is fundamentally a CD strategy, while an excessively large regularization strength causes underfitting with sufficient capacity. Thus, choosing a suitable value of λ results in an optimal trade-off between capacity and robustness and leads to the best possible results.

The PRReg objective offers several benefits. Firstly, it is model agnostic, which implies that it can be used with various multivariate forecasting models. Secondly, it is a modified version of the CD strategy that incorporates the correlations between different channels. Lastly, it outperforms the CI strategy. It is intuitive that

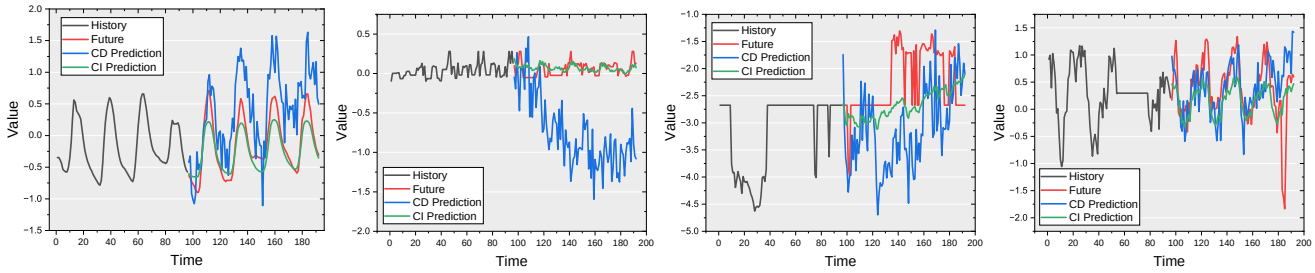


Figure 7: We select four representative examples to demonstrate the differences between CD and CI strategies. The experiment is conducted on ETTh2 with Linear model. (a) When both of them capture the correct trend and seasonal component. CD tends to generate "sharp" predictions, while the CI produces smoother ones. (b) The CD could predict wrong trends, while CI is less likely to do so. (c) When faced with abnormal series, CI are more robust with less oscillation. (d) The high capacity of CD models may be beneficial for capturing complex but predictable patterns. CI is unable to capture them.

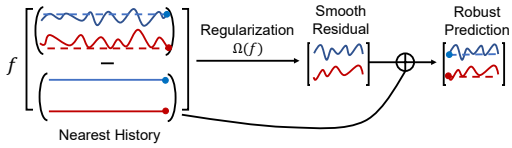


Figure 8: The idea of Predict Residuals with Regularization (PRReg). The input series is subtracted by the last value. Then the predictor is regularized to predict smoothed residual. Robust prediction is made by adding back the nearest history.

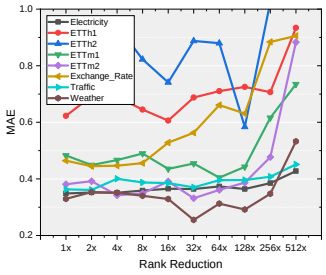


Figure 9: The MAE error of Transformer (CD) model with low rank linear layer on different datasets. We vary the rank reduction rate from 1x to 512x, gradually reducing the rank. The errors drop and rise with increasing reduction rate. Thus, suitable rank regularization helps.

we should not treat each channel independently since they represent the features of the same object. However, the CD strategy often exhibits inferior performance due to its lack of robustness. The PRReg objective successfully resolves this issue, striking a balance between capacity and robustness, thereby achieving superior results.

5.2 Some Other Factors

We list some of the factors that may influence the performance of CD or CI models by altering their capacity and robustness. It is important to note that capacity and robustness are intertwined in the model selection process, and increasing one often requires decreasing the other. Thus, these factors can impact CD and CI strategies in various ways. We list some factors only for inspiration.

