Transferability of Neural Forecast Models

Kin G. Olivares*[a], Cristian Challu*[a], Stefania La Vattiata[a],
Azul Garza[b], Max Mergenthaler[b], Artur Dubrawski[a]

[a] *Auton Lab, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA*
[b] *Nixtla, Pittsburgh, Pennsylvania, USA*

Keywords: Time Series, Deep Learning, Meta Learning, Transfer Learning, Domain Adaptation, Automatic Forecasting, Large Forecasting Models

1. Introduction

Neural networks are renowned for their capacity to learn hierarchical representations of inputs. They acquire higher-level features by composing simple functions across layers, learning intricate relationships, and enabling accurate predictions. In the forecasting field, these models excel due to their improved accuracy in large data settings and their ability to simplify the forecasting pipelines, as shown by their success in recent competitions (Makridakis et al., 2020, 2021) and their industry adoption (Wen et al., 2017; Lim et al., 2021).

Despite the recent successes of neural networks in time series forecasting, their full expressivity often remains underutilized in scenarios with small groups of series, and the considerable computational costs remain an adoption barrier. Transfer learning offers a solution by pre-training deep network models on large-scale datasets to capture expressive representations and domain knowledge. These pre-trained models can be fine-tuned on smaller datasets, facilitating effective generalization to specific forecasting tasks with limited data. In addition to knowledge sharing, transfer learning also enables lightning-fast predictions at a fraction of the computational cost while significantly enhancing the models' generalization capabilities (Yosinski et al., 2014; Zhuang et al., 2021).

Our objective in this work is to advance the applications of transfer learning in time series forecasting tasks. We aim to enhance our understanding of the technique and make these tools widely accessible. Our contributions are summarized as follows:

(i) **A Unified Model Implementation.** To ensure consistent experimental settings and facilitate direct comparisons, we introduced a unified implementation of various neural network-based forecasting models; allowing us to control variables such as architecture size, optimization, and standardize the transfer learning tasks.

*Shared First Authorship
Email addresses: kdgutier@cs.cmu.edu (Kin G. Olivares*), cchallu@cs.cmu.edu (Cristian Challu*)

(ii) **Accessible Transfer Learning Comparison.** We explore the impact of neural forecasting innovations on transfer learning tasks and conduct an extensive comparative analysis of various models and datasets with zero-shot and fine-tuning. We make the comparison experiments' code, along with the pre-trained models, publicly accessible.

(iii) **Transferability's Enabling Conditions Exploration.** We document insights from several transferability-enabling conditions. First, we focus on model-related aspects, such as architecture with an emphasis on the encoders, number of parameters, and forecasting strategy. Second, we propose a measure of the *distance* between the source and target tasks strongly correlating with the pre-trained models' performance.

2. Related Work

2.1. ***Transfer Learning Literature***

Transfer learning refers to the techniques that aim to improve a model on a target domain by transferring information or knowledge from another related training/source domain. It has been widely studied for tabular data, computer vision, and natural language processing. There are several different transfer-learning solutions, with data-based solutions focusing on transferring knowledge via the distribution adjustment and transformation of data; and model-based approaches focusing on model regularization, model ensembling, and model parameter sharing. For a throughout review we refer to the following surveys (Weiss et al., 2016; Zhuang et al., 2021).

Notable examples of the distribution adjustment solutions include sample weighting strategies for domain adaptation (Huang et al., 2006; Dai et al., 2007), feature transformation strategies based on minimization of distribution differences (Shen et al., 2018) and feature mappings (Wang et al., 2018). Notable examples of model-based approaches to transfer learning include model parameter sharing (Yosinski et al., 2014), model consensus regularization, and model ensembling (Yao & Doretto, 2010). In this work we mostly focus on model parameter sharing, which is the most popular transfer learning technique in neural network applications.

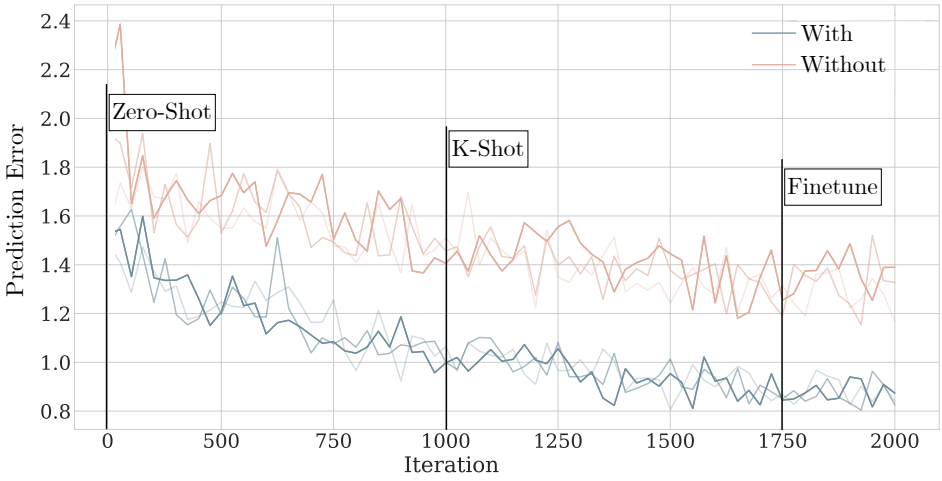


Figure 1: Transfer-learning prediction errors along the optimization trajectory of a model with and without pre-training (randomly initialized). Varying optimization time restrictions correspond to the zero-shot, k-shot, and finetune transfer-learning regimes.

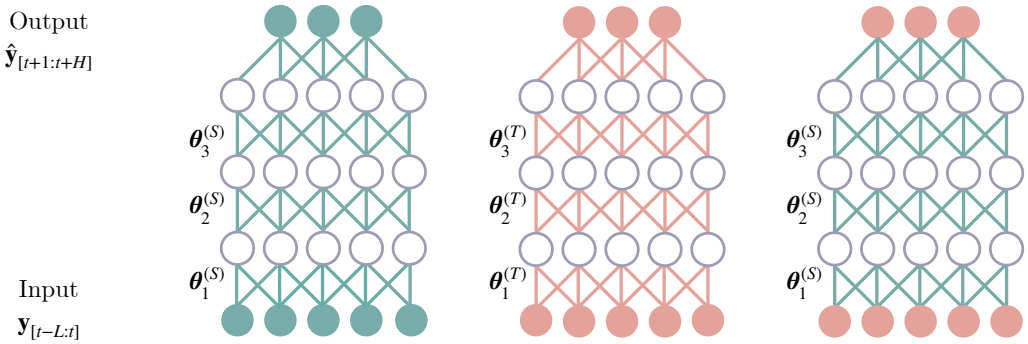
2.2. ***Cross and Transfer Learning Relationship***

In recent years, neural forecasting models have improved over classical methods, overcoming previous computational and accuracy limitations (Makridakis et al., 2018). This progress can be attributed to the widespread adoption of the cross-learning technique (Semenoglou et al., 2021) and the use of global models leveraging data from large collections of related time series. Notable applications include the M4 and M5 competition top performers (Smyl, 2019; Montero-Manso et al., 2020; Oreshkin et al., 2020) and popular industry models like `DeepAR`, `MQCNN`, and `TFT` (Salinas et al., 2020; Wen et al., 2017; Lim et al., 2021). This success has renewed interest in industry and academia, leading to numerous neural forecasting innovations (Längkvist et al., 2014; Benidis et al., 2020).

