Selego: robust variate selection for accurate time series forecasting

Selego: Robust Variate Selection for Accurate Time Series Forecasting

Anonymous

Abstract*—Integration of sensory technologies into critical applications, such as flight monitoring, building energy optimizations, and health monitoring, has highlighted the importance of effective forecasting models for multi-variate time series. To accomodate this demand, approaches to time series prediction have been extended from uni-variate prediction to multi-variate time series prediction. Naïve extensions of prediction techniques, however, lead to an unwelcome increase in the cost of model learning and, more importantly, a significant deterioration in the model performance due to the limited ability of the extended models to capture the intrinsic relationships among variates. For instance, while recurrent models, such as LSTM and RNN, aim to capture the temporal complexities in the data, their performance can deteriorate quickly. In this paper, we first argue that (a) one can learn a more accurate forecasting model by leveraging *temporal alignments* among variates to quantify the importance of the recorded variates with respect to a target variate. We further argue that, (b) for most applications, traditional time series similarity/distance functions, such as DTW, which require that variates have similar absolute patterns, are fundamentally ill-suited for this purpose and we instead need to quantify temporal correlation not in terms of series similarity, but in terms of temporal alignments of key "events" impacting these series. Finally, we argue that (c) while learning a temporal model using recurrence based techniques (such as RNN and LSTM – even when leveraging attention strategies) is difficult and costly, we can achieve better performance by coupling simpler CNNs with an *adaptive variate selection strategy*. Relying on these arguments, we propose a *Selego* framework[1] for variate selection and experimentally evaluate the performance of the proposed approach on various forecasting models, such as LSTM, RNN, and CNN, for different top-X% variates and different forecasting time in the future (lead) on multiple real-world datasets. Experiments show that the proposed framework can offer significant $(90-98\%)$ drops in the number of recorded variates that are needed to train predictive models, while simultaneously boosting accuracy.

***Index Terms*—Forecasting, recurrent and convolutional networks, variate selection**

I. Introduction

Recent advances in sensory technologies have enabled large scale integration of sensor networks in a wide variety of applications, prediction explainability [3], sub-sequence extraction for interpretable forecasting [16] and shaplet extraction [14]. This rapid integration, consequently, has lead to a significant explosion in the amount of temporal data being generated, both in terms of *depth* (length of time series) and *diversity* (type of time series). Sensor networks have enabled simultaneous recording a variety of attributes defining a system, leading to the generation of multi-variate time series - each variate corresponding to a different attribute being recorded.

[1]Selego is a word of *latin* origin meaning "*selection*".

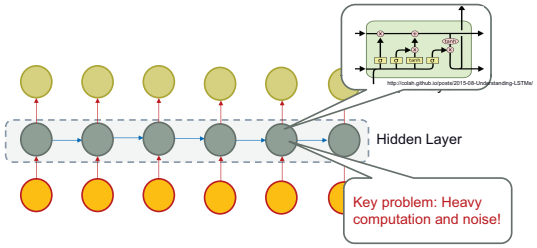


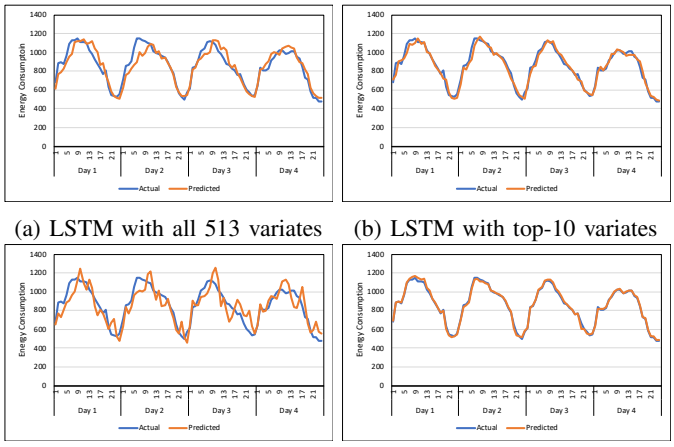
Fig. 1: Given a large number of input and output sequence pairs, recurrence-based models, such as LSTM, learn the underlying hidden model that relate input sequences to output sequences; but these models suffer from heavy computation and noise

A. Time Series Forecasting

With the increase in the availability of time series data, knowledge discovery tasks that rely on these data, such as forecasting, have become increasingly more feasible. The problem of time series forecasting involves learning a function f than can map the observations from the past $(t_1, t_2, \ldots, t-1)$ to the present (t) or the future. The problem involves a set of recorded variates $\mathbb{X} \in \boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_T$ that drive a set of target variates, $\mathbb{Y}$. Forecasting often involves predicting the target variate sufficiently ahead in the future, which we refer as "*lead*" (l) in the paper.

In practice, the forecasts are often imprecise due to various reasons and the function f is often learned with an error, ϵ. The ultimate goal of forecasting model learning task, therefore, is to minimize this error. To do so, various statistical and deep models have been proposed. Statistical forecasting models for time series, primarily SVR [13] and ARIMA [8], have helped reduce the forecasting error in various real-world applications, however, with the increase in the number of variates of the time series, SVR and ARIMA have fallen short in their ability to learn. Neural network-based techniques, such as recurrent neural networks (RNNs) [29] demonstrated that the shortcomings of SVR and ARIMA can be overcome by exploring deep features[2] and their evolution overtime by relying on the $(t-1)^{th}$ state of the network to learn the t^{th} state. Unfortunately, RNNs have proven ineffective on long time series due to catastrophic forgetting. Long-Short Term-Memory networks (LSTM) [20] have been relatively successful in their

[2]While the terms "variate" and "feature" are often used interchangeably, in this paper, we make a clear distinction: A "variate" is an input time series describing a time-varying property of the system being observed, whereas as "feature" is a temporal pattern extracted from a given time series and can be used to characterize that series. This distinction is critical.



