DyAt Nets: Dynamic Attention Networks for State Forecasting in Cyber-Physical Systems

Nikhil Muralidhar[1] , **Sathappah Muthiah**[1] and **Naren Ramakrishnan**[1]
[1]Computer Science Department, Virginia Tech, USA
{nik90, sathap1}@vt.edu, naren@cs.vt.edu

Abstract

Multivariate time series forecasting is an important task in state forecasting for cyber-physical systems (CPS). State forecasting in CPS is imperative for optimal planning of system energy utility and understanding normal operational characteristics of the system thus enabling anomaly detection. Forecasting models can also be used to identify sub-optimal or worn out components and are thereby useful for overall system monitoring. Most existing work only performs single step forecasting but in CPS it is imperative to forecast the next sequence of system states (i.e curve forecasting). In this paper, we propose DyAt (Dynamic Attention) networks, a novel deep learning sequence to sequence (*Seq2Seq*) model with a novel hierarchical attention mechanism for long-term time series state forecasting. We evaluate our method on several CPS state forecasting and electric load forecasting tasks and find that our proposed DyAt models yield a performance improvement of at least **13.69%** for the CPS state forecasting task and a performance improvement of at least **18.83%** for the electric load forecasting task over other state-of-the-art forecasting baselines. We perform rigorous experimentation with several variants of the DyAt model and demonstrate that the DyAt models indeed learn better representations over the entire course of the long term forecast as compared to their counterparts with or without traditional attention mechanisms. All data and source code has been made available online[1]

1 Introduction

Cyber physical systems like the electric grid and industrial and chemical plants often have a complex set of interdependent process that are operating simultaneously. In order to ensure uninterrupted and smooth function of the CPS system, it is important for system maintainers to quantify the system state well into the future so as to gain enough lead time to plan for contingencies. For example, electric utilities perform weekly forecasts to estimate the expected power demand over the coming week. Such a week long forecast affords the utility enough time to increase (or decrease) the power production to meet the expected future power demand, ensuring optimal grid operation [Carvallo *et al.*, 2017].

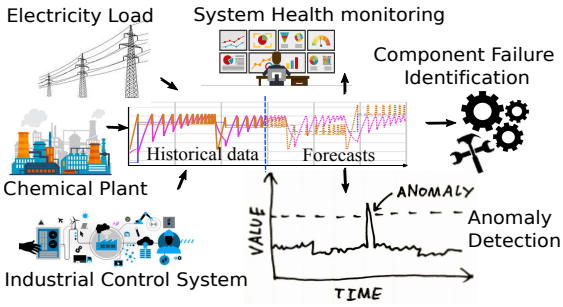


Figure 1: Time series forecasting systems are at the center of critical applications like system health monitoring, anomaly detection and component wear identification, in the context of various CPS systems like the electric grid, chemical and industrial plants.

CPS state forecasting for chemical and industrial plants allows system maintainers to estimate the operational load the plant will undergo over the course of the forecast and hence serve as an important tool in system efficiency and health monitoring. Critical infrastructure CPS also run the risk of cyber attacks from malicious entities whose goal is to disrupt optimal system operation. CPS state forecasting forms the core of early warning systems which alert system maintainers if the system has veered from its expected state of normal operation [Filonov *et al.*, 2016; Malhotra *et al.*, 2016]. Traditional forecasting models predict system state at the next time step given system measurements at the current time step. However, as mentioned previously, CPS state forecasting contexts require that the forecasting models yield predictions multiple steps into the future; CPS traditionally have components which share highly non-linear relationships evolving continuously and the underlying process might not be stationary or easy to define mechanistically. Hence, traditional auto-regressive models or dynamical systems like

[1]https://github.com/nmuralid1/DynamicAttentionNetworks

Kalman filters might be ineffective in capturing the data representation. However, deep learning models have proven effective in learning highly non-linear function spaces and require no stationarity assumptions to be satisfied or mechanistic representations of system operation to be provided. A popular deep learning architecture for sequence prediction (employed extensively for neural machine translation and abstractive text summarization) is the *Sequence to Sequence* (*Seq2Seq*) architecture a.k.a encoder-decoder architecture.