The cross-learning and transfer-learning via model parameter-sharing are closely related, with their main differences in two key aspects. Firstly, cross-learning allows applying the pre-trained global model to different target datasets. Secondly, transfer learning offers the flexibility to minimize the re-training procedure or entirely skip it, for which it is particularly suited for small dataset applications. On the time series front, the technique's use has been limited but has shown very promising early results in classification (Fawaz et al., 2018), anomaly detection (Wen & Keyes, 2019), contrastive learning pre-training (Eldele et al., 2021) and forecasting meta learning (Oreshkin et al., 2021).

In neural forecasting applications, an immediate approach for transfer learning via model parameter-sharing is using pre-trained neural networks that can be updated using the smaller target dataset. The updating degree determines the scenarios, depicted in Figure 1, ranging from zero-shot to k-shot and fine-tuning [1].

[1]It is important to note that in classification settings, the k-shot category refers to the number of labeled data on the target task; k-shot refers to the number of training iterations in the forecasting domain.



a) Pre-trained model on source dataset $D^{(S)}$

b) Trained model on target dataset $D^{(T)}$

c) Parameter based transfer-learning

Figure 2: The figure shows a three-layer fully connected network predictive function. Classic forecasting applications optimize distinct model parameters for source $D^{(S)}$ and target $D^{(T)}$ datasets, a) and b) columns. Parameter-based transfer-learning leverages source dataset knowledge by using a pre-trained model's parameters $\boldsymbol{\theta}_l^{(S)}$, to initialize another model's parameters $\boldsymbol{\theta}_l^{(T)}$ that can specialize on a target dataset.

3. Transfer Learning Notation

We now present a formal definition of transfer learning, which will be utilized throughout the rest of the paper.

Definition 3.1. (Domain and Task). Two components define a domain, a feature space $\mathcal{X}$ and a marginal probability distribution $\mathbb{P}(\mathbf{x})$, with realizations $\mathbf{x} \in \mathcal{X}$. A task $\mathcal{T} = \{\mathcal{Y},\ f(\cdot)\}$ is defined by two components, a dependent variable space $\mathcal{Y}$ and a *predictive function* $f : \mathcal{X} \mapsto \mathcal{Y}$. In the case of a simple forecasting task, the dependent variable space is $\mathcal{Y} = \{\mathbf{y}_{[t+1:t+H]}\}$, which corresponds to the set of H future values of a series $\mathbf{y}$ at time t, and the feature space $\mathcal{X} = \{\mathbf{y}_{[t-L:t]}\}$ is the set of L past observed values (lags) of the series[2]. The forecasting task is to learn or estimate the parameters $\boldsymbol{\theta}$ in the following regression:

$$f(\mathbf{x}, \boldsymbol{\theta}) = \mathbb{P}\left(\mathbf{y}_{[t+1:t+H]} |\ \mathbf{y}_{[t-L:t]}, \boldsymbol{\theta}\right) \tag{1}$$

Definition 3.2. (Transfer Learning). A training dataset D is defined as the set of realization pairs $D = \{(\mathbf{x}, \mathbf{y}) |\ \mathbf{x} \in \mathcal{X},\ \mathbf{y} \in \mathcal{Y}\}$. Transfer learning breaks the traditional assumption that the training and the test/target datasets are taken from the same domain. Formally, consider two different source and target dataset $(D^{(S)} \neq D^{(T)})$ along their corresponding learning tasks; transfer-learning is the process of improving the predictive function $f^{(T)}(\cdot)$ by using information directly from the source domain $D^{(S)}$ or indirectly through $f^{(S)}(\cdot)$.

In this work, we consider a case of *homogeneous transfer-learning* case where the (autoregressive) feature space $\mathcal{X}^{(S)} = \mathcal{X}^{(T)}$, and the forecasting tasks are the same $\mathcal{T}^{(S)} = \mathcal{T}^{(T)}$, but the distribution of the features are only related $\mathbb{P}^{(S)}(\mathbf{x}) \neq \mathbb{P}^{(T)}(\mathbf{x})$. We focus in particular in the parameter based transfer learning approach.

[2]Note that the feature space is not restricted to autoregressive features can also include exogenous variables predictors like static data, past information, or information available at the time of the predictions.

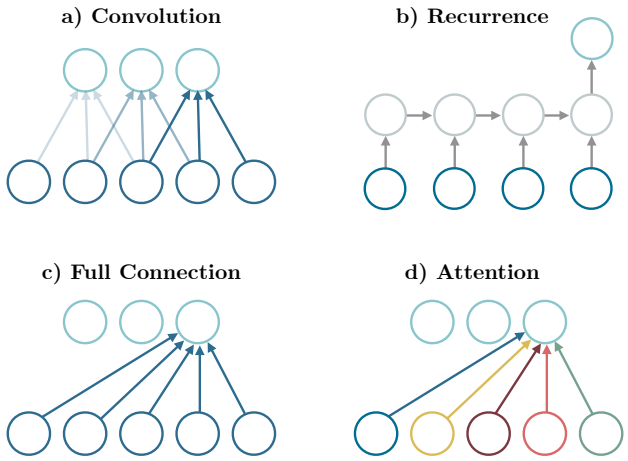


Figure 3: A taxonomy of neural forecasting models, defined by their fundamental building blocks.

3.1. ***Zero-shot, K-shot, and Finetuning***

In general the forecasting learning task can be translated into a function estimation problem using the classic Empirical Risk Minimization (ERM) framework (Vapnik, 1999), where the objective is to minimize the expected loss function at the dataset D level:

$$f^* := \underset{f \in \mathcal{F}(\Theta)}{\arg\min}\, \mathbb{E}_D \left[\mathcal{L}(\mathbf{y}, f(\mathbf{x}, \boldsymbol{\theta})) \right] \quad \Longleftrightarrow \quad \hat{\boldsymbol{\theta}} := \underset{\boldsymbol{\theta} \in \Theta}{\arg\min}\, \mathbb{E}_D \left[\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}(\mathbf{y}_{[t-L:t]} | \boldsymbol{\theta})) \right] \tag{2}$$

In this work pre-training and retraining the considered models is performed using stochastic gradient descent updates (Robbins & Monro, 1951). We used pre-trained model parameters $\boldsymbol{\theta}^{(S)} = \boldsymbol{\theta}_0^{(T)}$ as the initial condition, and update the parameters for the target task using the following rule:

$$\text{Sample } (\mathbf{x}, \mathbf{y}) \sim \mathbb{P}(D^{(T)}) \quad \text{update} \quad \boldsymbol{\theta}_k^{(T)} = \boldsymbol{\theta}_{k-1}^{(T)} - \alpha \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathbf{y}, f(\mathbf{x}, \boldsymbol{\theta})) \tag{3}$$

Where the re-trained model's parameters after k steps are denoted by $\boldsymbol{\theta}_k^{(T)}$, the learning rate is denoted by α. The difference between zero-shot, k-shot, and finetuning dwells in the amount of updating steps performed. Zero-shot requires no re-training, finetuned parameters are re-trained until an early stopping halt, and k-shot performs exactly k-updates.

