DEEPCAST: UNIVERSAL TIME-SERIES FORECASTER

Anonymous authors
Paper under double-blind review

ABSTRACT

Reliable and accurate time-series forecasting is critical in many fields including energy, finance, and manufacturing. Many time-series tasks, however, suffer from a limited amount of training data (i.e., the cold start problem) resulting in poor forecasting performance. Recently, convolutional neural networks (CNNs) have shown outstanding image classification performance even on tasks with small-scale training sets. The performance can be attributed to transfer learning through CNNs' ability to learn rich mid-level image representations. However, no prior work exists on general transfer learning for time-series forecasting. In this paper, motivated by recent success of transfer learning in CNN model and image-related tasks, we for the first time show how time-series representations learned with Long Short Term Memory (LSTM) on large-scale datasets can be efficiently transferred to other time-series forecasting tasks with limited amount of training data. We also validate that despite differences in time-series statistics and tasks in the datasets, the transferred representation leads to significantly improved forecasting results outperforming majority of the best time-series methods on the public M3 and other datasets. Our online universal forecasting tool, *DeepCast*, will leverage transfer learning to provide accurate forecasts for a diverse set of time series where classical methods were computationally infeasible or inapplicable due to short training history.

1 INTRODUCTION

Accurate time-series forecasting is critical for load forecasting, financial market analysis, anomaly detection, optimal resource allocation, budget planning, and other related tasks. While time-series forecasting has been investigated for a long time, the problem is still challenging, especially in applications with limited history (e.g., holidays, sporting events) where practitioners are forced to use adhoc machine learning approaches achieving poor forecasting performance Wu & Olson (2015).

Recently, time-series modeling based on a particular recurrent neural network, the Long Short Term Memory (LSTM) model (Hochreiter & Schmidhuber, 1997), has gained popularity due to its end-to-end modeling, ease of incorporating exogenous variables, and automatic feature extraction abilities (Assaad et al., 2008). Inspired by the success of deep convolutional neural network (CNN), Hermans & Schrauwen (2013) use stacked LSTM cells with different weight matrices in different layers for text prediction; Graves et al. (2013) use deep LSTM cells for speech recognition; Donahue et al. (2015) use deep LSTM and CNN for video recognition; Laptev et al. (2017) show that an LSTM forecasting model is able to outperform classical time series methods in cases with long, interdependent time series. The superior performance of deep LSTM structure on different tasks empirically prove its capability of modeling complex nonlinear feature interactions.

However, similar to training a deep CNN model, training a deep LSTM network needs updating the weight matrices for each LSTM cell, which requires a large amount of data across numerous dimensions. The data requirement hinders the application of deep LSTM model in time series forecasting. For example, recent results on time-series forecasting using LSTM only apply a single layer of LSTM (Bianchi et al., 2017).

Transfer learning (Pan & Yang, 2010) can address this problem. In transfer learning, we first train a base network on a base dataset and task, and then we repurpose the learned features, or transfer them, to a second target network to be trained on a target dataset and task. This process will tend to work if the features are general, meaning suitable to both base and target tasks, instead of specific to

the base task. When the target dataset is significantly smaller than the base dataset, transfer learning can be a powerful tool to enable training a large target network without overfitting.

Bengio (2012) shows preliminary results of using transfer learning on images. While the deep CNN models also suffer from requiring massive training data, Yosinski et al. (2014); Huang et al. (2013); Karpathy et al. (2014); Oquab et al. (2014) explore transfer learning on images, language and video. In image-based transfer learning (Bengio, 2012), deep neural networks exhibit a curious phenomenon: when trained on images, they all tend to learn first-layer features that resemble either Gabor filters or color blobs (Bengio, 2012). The appearance of these filters is so common that obtaining anything else on a natural image dataset causes suspicion of poorly chosen hyperparameters or a software bug. This phenomenon occurs not only for different datasets, but even with very different training objectives, including supervised image classification (Krizhevsky et al., 2012), and unsupervised learning of sparse representations (Le et al., 2011). Yosinski et al. (2014) quantitatively explore transferability of different layers for the image classification task. The authors discover that because finding the Gabor filters on the first layer seems to occur regardless of the exact cost function and natural image dataset, these first-layer features are called *general*. On the other hand, the features computed by the last layer of a trained network must depend greatly on the chosen dataset and task. For example, in a network with an *N*-dimensional softmax output layer that has been successfully trained toward a supervised classification objective, each output unit will be specific to a particular class. Yosinski et al. (2014) thus calls the last-layer features *specific*.

