Transformers in Time Series Forecasting: A brief Analysis of the Autoformer Transfer Learning Performance

Witesyavwirwa Vianney Kambale
Institute for Smart Systems Technologies
Universitaet Klagenfurt
Klagenfurt, Austria
witesyavwirwa.kambale@aau.at

David Krame Kadurha
Génie Electrique et Informatique
Université Libre des Pays des Grands Lacs (ULPGL)
Goma, Democratic Republic of Congo
kramedavid@gmail.com

Ali Deeb
Institute for Smart Systems Technologies
Universitaet Klagenfurt
Klagenfurt, Austria
ali.deeb@aau.at

Fadi Al Machot
Faculty of Science and Technology
Norwegian University of Life Sciences (NMBU)
Ås, Norway
fadi.al.machot@nmbu.no

Taha Bernabia
Institute of Maintenance and Industrial Security
University of Oran 2
Oran, Algeria
benarbia.taha@univ-oran2.dz

Kyandoghere Kyamakya
Institute for Smart Systems Technologies
Universitaet Klagenfurt
Klagenfurt, Austria
kyandoghere.kyamakya@aau.at

2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS) | 979-8-3503-2649-9/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICECS58634.2023.10382781

***Abstract*— For the task of long-term time-series forecasting, transformer models have shown great potential in achieving high prediction accuracy. For this reason, numerous transformer architectures designed for time series forecasting have been introduced in the literature. Nevertheless, the issue of a limited amount of training data in certain domains is a real challenge in deep learning. Transfer learning promises to be the solution. However, it is essential to develop transformer-based transfer learning techniques that can lead to the design of Transformer-based Time Series Pre-Trained Models (TS-PTMs) that can be used in this situation. This is why, in this paper, we discuss a brief performance analysis of transfer learning techniques applied to the Autoformer, a transformer model designed for time series forecasting, to draw attention to this area. Initial experimental results show potential transfer learning gain. However, given the complexity of transformer models, various transfer learning techniques need to be developed to advance research in this area.**

Keywords*— *Transformer, Time-Series forecasting, Transfer Learning, Pre-Trained Models, Autoformer, Deep Learning.

I. INTRODUCTION

A. Background and Motivation

Time series are ubiquitous. In today's world driven by data, time series data holds a pervasive presence across various domains, including finance, economics, engineering, climate modeling, and resource management. The ability to accurate predictions of future values in a time series is pivotal in empowering businesses and organizations to make well-informed decisions, enhance operational efficiency, and proactively anticipate forthcoming trends. Today, there is an increasing demand for long-term forecasting, especially in critical areas such as electricity consumption planning [1].

Lately, advanced deep learning models, notably Transformers [2], have emerged as powerful tools, potentially standing as the most successful sequence modeling architectures, that have demonstrated remarkable performances across a diverse spectrum of applications, including natural language processing (NLP) [3], speech recognition [4], and computer vision (CV) [5]. For the task of capturing long-range dependencies and modeling intricate patterns in sequential data, transformers have demonstrated exceptional capabilities. Building upon their success in the previously mentioned domains, researchers have embarked on exploring the application of Transformers in the area of time series forecasting. Numerous transformer-based architectures designed for time series forecasting have been developed, and a succinct survey of these is presented in [6], providing the latest insights in the field. Nevertheless, the persisting issue of limited training data and data (for training or for test) drawn from different distributions in certain domains remains a constant challenge in deep learning [7]. In addressing this challenge, transfer learning has been exploited in developing solutions like BERT [3] and GPT [8] in NLP.

Transfer learning, a methodology that harnesses knowledge acquired from one task to enhance performance in another task, has demonstrated its efficacy in enhancing the predictive capabilities of deep learning models. Transfer learning allows a model pre-trained on a large-scale dataset to be fine-tuned on a target task with a smaller dataset. This approach leverages the learned representations from the pre-training task, enabling the model to effectively capture high-level features and generalize well to the new task. Despite transfer learning's notable achievements in diverse domains, its application in time series forecasting, specifically with Transformers, is a relatively unexplored area.

In fact, as highlighted by the authors in [6], while large-scale pre-trained Transformer models have notably enhanced performance across numerous tasks within NLP [8] and CV [9], there are limited works on pre-trained Transformers for time series, with the existing studies predominantly focusing on time series classification [10]. Consequently, this survey [6] underscores the need for further investigation into the development of appropriate pre-trained Transformer models for different tasks in time series.