Seq2Seq models can also be adopted for time series sequence forecasting applications. Most existing work on deep learning for time series [Filonov *et al.*, 2016; Malhotra *et al.*, 2016] makes use of a traditional *Seq2Seq* architecture for forecasting future values of the time series without any modifications. Such traditional *Seq2Seq* architectures usually also employ recurrent units for each encoder and decoder step and an *attention mechanism* wherein the hidden state for each decoder unit is constructed as some combination of a set of hidden states from preceding units. Thus far, *Seq2Seq* models used for time series forecasting have employed standard attention mechanisms where each decoder unit only considers the hidden states from the encoder phase. Such an attention mechanism is effective for neural machine translation or abstractive text summarization because, in these applications, all the information required for sequence generation, is encapsulated in the encoding phase. However, such an encoder-focused attention mechanism is not optimal for time series forecasting where each decoder unit models one time step of a sequence in an ever-evolving temporal process, all of whose information is not captured by the encoder phase. Time series models are known to benefit significantly from knowledge of the recent past. Thus, traditional encoder-focused attention mechanisms in *Seq2Seq* forecasting models lose valuable information as states from previous decoders are not considered by an attention mechanism that only considers encoder hidden states. In this paper, we introduce two variants of novel *hierarchical dynamic attention models* for multivariate time series forecasting which addresses this issue. Our contributions are as follows:

- We introduce a novel dynamic attention (DyAt) mechanism for effective multivariate time series forecasting with *Seq2Seq* models.
- We develop two novel hierarchical dynamic attention models, *Hierarchical Dynamic Attention Networks* (DyAt-H) and *Hierarchical Dynamic Attention Networks with Max Pooling* (DyAt-MaxPool-H).
- We also characterize the performance of our hierarchical DyAt models relative to other state of the art time series forecasting models and discuss the properties responsible for our models yielding superior performance.

In section 2, we formally introduce the multivariate time series forecasting task in the context of *Seq2Seq* models and develop our DyAt models. The experimental setup is described in section 3, followed by discussion of experimental results in section 4. We then present a brief survey of related literature in section 5 followed by concluding remarks in section 6.

2 Problem Formulation

Let us consider a set $\mathcal{D} = \{X_1, X_2, .., X_m\}$ where each $X_i \in \mathbb{R}^{l \times k}$ represents a multivariate sequence of length l with k time series at each step of the sequence. Sequence forecasting applications in time series employ encoder-decoder (*Seq2Seq*) models wherein the encoder is supplied with a sequence X_i and the learning task is for the decoder to forecast the next k-variate sequence X_{i+1} of length l.

2.1 Seq2Seq Attention Models

Formally, in traditional *Seq2Seq* architectures with attention, the prediction at each decoder step is calculated as defined by [Luong *et al.*, 2015].

Let $E = \{h_1^e, h_2^e, .., h_n^e\}$ represent hidden states of the n encoder units and $D = \{h_1^d, h_2^d, .., h_q^d\}$ represent hidden states of the q decoder units of a *Seq2Seq* model, where each $h_i^* \in \mathbb{R}^{h \times 1}$ comes from a recurrent unit like LSTM, GRU, basic RNN etc. The prediction at each decoder is calculated as defined in Eq. 1.

$$\begin{aligned} h_i^d &= f(h_{i-1}^d, \hat{y}_{i-1}) \\ \alpha_{i*} &= \textit{score}(h_i^d, E) \\ \overline{\alpha}_{i*} &= \textit{Softmax}(\alpha_{i*}) \\ c_i &= \sum_{j=1}^{l} \overline{\alpha}_{ij} h_j^e \\ \tilde{h}_i^d &= \textit{tanh}(W_h * [c_i; h_i^d]) \\ \hat{y}_i &= g(\tilde{h}_i^d) \end{aligned} \tag{1}$$

In Eq. 1, f may be a recurrent unit like an RNN, LSTM or a GRU. h_{i-1}^d represents the hidden state from the previous decoder unit and $\hat{y}_{i-1} \in \mathbb{R}^{k \times 1}$ the k-variate prediction of the previous decoder unit passed as input to the current decoder unit. $\overline{\alpha}_{i*} \in \mathbb{R}^{l \times 1}$ is the attention weight vector for decoder i over the encoder hidden states (i.e $l = |E|$) and $c_i \in \mathbb{R}^{h \times 1}$ is the attentional context vector calculated as a linear combination over all E, and corresponding attention weights $\overline{\alpha}_{i*}$.

Attention Calculation. The attention energy vector α_{i*} for the i^{th} decoder unit can be calculated in multiple ways. The attention energies can be calculated purely based on the current hidden vector or as a function of the alignment between the current hidden vector and encoder states. In Eq. 2, we define two such variants of the scoring mechanism for calculating attention energies.

$$\text{score}(h_i, E) = \begin{cases} h_i^T W_\alpha h_j^e \quad \forall\, h_j^e \in E & \textbf{(Bilinear)} \\ W_\alpha h_i & \textbf{(Location-Based)} \end{cases} \tag{2}$$

The attention energies are then normalized using a *softmax* transformation, to obtain the attention weights $\overline{\alpha}_{i*} \in \mathbb{R}^{l \times 1}$ used to obtain the attentional context vector c_i.