Motivated by these findings, we investigate if transfer learning applies to time series. Until now, the success of model generalization in deep CNN models for image-related tasks still does not happen in deep LSTM models for time series-related tasks. Though Yang et al. (2017) proposed to use transfer learning for sequence tagging problems, the sequence tagging problem can be explicitly decomposed into different sub-problems and the transfer learning is also explicitly divided. Transfer learning has also been recently used in language translation where an auto-encoder architecture is typically used Mou et al. (2016). Previous work has shown that an auto-encoder is useful in time-series for feature extraction Laptev et al. (2017), but not in time-series forecasting.

In this paper, we explore if there are equivalent *general* and *specific* features for time-series forecasting using deep LSTM model. We are interested in this, to the extent that features within a deep LSTM network are general, we will be able to use them for transfer learning to do more accurate forecasting on short time-series.

To the best of our knowledge, our work makes the first attempt to present the evidence of transfer learning for time-series in neural nets and to quantify its applicability to real-world applications. Our contributions are four-fold:

1. We firstly demonstrate transfer learning for time-series forecasting.
2. We demonstrate use-cases and impacts of time-series transfer learning, including:
   (a) forecasting with limited history,
   (b) computational resource saving,
   (c) cross-domain learning capability.
3. We show the geometric interpretation of learned features in a deep LSTM model, inferring the theoretical support of the usage of transfer learning.
4. We publish an online tool that democratizes time-series forecasting through a public transfer learning model motivated by ImageNet (Krizhevsky et al., 2012).

2 TRANSFER LEARNING IN TIME-SERIES

Transfer learning involves the concepts of a task and of a domain. A domain D consists of a marginal probability distribution $P(X)$ over the feature space $\mathcal{X} = \{x_1, ..., x_n\}$. Thus, given a domain $\mathcal{D} = \{\mathcal{X}, P(\mathcal{X})\}$, a task T is composed of a label space $\mathcal{Y}$ and a conditional probability distribution $P(Y|X)$ that is usually learned from training examples consisting of pairs $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. Given a source domain $\mathcal{D}_S$ and a source task $\mathcal{T}_S$ as well as a target domain $\mathcal{D}_T$ and a target task $\mathcal{T}_T$, transfer learning aims to learn the target conditional probability distribution $P(Y_T|X_T)$ in $\mathcal{D}_T$ from the information learned from $\mathcal{D}_S$ and $\mathcal{T}_T$. In this paper we apply transfer learning to a time-series

domain and apply it to cases where $\mathcal{X}_S \neq \mathcal{X}_T$ and $P(Y_S|X_S) \neq P(Y_T|X_T)$ (e.g., target domains with limited training data and different time-series classes).

Time-series data can be decomposed into the summation of a (usually monotonic) trend component, a (periodic) seasonal component, a holiday component, a stationary (stochastic) component, and a (usually i.i.d.) error component. Most of the time-series forecasting models assume explicit models with hyper-parameters of the trend, seasonality, and the stochastic process:

$$y(t) = g(t; \alpha_i^t) + s(t; \alpha_i^s) + h(t; \alpha_i^h) + c(t; \alpha_i^c) + \epsilon(t; \alpha_i^\epsilon),$$

where $g(\cdot)$, $s(\cdot)$, $h(\cdot)$, $c(\cdot)$, $\epsilon(\cdot)$ represent the trend, seasonal, holiday, stationary, and error component, respectively. Then, these models usually use maximum likelihood estimation to determine the hyper-parameters α. However, the explicit model formulation of each component is still far from empirical understanding. For example, SJ & B (2017) propose to use only a piecewise-linear function and a logistic growth model for trending modeling, and a Fourier series with trigonometric function as basis for seasonality modeling. Moreover, all hyper-parameters need to be optimized for specific data sets.