(a) LSTM with all 513 variates (b) LSTM with top-10 variates

(c) CNN with all 513 variates (d) CNN with all top-10 variates

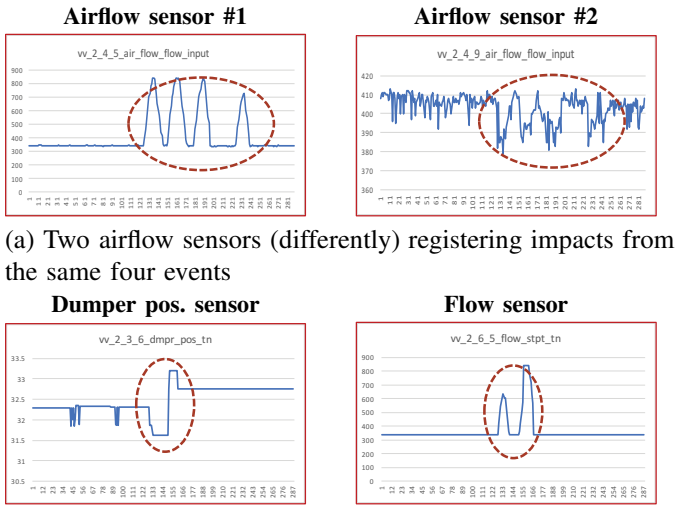
Fig. 2: Sample results: LSTM and CNN based building 1-hour energy consumption prediction results (a,c) using all 513 variates (100%) vs. (b,d) using only the top-10 (2%) variates (see Section IV for details): both models benefit significantly from variate selection

ability to remember and model time series, yet they also suffer from model complexity and are susceptible to noise (Figure 1). In fact, the quality of forecasting models often suffers from the *curse of dimensionality* – statistical or deep, the model accuracy depends on how well the model captures the rich and complex relationships among data variates and not every recorded variate is of equal importance to a target variate. Consequently, using all recorded variates to develop a forecasting model can hurt the overall model performance.

B. Our Contribution: the Selego Framework

As discussed above, naïve extensions of uni-variate forecasting techniques lead to both an increase in the cost of these models and, more importantly, a deterioration in the model performance, as a whole, due to the limited ability of the extended models to capture the intrinsic relationships among variates. In this paper,

- we first argue that one can learn a more accurate forecasting model by leveraging *temporal correlations* among variates to quantify the importance of the recorded variates with respect to a target variate and using these to help reduce the number of variates needed to train a model (Figure 2);
- we further argue that traditional time series similarity/distance functions, such as DTW, are fundamentally ill-suited for this purpose as recorded variates relevant for a particular task do not necessarily look similar to each other (Figure 3); instead, the relationship between two series needs to be quantified based on temporal alignments of the "key events" (or local patterns) identified on these series (Figures 4) *irrespective of how these key events themselves look*;
- we finally argue that, while trying to learn a temporal model for a multi-variate time series using recurrence based techniques (such as RNN and LSTM) is difficult and costly (*even when they are leveraging attention strategies*), we can achieve better performance by coupling simpler CNN based models with an *adaptive variate selection strategy* that captures the temporal evolution of the pairwise relationships among the variates.



(a) Two airflow sensors (differently) registering impacts from the same four events

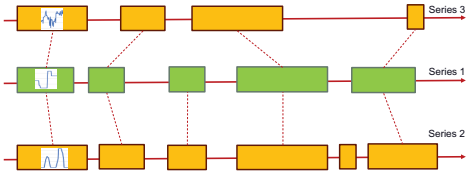
(b) Two different types of sensors (differently) registering impacts from the same two events

Fig. 3: (a) Two airflow sensor readings that have *aligned but not similar* temporal structures; (b) dumper and flow sensors that have *aligned but not similar* temporal structures; note that in these scenarios, the two temporal structures would be judged to be very different from each other under common distance (such as DTW) or similarity (such as Pearson's correlation) functions

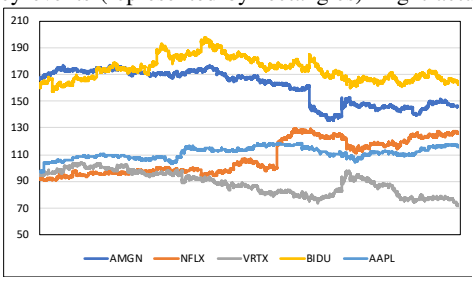
TABLE I: Top-10 variates selected for the NASDAQ (APPL) dataset by different variate selection strategies: Selego selects a larger number of technology and semi-conductor series (shown in bold) that are likely to register impact from similar events as the target series

DTW	PCA	Inf-FS	Selego
Amazon (Tech)	**Adobe (Tech)**	**Nvidia (Semi)**	**Google (Tech)**
Seagate (Semi)	American Air. (Travel)	**Netflix (Tech)**	Amgen (Pharma)
Fastenal (Distr.)	**Amazon (Tech)**	AutoDesk (Software)	**Netflix (Tech)**
Amgen (Pharma)	**Google (Tech)**	**West. Dig. (Semi)**	Vertex (Pharma)
CSX (Rail)	Kraft (Retail)	**Lam Research (Semi)**	**Baidu (Tech)**
Discovery (Comm.)	Ross (Retail)	CSX (Rail)	**Skyworks (Semi)**
Intel (Semi)	**Maxim (Semi)**	**Micron (Semi)**	**Broadcom (Semi)**
BBBY (Retail)	Whole Food (Retail)	Dish (Comm.)	**Facebook (Tech)**

Relying on these observations and arguments, in this paper, we propose a *Selego* framework for variate selection: traditional variate selection mechanisms either require a one-to-one alignment of data points or rely heavily on series similarity. We, however, highlight that what matters is not that the series are similar, but that the temporal scopes of the underlying patterns are temporally aligned. Selego, therefore, ranks variates based on the co-occurrence of key temporal features/events to select variates that have high impact on forecasting of a target variable (Table I). We experimentally validate the



(a) The relationship between two series needs to be quantified based on temporal alignments of the key events identified on these series: Series#1 aligns better with Series#2 than Series#3, irrespective of how the key events (represented by rectangles) might actually look



(b) NASDAQ-AAPL series along with the top-4 series that are temporally aligned with it; note that while these series show evidence of impact from similar external events, they do not necessarily look similar - yet, as we see in Section IV, these series prove to be more predictive than other series that are more similar to NASDAQ-AAPL based on DTW and PCA

Fig. 4: Temporal alignment of variates does not necessarily mean that they look similar

key contributions of Selego in Section IV and observe that Selego is able to identify impactful recorded variates for the target variate, and apply to various domains such as, building energy optimization [7], fuel consumption [18], stock price prediction [26], and brain signals [15]. Experiments show that the proposed framework can offer significant $(90-98\%)$ drop in the number of variates that are needed to train predictive models, while also boosting model accuracies.

C. Organization of the Paper

The paper is organized as follows: Section II presents the state-of-the-art. Section III discusses in details the proposed *Selego framework*, in Section IV we present the experimental evaluation of Selego, and in Section V we conclude.

II. Related Works

A. Time Series Similarity

There has been significant amount of research both into defining measures for comparing sequences, as well as into the development of efficient data structures and algorithms for implementing these core operations [2]. Euclidean distance and, more generally $L_p - norm$ measures, were among the first used to determine the similarity between two time series. Euclidean distance assumes a strict synchrony among time series, it is not suitable when two time series can have different speeds or are shifted in time. Other measures that require equal length and perfect synchrony across time series include cosine and correlation similarity [31], [32].