The focus of this work is on the performance analysis of learning techniques applied to Transformers for time series forecasting. The goal is to explore how pre-trained Transformers can be effectively utilized and fine-tuned to improve the accuracy and efficiency of time series forecasting. By harnessing the knowledge gained from diverse source tasks, we can potentially enhance the

modeling of temporal dependencies, capture intricate patterns, and overcome data scarcity challenges.

B. Problem Statement and Core Objectives

Extensive studies and experiments are needed to develop suitable pre-trained Transformer models tailored to different time series forecasting tasks.

Based on the above problem statement we define the following research objectives:

- To briefly review the current state-of-the-art transformer models developed for time-series forecasting and the state-of-the-art of pre-trained models thereof.
- To identify suitable techniques for conducting a transfer learning performance analysis for transformer models used in time series forecasting.
- To evaluate and compare the performance of transfer learning of transformer-based models with shallow neural networks for time series forecasting with respect to the transfer learning gain.

The rest of the paper is structured as follows. We provide a brief overview of the state-of-the-art transformers and pre-trained models for time series forecasting in Section 2. The models that we use in this study are briefly described in Section 3. We present some techniques for transfer learning in Section 4. Details of our experiments are provided in Section 5, and concluding remarks and future work are given in Section 6.

II. A Brief Summary Of The State-Of-The-Art Of Transformers In Time Series And Of Pre-trained Models Thereof

Researchers are working on developing new variants of Transformers for time series forecasting tasks, inspired by the impressive performance of Transformers in NLP and CV. These variants can be categorized as low-level or high-level modifications of the original vanilla Transformer. The work in [6] provides further insights into these modifications. However, in this section, we will briefly discuss six state-of-the-art Transformer variants for time series: LogTrans [11], Informer [12], Autoformer [13], Pyraformer [14], FEDformer [15], and Conformer [16].

LogTrans utilizes convolutional self-attention layers with a LogSparse design to capture local information and reduce space complexity. Informer, on the other hand, employs a combination of canonical convolutional layers and max-pooling layers to establish connections between self-attention blocks. Autoformer is known for utilizing autocorrelation to establish patch-level connections, but it is a handcrafted design that doesn't incorporate all the semantic information within a patch. On the other hand, Pyraformer applies a pyramidal attention module with inter-scale and intra-scale connections, achieving a linear complexity. FEDformer employs a Fourier enhanced structure to achieve a linear complexity. Finally, Conformer is a new model designed with an encoder-decoder architecture that incorporates linear complexity. This model is claimed to achieve linear complexity without sacrificing information utilization, unlike the other models mentioned [16].

The area of pre-trained Transformers models for time series forecasting is still unexplored, as confirmed by the statement in [6] that such models are scarce in the literature. However, two works can be mentioned: TrafficBERT [17] and TabBERT [18]. TrafficBERT is a BERT-based pre-trained model specifically designed for large-scale traffic data. It captures time-series information by utilizing multi-head self-attention instead of the commonly used recurrent neural network. Similarly, TabBERT is a BERT-based pre-trained architecture designed for tabular time series data.

A brief review of literature shows that Transformer-based pre-trained models are predominantly focused on NLP applications [10]. This fact is also pointed out by the authors of [19], a survey on pre-trained models for time series, who suggest that future research on TS-PTMs (Time Series Pre-Trained Models) should focus on developing suitable Transformer-based pre-training methods for time series forecasting and anomaly detection.

III. Presentation Of The Models Used In The Experiments

The following models are selected for the experiments on the Transfer Learning performance analysis: The Autoformer, the Vanilla Transformer, the LSTM and the MLP.

A. The Autoformer

The Autoformer is one of the transformers designed for long-term time series forecasting. Introduced for the first time by Wu et al. in [13], the Autoformer is based on two key concepts: the series auto-correlation and the series decomposition.

Traditionally, Transformer-based models have been built on the concept of multiple self-attention mechanisms, and this has offered transformer models an edge in the modelling of long-range dependencies for sequential data. But Wu et al. argue that forecasting long-term sequences is an extremely challenging task. Reasons are, determining temporal dependencies directly from the long-term time series is unreliable exercise given that dependencies with the data can be masked by interwoven temporal patterns. Also, Traditional Transformers utilizing self-attention mechanisms face computational challenges due to the quadratic complexity related to the sequence length. Even, some previously developed Transformers have attempted to improve the self-attention structure by using a sparse version instead. Although there's a noticeable enhancement in performance with this approach, these models continue to employ point-wise representation aggregation. Consequently, in spite of the enhance performance, information utilization is sacrificed because of the sparse point-wise connections that lead to a bottleneck in this case of long-term forecasting of time series. These reasons have led to the development of the Autoformer model. Below, we will succinctly discuss its main blocks. As previously mentioned, the architecture of the Autoformer comprises two main components: a series decomposition block and the Auto-Correlation mechanism.