Instead of defining the explicit model components, the LSTM model only consists of 5 different nonlinear components. Initially, the LSTM cell is designed to repeat infinitely, with a single set of hyperparameters, including four rectangular weight matrices W_f, W_h, W_u, W_o, acting on input vector x, serving for computing forget gate, candidate state, update gate, and output gate, respectively. Four square weight matrices R_f, R_h, R_u, R_o, acting on lagged output vector y_{t-1}, serving for the same computation procedures. By stacking LSTM cells to construct a deep LSTM model, we gain additional freedom by enabling different weight matrices in different LSTM layers. We hypothesize that the feature layers of LSTM model is a generalization of the trend, seasonality, and holiday representation, in analogy to the explicit models in traditional time-series modeling tasks. While the model representation itself rather a general feature for all time-series data, we anticipate that the representation learned from one dataset can be used for another dataset. Then, the lower levels of the LSTM model serves the role in analogy to the hyper-parameter optimization which varies for different datasets. The illustration of the generalization is shown in Fig. 1

After model training, a set of feature layers is typically *frozen* in order to avoid changing the learned weights. In this paper, we explicitly decompose the model into two types of layers: feature layers and predictive layers. After training, we typically *freeze* the feature layer weights to reuse them.

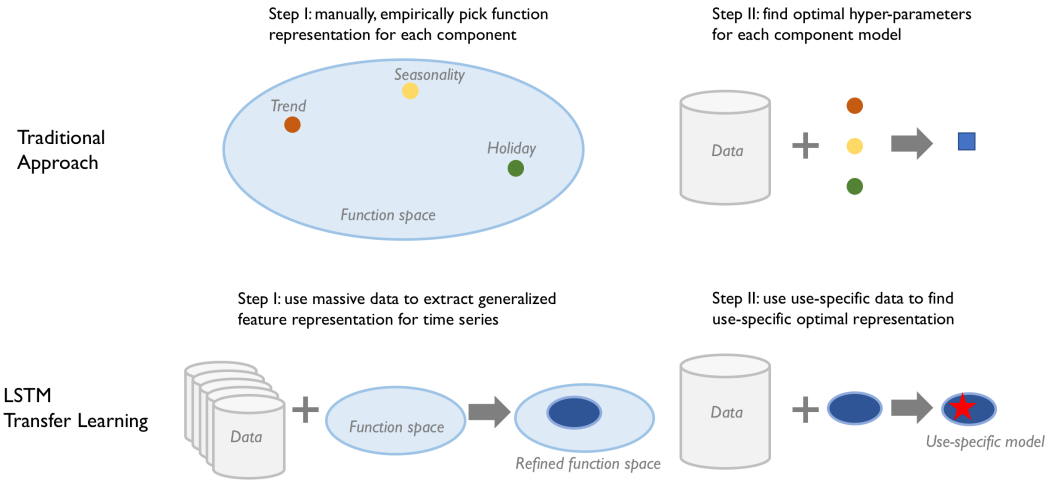


Figure 1: Transfer learning visualization

Visualizing the learned knowledge of a neural network is critical for interpretability and tuning of the network Zeiler & Fergus (2013). While in the past deep neural nets were often considered a black box, recent convolutional layers were successfully visualized Zeiler & Fergus (2013). Visualizing time-series data is different because we have numerical values instead of images. In section 4.3 we provide an attempt to interpret some of the learned features.

3 EXPERIMENTAL SETUP

Dataset: The dataset used for demonstrating transfer learning contains 116,000 anonymous residential scale electricity loads at hourly granularity from Pacific Gas and Electric Company (PG&E dataset). The data is from 08/01/2011 to 7/31/2012. The 116,000 time series are from 13 different climate zones and are with large diversity (Kwac et al., 2014). Besides the anonymous PG&E dataset, we also use a public available M3 dataset for validation. All time series are logarithm transformed and min-max scaled to $[0, 1]$ interval.

Deep LSTM Model: The deep LSTM model consists of 6 LSTM cells. Each LSTM cell has a 128-dimensional state vector. For a given timestamp t (hour), the input vector $\boldsymbol{x}$ consists of 60-hour historical electricity load: $\boldsymbol{x} = [l_t, \cdots, l_{t-59}]$, and the output vector $\boldsymbol{y}$ consists of 60-hour electricity load from time t: $\boldsymbol{y} = [l_{t+1}, \cdots, l_{t+60}]$. We train the network using 60 hours for the forecast horizon and 60 hours for the lookback.