4. Forecasting Baselines

4.1. ***Neural Forecast Models***

We chose well-performing neural forecasting architectures identified by their fundamental building blocks to establish experiment baselines: fully connected layers, recurrent and convolutional decoders, and attention mechanisms.

Table 1: Summary of datasets used in our empirical study. All the datasets are used in the forecasting transfer-learning experiments. We use larger datasets as source domains and smaller as target domains. Monash repository will be added for the final report.

Dataset	Transfer Purpose	# Series	Frequencies
`Wikipedia`	Source	5e5	{Daily}
`M4`	Source	1e5	{Yearly, Quarterly, Monthly, Daily}
`M3`	Source	3003	{Yearly, Quarterly, Monthly, Daily}
`M1`	Target	1001	{Yearly, Quarterly, Monthly}
`M3`	Target	3003	{Yearly, Quarterly, Monthly, Daily}
`Tourism`	Target	1311	{Yearly, Quarterly, Monthly}

In the fully connected architectures model group, we consider the basic multi-layer perceptron (`MLP`) (Rosenblatt, 1961), the neural basis expansion network (`NBEATS`) (Oreshkin et al., 2020) and the neural hierarchical interpolation network (`NHITS`) (Challu et al., 2023). For architectures with recurrent and convolutional encoders, we include the classic long-short-term memory network (`LSTM`) (Gers et al., 2000; Sak et al., 2014), a temporal convolution network (`TCN`) (van den Oord et al., 2016; Bai et al., 2018), and the deep autoregressive network (`DeepAR`) (Salinas et al., 2020). Finally, for architectures that integrate the attention mechanism, we consider the temporal fusion transformer (`TFT`) (Lim et al., 2021) and the patch time series transformer (`PatchTST`) (Nie et al., 2023).

All architectures are implemented and trained in PyTorch (Paszke et al., 2019), with the code publicly available at the NeuralForecast repository (Olivares et al., 2022).

4.1.1. Automated Statistical Forecast

We include in our experiments two workhorse statistical forecasting baselines: the Autoregressive Integrated Moving Average model (ARIMA) and the simple `Naive` baseline, which surprisingly provides accurate performance reference. For ARIMA, we rely on a modern implementation tool that automatically explores its hyperparameters (Hyndman & Khandakar, 2008; Garza et al., 2022).

5. Empirical Evaluation

5.1. ***Datasets***

All the datasets used in our empirical studies are publicly available and have been previously utilized in the neural forecast literature. In Table 1, we concisely summarize each dataset's sources, time frequencies, and purpose for our transfer learning tasks. Below we describe in greater detail the origins and dataset characteristics.

- Wikipedia (`Wikipedia`): This large dataset comprises time series data from Wikipedia articles' visits and logs. With approximately half a million series, this dataset provides valuable insights into the temporal patterns of Wikipedia usage.

Table 2: Model evaluation based on **sCRPS** on **zero-shot** forecasting transfer learning; lower values are better. The source dataset is the first vertical column, while the target dataset is the second column. Metrics are averaged over eight runs and standard deviation in brackets, best results are highlighted in bold.

		Freq	Naive	AutoARIMA	MLP	LSTM	TCN	DeepAR	TFT	Trans	NHITS	PatchTST
M3	M1	Y	0.184	0.160	0.116	0.216	0.264	0.198	**0.093**	0.393	0.130	0.134
		Q	0.102	**0.088**	0.124	0.149	0.115	0.095	0.125	0.148	0.105	0.112
		M	0.197	0.134	0.125	0.133	0.130	0.164	0.119	0.121	**0.107**	0.110
	Tour	Y	0.112	0.114	0.112	0.161	0.112	**0.096**	0.114	0.272	0.123	0.098
		Q	0.206	0.121	0.163	0.279	0.194	0.127	0.146	0.224	0.157	**0.109**
		M	0.412	**0.145**	0.202	0.200	0.207	0.208	0.197	0.182	0.172	0.173
M4	M1	Y	0.184	0.160	0.128	0.211	0.205	0.133	0.119	0.186	**0.118**	0.140
		Q	0.102	0.088	0.077	0.092	0.110	0.085	0.076	0.080	0.079	**0.072**
		M	0.197	0.134	0.118	0.127	0.133	0.160	0.119	**0.103**	0.111	0.110
	M3	Y	0.138	0.162	0.139	0.159	0.170	0.156	0.134	0.176	0.136	**0.122**
		Q	0.086	0.079	0.069	0.081	0.086	0.094	0.070	0.075	**0.066**	0.067
		M	0.138	0.088	0.087	0.093	0.099	0.106	0.088	0.085	**0.085**	0.085
		D	0.067	**0.053**	0.061	0.094	0.078	0.137	0.063	0.061	0.071	0.068
	Tour	Y	0.112	0.114	0.107	0.098	**0.091**	0.096	0.099	0.148	0.101	0.102
		Q	0.206	0.121	0.107	0.104	0.119	0.150	0.104	0.142	0.102	**0.096**
		M	0.412	**0.145**	0.170	0.189	0.198	0.217	0.164	0.151	0.148	0.169
Wiki	M3	D	0.067	**0.053**	0.068	0.082	0.085	0.147	0.064	0.069	0.066	0.079

- Makridakis Competitions (M1, M3, M4): The M4 dataset (Makridakis et al., 2020) is a comprehensive one hundred thousand time series collection from diverse domains such as finance, industry, macroeconomics, and microeconomics. It offers a wide range of frequencies from yearly to hourly intervals. The M3 dataset (Makridakis & Hibon, 2000) shares similarities with M4, although it consists of a smaller set of 3003 series. Notably, the M3 dataset (Fiorucci et al., 2016; Spiliotis et al., 2020; Hyndman & Khandakar, 2008) has been extensively utilized for research on statistical methods. Additionally, the M1 dataset (Makridakis et al., 1982) comprises 1001 time series representing demography, industry, and economics. M1 dataset's time series are only available in yearly, quarterly, and monthly frequencies.

- The Tourism dataset (Athanasopoulos et al., 2011) consists of yearly, quarterly, and monthly records of tourist visits and other indicators sourced from tourism bodies or academics. This dataset has been utilized in previous tourism forecasting studies.

- The Monash Time Series Forecasting Repository (Monash) is a comprehensive collection of time series datasets designed to facilitate the evaluation of global and multivariate forecasting algorithms. It includes 25 publicly available datasets, each with varying characteristics such as frequency, series length, and the presence of missing values (Godahewa et al., 2021). This repository provides researchers with a diverse benchmark for assessing algorithm performance across different time series scenarios.

Table 3: Model evaluation based on **sMAPE** on **zero-shot** forecasting transfer learning (lower is better). The source dataset is the first vertical column, while the target dataset is the second column. Metrics are averaged over eight runs and standard deviation in brackets, best results are highlighted in bold.