The ***Series decomposition block***, as an inner operation of the Autoformer, has the function of extracting the long-term stationary trend from predicted intermediate hidden variables progressively. It is implemented by adapting the moving average to smooth out periodic fluctuations and highlight the long-term trends. For length-L, input series $\chi \in \mathbb{R}^{L \times d}$, the process is computed as follows:

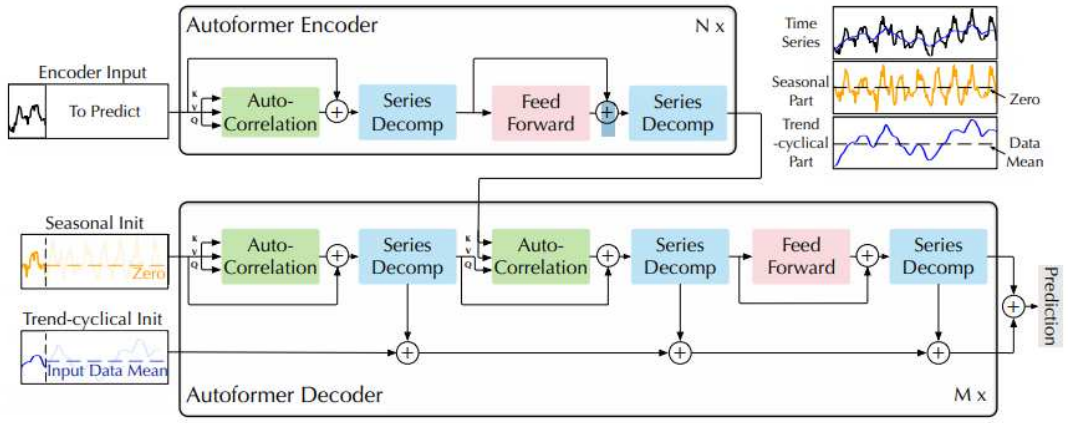

Fig 1. Autoformer architecture (source [13])

$$\chi_t = \text{AvgPool}\,(\text{Padding}(\chi)) \quad (1)$$
$$\chi_s = \chi - \chi_t \quad (2)$$

where $\chi_s, \chi_t \in \mathbb{R}^{L\times d}$ represent the seasonal and the extracted trend-cyclical parts, respectively. The AvgPool $(\cdot)$ is adopted as the moving average, utilizing the padding operation to keep the series length constant.

Regarding the ***Model inputs***, which are the inputs to the encoder block, there are I past steps $\chi_{en} \in \mathbb{R}^{I\times d}$. Because the Autoformer is regarded as a decomposition architecture, the input to the decoder is seen as containing both the seasonal part $\chi_{des} \in \mathbb{R}^{(\frac{I}{2}+O)\times d}$ and the trend-cyclic part $\chi_{det} \in \mathbb{R}^{(\frac{I}{2}+O)\times d}$ which are to be refined. Each initialization has two components: one decomposed from the latter half of encoder's input χ_{en} with length $\frac{I}{2}$ to give recent information, placeholders with length O filled with scalars, formulated as

$$\chi_{ens}, \chi_{ent} = \text{SeriesDecomp}(\chi_{en\,\frac{I}{2}:I}) \quad (3)$$
$$\chi_{des} = \text{Concat}(\chi_{ens}, \chi_0) \quad (4)$$
$$\chi_{det} = \text{Concat}(\chi_{ent}, \chi_{Mean}) \quad (5)$$

where $\chi_{ens}, \chi_{ent} \in \mathbb{R}^{\frac{I}{2}\times d}$ denote the seasonal and trend-cyclical parts of χ_{en} respectively and $\chi_e, \chi_{Mean} \in \mathbb{R}^{O\times d}$ denote the placeholders filled with zero and the mean of χ_{en}, respectively.