Next, the attentional hidden state $\tilde{h}_i^d$ of the i^{th} decoder is obtained by imposing a non-linear transfer function (hyperbolic tangent) over an affine transformation (with $W_h \in \mathbb{R}^{h \times 2h}$) of the concatenation of c_i and h_i^d. Finally, g is some function that calculates the i^{th} prediction $\hat{y}_i$ from the corresponding decoder attentional hidden state $\tilde{h}_i^d$. In our experiments we consider g to be a simple linear layer.

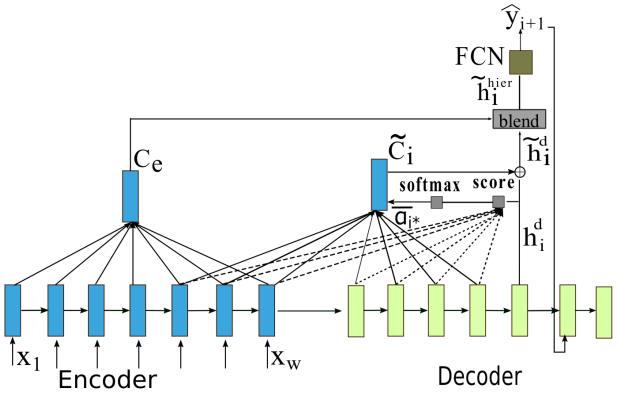


Figure 2: DyAt-H Model Representation.

2.2 Dynamic Attention Networks

In time series applications, we hypothesize that calculating attention just over the encoder hidden states is insufficient. This is because, unlike in the case of NMT where all the information required by the decoder sequence is contained in the encoder steps, the time series forecasting task models an evolving process and each step in the decoder phase also carries with it important information about function evolution. Hence, incorporating these signals into the attention mechanism is paramount to learning better time series forecasting models. To improve the learned representation of *Seq2Seq* architectures in the context of multivariate time series forecasting, we introduce the *Dynamic Attention network* (DyAt). A DyAt employs a *Seq2Seq* architecture with a novel dynamic attention mechanism.

Let us consider a set of hidden states $H_i = \{h_{(i-1):(i-l)}\}$ where each hidden state can either be an encoder or decoder hidden state from a previous time step, i.e., $(h_k \in E \vee h_k \in D)\ \forall\, h_k \in H_i$. The calculation of the context vector c_i and the corresponding attentional hidden state $\tilde{h}_i^d$ is modified as defined in Eq. 3.

$$\begin{aligned} \tilde{c}_i &= \sum_{j=1}^{l} \overline{\alpha}_{ij} h_j \\ \tilde{h}_i^d &= \tanh(W_h * [\tilde{c}_i; h_i^d]) \end{aligned} \tag{3}$$

Dynamic Attentional Context Vector. In Eq. 3, each candidate hidden state h_j in the linear combination to produce the *dynamic* attentional context vector $\tilde{c}_i$, no longer originates from the set of encoder hidden states E as is the case for the *static* attentional context vector c_i in Eq. 1. Rather, $\tilde{c}_i$ is derived from set H_i which consists of the past l hidden states irrespective of whether they are from the encoder or the decoder. Maintaining a dynamic candidate set of hidden states H_i for each decoder unit i, allows the model to more effectively attend to the model properties learned in the recent past as opposed to only being restricted to focus on the model properties learned during the encoder phase.

2.3 Hierarchical Dynamic Attention Networks

In the aforementioned DyAt model, as we move deeper into the decoder forecasting sequence, we can observe that the set H_i becomes dominated by a majority of hidden states from previous decoders and hence information from the encoder phase (which receives ground-truth input) considered in the decoder phase is continually reduced. Due to the lack of ground truth inputs in the decoder phase, the effect of erroneous predictions or bad hidden representations increases as we traverse deeper into the decoder sequence.

To alleviate this effect and encourage the DyAt models to learn a more holistic representation, we introduce *Hierarchical Dynamic Attention Networks* (DyAt-H) which also considers a concise representation of the encoder referred to as the encoder context vector c_e in addition to the *dynamic* attentional context vector $\tilde{c}_i$ at each decoder unit. A schematic representation of the DyAt-H model is presented in Fig. 2.

$$\begin{aligned} c_e &= \textit{contextGen}(E) \\ h_i^d &= f(h_{i-1}^d, \hat{y}_{i-1}) \\ \alpha_{i*} &= \textit{score}(h_i^d, H_i) \\ \overline{\alpha}_{i*} &= \textit{Softmax}(\alpha_{i*}) \\ \tilde{c}_i &= \sum_{j=1}^{l} \overline{\alpha}_{ij} h_j \quad \forall h_j \in H_i \\ \tilde{h}_i^d &= \tanh(W_h * [\tilde{c}_i; h_i^d]) \\ \tilde{h}_i^{hier} &= \textit{blend}(c_e, \tilde{h}_i^d) \\ \hat{y}_i &= g(\tilde{h}_i^{hier}) \end{aligned} \tag{4}$$