Low rank layer. The low rank assumption is widely used in robust learning [23, 41]. Following the approach proposed in [17], we replace the linear output projection of each attention layer with a low rank linear layer. Specifically, if the weight of the original linear layer is $W \in \mathbb{R}^{m\times n}$, we replace it with M_1M_2, where $M_1 \in \mathbb{R}^{m\times r}$ and $M_2 \in \mathbb{R}^{r\times n}$. We varied the rank reduction rate in 2, 4, 8, 16, 32, 64, 128, 256, 512, which means that if the rate is 2, $r = \lfloor \frac{\min m,n}{2} \rfloor$. Figure 9 presents the results of our experiments. As the rank reduction rate increases, the error initially drops and then rises. Low rank regularization reduces the capacity and increases the robustness of a model. Thus, an appropriate choice of the rank can help a CD model perform better.

Robust loss. The Mean Absolute Error (MAE), also known as the L-1 loss, has been demonstrated to be resilient to noisy labels [14, 15]. Hence, by applying the L-2 loss to a model trained using the CD strategy, its robustness can be improved. To avoid ambiguity, we will refer to the loss used during training as L-2/L-1, and the evaluation metric as Mean Squared Error (MSE)/MAE. The outcomes of applying L-1 and L-2 losses to the Transformer (CD) are presented in Table 5. We observe that L-1 loss enhances the robustness of the model, resulting in more accurate predictions.

Length of look-back windows. The length of the look-back window determines the amount of memory that a forecasting model can utilize. In the context of multiple series forecasting, increasing the memory capacity can improve the performance of the global model [27]. Our study demonstrates that the window length also affects the performance of CD and CI strategies in different ways. We varied the sequence length of the input look-back window from 48 to 432, while keeping the horizon fixed at 48. We selected Linear and Transformer models as representative methods to illustrate this phenomenon. The results of comparing the performance of these two models with CD and CI strategies on some datasets are presented in fig. 10. A longer length of the look-back window provides more information about the historical data but also increases the capacity of the model. For CD models, when the history is not too short, increasing the length often leads to worse performance. On the other hand, CI always benefits from longer window lengths.

Table 4: Comparison among forecasters trained with PRReg (varying λ), CD and CI strategies. The base forecasters are Linear and Transformer. Performance is measured by MSE. The best results of each setting (row) are marked bold. PRReg is able to surpass CD and CI if the λ is selected properly. Meaning that it produces a suitable balance between capacity and robustness.

	Model	CD	CI	PRReg (ours)						
				$\lambda = 10^{-6}$	$\lambda = 10^{-5}$	$\lambda = 10^{-4}$	$\lambda = 10^{-3}$	$\lambda = 10^{-2}$	$\lambda = 10^{-1}$	$\lambda = 1$
Electricity	Linear	-	-	-	-	-	-	-	-	-
	Transformer	0.250	**0.185**	0.218	0.219	0.227	0.269	0.378	0.742	1.527
ETTh1	Linear	0.402	0.345	0.346	0.345	0.344	**0.342**	0.355	0.426	0.737
	Transformer	0.861	0.655	0.624	0.624	0.623	0.625	**0.539**	0.744	1.164
ETTh2	Linear	0.711	**0.226**	0.335	0.329	0.296	0.248	0.239	0.259	0.298
	Transformer	1.031	0.274	0.319	0.319	0.323	0.373	0.424	**0.273**	0.332
ETTm1	Linear	0.404	0.354	0.315	0.315	0.314	**0.311**	0.318	0.378	0.668
	Transformer	0.458	0.379	0.374	0.374	0.375	**0.349**	0.370	0.536	1.148
ETTm2	Linear	0.161	0.147	0.142	0.142	0.140	**0.136**	0.141	0.163	0.195
	Transformer	0.281	0.148	0.171	0.164	0.160	**0.144**	0.149	0.182	0.211
Exchange_Rate	Linear	0.119	0.051	0.050	0.050	0.049	0.046	0.043	**0.042**	**0.042**
	Transformer	0.511	0.101	0.098	0.098	0.092	0.074	0.056	**0.044**	**0.044**
Weather	Linear	0.142	0.169	0.133	0.133	0.133	0.132	**0.131**	0.141	0.169
	Transformer	0.251	**0.168**	0.189	0.191	0.195	0.180	0.189	0.297	0.193
ILI	Linear	2.343	2.847	2.599	2.599	2.597	2.581	2.467	**2.299**	2.693
	Transformer	5.309	4.307	3.258	3.258	3.257	**3.254**	3.310	3.848	4.793

Table 5: Performance of Transformer (CD) with L-1 and L-2 loss. When using the L-1 loss which is more robust, Transformer (CD) forecast more accurately.