In most applications, when comparing two sequences or time series, exact alignment is not required. Instead, whether two sequences are going to be treated as matching or not depends on the amount of difference between them; thus, this difference needs to be quantified. In the 1970s, Sakoe [30] and in the 1990s, Berndt [6] proposed an edit distance like dynamic time warping (DTW) technique to find an optimal alignment between two given (time-dependent) sequences under certain restrictions. Intuitively, DTW considers all possible warping paths that can warp (or transform) one series into the other, and picks the warping path that has the lowest cost. DTW has found wide acceptance, and the last two decades have seen several innovations [10], [21]. For example, while the original DTW is not metric (does not satisfy the triangular inequality), [10] proposed an extended version of DTW that does satisfy the triangular inequality.

B. Time Series Modeling

Time series modeling and forecasting have received significant interest with approaches ranging from statistical models [8], [13], deep models [20], [29], and more recently attention networks [4]. Kernel-based methods, ensemble learning, such as decision trees and random forest [22], [28] have been employed as well. Many of these methods make the intrinsic assumption of a predefined relationship between the observed variates and the target variable, yet this assumption may not always be applicable [26].

Deep networks have proven particularly successful in working with large volume and variety of data by mapping them to deep features (non-linearly dependent) through the use of combinations of linear and non-linear operations [12]. Deep recurrent networks (RNNs) have shown success in Non-linear AutoRegressive eXogenous models (NARX), i.e. aimed a predicting value at t, based on the past observation [29]. RNNs' success is often limited to the length of the time series as it suffers from the problem of catastrophic forgetting [4]. Long-short Term Memory (LSTM) were proposed as a solution to the RNNs' short coming which introduced a cell state in addition to the hidden state to remember the past patterns [20]. While LSTM has proven great success, such as in speech translation, voice recognition, and video processing (LSTM-CNN), they are still vulnerable to learning noisy models in presence of large-number of input features which can deteriorate the model performance. [19] has shown that after a certain depth and width the performance of the deep network degrades and the model starts to become noisy, this degradation is credited to the increasing abstractness in the deeper network layers.

High-dimensionality often proves fatal for conventional approaches to time series analysis, such as Dynamic Time Warping (DTW), Segregated ApproXimation (SAX), AutoRegressive Integrated Moving Average (ARIMA), and Support Vector Regression (SVR), where the computational cost increases with the addition of new variates and observations, while the accuracy drops simultaneously. This can prove catastrophic in crucial applications, such as medical domain

(seizure prediction) or manufacturing (supply chain), where it is desired to predict events sufficiently ahead in the future. A potential solution to address the increasing dimensionality of the data is through the use of attention mechanism employed at every hidden layer in the network [4], [26], [35], aimed by determining the subset of important input variates during network training. However, the success of attention mechanism heavily relies on the design of the network architecture. Search for high-performing network architecture in itself is a complex process [5], [17], [36].

[17] demonstrated that salient localized features extracted *before* training a NN-based model, which have been largely ignored by many works in the domain, can improve accuracy by highlighting key insights in the data, In this paper, we build upon a similar idea and explore a variate selection mechanism to help improve the forecasting accuracy (reduce error) by intelligently understanding and quantifying the inter-variate relationships as a function of local temporal features extracted from individual variates. In particular, we show that pre-model training variate selection can reduce the model noise and help learn a robust model.

III. Selego: Robust Variate Selection for Time Series Forecasting

In this section, we present the proposed *Selego framework* which leverages salient localized temporal events to select subset of variates from a multi-variate time series.

A. Uni- and Multi-Variate Time Series

A uni-variate time series (UVTS) is a sequence of ordered pairs of observations and time at which observations were recorded for a given attribute (variate),

$$\boldsymbol{T} = \left[(\mathrm{v}_1, \mathrm{t}_1), (\mathrm{v}_2, \mathrm{t}_2), \ldots, (\mathrm{v}_T, \mathrm{t}_T) \right]. \tag{1}$$

While in general the temporal separation between two consecutive timestamps can be non-periodic, in this paper we assume that timestamps recorded in a UVTS are periodic in nature. We denote the prefix of $\boldsymbol{T}$ until time t as $\boldsymbol{T}_{[t]}$, whereas we denote the value of $\boldsymbol{T}$ at time t as $\boldsymbol{T}_{(t)}$.

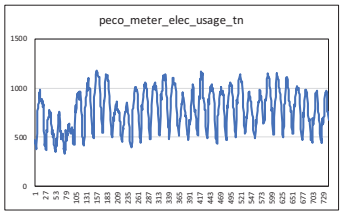
A multi-variate time series (MVTS), $\mathbb{T}$, is a set of uni-variate time series, s.t.

$$\mathbb{T} = \{\boldsymbol{T}_1, \boldsymbol{T}_2, \ldots, \boldsymbol{T}_K\} \tag{2}$$

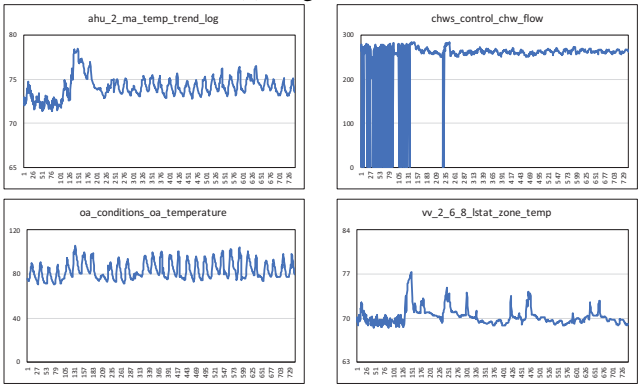
where, K is the number of variates, $\mathbb{T} \in \mathbb{R}^{K \times T}$, and $\boldsymbol{T}_i \in \mathbb{R}^{1 \times T}$.

B. Time Series Forecasting Problem

Time series forecasting involves learning a function f that can map historical observations at time $1, 2, \ldots, t$ to the future observations at time $t+l$; we refer to the value of l as the "*lead time*". The problem involves a set of source variates $\mathbb{X} \subseteq \mathbb{T}$ that drive a set of target variates, $\mathbb{Y} \subseteq \mathbb{T}$; i.e. $f : \mathbb{X}^{\{t\}} \rightarrow \mathbb{Y}^{(t+l)}$.