In DyAt-H models, at the outset of the decoder phase, we generate $c_e \in \mathbb{R}^{h \times 1}$ using the *contextGen(·)* function as defined in Eq. 4. Context vector c_e is calculated using all encoder hidden states E and is calculated only once per decoder phase. We experimented with two variants for generating c_e, detailed in Eq. 5.

$$\text{contextGen}(E) = \begin{cases} \text{MaxPool1D}(W_c^T E) & \textbf{(Learn)} \\ \text{mean}(E) & \textbf{(Mean)} \end{cases} \tag{5}$$

The attention energies (α_{i*}), attention weights ($\overline{\alpha}_{i*}$), *dynamic* attentional context vector ($\tilde{c}_i$) and *dynamic* attentional hidden state ($\tilde{h}_i^d$) for the i^{th} decoder unit are all calculated as detailed in section 2.1 and section 2.2. Finally, the *blend* function is used to produce the hierarchical attentional hidden state $\tilde{h}_i^{hier}$ by combining the encoder context vector c_e with the dynamic attentional hidden state $\tilde{h}_i^d$ which is then supplied to the *g(·)* function to obtain the k-variate prediction at time step i. The blend function can take multiple forms, it can be an affine transform over the concatenation of the two vectors or the result of a pooling operation over the two. For our experiments, we use mean-pooling.

It can be argued that attention mechanisms (static or dynamic) are redundant as standard recurrent units are designed to propagate model properties effectively to subsequent recurrent units. However, as we showcase in our results, in long-term forecasting tasks, hierarchical dynamic attention mechanisms do indeed provide forecasting models the added ability to focus on relevant parts across the entire sequence, both in the immediate and the distant past.

3 Experimental Setup

We demonstrate the forecasting performance of DyAt and DyAt-H on several state of the art datasets given below:
Electricity[2]: This dataset contains electricity consumption in kilo-watt hour (kWh) measured in 15 minute intervals from 2011-14 for n=370 clients. For our purposes, we randomly choose 20 clients to train and test our models.
Tennessee Eastman Challenge Process (TEP)[3], represents a CPS performing a chemical process. It is a popular benchmark dataset in process control studies and for system identification, forecasting and anomaly detection applications [Filonov *et al.*, 2017; Juricek *et al.*, 2001].
Gasoil Heating Loop (GHL)[4] released by [Filonov *et al.*, 2016], is another benchmark dataset for multivariate system state forecasting and anomaly detection in CPS. It represents a chemical process of a fluid (gasoil), being heated to a particular temperature and transported over to a collection tank once the fluid is at the specified temperature.

Dataset Name	Time Steps	Time Series
Electricity	105,216	20
TEP	320,000	41
GHL	200,000	5

Both GHL and TEP were created via process based simulations. Different time series within them show different periodicity, trends and inter-dependencies. For example, in GHL the receiving tank temperature and the heating tank temperature are correlated. In the Electricity dataset, given that each time series corresponds to the electricity consumption of one client, they typically have a periodicity of 24 hours. Thus, evaluating our model on these datasets corresponding to diverse and complex scenarios showcases the effectiveness and generalization capacity of our model architecture.

3.1 Methods for Comparison

Traditional Regression Models

1. *GPR* stands for Gaussian process regression for time series modeling [Roberts *et al.*, 2013].
2. *LSVR* represents the vector-autoregression model with linear support vector regression objective function. [Vapnik *et al.*, 1997].

Deep Learning Models

3. We also compare our models against the model proposed by [Filonov *et al.*, 2016]. The model is a two layer LSTM *Seq2Seq* model with static attention.
4. *LSTNet*, proposed by [Lai *et al.*, 2018], is a state-of-the-art multivariate forecasting model consisting of recurrent, convolutional and auto-regressive components.
5. *S2S* represents a traditional *Seq2Seq* model.
6. *S2S + Attn.* represents a *Seq2Seq* model with the traditional static attention mechanism.

[2]https://tinyurl.com/yyr4lo9j
[3]https://tinyurl.com/y2ej46hr
[4]https://tinyurl.com/yy8vtla5

Proposed Dynamic Attention Models

7. *DyAt*: A *Seq2Seq* model with a dynamic attention mechanism without the hierarchical component, as outlined in section 2.1 and 2.2. Attention weights are calculated using the location-based *score(·)* function (Eq. 2) .
8. *DyAt-H*: A *Seq2Seq* model with a hierarchical dynamic attention mechanism as outlined in Eq. 4. The mean based *contextGen(·)* function is employed with the location based *score(·)* function.
9. *DyAt-MaxPool-H*: Another *Seq2Seq* model with a hierarchical dynamic attention mechanism similar to DyAt-H, except in this case, the *learn* based *contextGen(·)* function is employed to generate the encoder context vector c_e and the bilinear *score(·)* function is employed instead of the location-based variant.