		Freq	Naive	AutoARIMA	MLP	LSTM	TCN	DeepAR	TFT	Trans	NHITS	PatchTST
M3	M1	Y	22.43	19.53	**15.96**	36.51	32.49	29.28	18.08	51.48	17.15	16.50
		Q	18.38	17.57	18.15	24.46	22.93	18.31	17.42	23.24	17.30	**15.74**
		M	18.67	13.72	16.35	16.31	16.37	17.27	15.40	15.37	14.43	**13.27**
	Tour	Y	34.80	39.76	36.02	60.18	37.79	**29.55**	35.41	106.13	38.59	32.34
		Q	54.19	49.74	58.02	80.07	70.30	48.17	52.55	78.67	56.16	**46.76**
		M	72.24	65.42	65.74	64.77	64.78	71.47	**63.52**	66.31	66.28	63.76
M4	M1	Y	22.43	19.53	15.70	24.15	25.41	22.94	15.62	21.50	**15.62**	16.52
		Q	18.38	17.57	15.53	17.09	18.39	17.98	15.48	16.47	15.33	**15.18**
		M	18.67	13.72	13.38	15.78	16.01	17.62	13.24	12.94	**12.71**	12.80
	M3	Y	17.88	18.13	16.12	19.55	21.33	20.59	15.64	22.87	15.68	**15.30**
		Q	11.32	10.57	9.32	10.47	11.20	12.46	9.37	10.37	**8.95**	9.06
		M	16.85	13.47	13.46	13.60	14.48	15.40	13.24	13.13	12.97	**12.82**
		D	8.56	**6.91**	8.44	10.79	10.03	15.54	8.47	8.27	9.55	8.88
	Tour	Y	34.80	39.76	35.25	31.38	32.15	**29.50**	31.74	51.32	31.19	31.13
		Q	54.19	49.74	45.41	44.32	44.92	50.70	45.49	51.47	45.09	**43.62**
		M	72.24	65.42	64.80	62.88	65.09	71.12	**62.40**	63.96	63.26	62.85
Monash	M3	Y	17.88	18.13	16.11	18.39	19.20	20.63	15.60	20.42	15.33	**15.25**
		Q	11.32	10.57	9.69	10.02	10.98	11.86	9.25	9.93	**8.92**	**8.92**
		M	16.85	13.47	13.32	13.97	14.24	15.17	12.75	12.84	12.68	**12.60**
		D	8.56	**6.91**	7.70	9.13	9.33	15.42	7.39	8.38	8.76	8.31
	Tour	Y	34.80	39.76	34.27	31.30	31.29	30.41	29.70	48.28	29.43	**29.40**
		Q	54.19	49.74	44.16	44.21	44.25	49.13	43.95	47.81	**43.41**	43.56
		M	72.24	65.42	64.70	62.96	64.51	70.58	**62.14**	64.09	62.30	62.22
Wiki	M3	D	8.56	**6.91**	7.91	9.15	9.57	16.71	7.45	7.91	7.72	9.88

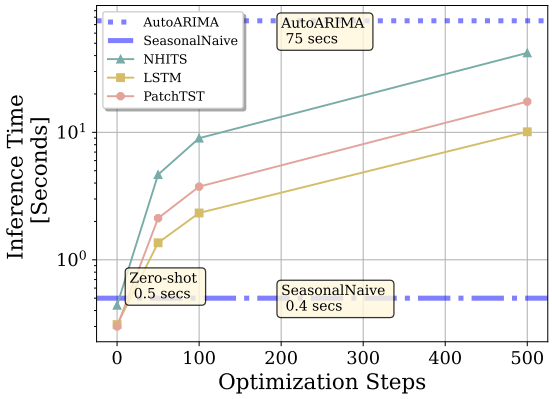


Figure 4: Inference time comparison for selected neural method and baselines. Values reported correspond to the average inference time for the complete target datasets from Table 3.

5.2. ***Evaluation Metrics***

To assess the models' probabilistic and point forecast accuracy, we employ the Scaled Continuous Ranked Probability Score (sCRPS; Makridakis et al. 2022) and the Symmetric Mean Average Percentage Error (sMAPE; Hyndman & Koehler 2006).

$$\begin{aligned} \text{sCRPS}(\hat{\mathbb{P}}, \mathbf{y}) &= \frac{2}{H \times |[i]|} \sum_{i} \sum_{\tau=t+1}^{t+H} \frac{\int_0^1 \text{QL}(\hat{\mathbb{P}}_{i,\tau}, y_{i,\tau})_q dq}{\sum_i |y_{i,\tau}|} \\ \text{sMAPE}(\mathbf{y}, \hat{\mathbf{y}}) &= \frac{200}{H \times |[i]|} \sum_{i} \sum_{\tau=t+1}^{t+H} \frac{|y_\tau - \hat{y}_\tau|}{|y_\tau| + |\hat{y}_\tau|} \end{aligned} \tag{4}$$

where $\text{QL}(\hat{\mathbb{P}}_{i,\tau}, y_{i,\tau})_q$ stands for the quantile loss at the q level, between the estimated forecast probability $\hat{\mathbb{P}}_{i,\tau}$ and the observation $y_{i,\tau}$. We use a Riemann approximation to the sCRPS with dq quantile intervals of 1 percent.

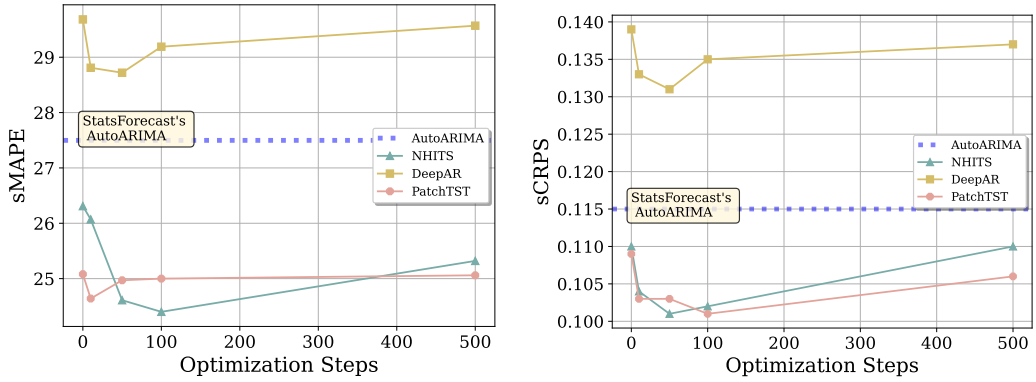
In addition to measuring forecast accuracy, we also evaluate the computational time complexity of the methods. We conduct the experiments on an EC2 g5.4xlarge instance with NVIDIA Tesla M60 GPUs and 16 CPU cores.

5.3. ***Key Results***

As mentioned in Section 3, we define a transfer-learning task by combining a source and a target dataset. We report the performance of zero-shot models, that is, pre-trained models with no additional training. We use large datasets as sources, for which we select `Wikipedia`, and `M4`. The target datasets, `M3`, `M1`, and `Tourism` are smaller.

5.3.1. Zero-shot

Our findings summarized in Table 3 and Table 3 confirm that neural networks can acquire general forecasting knowledge and effectively apply it in zero-shot transfer. Pre-training on larger source datasets improves performance on downstream target tasks. In addition, the



(a) *sMAPE* (b) *sCRPS*

Figure 5: Finetuning forecasting performance for selected neural forecast models, measured by average sMAPE and sCRPS across all transfer-learning tasks.

latest architectures, including `NHITS`, `TFT`, and `PatchTST`, are better suited for leveraging large datasets' information. The best pre-trained `PatchTST` model shows improvements over automated `AutoARIMA` of 8.89% on average across all tasks. On the other side, other architectures like `LSTM` do not manage to improve on `AutoARIMA`'s performance on average.