(a) Target variate



(b) top variates for the energy usage variate

Fig. 5: (a) A sample building energy usage series and (b) several variates that, together, predict it well – note that these series are temporally aligned with the target series but are not necessarily similar to it (see Section IV for details)

C. NN-based Forecasting Models

While Selego has wide applicability, in this paper, we explore its use within the context of neural-network (NN) based forecasting models. In particular, we consider two distinct approaches:

Convolutional neural models (CNNs): Modern neural networks leverage *depth* and *width* of their models to learn complex patterns in the data in the form of deep features [33]. Convolutional neural networks (CNN), achieve this by repeatedly applying convolution operations (complemented with non-linear activation functions and pooling operations that scale the data) to identify multi-scale patterns of different complexities that characterize the input data. The patterns discovered by the CNN depend on the task. In the case of a time series *prediction* problem, where the goal is to discover a function of the form $f : \mathbb{X}^{\{t\}} \rightarrow \mathbb{Y}^{(t+l)}$ the model training would be carried out by providing as input prefixes, $\mathbb{X}^{\{t\}}$, of the input series up to time t and as output the values, $\mathbb{Y}^{(t+l)}$, at time $t+l$ of the target series. Given these, the CNN training process would recover patterns (of varying sizes and complexities) that are useful in the prediction task.

Recurrent neural models (RNNs and LSTMs): Convolutional neural networks (CNN) lack the ability to memorize temporal patterns over time. To counter this, recurrent networks (RNN) introduced a memorization block in form of a recurrent connection to remember the pattern at time $t-1$ to help inform the network at time t [29]. RNNs, however, suffer from frivolous propagation artifacts that may introduce noise and prevent effective memorization. LSTM [20] extends

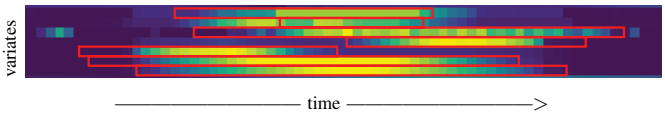


Fig. 6: Sample localized temporal patterns (red boxes) of varying temporal scopes on a multi-variate time series: each row of pixels is a separate variate and pixels with lighter colors have higher values

RNNs with the ability to forget and has been shown to be more effective than the conventional feed-forward neural networks and recurrent neural networks in effective pattern memorization.

Attention mechanisms: As the number of variates and lengths of the time series increase, it is becoming increasingly difficult to learn an effective model, especially for recurrent models [4]. CNN, RNN, and LSTM-based solutions tend to fail when the events that are being inferred are rare or when detecting and predicting anomalies. One difficulty with neural network based inference is that a large number of model parameters need to be learned from data. This is especially problematic for sparse and noisy data sets where it is difficult to learn these model parameters for accurate inference. Recent research [4], [26] has shown that *attention mechanisms*, which help the neural network to focus on different aspects of the data at different stages of inference, have the potential to alleviate this difficulty to some degree. The challenge with such attention mechanisms, however, is that the attention model itself needs to be constructed carefully from data to ensure that the model is able to learn to focus on the most relevant patterns, without mistakenly ignoring patterns critical for the inference task.

This motivates the need for variate selection: As we have seen in Figure 2 (and as we experimentally validate in Section IV), variate selection can potentially boost the predictive model accuracies. Yet, as we also see in Figure 5, the subset of the variates that help predict a time series do not necessarily look like the target series. Instead, as we argued in Section I-B, the subset of the variates to be used must present evidence of impact from events that drive the shape of the target series – often in the form of patterns that are temporally aligned with the patterns of the target series.

D. Robust Localized Temporal Patterns/Features

In this paper, we recognize that in many cases the two variates (time series) that are being compared carry sufficient structural evidences (in the form of *salient temporal patterns (or features)*) that can be used for helping identify *locally relevant* alignments between the two series. Such localized patterns, e.g. SIFT [23], have been shown to be highly effective for image retrieval and object detection applications, as well as neural network hyper-parameter search [17]. [9] has shown that localized temporal patterns can also be used to speed up expensive time series operations, such as DTW computation. In this section, we describe the localized feature extraction process (consisting of "scale-space generation" and "extrema detection" steps) used by Selego to identify key intervals (or "robust localized temporal patterns") in the individual variates.

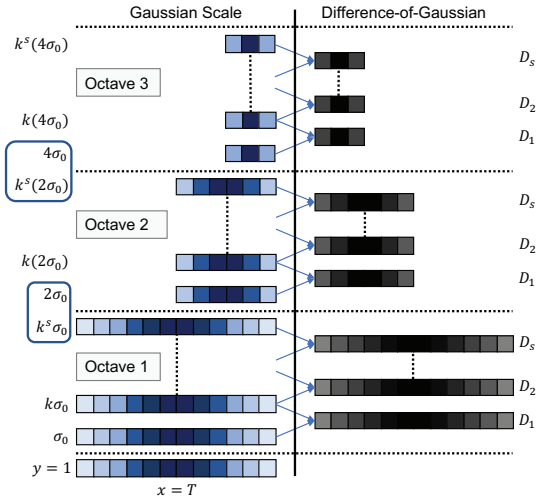


Fig. 7: Generating DoG for a single variate ($y = 1$) taken from a multi-variate time series. The length of the input time series is T

1) Temporal Scale-Space Generation: Temporal features of interest can be of different lengths (Figure 6). Based on the argument that the interesting events will be maximally different from the overall pattern in their local neighborhoods, Selego searches for those points that have largest variations with respect to both time and scale. Therefore, the first step of the process is to create a scale-space consisting of multiple smoothed versions of a given series – each resulting series is then subtracted from the series in the adjacent temporal scale to obtain what we refer to as the difference-of-Gaussian (DoG) series.

Intuitively, the smoothing process can be seen as generating a multi-scale representation of the given series and thus the differences between smoothed versions of a given series correspond to differences between the same series at different scales. Let $\boldsymbol{T}_v$ represent a uni-variate time series, s.t. $\boldsymbol{T}_v \in \mathbb{T}[v, *]$, and $\boldsymbol{T}_v^{(t,\sigma)}$ represents the smoothed version of $\boldsymbol{T}_v$ through convolution with the Gaussian function along the temporal dimension:

$$\mathbf{G}(\mathrm{t}, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-t^2}{2\sigma^2}} \tag{3}$$

such that

$$\boldsymbol{T}_v^{(\mathrm{t},\sigma)} = \mathbf{G}(\mathrm{t}, \sigma) * \boldsymbol{T}_v. \tag{4}$$

Gaussian smoothing is used to create a multi-scale representation of a given series ($\boldsymbol{T}_v$). As shown in Figure 7, the scale space is created by first applying an initial smoothing with parameter σ_0 and then adding L layers of smoothing, where the s^{th} smoothing layer is Gaussian smoothed at level $\kappa^s \times \sigma_0$, where κ is a constant multiplicative factor). Note that, for efficiency, we organize the scales into octaves with increasingly shorter lengths by sub-sampling the series; but this detail is not critical for our discussion.