Evaluation Metrics: We evaluate our models using two metrics: (1) Mean-Squared Error (MSE) and (2) Weighted MSE, wherein the later decoder steps are weighted higher. The two metrics are formally defined as

$$\begin{aligned} MSE &= \frac{1}{kml}\sum_{i=1}^{m}\sum_{j=1}^{l}\sum_{q=1}^{k}(\hat{y}_{ijq} - y_{ijq})^2 \\ WMSE &= \sum_{j=1}^{l}\frac{j}{\sum_{p=1}^{l} p}\left(\frac{1}{km}\sum_{i=1}^{m}\sum_{q=1}^{k}(\hat{y}_{ijq} - y_{ijq})^2\right) \end{aligned} \tag{6}$$

3.2 Experimental Setting & Model Architecture

We arbitrarily choose 4 different forecasting sequence lengths 90, 110, 130 to test model performance on long-term forecasting and length 10 for short-term forecasting. We don't consider sequence length to be dataset specific, in an effort to maintain uniform experimental settings across datasets but, in practice, experts can provide optimal values of sequence lengths for forecasting tasks. We employ *Seq2Seq* models with a single hidden layer, GRU as the recurrent unit and set the encoder and decoder phases to have the same sequence length (l) for simplicity. However, our proposed approach is applicable even if this constraint is relaxed. All *Seq2Seq* models are trained with early stopping for a maximum of 200 epochs. Finally, each experiment has a 80/20 training/validation split and a separate holdout test set.

4 Results &Discussion

We evaluate the forecasting performance of our DyAt models in four experimental settings by varying the length of the prediction sequence between 10 and 130 time steps for all the models. For each setting, we evaluate the models on three different datasets. We also compare our DyAt models to several state of the art forecasting models as outlined in section 3.1.

The forecasting performance results have been detailed in Tab. 1. The best performing hierarchical DyAt model (either DyAt-H or DyAt-MaxPool-H) was able to achieve at least a **17.82**% improvement over the state of the art [Filonov *et al.*, 2016] *Seq2Seq* forecasting model and **21.88**% improvement over the LSTNET model proposed by [Lai *et al.*, 2018] across all the datasets and sequence lengths. The DyAt-MaxPool-H model outperforms all other models in the CPS

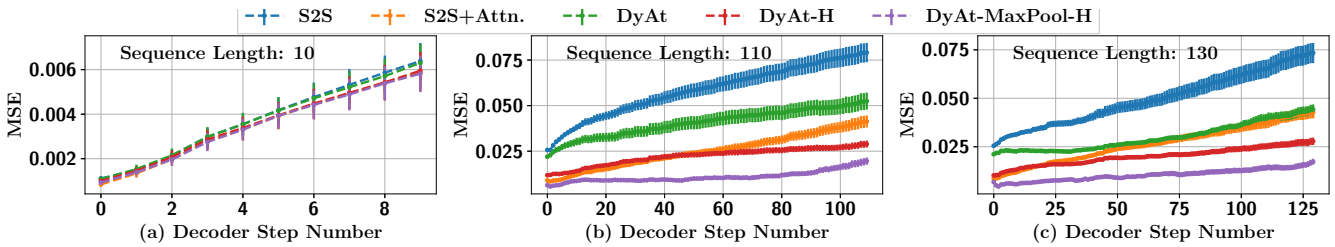
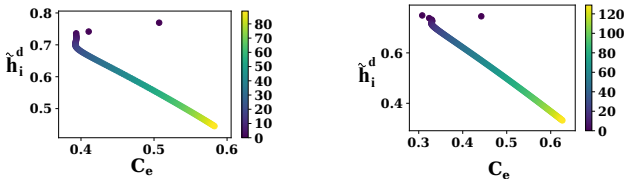


Figure 3: Per Decoder MSE. for different sequence lengths. Both hierarhical models – DyAt-H and DyAt-MaxPool-H show superior performance than other methods for long-term forecasts. We only show results for seq. len. 110, 130 due to space constraints. For the short sequence length (10), the proposed models perform better than the others but the difference is minimal.