Metric	MAE		MSE	
Loss	L-2	L-1	L-2	L-1
Electricity	0.352	**0.347**	**0.250**	0.253
ETTh1	0.734	**0.604**	0.861	**0.669**
ETTh2	0.829	**0.536**	1.031	**0.463**
ETTm1	0.458	**0.421**	0.458	**0.403**
ETTm2	0.404	**0.328**	0.281	**0.210**
Exchange_Rate	0.571	**0.451**	0.511	**0.316**
Traffic	0.364	**0.347**	**0.645**	0.668
Weather	0.343	**0.274**	0.251	**0.229**
ILI	1.508	**1.373**	5.309	**4.457**

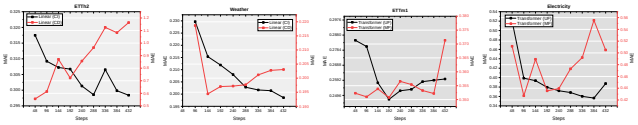


Figure 10: MAE performance of Linear and Transformer with CI and CD strategy on certain datasets. The X-axis represents the length of look-back window. The capacity of both CI and CD model are increased when we input a longer window. We can see that a longer window may do harm to the performance of CD models, while CI can benefit from it.

6 DISCUSSION ABOUT LIMITATIONS

It is important to stress that the conclusions drawn in this paper are closely tied to the characteristics of the datasets employed. While the Channel Independent (CI) training approach generally outperforms the Channel Dependent (CD) strategy, there are exceptions. For instance, as indicated in Table 2 for the ILI dataset, CD performs better on average. Nonetheless, the analysis of CI and CD provides valuable insights into the peculiarities of real-world time series and how different strategies can leverage them.

Analyses of this paper may also be limited to numerical channels. Nevertheless, it is possible to handle numerical and non-numerical features separately. We defer an analysis of strategies on more general types of time-series data to future research.

7 CONCLUSION

Recent years have seen the emergence of several methods for long-term Multivariate Time Series Forecasting (MTSF), with some adopting the channel independent (CI) strategy to achieve good performance. By the analyses of this paper, we show the performance boost generated by these methods is often not due to their design, but rather to the training strategy. Despite lower model capacity, the CI strategy exhibits higher robustness, making it better suited for non-stationary time series in practice. We hope that this article will alert the researchers the characteristics of MTSF benchmarks and inspire researchers to better deal with multivariate time series forecasting problems.

ACKNOWLEDGMENTS

This research was supported by NSFC (61773198, 62006112,61921006), Collaborative Innovation Center of Novel Software Technology and Industrialization, NSF of Jiangsu Province (BK20200313).

REFERENCES

[1] OD Anderson. 1976. Time-Series. 2nd edn.

[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. 2018. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *CoRR* abs/1803.01271 (2018).

[3] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

[4] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *SIGKDD*. 785–794.

[5] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. In *SSST@EMNLP*, Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi (Eds.). Association for Computational Linguistics, 103–111.

[6] Razvan-Gabriel Cirstea, Darius-Valer Micu, Gabriel-Marcel Muresan, Chenjuan Guo, and Bin Yang. 2018. Correlated time series forecasting using multi-task deep neural networks. In *ICKM*. 1527–1530.

[7] Yue Cui, Kai Zheng, Dingshan Cui, Jiandong Xie, Liwei Deng, Feiteng Huang, and Xiaofang Zhou. 2021. METRO: A Generic Graph Neural Network Framework for Multivariate Time Series Forecasting. *Proc. VLDB Endow.* 15, 2 (2021), 224–236.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Minneapolis, MN, USA, June 2-7, 2019*. 4171–4186.

[9] Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition. In *ICASSP*. IEEE, 5884–5888.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*.

[11] Shereen Elsayed, Daniela Thyssens, Ahmed Rashed, Lars Schmidt-Thieme, and Hadi Samer Jomaa. 2021. Do We Really Need Deep Learning Models for Time Series Forecasting? *CoRR* abs/2101.02118 (2021).