The ***Encoder*** block of the Autoformer focuses on modeling the seasonal part. The encoder's output contains past seasonal information and will serve as cross information to aid the decoder in enhancing prediction outcomes. Assume there are N encoder layers, the overall equations for l-th encoder layer can be summarized as $\chi_{en}^{l} = \text{Encoder}(\chi_{en}^{l-1})$ where $l \in \{1, \ldots, \text{N}\}$. More details are found in [13].

The ***Decoder*** block consists of two parts: the accumulation structure for trend-cyclical components and the stacked Auto-Correlation mechanism for seasonal components, as shown in Figure 1. Each decoder layer has both the inner Auto-Correlation and the encoder-decoder Auto-Correlation. These functions are for refining the prediction and utilizing past seasonal information, respectively. Assume there are M decoder layers. With the latent variable χ_{en}^{N} from the encoder, the equation of l-th decoder layer can be summarized as $\chi_{de}^{l} = \text{Decoder}(\chi_{de}^{l-1}, \chi_{en}^{N})$ where $l \in \{1, \ldots, \text{M}\}$. Additional details are found in [13].

An innovative aspect of the Autoformer is the concept of the ***Auto-Correlation*** mechanism. This is a novel mechanism designed to replace the standard self-attention mechanism used in the vanilla transformer. It is based on the series periodicity and conducts the dependencies discovery and representation aggregation at the sub-series level. The Auto-Correlation mechanism is designed to empower the Autoformer with progressive decomposition capacities for complex time series. It draw inspiration from the stochastic process theory and is conducted at the series level.

B. The Vanilla Transformer

We used a Vanilla Transformer as described in [2] and [20], but with the following specifics: 2 encoder-layers and 1 decoder-layer. The hidden dimension of Transformer is set to 512 and the attention is made of 8 heads. The attention function used was the classical scaled dot product attention. For the positional embedding, the classical sine-cosine base positional encoder was used. The dimension of the feed forward layer was set to 2048, and a dropout rate of 0.05 was used. GeLu was used as the activation function. These parameter choices were made to align with those set for the Autoformer model.

C. The LSTM

The next model we use is the LSTM. The LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) architecture that is designed to handle and process sequential data, such as time series, or any other form of data with a temporal dependency. In this work, it is used just for comparison with the Transformer performance. It is set in shallow configuration with 1000 neurons in the hidden layer.

D. The MLP

Finally, we consider using also the MLP. The MLP (Multilayer Perceptron) is a type of artificial neural network (ANN) that consists of multiple layers of interconnected nodes called neurons. MLPs are known to be versatile and have been widely used in various domains for tasks including regression. In this work, it is used just for comparison with the

performance of the Transformer. It is set in shallow configuration with 1000 neurons in the hidden layer.

IV. Transfer Learning Techniques

The following transfer learning techniques are considered for our experiments.

A. Feature extraction

In this technique, the pre-trained Transformer model is used as a fixed feature extractor. Earlier layers of the model are frozen and only the latest layers are fine-tuned on the target domain dataset. This strategy is suitable when the pre-trained model captures relevant temporal patterns that can be generalized to the target task. Feature extraction is implemented in this work by freezing a portion of the trainable part of the model and the other trainable portion fine-tuned by the target data. The part of the Autoformer's architecture that was frozen is the Auto-Correlation block of the encoder, which plays the role of the attention mechanism in the encoder. For the Vanilla transformer, the MHA(Multi-head attention) of the encoder block is frozen, and for the LSTM and MLP, the weights from the input to the hidden layer are frozen.

B. Fine-tuning

This technique entails updating the weights of pre-trained Transformer models in the target domain dataset. This strategy allows the model to adapt to the specific characteristics of the target task while preserving the knowledge gained from the source domain dataset. Depending on the transferability of the learned representations, fine-tuning can be performed on the model as a whole or on individual layers. Fine-tuning is implemented in this work by continuing the training of the model, pre-trained in the source model, in the target domain.

C. Freezing the whole Model.

This is a very naive technique whereby the model, pre-trained in the source domain, is frozen and simply used in the target domain for test.

V. Experiments

A. Experiments setup/Details

To assure comparability with [13], in the experiments we have set the lookback length to 96. All the models are evaluated on the following prediction horizons {96, 192, 384, 768}. All the models were trained on 10 epochs, with an early stopping patience set to 5. However, for fine-tuning, we used 3 epochs with an early stopping patience of 2. In all models, the Adam optimizer was employed with a learning rate of 0.0001, consistent with the Conformer training parameters. During training, Mean Squared Error (MSE) was used as the loss function. However, for testing, both MSE and Mean Absolute Error (MAE) metrics were employed, following the conventional practice for time series forecasting with transformers. Training of some models was done on Nvidia GPU and other models on Google Colab.