(a) Sequence Length = 90 (b) Sequence Length = 130

Figure 4: Characterization of the evolution of $\tilde{h}_i^{hier}$ (hierarchical attentional hidden state) in DyAt-MaxPool-H for different sequence lengths for the TEP dataset.The y-axis shows the alignment of $\tilde{h}_i^{hier}$ with dynamic attentional vector $\tilde{h}_i^d$ in terms of cosine similarity and x-axis its alignment with encoder context vector c_e. The colors indicate the decoder step number, with purple pointing to the first decoder unit and yellow the last. We see that the model starts to rely more (is more similar to) on c_e as we move towards later decoder states showing the holistic representative capacity of hierarchical dynamic attention models, leading to better forecasting performance.

state forecasting tasks on the GHL, TEP datasets (minimum performance improvement **13.69**%). However, in the case of the electric load forecasting, DyAt-H is the best performing model (minimum performance improvement **18.83**%). We believe this can be attributed to the well known property of the maxpool operation (used to calculate the encoder context vector c_e in the DyAt-MaxPool-H model) leading to order invariance [Scherer *et al.*, 2010]. Due to this, the short-term cyclical changes in the electricity dataset are not captured as effectively by the context vector c_e in the case of DyAt-MaxPool-H leading to the drop in performance relative to DyAt-H which employs a *mean* operation to generate c_e instead. In the case of CPS state forecasting, the transitions of state are much more gradual and the data is devoid of significant short-term cyclical components. In order to showcase the effectiveness of the DyAt mechanisms, we also compare their performance to standard *Seq2Seq* architectures with and without the traditional attention mechanisms. Our DyAt-H and DyAt-MaxPool-H models achieve a minimum improvement of **31.33**% over the *Seq2Seq*(S2S) model without attention and a minimum performance improvement of **13.69**% over traditional *Seq2Seq* attention (S2S + Attn.) models.

4.1 Sequence Forecasting Performance Inspection

To further showcase the sequence forecasting performance of DyAt models, we also calculate the MSE at each decoder time step and plot the average MSE per decoder unit for each *Seq2Seq* model in Table 1. We showcase results on the GHL dataset and notice that DyAt-H and DyAt-MaxPool-H, comfortably outperform the other models as depicted in Fig. 3. We notice that although in the short-term forecasting tasks (sequence length = 10) there doesn't seem to be too much disparity in forecasting performance over the entire sequence length, the disparity in performance increases in the case of longer sequences and the performance gap between DyAt-H, DyAt-MaxPool-H models and others increases as we move deeper into the prediction sequence. This showcases the ability of the proposed hierarchical models to reduce the effect of error-propagation to subsequent steps. The hierarchical nature of DyAt-H and DyAt-MaxPool-H models helps them consider the rich information from the encoder phase in addition to the dynamic attention mechanism which allows them to consider representations from the immediate past. Hence the hierarchical attentional hidden state $\tilde{h}_i^{hier}$ at each decoder captures this rich representation from both the encoder phase as well as the recent past leading to better long-term forecasts.

4.2 Evolution of Hierarchical Attentional Vector

As outlined in section 2.3, the hierarchical attentional hidden state $\tilde{h}_i^{hier}$ is calculated as a function of the encoder context vector c_e and the dynamic attentional hidden state $\tilde{h}_i^d$ at each decoder. In order to further understand the effect of the hierarchical model, we inspected the evolution of $\tilde{h}_i^{hier}$ over the course of a prediction sequence with respect to the encoder context vector c_e and the dynamic attentional hidden state at each decoder $\tilde{h}_i^d$. We calculate the cosine similarity of $\tilde{h}_i^{hier}$ and c_e, let's call this Sim_{c_e} and similarly $Sim_{\tilde{h}_i^d}$ for each prediction sequence. We then average the cosine similarity over all batches for each decoder step and plot the $Sim_{\tilde{h}_i^d}$ vs. Sim_{c_e} values in Fig. 4. We only show results for sequence length 90 and 130 due to space limitations. We notice a consistent pattern for all long-term forecasting tasks (90, 110, 130) wherein as the forecast moves further into the decoder phase, the $\tilde{h}_i^{hier}$ starts to rely more heavily on the encoder context vector c_e in addition to the dynamic attentional hidden state $\tilde{h}_i^d$. Throughout the decoding sequence, we notice that $\tilde{h}_i^{hier}$ is influenced significantly by both $\tilde{h}_i^d$ and c_e. This ability of the hierarchical DyAt models to utilize both encoder and dynamic attentional representations leads to their superior performance.