Fig. 8: A sample candidate feature point,$\mathcal{F}$, (solid black) and its neighbors in adjacent scales "$s+1$" (red) and "$s-1$" (yellow) and in time "$t-1$" (blue) and "$t+1$" (green)

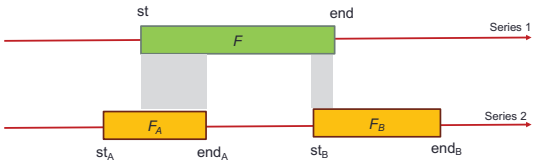


Fig. 9: The temporal alignment between two features depends on degree of overlap between their scope: in this example, the feature $\mathcal{F}$ on Series #1 (highlighted in green) is better aligned with feature $\mathcal{F}_A$ than with feature $\mathcal{F}_B$ on Series #2

2) Extrema Detection: In this step, we search for points of interest, $\langle t, s\rangle$ across multiple scales of the given time series, v, by searching over multiple scales and locations of the given series (here s denotes the corresponding scale).

The search of local extrema (features) is performed by comparing the immediate neighbors (see Figure 8) along both time and scale in the Difference-of Gaussian (DoG) representation, $\mathbf{D}_v^{(\mathrm{t},\sigma)}$, of the input series, $\boldsymbol{T}_v$:

$$\mathbf{D}_v^{(\mathrm{t},\sigma)} = \boldsymbol{T}_v^{(\mathrm{t},\kappa\sigma)} - \boldsymbol{T}_v^{(\mathrm{t},\sigma)}. \tag{5}$$

This enables the algorithm to prune features that are similar to their local neighborhood both in scale and time and, thus, highlight regions of the time series that are distinct from their local neighborhood. More specifically, a pair $\langle t, s\rangle$ on variate v, is an extremum if it is maximum across its *eight* neighbors – *three* per each neighboring scales ($s-1$ and $s+1$) and *two* in time ($t-1$ and $t+1$):

$$max \begin{pmatrix} \boldsymbol{D}_v^{t-1,\kappa^{s+1}\sigma} & \boldsymbol{D}_v^{t,\kappa^{s+1}\sigma} & \boldsymbol{D}_v^{t+1,\kappa^{s+1}\sigma} \\ \boldsymbol{D}_v^{t-1,\kappa^{s}\sigma} & \boldsymbol{D}_v^{t,\kappa^{s}\sigma} & \boldsymbol{D}_v^{t+1,\kappa^{s}\sigma} \\ \boldsymbol{D}_v^{t-1,\kappa^{s-1}\sigma} & \boldsymbol{D}_v^{t,\kappa^{s-1}\sigma} & \boldsymbol{D}_v^{t+1,\kappa^{s-1}\sigma} \end{pmatrix}. \tag{6}$$

In other words, $\langle t, s\rangle$ is designated as an extremum if it is greater than $\Theta\%$ of the maximum of its *8* scale-time neighbors in DoG ($\boldsymbol{D}$).

Note that each identified feature has an associated temporal feature scope, defined by the temporal scale (s) in which it is located. Since under Gaussian smoothing *three* standard deviation would cover $\sim 99.73\%$ of the original temporal points that have contributed to the feature, the radius of the feature is set to 3σ: More specifically, each key temporal feature, $\mathcal{F}$, can be written as triple, $\langle v, t, s\rangle$ and would cover a time interval

$$t\_scope(\langle v, t, s\rangle) = [t - 3\kappa^s\sigma_0, t + 3\kappa^s\sigma_0)]$$

on variate v.

E. Measuring Feature Alignment

Once these key features are extracted, the Selego framework relies on the co-occurrence of salient temporal features to quantify the degree of temporal alignment among variates (Figure 9). Therefore, we first propose a feature overlap measure, *interval alignment*, to measure the temporal overlap (feature co-occurrence) between the features on different variates in the same multi-variate time series. Let $\mathcal{F}_1\langle v_1, t_1, s_1\rangle$ and $\mathcal{F}_2\langle v_2, t_2, s_2\rangle$, be two features; the *interval alignment* (IA) between two features is defined as follows:

$$\mathtt{IA}(\mathcal{F}_1, \mathcal{F}_2) = \begin{cases} overlap(\mathcal{F}_1, \mathcal{F}_2), & overlap(\mathcal{F}_1, \mathcal{F}_2) \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where

$$overlap(\mathcal{F}_1, \mathcal{F}_2) = min(t_{end,1}, t_{end,2}) - max(t_{st,1}, t_{st,2}). \tag{8}$$

Here, $t_{st,i}$ and $t_{end,i}$ represents the start and end time of the feature, $\mathcal{F}_i$, respectively.

It is important to note that the magnitude of IA is likely to be larger for feature pairs that are identified in higher scales, since the $overlap()$ function measures the absolute (not relative) amount of overlap between two feature intervals and since the features at larger octaves/scales have larger scopes. This choice reflects the fact that a large overlap between two features with large scopes is a clearer evidence of temporal alignment between the corresponding variates.

F. Measuring Variate Alignment

Let us be given two variates, $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$, of a multi-variate time series, $\mathbb{T}$, and the corresponding temporal features set, $\mathbb{F}_i = \{\mathcal{F}_{i,1}, \mathcal{F}_{i,2}, \ldots, \mathcal{F}_{i,|\mathbb{F}_i|}\}$ and $\mathbb{F}_j = \{\mathcal{F}_{j,1}, \mathcal{F}_{j,2}, \ldots, \mathcal{F}_{j,|\mathbb{F}_j|}\}$ respectively – here $|\mathbb{F}|$ represents the number of features in the set $\mathbb{F}$. The feature-based temporal alignment of variate $\boldsymbol{T}_i$ against variate $\boldsymbol{T}_j$ is defined as follows:

$$\mathtt{TA}(\boldsymbol{T}_i|\boldsymbol{T}_j) = \frac{\sum_{m=1}^{|\mathbb{F}_i|} max_{n\in(1,\ldots,|\mathbb{F}_j|)}\, \mathtt{IA}(\mathcal{F}_{i,m}, \mathcal{F}_{j,n})}{|\mathbb{F}_i|}. \tag{9}$$

Given this, we then define the variate alignment (VA) between the two variates as

$$\mathtt{VA}(\boldsymbol{T}_i, \boldsymbol{T}_j) = \mathtt{TA}(\boldsymbol{T}_i|\boldsymbol{T}_j) + \mathtt{TA}(\boldsymbol{T}_j|\boldsymbol{T}_i). \tag{10}$$

It is important to note that, while this measure seeks maximal temporal alignment between features of the variates $\boldsymbol{T}_i$ and $\boldsymbol{T}_j$, this does not imply that the time series will actually be similar – this is because, the variate alignment function, VA, and its various components do not consider how the individual features/patterns look; instead, they focus only on whether the features/patterns are temporally aligned or not.