B. Description of the Datasets

We conduct the experiments on three datasets including 2 transformers-related benchmark datasets.

ECL: is a processed dataset contains the hourly electricity consumption of 321 clients [21]. We use 'MT 321' as the source dataset and 'MT 319' as the target dataset. The train/val/test split done was 12/4/4 months.

ETT: records the electricity transformer temperature[22]. Every data point consists of six power load features and the target value was OT "oil temperature". This dataset is separated into {ETTh1, ETTh2} and {ETTm1, ETTm2} for 1-hour-level and 15-minute-level observations, respectively. We use ETTh1 as the source dataset and ETTh2 as the target dataset. The train/val/test split done was 12/4/4 months for ETTh1 and ETTh2.

PJM: A set of ten datasets were selected from the PJM Hourly Energy Consumption Data [23]. The datasets were pre-processed and normalized using minmax scaler. For our experiments two datasets were selected t_01 as the source dataset and t_04 as the target dataset. The Pearson and DTW distances of the two time series were calculated (PD = 0.9105, DTW=4300.78). Technically, the aim is to use various distances in the experiments to also check its impact on the transfer learning process. The train/val/test split done was 48/5/5 months.

C. Analysis and discussion of Results

The results of the experiments are presented in Table I, Table II, Table III, and Table IV. We use the abbreviations (S), (T), and (TL) to denote performance in the source domain, target domain, and transfer learning, respectively. In the experiments, we considered the following three transfer learning (TL) techniques: freezing the whole model, fine-tuning, and feature extraction, as discussed in Section 4. The technique of freezing the whole model resulted in noticeable negative transfer outcomes across all four models, prompting us to omit the results. Table I shows the results of the fine-tuning TL technique, while Table III presents the TL gain analysis of this technique. Table II displays the results of the feature extraction TL technique, and Table IV presents the TL gain analysis of this method. The TL gain is computed as a percentage change, providing a scaled value insight into the degree of transferability. Regarding the two transfer learning techniques, the Autoformer performs equally well in both the fine-tuning and feature extraction techniques, with slightly higher gains observed in the feature extraction method. However concerning the datasets, while the Autoformer performs well on the ECL and PJM datasets, it is noted that the Autoformer shows no gain for the ETTh1/2 datasets. A plausible explanation is that since PJM and ECL are both electricity-related datasets, they likely share some similar patterns that the Autoformer was able to capture. Also, the Autoformer had the same hyper-parameter setup for all datasets for comparability. In terms of comparison with other models, the MLP is the worst performer in terms of transfer learning gain. This is likely because this model is not designed for long-term forecasting. Given that these are initial experiments, further actions are to be considered to maximize the TL gains. For instance, further parameter tuning adjustments can be explored for the fine-tuning TL technique in hopes of identifying a clearer trend in the TL gain results. Moreover, additional analysis is needed on the Autoformer architecture, or any other transformer architecture for that matter, to determine which parts of the architecture can be frozen to achieve better TL gain and a more consistent trend in the results.