[12] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. 2019. Unsupervised Scalable Representation Learning for Multivariate Time Series. In *NeurIPS*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 4652–4663.

[13] Jan Gasthaus, Konstantinos Benidis, Yuyang Wang, Syama Sundar Rangapuram, David Salinas, Valentin Flunkert, and Tim Januschowski. 2019. Probabilistic forecasting with spline quantile function RNNs. In *AISTATS*. PMLR, 1901–1910.

[14] Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. 2017. Robust Loss Functions under Label Noise for Deep Neural Networks. In *AAAI*, Satinder Singh and Shaul Markovitch (Eds.). 1919–1925.

[15] Aritra Ghosh, Naresh Manwani, and P. S. Sastry. 2015. Making risk minimization tolerant to label noise. *Neurocomputing* 160 (2015), 93–107.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*.

[18] Rob J Hyndman and George Athanasopoulos. 2018. *Forecasting: principles and practice*. OTexts.

[19] Gwilym M Jenkins and MB Priestley. 1957. The spectral analysis of time-series. *Journal of the Royal Statistical Society: Series B (Methodological)* 19, 1 (1957), 1–12.

[20] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*. 95–104.

[21] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. 2017. Time-series extreme event forecasting with neural networks at uber. In *ICML*, Vol. 34. 1–5.

[22] Bryan Lim and Stefan Zohren. 2020. Time Series Forecasting With Deep Learning: A Survey. *CoRR* abs/2004.13408 (2020).

[23] Guangcan Liu, Zhouchen Lin, and Yong Yu. 2010. Robust Subspace Segmentation by Low-Rank Representation. In *ICML*, Johannes Fürnkranz and Thorsten Joachims (Eds.). Omnipress, 663–670.

[24] Shizhan Liu, Hang Yu, Cong Liao, Jianguo Li, Weiyao Lin, Alex X Liu, and Schahram Dustdar. 2022. Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting. In *ICLR*.

[25] Henrik Madsen. 2007. *Time series analysis*. Chapman and Hall/CRC.

[26] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of machine learning*. MIT press.

[27] Pablo Montero-Manso and Rob J Hyndman. 2021. Principles and algorithms for forecasting groups of time series: Locality and globality. *International Journal of Forecasting* 37, 4 (2021), 1632–1653.

[28] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2022. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. *CoRR* abs/2211.14730 (2022).

[29] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *ICLR*.

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *NeurIPS*. 8024–8035.

[31] Stephan Rabanser, Tim Januschowski, Valentin Flunkert, David Salinas, and Jan Gasthaus. 2020. The Effectiveness of Discretization in Forecasting: An Empirical Study on Neural Time Series Models. *CoRR* abs/2005.10111 (2020).

[32] Syama Sundar Rangapuram, Matthias W. Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. 2018. Deep State Space Models for Time Series Forecasting. In *NeurIPS*, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 7796–7805.

[33] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[34] Slawek Smyl. 2020. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting* 36, 1 (2020), 75–85.

[35] George Udny Yule. 1927. On a method of investigating periodicities in disturbed series, with special reference to Wolfer's sunspot numbers. *Philosophical Transactions of the Royal Society of London Series A* 226 (1927), 267–298.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS* 30 (2017).

[37] Gilbert Thomas Walker. 1931. On periodicity in series of related terms. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 131, 818 (1931), 518–532.

[38] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *NeurIPS*. 101–112.

[39] Xinle Wu, Dalin Zhang, Chenjuan Guo, Chaoyang He, Bin Yang, and Christian S. Jensen. 2021. AutoCTS: Automated Correlated Time Series Forecasting. *Proc. VLDB Endow.* 15, 4 (2021), 971–983.

[40] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. 2020. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash (Eds.). ACM, 753–763.

[41] Xiaoming Yuan and Junfeng Yang. 2009. Sparse and low-rank matrix decomposition via alternating direction methods. *preprint* 12, 2 (2009).

[42] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2022. Are Transformers Effective for Time Series Forecasting? *CoRR* abs/2205.13504 (2022).

[43] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *AAAI*, Vol. 35. 11106–11115.

[44] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *ICML*.