G. Variate Alignment Graph and top-k Variate Selection

Let $\mathbb{T}$ be, as described in Section III-A, a multi-variate time series, s.t.

$$\mathbb{T} = \{\boldsymbol{T}_1, \boldsymbol{T}_2, \ldots, \boldsymbol{T}_K\}, \tag{11}$$

where K is the number of variates. As formulated in Section III-B, let the task be to learn a function $f : \mathbb{X}^{\{t\}} \rightarrow \mathbb{Y}^{(t+l)}$ to forecast, with lead time l, a set of target variates, $\mathbb{Y} \subseteq \mathbb{T}$ using a set of source variates $\mathbb{X} \subseteq \mathbb{T}$. To address this task effectively Selego creates a *variate alignment graph*, $G_{\mathbb{X},\mathbb{Y},l}$, and uses this graph to help select top-k variates in $\mathbb{X}$ to be used for training a predictive model.

1) Lead-l Variate Alignment Graph: Lead-l variate alignment graph, $G_{\mathbb{X},\mathbb{Y},l}(V, E, w_l)$, is a weighted graph where

- $V = \mathbb{X} \cup \mathbb{Y}$,
- $E = E_{XX} \cup E_{YY} \cup E_{XY}$, where
  - $E_{XX} = \{\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle \mid \boldsymbol{T}_n, \boldsymbol{T}_m \in \mathbb{X}\}$,
  - $E_{YY} = \{\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle \mid \boldsymbol{T}_n, \boldsymbol{T}_m \in \mathbb{Y}\}$,
  - $E_{XY} = \{\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle \mid \boldsymbol{T}_n \in \mathbb{X}, \boldsymbol{T}_m \in \mathbb{Y}\}$,
- for all $\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle \in E_{XX} \cup E_{YY}$, the edge weight is computed as $w_l(\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle) = \texttt{VA}(\boldsymbol{T}_n, \boldsymbol{T}_m)$, and
- for all $\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle \in E_{XY}$, the edge weight is computed as

$$w_l(\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle) = \texttt{VA}(\boldsymbol{T}_n^{\{l\}}, \boldsymbol{T}_m^{(l)});$$

here $\boldsymbol{T}^{(l)}$ is the l-step back-shifted version of $\boldsymbol{T}$:

$$\boldsymbol{T}^{(l)} = [(\mathrm{v}_{l+1}, \mathrm{t}_1), (\mathrm{v}_{l+2}, \mathrm{t}_2), \ldots, (\mathrm{v}_T, \mathrm{t}_{T-l})], \quad (12)$$

whereas $\boldsymbol{T}^{\{l\}}$ is the l-step shortened version of $\boldsymbol{T}$:

$$\boldsymbol{T}^{\{l\}} = [(\mathrm{v}_1, \mathrm{t}_1), (\mathrm{v}_2, \mathrm{t}_2), \ldots, (\mathrm{v}_{T-l}, \mathrm{t}_{T-l})]. \quad (13)$$

Intuitively, the weight $w_l(\langle \boldsymbol{T}_n, \boldsymbol{T}_m \rangle)$ for an edge crossing the source and target variates represents the temporal alignment among series where the target variates are shifted l steps backwards. The graph $G_{\mathbb{X},\mathbb{Y},l}(V, E, w_l)$ represents all lead-l alignments between source and target variate pairs (along with synchronous variate alignments for source pairs and target pairs).

2) Top-k Variate Selection: Given the lead-l variate alignment graph, $G_{\mathbb{X},\mathbb{Y},l}(V, E, w_l)$, the k source variates, $\hat{\mathbb{X}}$, to be used for training can be selected using various node selection strategies, including random walk based techniques, such as Personalized PageRank [34], which would rank the nodes in a graph with respect to a given seed node set (the target nodes $\mathbb{Y}$ in this case) through a random walk that would emphasize those nodes that are quickly reachable from the seed nodes over a large number of paths, against those that are poorly connected to the seed nodes.

In order to prevent the specific variate ranking strategy to cloud the results and assess the general applicability of the variate selection approach to multi-variate time series forecasting, in the experiments reported in Section IV, we use a simpler strategy, where we only consider the edge set, E_{XY}, between source/target pairs and rank the target variates in $\mathbb{X}$ according to their average edge weights towards the source variates in $\mathbb{Y}$ to select the top-k target variates, $\hat{\mathbb{X}}$.

H. Lead-l Model Training

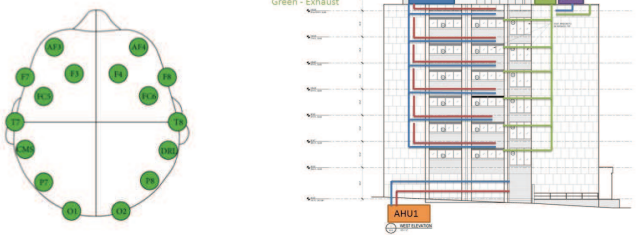
Once the top-k subset, $\hat{\mathbb{X}}$, of source variates are selected, in the final step we train the model (CNN, RNN, or LSTM) to learn a function f_l which forecasts the values, l step in the future, of the target variables, $\mathbb{Y}$, using source variables, $\hat{\mathbb{X}}$. More specifically, the training process seeks the function

$$f_l : \hat{\mathbb{X}}^{\{l\}} \rightarrow \mathbb{Y}^{(l)}, \quad (14)$$

where $\mathbb{Y}^{(l)}$ is the l-step back-shifted version of the target variates[3] in $\mathbb{Y}$ and $\hat{\mathbb{X}}^{\{l\}}$ is the l-step shortened version of the k source variates in $\hat{\mathbb{X}}$.

TABLE II: Overview of multi-variate time series datasets

Datasets	NASDAQ	EEG (O1, O2)	FC	BE
# of Variates	81	80 (=5× 16)	157	390
# of Timestamps	210	19	212	24
Time Unit	1 minute	5 seconds	1 minute	1 hour



(a) Brain sensors (b) Building energy sensors

Fig. 10: (a) Positions of the brain sensors for the EEG-BCI data set [15]; (b) supply, return, and exhaust readings for the various heating units on the building energy data

IV. Experiments

In this section, we experimentally evaluate the validity of the key arguments presented in Section I-B and assess the effectiveness of the Selego framework against alternative variate selection strategies, for various forecasting models. We implemented Selego in Python environment (3.5.2) using Keras Deep Learning Library (2.2.4-tf) [11] with TensorFlow Backend (1.14.0) [1]. All forecasting models were trained on an Intel Xeon E5-2670 2.3 GHz Quad-Core Processor with 32GB RAM equipped with Nvidia Tesla P100 GPU with 16 GiB GDDR5 RAM with CUDA-10.0 and cuDNN v7.6.4[4]. The variate selection processes were executed on MATLAB R2018b U5 (9.5.0.1178774) on MacOS 10.14.6 with 2.9Hz Intel Core i5 equipped with NVIDIA GeForce GT 750M 1GB graphics card.