TABLE I. FINE-TUNING TL TECHNIQUE RESULTS

Models		Autoformer(S)		Autoformer(T)		Autoformer(TL)		VanillaTrans(S)		VanillaTrans(T)		VanillaTrans(TL)		LSTM(S)		LSTM(T)		LSTM(TL)		MLP(S)		MLP(T)		MLP(TL)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	0.2812	0.3975	0.3474	0.4530	0.2942	0.4108	0.1686	0.3059	0.3029	0.4086	0.2869	0.3974	0.4557	0.5005	0.5107	0.5488	0.6426	0.6288	0.6279	0.6415	0.7305	0.6743	0.7369	0.6759
	192	0.3888	0.4822	0.4263	0.5082	0.3893	0.4827	0.1897	0.3298	0.3210	0.4250	0.2980	0.4063	0.5270	0.5589	0.6684	0.6429	0.6694	0.6576	0.7698	0.7012	0.8271	0.7238	0.8255	0.7233
	384	0.4874	0.5445	0.4777	0.5345	0.4236	0.4914	0.2116	0.3457	0.3240	0.4302	0.3200	0.4269	0.6351	0.6504	0.6933	0.6567	0.6897	0.6566	0.8499	0.7313	0.8814	0.7503	0.8823	0.7507
	768	0.5354	0.5656	0.5203	0.5518	0.6077	0.6161	0.2647	0.3951	0.3305	0.4274	0.3281	0.4293	0.7022	0.6851	0.7818	0.6982	0.8779	0.7496	0.9452	0.7641	0.9129	0.7666	0.9113	0.7661
ETTh1/2	96	0.0913	0.2403	0.1476	0.3000	0.2048	0.3539	0.5017	0.6393	0.1918	0.3550	0.2394	0.3951	0.3011	0.4774	0.2798	0.4293	0.2921	0.4378	0.3478	0.5223	0.2346	0.3907	0.2350	0.3901
	192	0.0967	0.2400	0.1842	0.3376	0.2053	0.3556	0.4446	0.6028	0.2756	0.4272	0.2305	0.3889	0.4536	0.5893	0.3113	0.4490	0.3935	0.5097	0.9744	0.8493	0.9053	0.7620	0.9234	0.7740
	384	0.1114	0.2592	0.2270	0.3798	0.2561	0.4053	0.2757	0.4508	0.2189	0.3792	0.2442	0.4010	1.3731	1.1210	0.4684	0.5625	0.4756	0.5660	1.3895	1.0778	1.3065	0.9732	1.2591	0.9526
	768	0.1288	0.2866	0.2754	0.4163	0.2841	0.4289	0.5943	0.7146	0.2262	0.3946	0.2434	0.4076	0.7271	0.7920	0.5123	0.5893	0.6272	0.6569	1.5214	1.1643	1.3263	1.0086	1.2845	0.9898
PJM	96	0.1725	0.3245	0.3403	0.4616	0.2354	0.3715	0.1468	0.2986	0.1932	0.3289	0.2031	0.3423	0.1901	0.3402	0.3164	0.4486	0.2663	0.4061	0.3090	0.4490	0.4341	0.5363	0.4350	0.5367
	192	0.2117	0.3519	0.3869	0.4956	0.3059	0.4303	0.1537	0.2983	0.2526	0.3882	0.2538	0.3863	0.2177	0.3693	0.3333	0.4607	0.2929	0.4301	0.3754	0.4960	0.4958	0.5711	0.4959	0.5711
	384	0.3165	0.4539	0.3753	0.4845	0.4727	0.5462	0.1438	0.3034	0.2599	0.3985	0.3168	0.4246	0.3112	0.4411	0.4391	0.5265	0.4446	0.5348	0.4033	0.5174	0.5262	0.5897	0.5262	0.5897
	768	0.3131	0.4474	0.4962	0.5528	0.4783	0.5457	0.1555	0.3099	0.2354	0.3834	0.2388	0.3778	0.3781	0.4947	0.4668	0.5428	0.5166	0.5837	0.4034	0.5195	0.5178	0.5851	0.5177	0.5851