These results also show that, in most cases, the accuracy of transferred neural forecast models improves when the source dataset is larger. For example, the best pre-trained model on Tourism monthly is the `NHITS`, with a sCRPS of 0.172 when trained in M3 and 0.148 when trained in M4, representing a 14% improvement. In section 6.2 we explain this finding based on the *distance* between tasks.

5.3.2. Inference Time

One of the most significant advantages of zero-shot transfer learning is the ability to produce forecasts from a model with a fraction of the computational cost of complete training procedures. In Figure 4, we report the average inference time to forecast a complete target dataset. Zero-shot inference requires almost the same computational time as the simplest baseline, the `Naive`, and 145 times less time to fit and predict with the highly efficient `AutoARIMA` from StatsForecast. Moreover, neural methods can be fine-tuned for up to 500 training iterations, requiring less time than the `AutoARIMA`.

5.3.3. Finetuning

Figure 5 presents the results of fine-tuning neural methods in the target task for multiple optimization steps. The results show that it is possible to improve the zero-shot performance by customizing the pre-trained models on the task at hand. Updating the model weights on the smaller datasets tends to be beneficial. However, the benefits diminish with the training iterations, and in the case of large models, there is a potential risk of overfitting, for which early stopping is necessary. In this experiment, the finetune `NHITS` model achieves average sMAPE improvements considering all tasks of 11.26% over the `AutoARIMA` with just 100 optimization steps and under 10 seconds of fine-tuning per task.

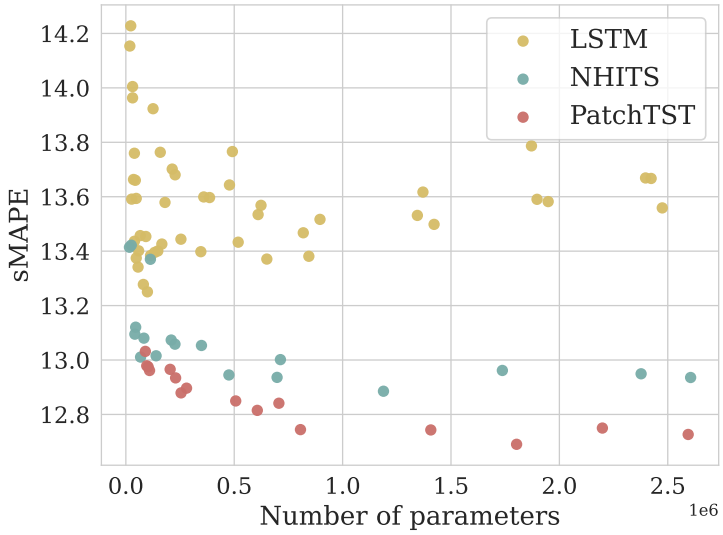


Figure 6: sMAPE performance of zero-shot models as a function of model size. While `LSTM` exhibits a clear bias-variance tradeoff, modern architectures like `NHITS` and `PatchTST` show no significant performance degradation with larger models, suggesting they utilize the pre-training dataset more effectively.

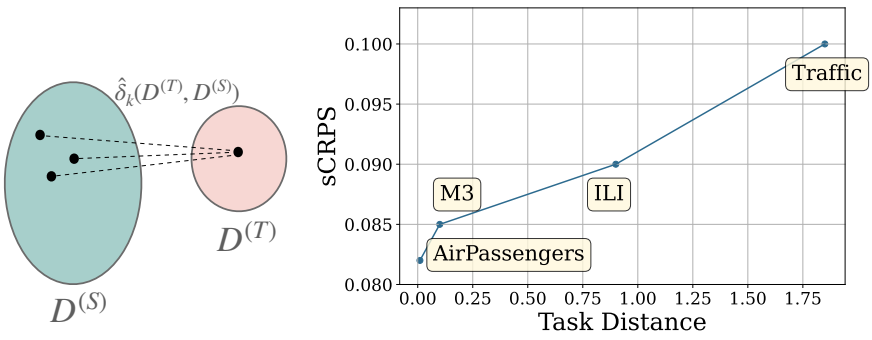
6. Transferability Enabling Conditions.

6.1. Model Size effects on Zero-Shot Performance.

Across machine learning domains, a direct relationship has been observed between model size and model performance. This trend is evident in large language models (Kaplan et al., 2020), image processing, and speech recognition (Hestness et al., 2017). In this section, we investigate the effects of model size on zero-shot forecasting models.

In this experiment, we use default parameters from the NeuralForecast library (Olivares et al., 2022), only varying the number of parameters of the models by increasing the hidden size of the units on three models: 1) `NHITS` (Challu et al., 2023), 2) `PatchTST` (Nie et al., 2023), and 3) `LSTM` (Sak et al., 2014; Zhou et al., 2019). In a similar fashion to the experiment in Section 6.4 we used the monthly `M4` data set using a random split of 80 percent pre-train and 20 percent test series, and measure the precision of the forecast using the *symmetric Mean Average Percentae Error* (sMAPE) in Equation 4.

Figure 6 highlights the performance gains associated with over-parameterization in zero-shot forecasting on the monthly `M4` dataset. The difference in accuracy between the smallest and the largest models is 4.25% for `LSTM`, 3.35% for `NHITS`, and 3.05% for `PatchTST`. These results suggest that modern architectures like `PatchTST` and `NHITS` utilize their parameters more efficiently and exhibit less overfitting compared to `LSTM`. However, optimizing larger models requires careful tuning and the application of early stopping (Yao et al., 2007), as prolonged training can degrade performance.



(a) *Task Distance* (b) *Distance Effects*

Figure 7: Transferability decreases as the distance between task increases. (a) Distance nearest neighbor estimation between forecasting tasks. (b) `NHITS` zero-shot performance as a function of task distance.

6.2. Effects of Tasks Distance.

In this section, we explore how differences in the temporal patterns of the time series between the source and target datasets can affect the transferability of pre-trained models, regardless of their architecture. We first propose a measure of the *distance* between the source and target datasets that strongly correlates with their transferability and, therefore, the performance of pre-trained models.

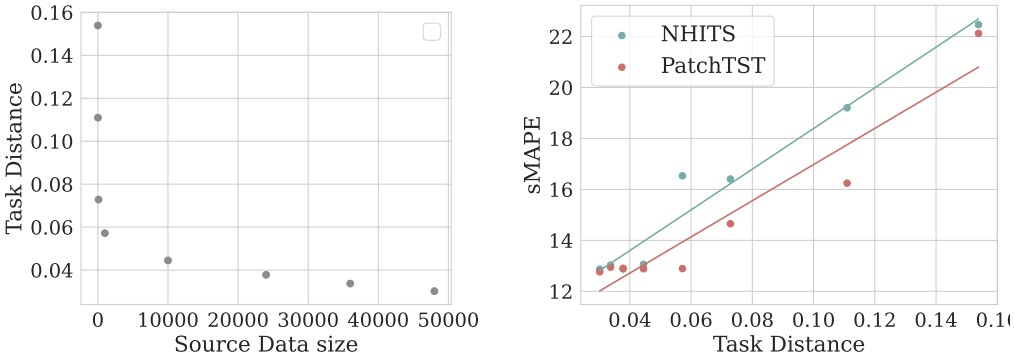
Let a window of autoregressive features and forecast horizon be denoted $\mathbf{w} = \mathbf{y}_{[t-L:t+H]}$. Let the source domain and target forecasts datasets be $D^{(S)} = \{(\mathbf{y}_{[t-L:t]}, \mathbf{y}_{[t+1:t+H]}) \,|\mathbf{y} \in \mathcal{Y}^S\}$ and $D^{(T)} = \{(\mathbf{y}_{[t-L:t]}, \mathbf{y}_{[t+1:t+H]}) \,|\mathbf{y} \in \mathcal{Y}^T\}$. We estimate the distance between the target and the source tasks as follows:

$$\hat{\delta}_k \left(D^{(T)},\ D^{(S)}\right) = \frac{1}{k \times |\mathcal{W}_t|} \sum_{\mathbf{w} \in \mathcal{W}_t} \sum_{\mathbf{w}_\kappa \in \mathrm{N}_k(\mathbf{w})} ||\mathbf{w} - \mathbf{w}_\kappa||_2 \tag{5}$$

where $\mathrm{N}_k(\mathbf{w})$ is the neighborhood of $\mathbf{w}_\kappa$ closest source windows to the $\mathbf{w}$ target window. This proposed distance is motivated by the fact that most deep-learning models can be seen as functions that map a fixed amount of historical values (lags) L, to the forecasting horizon of interest H. By setting the window size $\mathbf{w}$ to $L + H$, $\hat{\delta}_k \left(D^{(T)},\ D^{(S)}\right)$ simultaneously measures if the source dataset contains windows with similar input-to-output mappings as the target dataset.

For this experiment, we selected additional target datasets exhibiting varying differences in frequency, seasonality, and trends from the nature of the source datasets in Table 1.

- Box-Jenkins Airline Passengers (`AirPassengers`): This classic dataset (Box et al., 2015) captures monthly totals of international passengers from 1949 to 1960, exhibiting prominent trends and seasonal patterns.
- Influenza-like Illness (`ILI`): This dataset reports weekly recorded influenza-like illness patients from the Centers for Disease Control and Prevention of the United States from 2002 to 2021 (US-CDC, 2017).



(a) *Task Distance vs Dataset Size* (b) *Performance vs Task Distance*

Figure 8: Relationship between tasks distance, model performance and dataset sizes. (a) As the dataset size increases the minimum distance between series in the dataset of an out of sample task decreases. (b) `NHITS`/PatchTST zero-shot performance as a function of task distance.

- San Francisco Bay Area Highway Traffic (`Traffic`): The California Department of Transportation collected this dataset (Dua & Graff, 2017), reporting hourly road occupancy rates of 862 sensors from January 2015 to December 2016.

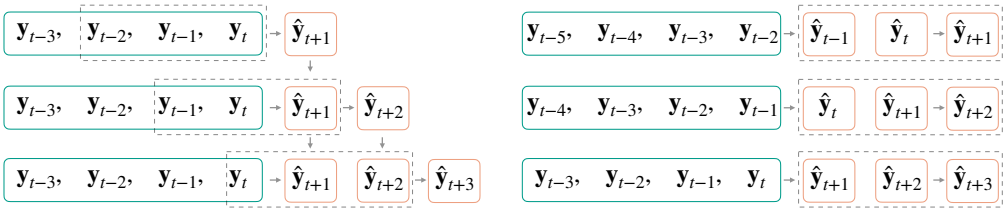
Figure 7 presents the zero-shot performance of the `PatchTST` pre-trained in `M4`, the model with the best overall performance, against the task distance for four target datasets. Our experiment shows that the zero-shot performance of `PatchTST` declines as the target datasets deviate from the source dataset. We expect reasonable transfer learning performance when the source domain contains some highly similar series to the target task.

6.3. Another Example of the Effects of Tasks Distance.

To expand on Section 6.2 and explore the conditions that allow transferability, we examine the relationship between the size of the pre-training dataset and the zero-shot performance of the models. For this analysis, we used the monthly `M4` dataset and created subsets by incrementally sampling between one and fifty thousand series with uniform probability.

We measured the task distance of the monthly `M4` dataset relative to the monthly `M3` dataset. As shown in Figure 7, there is a clear inverse relationship between task distance (defined by Equation 5) and dataset size: adding more series to the pre-training dataset reduces the minimum distance of an out-of-sample task to the pre-training data.

Consistent with Figure 7(b), the performance of an out-of-sample task improves as its distance from the source dataset decreases. This is intuitive, as smaller distances reflect greater similarity between the forecast series and the pre-training series, resulting in better performance. Figure 8(b) further illustrates the strong link between task distance and the inclusion of samples from the `Monash` dataset in the pre-training data. These findings highlight an effective way for improving the performance of pre-trained models: increasing the size of pre-trained datasets to better support a wider range of unseen tasks.



(a) *Recursive Forecast* (b) *Multi-Step Forecast*

Figure 9: Illustration of the recursive and multi-step forecast strategies.

Table 4: Empirical evaluation of the *symmetric Mean Absolute Percentage Error* (sMAPE) for zero-shot models on the monthly `M4` dataset. The third column shows the percentage difference in performance between models trained on the full dataset and those trained on constrained data. Lower values indicate better performance, with the best results highlighted in bold.

Model	Recursive	Multi-Step	Difference
`NHITS`	13.16	**12.97**	1.47
`MLP`	17.85	**13.46**	24.59
`TCN`	15.03	**14.48**	3.67
`PatchTST`	13.08	**12.82**	1.95
`Transformer`	15.03	**13.13**	12.64

6.4. Recursive vs Multi-Step Forecast Strategy effects on Zero-Shot Performance

In this experiment, we investigate the impact of forecasting strategies on transferability as an enabling condition. Specifically, we compare two primary strategies: recursive and multi-step forecasting. The key distinction between these approaches lies in how the forecasts are generated. The recursive strategy uses the model, producing forecasts for successive horizons based on its own prior output. In contrast, the multi-step strategy generates the entire forecast window in a single pass (Bao et al., 2014; Amir & Souhaib, 2016).

In this experiment, we used the monthly `M4` dataset, using a random split between 80 percent pre-train and 20 percent test series. In addition, we use the last 12 months of the pre-train series as the validation signal for early stopping. We train five models with distinct forecasting strategies: 1) `NHITS` (Challu et al., 2023), 2) `MLP` (Rosenblatt, 1961), 3) `TCN` (Bai et al., 2018; Wen et al., 2017), 4) `PatchTST` (Nie et al., 2023), and 5) a simple `Transformer`. For all models, we use their default parameters as implemented in the NeuralForecast library (Olivares et al., 2022), and measure the accuracy of the forecast using *symetric Mean Average Percengage Error* (sMAPE) in Equation 4.

Table 4 confirms observations from the ML community that the multistep forecast strategy is generally preferred compared to the recursive alternative. With zero-shot accuracy percentage improvements ranging from 1.47 to 24.59 percent. A careful revisit of the multi-step forecast strategy with respect to smaller or bigger dataset sizes could cement our findings, yet from a lot of our practical experience, the multi-step forecast strategy is preferred.

7. Discussion

The default approach in Natural Language Processing (NLP) uses pre-trained models due to the considerable advantages in accuracy and efficiency. Large language models can be used without needing to train them from scratch, saving considerable computation and time; such models trained on large and diverse datasets can learn complex representations useful for plenty of smaller tasks. The goal of this goal is to serve as a starting point for the broader adoption of pre-trained models for the forecasting task, as we believe that pre-trained models and transfer learning will become the default solution for forecasting projects.