A. Datasets

As we summarize in Table II, to evaluate the application of the proposed Selego framework, we consider four diverse real-world datasets from a variety of domains:

– **NASDAQ (S&P and APPL)**: [26], comprises of prominent NASDAQ stocks under S&P Index, stock prices and index is recorded for 105 days from July 26, 2016 - Dec

[3]Without loss of generality, in the experiments reported in Section IV, we consider target sets each with a single variate (i.e., $|\mathbb{Y}| = 1$).

[4]Results presented in this paper were obtained using NSF testbed: "*Chameleon: A Large-Scale Re-configurable Experimental Environment for Cloud Research*"

TABLE III: Lead (l) and top $X\%$ variate selection configurations (since EEG-BCI and BE data sets have a maximum temporal length of 19 and 24, respectively, the value of $l = 50$ is incompatible with those data sets)

Datasets	Lead (l)	Top % Variates
NASDAQ	1, 5, 10, 50	10, 50, 90, 100
EEG-BCI	1, 5, 10	10, 50, 90, 100
FC	1, 5, 10, 50	5, 10, 50, 90, 100
BE	1, 5, 10	2, 10, 50, 90, 100

22, 2016. We explore two target variables for this dataset, the S&P Index and APPL.

– **EEG-BCI**: [15] records brain signals, using the BCI System, for 30 subjects while they are performing visual activities. As we see in Figure 10, there are 16 EEG sensors placed on the subjects. The time domain signal from each sensor is transformed into 5 frequency bands, leading into a total of 80 variates. Among these, we consider the observed responses from the left and right *occipital lobes (O1,O2)* of the subjects as to 10 ($= 2 \times 5$) target variates.

– **Fuel Consumption (FC)**: This is a proprietary dataset, comprising of $\sim$ 500 variates for various flights averaging for 3.5 hours from takeoff to landing. Here, we forecast the fuel consumption for the flights using ~ 157 (non-categorical) variates that are not directly measuring aspects of fuel consumption.

– **Building Energy (BE)**: This is a proprietary dataset with 512 variates recording various indoor (e.g. heating, cooling, airflow) and outdoor sensor readings for 30 consecutive days at a resolutions of 1 hour. For this data set, we consider 390 non-categorical variates and select as the target variable the overall power consumption for the whole building.

B. Alternative Variate Selection Methods

In addition to Selego, we consider the following feature selection strategies:

- As discussed in Section II-A, **DTW** [6] is a widely-used elastic distance measure which accounts for differences in speed of patterns across two time series. In this case, top-k variates are selected by inversely sorting the source variates based on their DTW distances to the target variate. Note that, unlike Selego (which emphasizes temporal alignments of key events), DTW distance gives precedence to variates that have similar shapes.
- In **PCA**, instead of comparing variates in the temporal domain, we compare them in a latent space. More specifically, we first create a variate-variate co-variance matrix, C, which is then decomposed into $C = USU^T$ using PCA [25] based decomposition. Here, U is a factor matrix, where the rows correspond to source and target variates and columns correspond to latent basis vectors. Given this, the top k variates are selected by computing the dot product of the rows of U corresponding to source variates with the row of U corresponding to the target variate and ranking the source variates in decreasing order of dot-product match.
- **Inf-FS** [27] is a recently proposed feature selection strategy which ranks input variates based on a random-walk on their transition graph representing the inverse (Spearman) correlation between the variates.

C. NN-based Models

As described in Section III-C, variate selection can be used within the context of various neural-network (NN) based models. In this section, we consider CNN, RNN, and LSTM-based models (both attentioned and without attention):

Recurrent neural models: We consider two widely-used recurrent models (LSTM [20] and RNN [29]). As the default model architecture, we consider a model with 1 hidden layer with 200 computational units (LSTM, RNN) – the hidden activations were "*tanh*" and "*hard sigmoid*" for RNN and LSTM respectively. '*Linear*'" activation was used as output activation for all models. Models were trained for 200 epochs with batch size of 1, using mean absolute error ("*mae*") and "RMSProp" as model loss and optimizer[5].

Convolutional neural models: In addition to recurrent models, we also consider convolutional kernels as simple (non-sequential) model. In particular, CNN sees the entire temporal length at any given instance opposed to recurrent models where only one time instance (in sequence) is available at a time. To ensure fair comparison against LSTM and RNN experiments, we considered a CNN model with 1 hidden layer with 200 computational units, with linear activation function. The model was trained for 200 epochs with batch size of 1, using "*mae*" and "RMSProp" as model loss and optimizer.

Attentioned models [4]: We also considered attentioned versions of the CNN, RNN, and LSTM models. In particular, we applied the [4] encoder-decoder based attention module, which used encoder to map an input sequence $(\boldsymbol{T}^1 \dots \boldsymbol{T}^T)$ to a sequence of continuous representation $\boldsymbol{Z} = (\boldsymbol{Z}^1 \dots \boldsymbol{Z}^T)$ and decoder generates an output sequence $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{Y}}^1 \dots \hat{\boldsymbol{Y}}^T)$ one element at a time, i.e. applying *fine-grain* attention.

D. Data Normalization

In the experiments, we considered three alternative normalization strategies:

- **No normalization:** In this case, we use the input time series as is.
- **Min-Max normalization:** In this case, each uni-variate time series is independently scaled such that the minimum value is equal to 0.0 and the maximum value is equal to 100.0.
- **Z-normalization:** In this case, we use the well-known Z-normalization strategy [24] to normalize each uni-variate time series.

Note that the variate selection and the NN-based model training steps *do not necessarily need to rely on the same normalization strategy*.

[5]We report the best model performance across 200 epochs.