Data Separation: The PG&E dataset is separated into four parts for experiments. In particular, we randomly split the dataset to two subgroups, A and B, with same size. Then, we divide each subgroup into first six-month data and second six-month data. The four sub-groups of data is labelled with A1, A2, B1, and B2, shown in Table 1.

	Group A (58,000 customers)	Group B (58,000 customers)
08/01/2011 - 01/31/2012	A1	B1
02/01/2012 - 07/31/2012	A2	B2

Table 1: Data separation

In the following experiments, the whole subset A1 is used to train the base deep LSTM model, called `Base`. Then, for any time series b^i in subset B, we use $b_1^i \in$ B1 for training, and $b_2^i \in$ B2 for testing. The transfer learning is implemented as: given n as the transferred layer from `Base`, we *freeze* the first n layers of the `Base` model, and use b_1^i to train the other $6 - n$ layers. If $n = 6$, then we only use the *specific* data b_1^i to train the last fully connected layer. We call the transfer learning model as `AnB` for a given n. We can also initialize the deep LSTM model with randomly chosen hyperparameters, and only use b_1^i to train the deep LSTM model. We call the model `Single`. All the three models are tested using $b_2^i \in$ B2. The training data and test data setup is shown in Table 2. The symmetric mean absolute percentage error (SMAPE) is used for performance evaluation.

	`Base`	`AnB`	`Single`
Training data	A1	A1 + b_1^i	b_1^i
# *frozen* layers	-	n	-
Test data	b_2^i	b_2^i	b_2^i

Table 2: Training and test data for i-th customer for different deep LSTM models

4 RESULTS AND DISCUSSION

In this section, we systematically analyze the feature transferability and their stability among diverse time series. We also provide a geometric understanding to explain the observed transfer learning for deep LSTM model. Then, we show that by using transfer learning, we can use deep LSTM model for accurate time-series forecasting with limited history at a very small computational cost.

4.1 TIME-SERIES FEATURE TRANSFERABILITY

First, we compare the performance of the `AnB` model and the `Single` model on all customers from subgroup B, to show the improvements using transfer learning. We also use the classical forecasting method HoltWinters as a baseline. In this experiment, we fix the number of freeze layer at 3. The result is shown in 2a. When the training size is very small, transfer learning provides substantial performance improvement of the `A3B` model over the `Single` model. In particular, the SMAPE