While pre-trained models show impressive results, they still have limitations. A potential challenge can emerge when the target domain differs substantially from the source domains, distribution shifts, or the presence of very particular exogenous variables. Forecasting domain adaptation and heterogeneous transfer learning are exciting lines of research, as they will allow pre-trained models to be of service beyond simple univariate forecasting tasks. Some neural forecasting architectures have begun incorporating a shared latent feature space, enabling their application across diverse input scenarios. However, further research is required to unlock their potential and broaden their applicability.

Most research on time series transfer-learning research has centered around a single point or probabilistic forecasting task, yet there needs to be more research on the study of pre-trained models' modularity. Models able to separate their components and adapt their outputs for various tasks will be the most successful. Examples of desirable model flexibility are the ability to quickly switch forecast horizons, temporal frequencies, and point and probabilistic outputs.

8. Conclusion

In this work, we study the transferability of neural network-based forecasting models. For our experiments, we pre-trained a collection of well-performing architectures on several large time series datasets. We applied the models after zero-shot, k-shot, and fine-tuning on smaller datasets. We confirmed that pre-trained zero-shot models can significantly improve accuracy and inference time on widely adopted statistical forecasting tools.

In addition, we investigated transferability's enabling conditions. On the model side, we found that architecture, number of learnable parameters, and forecasting strategy play crucial roles in leveraging the information from large source datasets. Additionally, fine-tuning improves accuracy over zero-shot, but runs the risk of overfitting if done without early stopping. With respect to the data, we propose a novel metric to measure the distance between the source and target datasets that is highly correlated with the transferability of the forecasting models. We demonstrate that transfer learning accuracy increases as the distance between the forecasting tasks decreases, showing the importance of using large diverse datasets for pre-training models.

References

Amir, A., & Souhaib, B. (2016). A bias and variance analysis for multistep-ahead time series forecasting. *IEEE transactions on neural networks and learning systems*, *27*, 2162–2388. URL: https://pubmed.ncbi.nlm.nih.gov/25807572/.

Athanasopoulos, G., Hyndman, R. J., Song, H., & Wu, D. C. (2011). The tourism forecasting competition. *International Journal of Forecasting*, *27*, 822–844. URL: https://www.sciencedirect.com/science/article/pii/S016920701000107X. doi:https://doi.org/10.1016/j.ijforecast.2010.04.009. Special Section 1: Forecasting with Artificial Neural Networks and Computational Intelligence Special Section 2: Tourism Forecasting.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *Computing Research Repository*, *abs/1803.01271*. URL: http://arxiv.org/abs/1803.01271. arXiv:1803.01271.

Bao, Y., Xiong, T., & Hu, Z. (2014). Multi-step-ahead time series prediction using multiple-output support vector regression. *Neurocomputing*, *129*, 482–493. URL: https://www.sciencedirect.com/science/article/pii/S092523121300917X. doi:https://doi.org/10.1016/j.neucom.2013.09.010.

Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, B., Maddix, D., Turkmen, C., Gasthaus, J., Bohlke-Schneider, M., Salinas, D., Stella, L., Callot, L., & Januschowski, T. (2020). Neural forecasting: Introduction and literature overview. *Computing Research Repository*, . URL: http://arxiv.org/abs/2004.10240.

Box, G., Jenkins, G., Reinsel, G., & Ljung, G. (2015). *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. Wiley. URL: https://books.google.com/books?id=rNt5CgAAQBAJ.

Challu, C., Olivares, K. G., Oreshkin, B. N., Garza, F., Mergenthaler, M., & Dubrawski, A. (2023). N-HiTS: Neural Hierarchical Interpolation for Time Series forecasting. In *The Association for the Advancement of Artificial Intelligence Conference 2023 (AAAI 2023)*. URL: https://arxiv.org/abs/2201.12886.

Dai, W., Yang, Q., Xue, G.-R., & Yu, Y. (2007). Boosting for transfer learning. In *Proceedings of the 24th International Conference on Machine Learning* ICML '07 (p. 193–200). New York, NY, USA: Association for Computing Machinery. URL: https://doi.org/10.1145/1273496.1273521. doi:10.1145/1273496.1273521.

Dua, D., & Graff, C. (2017). UCI machine learning repository. URL: http://archive.ics.uci.edu/ml.

Eldele, E., Ragab, M., Chen, Z., Wu, M., Kwoh, C. K., Li, X., & Guan, C. (2021). Time-series representation learning via temporal and contextual contrasting. URL: https://arxiv.org/abs/2106.14112. arXiv:2106.14112.

Fawaz, H. I., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2018). Transfer learning for time series classification. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE. doi:10.1109/bigdata.2018.8621990.

Fiorucci, J. A., Pellegrini, T. R., Louzada, F., Petropoulos, F., & Koehler, A. B. (2016). Models for optimising the theta method and their relationship to state space models. *International Journal of Forecasting*, *32*, 1151–1161. URL: https://www.sciencedirect.com/science/article/pii/S0169207016300243. doi:https://doi.org/10.1016/j.ijforecast.2016.02.005.

Garza, F., Canseco, M. M., Challú, C., & Olivares, K. G. (2022). StatsForecast: Lightning Fast Forecasting with Statistical and Econometric Models. PyCon Salt Lake City, Utah, US 2022. URL: https://github.com/Nixtla/statsforecast.

Gers, F. A., Cummins, F., & Schmidhuber, J. (2000). Learning to forget: continual prediction with LSTM. *Neural Computation*, *12*, 2451–2471. URL: https://digital-library.theiet.org/content/conferences/10.1049/cp_19991218.

Godahewa, R., Bergmeir, C., Webb, G. I., Hyndman, R. J., & Montero-Manso, P. (2021). Monash time series forecasting archive. In *Neural Information Processing Systems Track on Datasets and Benchmarks*. URL: https://forecastingdata.org/.

Hestness, J., Narang, S., Ardalani, N., Diamos, G. F., Jun, H., Kianinejad, H., Patwary, M. M. A., Yang, Y., & Zhou, Y. (2017). Deep learning scaling is predictable, empirically. *CoRR*, *abs/1712.00409*. URL: http://arxiv.org/abs/1712.00409. arXiv:1712.00409.

Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., & Smola, A. (2006). Correcting sample selection bias by unlabeled data. In B. Schölkopf, J. Platt, & T. Hoffman (Eds.), *Advances in Neural Information Processing Systems*. MIT Press volume 19. URL: https://proceedings.neurips.cc/paper_files/paper/2006/file/a2186aa7c086b46ad4e8bf81e2a3a19b-Paper.pdf.

Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for r. *Journal of Statistical Software, Articles*, *27*, 1–22. URL: https://www.jstatsoft.org/v027/i03. doi:10.18637/jss.v027.i03.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*, 679 – 688. URL: http://www.sciencedirect.com/science/article/pii/S0169207006000239. doi:https://doi.org/10.1016/j.ijforecast.2006.03.001.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. *CoRR*, *abs/2001.08361*. URL: https://arxiv.org/abs/2001.08361. arXiv:2001.08361.

Lim, B., Arık, S. O., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, *37*, 1748–1764. URL: https://www.sciencedirect.com/science/article/pii/S0169207021000637. doi:https://doi.org/10.1016/j.ijforecast.2021.03.012.