TABLE II. FEATURE EXTRACTION TL TECHNIQUE RESULTS

Models		Autoformer(S)		Autoformer(T)		Autoformer(TL)		VanillaTrans(S)		VanillaTrans(T)		VanillaTrans(TL)		LSTM(S)		LSTM(T)		LSTM(TL)		MLP(S)		MLP(T)		MLP(TL)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	0.2812	0.3975	0.3474	0.4530	0.2586	0.3790	0.1686	0.3059	0.3029	0.4086	0.2853	0.3951	0.4557	0.5005	0.5107	0.5488	0.6171	0.6152	0.6279	0.6415	0.7305	0.6743	0.7302	0.6741
	192	0.3888	0.4822	0.4263	0.5082	0.3763	0.4736	0.1897	0.3298	0.3210	0.4250	0.3011	0.4092	0.5270	0.5589	0.6684	0.6429	0.6426	0.6410	0.7698	0.7012	0.8271	0.7238	0.8255	0.7233
	384	0.4874	0.5445	0.4777	0.5345	0.4236	0.4914	0.2116	0.3457	0.3240	0.4302	0.3270	0.4299	0.6351	0.6504	0.6933	0.6567	0.6725	0.6479	0.8499	0.7313	0.8814	0.7503	0.8815	0.7503
	768	0.5354	0.5656	0.5203	0.5518	0.5742	0.5957	0.2647	0.3951	0.3305	0.4274	0.3236	0.4248	0.7022	0.6851	0.7818	0.6982	0.7570	0.6868	0.9452	0.7641	0.9129	0.7666	0.9113	0.7661
ETTh1/2	96	0.0913	0.2403	0.1476	0.3000	0.2048	0.3539	0.5017	0.6393	0.1918	0.3550	0.2394	0.3951	0.3011	0.4774	0.2798	0.4293	0.2921	0.4378	0.3478	0.5223	0.2346	0.3907	0.2538	0.4056
	192	0.0967	0.2400	0.1842	0.3376	0.2053	0.3556	0.4446	0.6028	0.2756	0.4272	0.2305	0.3889	0.4536	0.5893	0.3113	0.4490	0.3303	0.4637	0.9744	0.8493	0.9053	0.7620	0.9234	0.7740
	384	0.1114	0.2592	0.2270	0.3798	0.2544	0.4048	0.2757	0.4508	0.2189	0.3792	0.2442	0.4010	1.3731	1.1210	0.4684	0.5625	0.5966	0.6443	1.3895	1.0778	1.3065	0.9732	1.3089	0.9747
	768	0.1288	0.2866	0.2754	0.4163	0.2769	0.4228	0.5943	0.7146	0.2262	0.3946	0.2434	0.4076	0.7271	0.7920	0.5123	0.5893	0.5877	0.6324	1.5214	1.1643	1.3263	1.0086	1.3265	1.0088
PJM	96	0.1725	0.3245	0.3403	0.4616	0.2354	0.3715	0.1468	0.2986	0.1932	0.3289	0.2032	0.3427	0.1901	0.3402	0.3164	0.4486	0.2663	0.4061	0.3090	0.4490	0.4341	0.5363	0.4350	0.5367
	192	0.2117	0.3519	0.3869	0.4956	0.3059	0.4303	0.1537	0.2983	0.2526	0.3882	0.2464	0.3853	0.2177	0.3693	0.3333	0.4607	0.2842	0.4223	0.3754	0.4960	0.4958	0.5711	0.4959	0.5711
	384	0.3165	0.4539	0.3753	0.4845	0.5907	0.6109	0.1438	0.3034	0.2599	0.3985	0.3173	0.4255	0.3112	0.4411	0.4391	0.5265	0.4325	0.5266	0.4033	0.5174	0.5262	0.5897	0.5262	0.5897
	768	0.3131	0.4474	0.4962	0.5528	0.4785	0.5456	0.1555	0.3099	0.2354	0.3834	0.2388	0.3778	0.3781	0.4947	0.4668	0.5428	0.4464	0.5330	0.4034	0.5195	0.5178	0.5851	0.5177	0.5851

TABLE III. FINE-TUNING TL GAIN (%)

Models		Autoformer	VanillaTrans	LSTM	MLP
ECL	96	15.33	5.28	-25.83	-0.88
	192	8.68	7.17	-0.15	0.19
	384	11.33	1.23	0.52	-0.10
	768	-16.80	0.73	-12.29	0.18
ETTh1/2	96	-38.75	-24.82	-4.40	-0.17
	192	-11.42	16.36	-26.41	-2.00
	384	-12.79	-11.56	-1.54	3.63
	768	-3.14	-7.60	-22.43	3.15
PJM	96	30.83	-5.12	15.83	-0.21
	192	20.95	-0.48	12.12	-0.02
	384	-25.95	-21.89	-1.25	0.00
	768	3.60	-1.44	-10.67	0.02

TABLE IV. FEATURE EXTRACTION TL GAIN (%)

Models		Autoformer	VanillaTrans	LSTM	MLP
ECL	96	25.57	5.81	-20.83	0.04
	192	11.73	6.20	3.86	0.19
	384	11.33	-0.93	3.00	-0.01
	768	-10.37	2.09	3.17	0.18
ETTh1/2	96	-38.75	-24.82	-4.40	-8.18
	192	-11.42	16.36	-6.10	-2.00
	384	-12.07	-11.56	-27.37	-0.18
	768	-0.55	-7.60	-14.72	-0.02
PJM	96	30.83	-5.18	15.83	-0.21
	192	20.95	2.45	14.73	-0.02
	384	-57.38	-22.09	1.50	0.00
	768	3.56	-1.44	4.37	0.02

VI. Conclusion And Future Work

This paper has provided a brief performance analysis of transfer learning techniques applied to Transformers for time series forecasting to draw attention to this area. Experimental results of the Autoformer model have been presented in comparison to the vanilla transformer, LSTM, and MLP. These preliminary results, though promising, reveal that various experiments need to be done in this field. Specific insights from the experiments show that further parameter tuning adjustments can be explored for the fine-tuning TL technique in hopes of identifying a clearer trend in the TL gain results. Also, thorough analysis of the transformer architecture is needed in order to determine which parts of the architecture can be frozen to achieve better TL gain and a more consistent tendency in the results. As another further work, a systematic study of the TL performance analysis [24] of main Transformers designed for time series forecasting may yield interesting findings. The aim of this work is to draw attention to this area by encouraging future research on TS-PTMs, focusing on designing suitable Transformer-based pre-training techniques [19] that can be useful in domains where a limited amount of data is experienced.