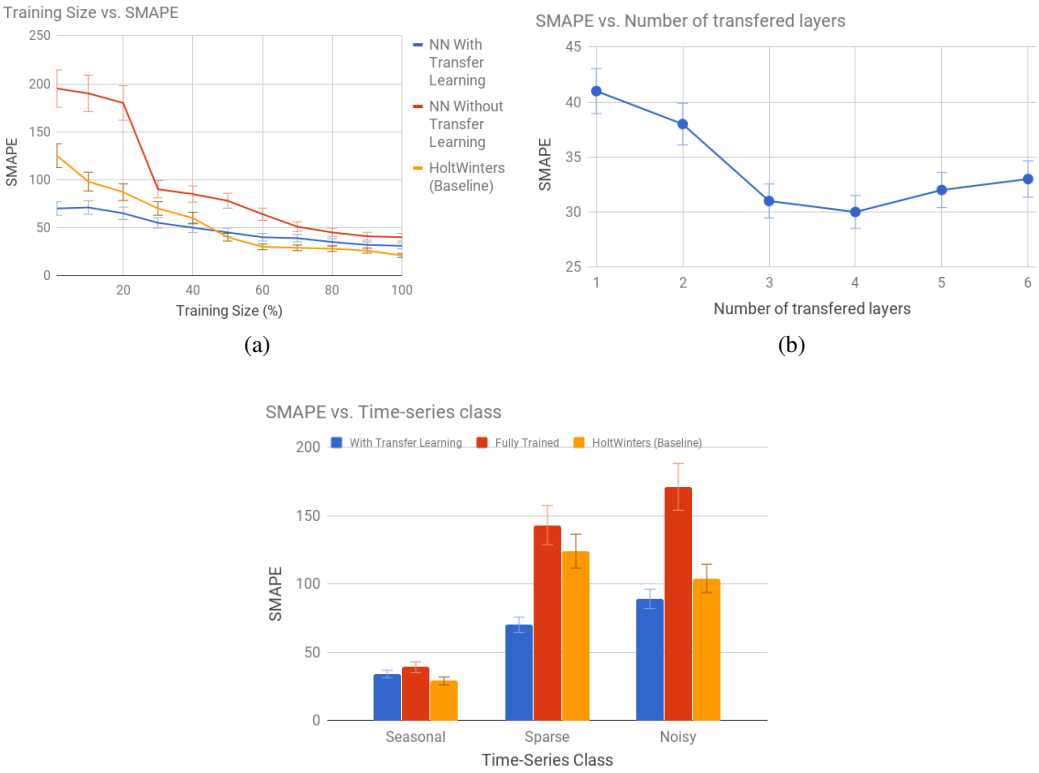


Figure 2: **(a)** Performance comparison of deep LSTM models between training with single time series and training with transfer learning. The x-axis is the training size; the y-axis is SMAPE. The red curve represents single time series-trained model; the blue curve represents model using transfer learning. The performance gap is huge for short training sizes. When training size increases, the performance difference shrinks. Each round dot represents the mean SMAPE of all customers from B2, and the error bar illustrates the standard deviation of SMAPE over 58,000 customers. A decreasing-saturating-increasing trend is observed. **(b)** Transfer learning performance for models with different transferred layers. The x-axis is the number of transferred layers from the base model, the y-axis is SMAPE. The meaning of round dots and error bars is the same as the meaning in (a). **(c)** Transfer learning performance for different types of time series. The y-axis is SMAPE. Transfer learning brings substantial improvements for sparse and noisy time series. The meaning of the bars and error bars is the same as the meaning of round dots and error bars in (a).

drops from around 200% to less than 75%. While increasing the training size helps both two model get smaller SMAPE, the performance difference between the two models also shrinks. When the training size is large enough, the performance of the two model converges, with a small learning gap. We refer to the final gap between the two networks when full data is available as the "transfer learning gap" or *TLG*. In our future work we will look at *TLG* in terms of data complexity.

Furthermore, by fixing the training size, we can also control the number of frozen layers, n, in the transfer learning model `AnB`, and investigate the performance improvement of different n's. The result is shown in Figure 2b. As we increase the depth of transfered layers, the performance gain diminishes. We also reveal that there is a diminishing rate of return on forecasting performance as a function of the number of trainable layers.

In our experiments we used simple NULL count and *VAR* computation approaches to do threshold classification of time-series into sparse and noisy classes. In Figure 2c different time-series class performance is shown. Three classes of time-series are used as input: (i) seasonal, (ii) noisy and (iii) sparse. A model that uses transfer learning always outperforms an equivalent model with lim-