	Sequence Length	10		90		110		130	
	Metric	MSE	WMSE	MSE	WMSE	MSE	WMSE	MSE	WMSE
Dataset Name	Model								
GHL	GPR [Roberts *et al.*, 2013]	0.01736	0.01806	0.08829	0.08866	0.09126	0.09139	0.09291	0.09289
	LSVR [Vapnik *et al.*, 1997]	0.08156	0.08758	0.09475	0.09631	0.09507	0.09635	0.09528	0.09635
	LSTNET [Lai *et al.*, 2018]	0.00356	0.00430	0.01769	0.01947	0.02146	0.02399	0.02451	0.02595
	Filonov et. al. [2016]	0.00335	0.00404	0.01931	0.02001	0.02736	0.03207	0.02675	0.02914
	S2S	0.00341	0.00425	0.03391	0.03894	0.04890	0.05659	0.04196	0.04905
	S2S+Attn.	0.00324	0.00406	0.02106	0.02615	0.02412	0.02946	0.02660	0.03179
	DyAt	0.00343	0.00425	0.02171	0.02578	0.03971	0.04352	0.02753	0.03079
	DyAt-H	0.00327	0.00406	0.01722	0.01964	0.02223	0.02492	0.01927	0.02172
	DyAt-MaxPool-H	**0.00322**	**0.00402**	**0.01043**	**0.01238**	**0.01071**	**0.01229**	**0.01091**	**0.01215**
Electricity	GPR [Roberts *et al.*, 2013]	0.01474	0.01511	0.01716	0.01723	0.01717	0.01728	0.01718	0.01714
	LSVR [Vapnik *et al.*, 1997]	0.01029	0.01149	0.02254	0.02412	0.02176	0.02194	0.02128	0.02167
	LSTNET [Lai *et al.*, 2018]	**0.00795**	**0.00899**	0.01670	0.01638	0.01642	0.01705	0.01686	0.01622
	Filonov et. al. [2016]	0.01030	0.01111	0.01671	0.01678	0.01667	0.01675	0.01681	0.01680
	S2S	0.01001	0.01083	0.01756	0.01765	0.04014	0.03841	0.01852	0.01812
	S2S+Attn.	0.00829	0.00918	0.01569	0.01583	0.01519	0.01551	0.01543	0.01555
	DyAt	0.00856	0.00939	0.01304	0.01299	0.01083	0.01115	0.01312	0.01363
	DyAt-H	0.00823	0.00914	**0.01271**	**0.01256**	**0.01071**	**0.01103**	**0.01140**	**0.01201**
	DyAt-MaxPool-H	0.00852	0.00950	0.01336	0.01284	0.01282	0.01268	0.01459	0.01448
TEP	GPR [Roberts *et al.*, 2013]	0.00600	0.00620	0.01060	0.01100	0.01120	0.01130	0.01140	0.01140
	LSVR [Vapnik *et al.*, 1997]	0.00770	0.00800	0.00820	0.00830	0.00823	0.00827	0.00810	**0.00810**
	LSTNET [Lai *et al.*, 2018]	0.01612	0.01613	0.02239	0.02237	0.02126	0.02126	0.02550	0.02489
	Filonov et. al. [2016]	0.00365	0.00369	0.00734	0.00790	0.00811	0.00852	0.00878	0.00904
	S2S	0.00357	0.00361	0.01033	0.01060	0.01023	0.01052	0.01054	0.01072
	S2S+Attn.	0.00312	0.00313	0.00732	0.00799	0.00813	0.00869	0.00844	0.00887
	DyAt	0.00315	0.00319	0.00685	0.00766	0.00782	0.00841	0.00840	0.00878
	DyAt-H	0.00303	0.00306	0.00679	0.00764	0.00778	0.00841	0.00841	0.00884
	DyAt-MaxPool-H	**0.00283**	**0.00286**	**0.00627**	**0.00726**	**0.00683**	**0.00784**	**0.00716**	0.00814

Table 1: Forecasting performance comparison in terms of mean squared error (MSE) and weighted mean squared error (WMSE), of our methods (DyAt, DyAt-H, DyAt-MaxPool-H) with several state of the art baselines. We notice that the DyAt-MaxPool-H model produces the best forecasts for the GHL, TEP datasets (i.e CPS state forecasting), while DyAt-H yields the best long-term electric load forecasts.

5 Related Work

Time Series Forecasting Methods: Time series forecasting is a well researched topic, with a variety of models like auto-regressive models (AR, MA, ARIMA) and state-space models like Kalman and particle filters, detailed by [De Gooijer and Hyndman, 2006]. Such traditional models however, require certain domain rules governing process state transition (as in the case of Kalman filters) to be known or others like the auto-regressive models require certain assumptions made regarding the properties of the time series themselves to hold.