Acknowledgment

The authors would like to express their gratitude to OeAD – Austria's Agency for Education and Internationalization, for the financial support provided through the OeAD Sonderstipendien, Universität Klagenfurt.

References

[1] G. R. Esteves, B. Q. Bastos, F. L. Cyrino, R. F. Calili, and R. C. Souza, "Long term electricity forecast: a systematic review," in *Procedia Computer Science*, vol. 55, 2015, pp. 549-558.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in Advances in Neural Information Processing Systems, vol. 30, 2017.

[3] M. V. Koroteev, "BERT: a review of applications in natural language processing and understanding," *arXiv preprint arXiv:2103.11943*, 2021.

[4] Y. Wang et al., "Transformer-based acoustic modeling for hybrid speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 6874-6878.

[5] J. Bi, Z. Zhu, and Q. Meng, "Transformer in computer vision," in *2021 IEEE International Conference on Computer Science, Electronic*

Information Engineering and Intelligent Control Technology (CEI), September 2021, pp. 178-188..

[6] Q. Wen et al., "Transformers in time series: A survey," *arXiv preprint arXiv:2202.07125*, 2022.

[7] F. Zhuang et al., "A comprehensive survey on transfer learning," in *Proceedings of the IEEE*, vol. 109, no. 1, 2020, pp. 43-76.

[8] D. Rothman and A. Gulli, "Transformers for Natural Language Processing: Build, train, and fine-tune deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, and GPT-3," Packt Publishing Ltd, 2022.

[9] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, et al., "Pre-trained image processing transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12299-12310.

[10] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 2114-2124.

[11] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y. X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *Advances in neural information processing systems*, vol. 32, 2019.

[12] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 11106-11115.

[13] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," in *Advances in Neural Information Processing Systems*, vol. 34, 2021, pp. 22419-22430.

[14] S. Liu, H. Yu, C. Liao, J. Li, W. Lin, A. X. Liu, and S. Dustdar, "Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting," in *International conference on learning representations*, 2021.

[15] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International Conference on Machine Learning*, 2022, pp. 27268-27286.

[16] Y. Li, X. Lu, H. Xiong, J. Tang, J. Su, B. Jin, and D. Dou, "Towards Long-Term Time-Series Forecasting: Feature, Pattern, and Distribution," *arXiv preprint arXiv:2301.02068*, 2023.

[17] K. Jin, J. Wi, E. Lee, S. Kang, S. Kim, and Y. Kim, "TrafficBERT: Pre-trained model with large-scale data for long-range traffic flow forecasting," *Expert Systems with Applications*, vol. 186, 2021, p. 115738.

[18] I. Padhi, Y. Schiff, I. Melnyk, M. Rigotti, Y. Mroueh, P. Dognin, J. Ross, R. Nair, and E. Altman, "Tabular transformers for modeling multivariate time series," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2021, pp. 3565-3569.

[19] Q. Ma, Z. Liu, Z. Zheng, Z. Huang, S. Zhu, Z. Yu, and J. T. Kwok, "A Survey on Time-Series Pre-Trained Models," *arXiv preprint arXiv:2305.10716*, 2023.

[20] N. Wu, B. Green, X. Ben, and S. O'Banion, "Deep transformer models for time series forecasting: The influenza prevalence case," *arXiv preprint arXiv:2001.08317*, 2020.

[21] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 95–104.

[22] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11 106–11 115.

[23] PJM Interconnection. Hourly Energy Consumption. https://www.kaggle.com/datasets/robikscube/hourly-energy-consumption?resource=download.

[24] W. V. Kambale, A. Deeb, T. Benarbia, M. Salem, F. Al Machot, and K. Kyamakya, "Ensemble Transfer Learning for Time Series Forecasting: a Sensitivity Analysis Framework for a Shallow Neural Network," presented at *27th Int. Conf. CSCC*, to be published in IEEE Xplore, 2023.