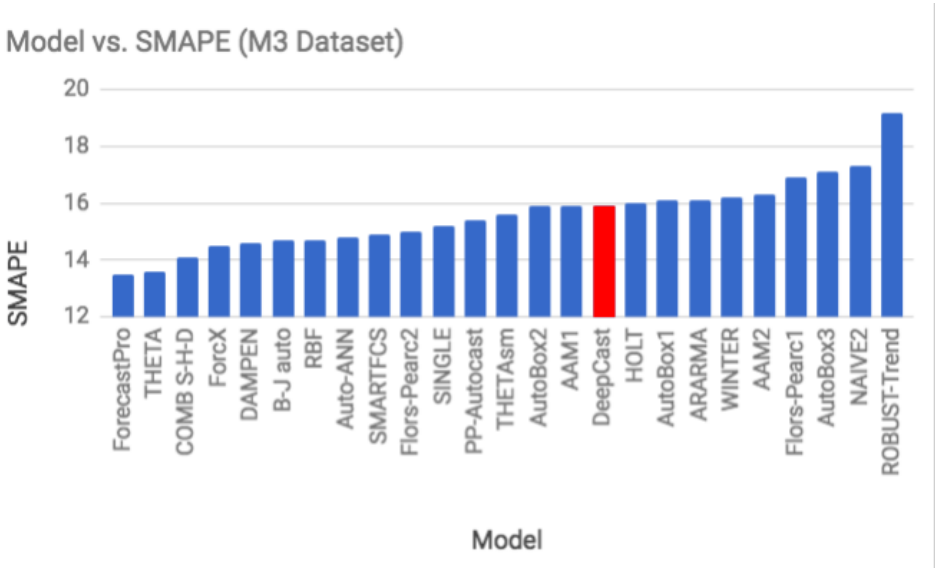


Figure 3: The comparison of forecasting performance on M3 public dataset between the base deep LSTM model trained with only A1 (6-month time-series data of 58,000 PG&E customers) and other main-stream forecasters trained with time series of the same id with test time series in M3 dataset. The transfer learned model has competitive performance (15.8% SMAPE) relative to other specialized models.

ited history, however, its performance is highest for noisy data. This shows that the nonlinearity presented in the noisy data are modeled better by the transfer learned model.

Figure 3 demonstrates the performance of the presented transfer learned model (*DeepCast* in the Figure) against the best forecasting models on the publicly available M3 dataset. Note that we trained a single model using the PG&E dataset instead of M3 leveraging transfer learning while other approaches trained a single specialized model per time-series in M3 (3K total). Transfer learning shows competitive results while using only a fraction of the training and inference cost (see Section 4.3).

4.2 Geometric Interpretation

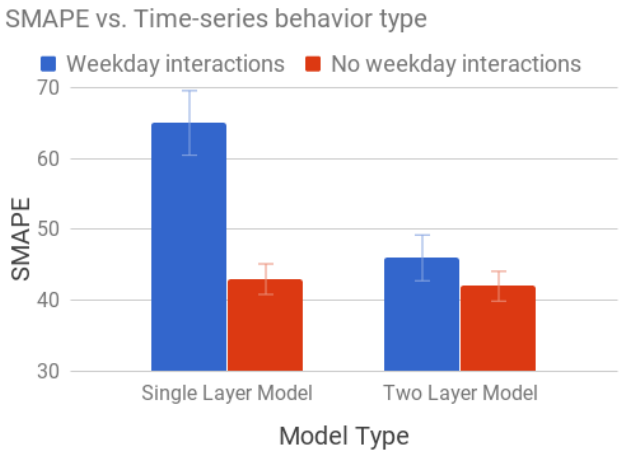
Figure 4a shows the model performance with and without daily interactions. Without daily interactions we found different architectures behave similarly. When daily interactions exist, however, we found that a deeper network works best.

This result is consistent with our geometric interpretation of the layer embeddings shown in Figure 4b where one can observe clusters caused by projecting the layer activations onto a *2-D* plane. One interpretation of these specific results is that the first layer learns individual weekday features while the second layer learns the interaction between the days. Visualizing the learned features as a geometric shape, similar to the ConvNet visualization for images Zeiler & Fergus (2013), is part of our future work.