Längkvist, M., Karlsson, L., & Loutfi, A. (2014). A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, *42*, 11–24. URL: https://www.sciencedirect.com/science/article/pii/S0167865514000221. doi:https://doi.org/10.1016/j.patrec.2014.01.008.

Makridakis, S., Andersen, A., Carbone, R., Fildes, R., Hibon, M., Lewandowski, R., Newton, J., Parzen, E., & Winkler, R. (1982). The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting*, *1*, 111–153. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.3980010202. doi:https://doi.org/10.1002/for.3980010202. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.3980010202.

Makridakis, S., & Hibon, M. (2000). The M3-competition: results, conclusions and implications. *International Journal of Forecasting*, *16*, 451–476. URL: https://www.sciencedirect.com/science/article/pii/S0169207000000571. doi:https://doi.org/10.1016/S0169-2070(00)00057-1. The M3- Competition.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and machine learning forecasting methods: Concerns and ways forward. *PLoS One*, *13(3)*, e0194889. URL: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0194889.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, *36*, 54–74. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301128. doi:https://doi.org/10.1016/j.ijforecast.2019.04.014. M4 Competition.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2021). Predicting/hypothesizing the findings of the M5 competition. *International Journal of Forecasting*, . URL: https://www.sciencedirect.com/science/article/pii/S0169207021001631. doi:https://doi.org/10.1016/j.ijforecast.2021.09.014.

Makridakis, S., Spiliotis, E., Assimakopoulos, V., Chen, Z., Gaba, A., Tsetlin, I., & Winkler, R. L. (2022). The m5 uncertainty competition: Results, findings and conclusions. *International Journal of Forecasting*, *38*, 1365–1385. URL: https://www.sciencedirect.com/science/article/pii/S0169207021001722. doi:https://doi.org/10.1016/j.ijforecast.2021.10.009. Special Issue: M5 competition.

Montero-Manso, P., Athanasopoulos, G., Hyndman, R. J., & Talagala, T. S. (2020). Fforma: Feature-based forecast model averaging. *International Journal of Forecasting*, *36*, 86–92. URL: https://www.sciencedirect.com/science/article/pii/S0169207019300895. doi:https://doi.org/10.1016/j.ijforecast.2019.02.011. M4 Competition.

Nie, Y., Nguyen, N. H., Sinthong, P., & Kalagnanam, J. (2023). A time series is worth 64 words: Long-term forecasting with transformers. arXiv:2211.14730.

Olivares, K. G., Challú, C., Garza, F., Canseco, M. M., & Dubrawski, A. (2022). NeuralForecast: User

friendly state-of-the-art neural forecasting models. PyCon Salt Lake City, Utah, US 2022. URL: https://github.com/Nixtla/neuralforecast.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. W., & Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. *Computer Research Repository*, *abs/1609.03499*. URL: http://arxiv.org/abs/1609.03499. arXiv:1609.03499.

Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2020). N-BEATS: neural basis expansion analysis for interpretable time series forecasting. In *8th International Conference on Learning Representations, ICLR 2020*. URL: https://openreview.net/forum?id=r1ecqn4YwB.

Oreshkin, B. N., Carpov, D., Chapados, N., & Bengio, Y. (2021). Meta-learning framework with applications to zero-shot time-series forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*, 9242–9250. URL: https://ojs.aaai.org/index.php/AAAI/article/view/17115. doi:10.1609/aaai.v35i10.17115.

Paszke et al. (2019). Pytorch: An imperative style, high-performance Deep Learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 32* (pp. 8024–8035). Curran Associates, Inc. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Robbins, H., & Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, (pp. 400–407).

Rosenblatt, F. (1961). *Principles of neurodynamics. perceptrons and the theory of brain mechanisms*. Technical Report Cornell Aeronautical Lab Inc Buffalo NY.

Sak, H., Senior, A. W., & Beaufays, F. (2014). Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *Computing Research Repository*, *abs/1402.1128*. URL: http://arxiv.org/abs/1402.1128. arXiv:1402.1128.

Salinas, D., Flunkert, V., Gasthaus, J., & Januschowski, T. (2020). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, *36*, 1181–1191. URL: https://www.sciencedirect.com/science/article/pii/S0169207019301888. doi:https://doi.org/10.1016/j.ijforecast.2019.07.001.

Semenoglou, A.-A., Spiliotis, E., Makridakis, S., & Assimakopoulos, V. (2021). Investigating the accuracy of cross-learning time series forecasting methods. *International Journal of Forecasting*, *37*, 1072–1084. URL: https://www.sciencedirect.com/science/article/pii/S0169207020301850. doi:https://doi.org/10.1016/j.ijforecast.2020.11.009.

Shen, J., Qu, Y., Zhang, W., & Yu, Y. (2018). Wasserstein distance guided representation learning for domain adaptation. arXiv:1707.01217.

Smyl, S. (2019). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, *36*, 75–85. URL: https://www.sciencedirect.com/science/article/abs/pii/S0169207019301153. doi:10.1016/j.ijforecast.2019.03.017.

Spiliotis, E., Assimakopoulos, V., & Makridakis, S. (2020). Generalizing the Theta method for automatic forecasting. *European Journal of Operational Research*, *284*, 550–558. URL: https://www.sciencedirect.com/science/article/pii/S0377221720300242. doi:https://doi.org/10.1016/j.ejor.2020.01.007.

US-CDC (2017). Influenza-like illness patients records. URL: https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

Vapnik, V. (1999). *The Nature of Statistical Learning Theory*. Springer Science & Business Media.

Wang, J., Chen, Y., Hao, S., Feng, W., & Shen, Z. (2018). Balanced distribution adaptation for transfer learning. *CoRR*, *abs/1807.00516*. URL: http://arxiv.org/abs/1807.00516. arXiv:1807.00516.

Weiss, K., Khoshgoftaar, T. M., & Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, *3*, 9. doi:10.1186/s40537-016-0043-6.

Wen, R., Torkkola, K., Narayanaswamy, B., & Madeka, D. (2017). A Multi-horizon Quantile Recurrent Forecaster. In *31st Conference on Neural Information Processing Systems NIPS 2017, Time Series Workshop*. URL: https://arxiv.org/abs/1711.11053. arXiv:1711.11053.

Wen, T., & Keyes, R. (2019). Time series anomaly detection using convolutional neural networks and

transfer learning. arXiv:1905.13628.

Yao, Y., & Doretto, G. (2010). Boosting for transfer learning with multiple sources. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 1855–1862). doi:10.1109/CVPR.2010.5539857.

Yao, Y., Rosasco, L., & Andrea, C. (2007). On early stopping in gradient descent learning. *Constructive Approximation*, *26(2)*, 289–315. URL: https://doi.org/10.1007/s00365-006-0663-2.

Yosinski, J., eff Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? arXiv:1411.1792.

Zhou, S., Zhou, L., Mao, M., Tai, H., & Wan, Y. (2019). An optimized heterogeneous structure LSTM network for electricity price forecasting. *IEEE Access*, *7*, 108161–108173. doi:10.1109/ACCESS.2019.2932999.

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2021). A comprehensive survey on transfer learning. *Proceedings of the IEEE*, *109*, 43–76. URL: https://arxiv.org/abs/1911.02685. doi:10.1109/JPROC.2020.3004555.