Deep Learning for Time series Forecasting: Traditional models fail in highly non-linear time series spaces. Recently, deep learning models have proven effective in learning complex function representations. Hence, we employ deep neural network models for the task of time series forecasting. Recently, [Lai *et al.*, 2018] proposed a deep learning model (LSTNET) employing convolutional, recurrent and auto-regressive methods for multivariate time series forecasting in the context of solar and electric load forecasting. They showcased the performance of LSTNET for both short and long-term forecasting tasks. [Romeu *et al.*, 2013] used stacked de-noising autoencoders for indoor temperature forecasting. [Qiu *et al.*, 2014] propose an ensemble of deep belief networks used in conjunction with a support vector regressor (SVR) for time series forecasting and other regression tasks.

Deep Learning for Sequence Modeling: Seq2Seq models are popular in the natural language processing domain. They were initially proposed for the sequential data modeling task of neural machine translation (NMT) [Cho *et al.*, 2014b; Sutskever *et al.*, 2014; Sennrich *et al.*, 2016; Vaswani *et al.*, 2017]. Variants of *Seq2Seq* models have been used for NMT and text summarization [Cho *et al.*, 2014a; Lin *et al.*, 2018; Nallapati *et al.*, 2016; Gehring *et al.*, 2017]. To the best of our knowledge, all *Seq2Seq* models used for time series forecasting in CPS, employ traditional attention mechanisms. We are the first to introduce dynamic attention for multivariate time series forecasting, through our novel DyAt models.

6 Conclusion

In this paper, we developed a novel dynamic attention mechanism and two novel hierarchical dynamic attention models for long-term multivariate time series forecasting and evaluated them on many CPS state forecasting and load forecasting datasets. We demonstrated that our hierarchical dynamic attention models achieved significant improvement in forecasting performance over state of the art time series forecasting baselines. Moving forward, we wish to incorporate more sophisticated attention mechanisms into the DyAt-H and DyAt-MaxPool-H models and inspect their performance in spatiotemporal forecasting applications.

Acknowledgments

This research was supported in part by the National Science Foundation via grants DGE-1545362, IIS-1633363.

References

[Carvallo *et al.*, 2017] Juan Pablo Carvallo, Peter H Larsen, Alan H Sanstad, and Charles A Goldman. Load forecasting in electric utility integrated resource planning. Technical report, Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), 2017.

[Cho *et al.*, 2014a] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014.

[Cho *et al.*, 2014b] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.

[De Gooijer and Hyndman, 2006] Jan G De Gooijer and Rob J Hyndman. 25 years of time series forecasting. *International journal of forecasting*, 2006.

[Filonov *et al.*, 2016] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv preprint arXiv:1612.06676*, 2016.

[Filonov *et al.*, 2017] Pavel Filonov, Fedor Kitashov, and Andrey Lavrentyev. Rnn-based early cyber-attack detection for the tennessee eastman process. *arXiv preprint arXiv:1709.02232*, 2017.

[Gehring *et al.*, 2017] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017.

[Juricek *et al.*, 2001] Ben C Juricek, Dale E Seborg, and Wallace E Larimore. Identification of the tennessee eastman challenge process with subspace methods. *Control Engineering Practice*, 9(12):1337–1351, 2001.

[Lai *et al.*, 2018] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao 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*, pages 95–104. ACM, 2018.

[Lin *et al.*, 2018] Junyang Lin, SUN Xu, Shuming Ma, and Qi Su. Global encoding for abstractive summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 163–169, 2018.

[Luong *et al.*, 2015] Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[Malhotra *et al.*, 2016] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148*, 2016.

[Nallapati *et al.*, 2016] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, 2016.

[Qiu *et al.*, 2014] Xueheng Qiu, Le Zhang, Ye Ren, Ponnuthurai N Suganthan, and Gehan Amaratunga. Ensemble deep learning for regression and time series forecasting. In *2014 IEEE symposium on computational intelligence in ensemble learning (CIEL)*, pages 1–6. IEEE, 2014.

[Roberts *et al.*, 2013] Stephen Roberts, Michael Osborne, Mark Ebden, Steven Reece, Neale Gibson, and Suzanne Aigrain. Gaussian processes for time-series modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984):20110550, 2013.

[Romeu *et al.*, 2013] Pablo Romeu, Francisco Zamora-Martínez, Paloma Botella-Rocamora, and Juan Pardo. Time-series forecasting of indoor temperature using pre-trained deep neural networks. In *International Conference on Artificial Neural Networks*, pages 451–458. Springer, 2013.

[Scherer *et al.*, 2010] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International Conference on Artificial Neural Networks*, pages 92–101. Springer, 2010.

[Sennrich *et al.*, 2016] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1715–1725, 2016.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112, 2014.

[Vapnik *et al.*, 1997] Vladimir Vapnik, Steven E Golowich, and Alex J Smola. Support vector method for function approximation, regression estimation and signal processing. In *Advances in Neural Information Processing Systems*, pages 281–287, 1997.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.