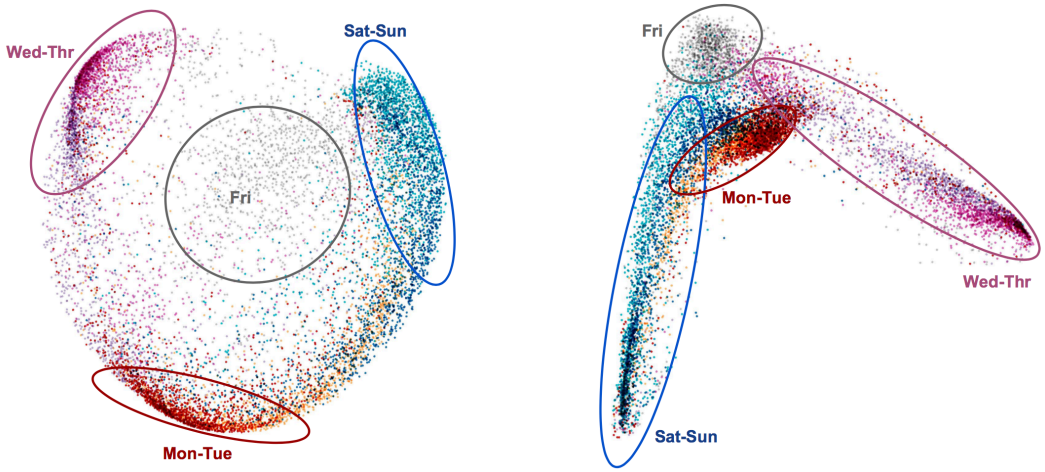
4.3 Computation resources

A major motivation for this work was to cut the training and inference cost of time-series models in production. The current state of the art is to train a single model per time-series. This is computationally unsustainable as the number of time-series increases. The transfer learning approach presented in this paper is able to alleviate the computational burden while providing competitive results.

With transfer learning, it is possible to train a single model that does inference on N time-series where N can be in the thousands (or hundreds of thousands). This results in many orders of magni-



(a)



(b)

Figure 4: **(a)** Model performance with and without weekday interactions. **(b)** Training set of time series, visualized in the embedding space. Each point represents a 28-day segment, colored by the day of the week of the last day. We evaluate the cell states of the two LSTM layers, where the first layer with dimension 128 is plotted on the left, and second layer with dimension 32 is plotted on the right. PCA is used to project into 2D space for visualization.

tude reduction if resources needed for training, inference and storage with a small performance hit (see Figure 5).

5 Democratizing Forecasting

Time-series forecasting has been primarily accessible to experts Dannecker et al. (2013). With a growing importance of the time-series forecasting field, however, ability for non-experts to generate reasonable forecasts becomes increasingly important. Previous work Hyndman et al. (2007) aims to provide automated time-series model selection, however, these techniques do not scale for millions of time-series due to per-timeseries model retraining.

Our goal is to use transfer learning to democratize the time-series forecasting field making non-experts get decent results competitive with the experts in the field. Figure 6 provides an overview of our online forecasting tool that accepts a time-series data and using the novel time-series transfer

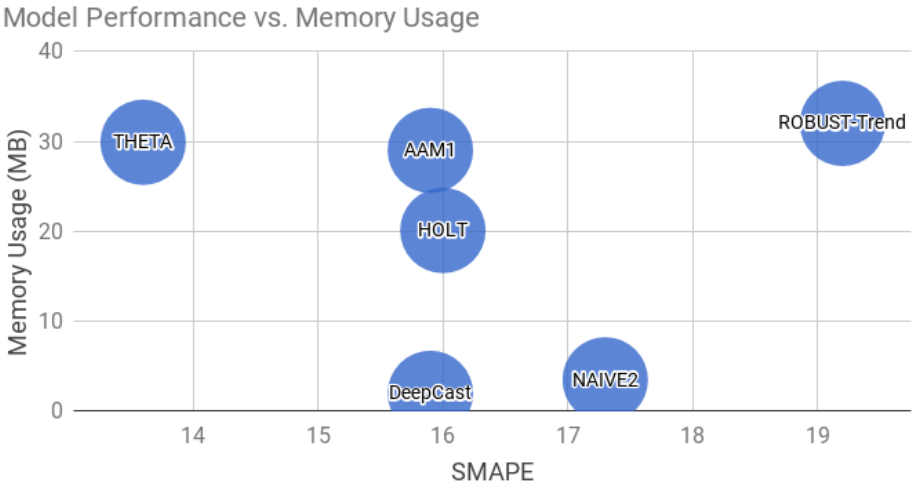


Figure 5: The presented Transfer Learned Model excels in compute power savings relative to the performance hit.

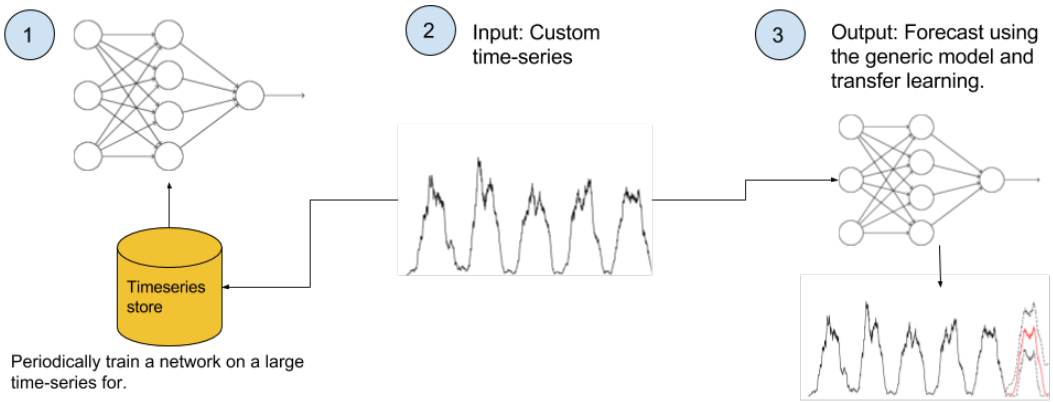


Figure 6: Our online forecasting tool will use the transfer learning technique for time-series to democrotize the time-series forecasting process provide high quality forecasts for everyone with dramatically low computational cost.

learning strategy presented, provides a forecast. This tool will allow ImageNet-like data collection Krizhevsky et al. (2012) for an important and niche market that time-series forecasting is becoming.

6 Conclusion

It is well known that transfer learning exists for images. In this paper, we demonstrated transfer learning for time series. We have shown a dramatic forecasting accuracy improvement with transfer learning under small to medium training data size conditions. Furthermore, we have identified compute cost improvements when using the transfer learning approach. An online universal forecasting tool, *DeepCast*, is proposed to provide a benchmark of transfer learning of LSTM for time-series forecasting.

For our future work, we will compare transfer learning across different architectures in terms of stability and applicability for unseen target classes focusing more on theoretical guarantees of transfer learning.

REFERENCES

Mohammad Assaad, Romuald Boné, and Hubert Cardot. A new boosting algorithm for improved time-series forecasting with recurrent neural networks. *Information Fusion*, 9(1):41–55, 2008.

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. *ICML Workshop on Unsupervised and Transfer Learning*, pp. 17–36, 2012.

Filippo Maria Bianchi, Enrico Maiorino, Michael C Kampffmeyer, Antonello Rizzi, and Robert Jenssen. An overview and comparative analysis of recurrent neural networks for short term load forecasting. *arXiv preprint arXiv:1705.04378*, 2017.

Lars Dannecker, Robert Lorenz, Philipp Rösch, Wolfgang Lehner, and Gregor Hackenbroich. Efficient forecasting for hierarchical time series. *ACM International Conference on Information & Knowledge Management*, pp. 2399–2404, 2013.

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2625–2634, 2015.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649, 2013.

Michiel Hermans and Benjamin Schrauwen. Training and analysing deep recurrent neural networks. *Advances in Neural Information Processing Systems*, pp. 190–198, 2013.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computing*, 9(1): 41–55, 1997.

Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong. Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7304–7308, 2013.

Rob J Hyndman, Yeasmin Khandakar, et al. Automatic time series for forecasting: the forecast package for R. (6/07), 2007.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. *IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

Jungsuk Kwac, June Flora, and Ram Rajagopal. Household energy consumption segmentation using hourly data. *IEEE Transactions on Smart Grid*, 5(1):420–430, 2014.

Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. Time-series extreme event forecasting with neural networks at Uber. *International Conference on Machine Learning*, 2017.

Quoc V Le, Alexandre Karpenko, Jiquan Ngiam, and Andrew Y Ng. ICA with reconstruction cost for efficient overcomplete feature learning. *Advances in Neural Information Processing Systems*, pp. 1017–1025, 2011.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. How transferable are neural networks in NLP applications. *arXiv preprint arXiv:1603.06111*, 2016.

Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1717–1724, 2014.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

Taylor SJ and Letham B. Forecasting at scale. *PeerJ Preprints 5:e3190v2*, 2017.

Desheng Dash Wu and David L Olson. Financial risk forecast using machine learning and sentiment analysis, 2015.

Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*, 2017.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in Neural Information Processing Systems*, pp. 3320–3328, 2014.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks, 2013. URL `http://arxiv.org/abs/1311.2901`.