TABLE IV: Average MAE scores under different normalization strategies (w/o variate selection, lead-1 prediction): Z-normalization leads to the best overall accuracy accross NN-models and data set

		Prediction Error (MAE) as a Function of the Normalization Strategy (w/o Variate Selection)											
		LSTM			RNN			CNN			Overall Average MAE		
		No-Norm	Min-Max	Z-Norm	No-Norm	Min-Max	Z-Norm	No-Norm	Min-Max	Z-Norm	No-Norm	Min-Max	Z-Norm
100% Variates Lead-1	Building	659.24	45.20	57.89	211.96	48.79	55.60	44.60	53.51	79.97	305.3	49.2	64.5
	Aviage	1329.09	157.50	221.32	1189.97	705.24	276.81	315.17	323.36	232.46	944.7	395.4	243.5
	Nasdaq	4312.62	8.70	14.20	2039.96	17.51	15.11	18.72	42.75	8.92	2123.8	23.0	12.7
	Apple	6.68	1.56	1.51	6.45	1.35	1.60	2.13	1.69	0.77	5.1	1.5	1.3
	EEG-O1	1516.36	17.17	14.59	911.36	15.84	15.42	23.67	11.33	2.54	817.1	14.8	10.9
	EEG-O2	1525.88	16.89	15.65	912.30	12.70	16.74	25.44	11.96	2.60	821.2	13.9	11.7
										Average	836.2	82.9	57.4

E. Experiment Parameters

To assess the effectiveness of variate selection strategies in different settings, we explored various top-X% of variates selections and different temporal "*lead*" conditions. We varied the ratio of the selected source variates from 2% to 100% of variates in the data set (excluding the target variable) to demonstrate how Selego performs with different numbers of variates – note that $k = \lceil num\_variates \times (X/100) \rceil$. We trained forecasting models for varying leads from $l = 1$ to $l = 50$. i.e. $t + 1$, to distinct future, i.e. $t + 50$. Table III lists the configurations specific to each dataset.

To extract the key patterns using Selego, we set σ_0 to 0.5, the maximum number of scales to 9, and κ to $\sqrt[3]{2}$ – this leads local features of sizes $3 = (6 \times 0.5)$ time units to $24 = (6 \times ((\sqrt[3]{2})^9 \times 0.5))$ time units for computing variate alignments.

We use 70% of the available samples for training, 10% for validation, and 20% for testing.

F. Evaluation Metrics

We report the model forecasting error using *mean absolute error* (MAE):

$$MAE(\boldsymbol{Y}_{true}, \boldsymbol{Y}_{pred}) = \frac{1}{T} \sum_{t=1}^{T} \left| \boldsymbol{Y}_{true}[t] - \boldsymbol{Y}_{pred}[t] \right|, \quad (15)$$

here, $\boldsymbol{Y}_{true}$ and $\boldsymbol{Y}_{pred}$ are the true and predicted values of the target variable and T is the length of the time series. Note that, if the data is normalized, we bring the data back to the original value range before computing the MAE. We use the resulting MAE values in two different ways:

- For comparing the accuracy performance for a given approach under various problem settings, we compute and report the *average MAE* for all testing instances for each configuration.
- For comparing the alternative variate selection strategies we compute and report

$$DCG_{avg,S}(D) = \frac{1}{l} \times \sum_{l} DCG_S(D, l),$$

where S is a variate selection strategy, D is a data set, l is the forecasting lead, and $DCG_S(D, l)$ is defined as

$$DCG_S(D, l) = \sum_{i=1..|S|} \frac{rank\_count(D, S, l, i)}{log_2(i + 1)}.$$

Here $rank\_count(D, S, l, i)$ is the number of problem configurations (model, number of variates etc.) in which the variate selection strategy provides the i^{th} best (i.e., lowest) MAE among all available strategies. Intuitively, the higher the $DCG_{avg,S}(D)$ value is, the better performing is the variate selection strategy S for data set D.

G. Results and Discussions

As we discussed in Section IV-D, the variate selection and the NN-based model training steps do not necessarily need to rely on the same normalization strategy. Therefore, before investigating the impact of variate selection strategies on forecasting accuracies, in Table IV, we first consider the impact of data normalization on model accuracy when no variate selection is applied. As we see in the table, the Z-normalization strategy leads to the best overall accuracy accross NN-models and data set (even for the building energy data where min-max normalization provides better result, the difference is relatively minor). Therefore, in the rest of this section, we will train NN-models on Z-normalized data by default (through variate selection process may be applied on non-normalized, min-max normalized, or Z-normalized data).

1) Impact of Variate Selection on Forecasting Accuracy: In Table V, we present average MAE values for different degrees of variate selection, learning models, data normalization strategies, and forecasting leads (the MAE scores included in this table are averages of MAEs for the four variate selection strategies). From this table, we see that *CNN with tight variate selection provides the best overall results*:

- It is interesting to note that, even though it is not often the best option when considering all 100% of the variates, CNN-based models become highly effective when we are able to select and focus only the relevant variates through the variate selection strategy; this confirms our argument that, when coupled with variate selection, CNNs could be more effective than sequence-aware recurrent networks (such as RNN and LSTM) that attempt to learn temporal patterns (through recurrence) but have difficulties in achieving this task in practice.
- As expected, when using all 100% variates, attention technique may be used to help reduce MAE, but its impact on accuracy is limited and in some cases (especially

TABLE V: Average MAE values for different degrees of variate selection, learning models, data normalization strategies, and forecasting leads (the MAE scores are averages of MAEs for the four variate selection strategies

			Average MAE (average for all Four Variate Selection Strategies)																
			LSTM				LSTM (Att)	RNN				RNN(Att)	CNN					CNN(Att)	
			10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	
NASDAQ-INDEX	No Norm	Lead-1	7.7	10.8	12.1	14.2	28.4	2.2	17.0	19.9	15.1	13.9	1.2	11.2	7.9	8.9	10.3	
		Lead-5	18.7	17.3	36.3	10.2	34.1	3.6	11.1	10.2	18.2	14.8	1.5	8.2	8.1	12.3	11.1	
		Lead-10	18.1	19.0	43.3	8.3	38.7	2.1	5.7	10.7	18.7	18.4	1.4	4.5	7.8	11.3	11.0	
		Lead-50	12.0	17.8	27.6	12.2	46.7	2.7	4.8	8.5	14.5	18.7	1.0	4.9	6.7	12.1	20.7	
	Z Norm	Lead-1	20.0	13.5	16.0	14.2	28.4	2.3	11.1	9.9	15.1	13.9	1.3	12.9	8.3	8.9	10.3	
		Lead-5	4.7	17.4	26.7	10.2	34.1	1.6	8.2	10.5	18.2	14.8	0.7	9.2	9.9	12.3	11.1	
		Lead-10	14.9	14.8	51.5	8.3	38.7	3.1	8.8	9.0	18.7	18.4	1.4	7.7	8.3	11.3	11.0	
		Lead-50	6.0	15.0	27.5	12.2	46.7	1.8	5.2	9.3	14.5	18.7	0.9	6.4	9.0	12.1	20.7	
			10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	
NASDAQ-APPLE	No Norm	Lead-1	1.0	1.3	1.2	1.5	2.1	0.1	1.3	1.4	1.4	1.2	0.2	0.5	0.8	0.9	0.8	
		Lead-5	3.3	1.6	1.6	1.3	1.6	0.4	1.4	1.8	1.3	1.7	0.2	0.8	1.4	0.9	1.1	
		Lead-10	3.1	2.3	1.8	1.0	2.5	0.2	1.6	1.7	1.6	1.5	0.1	0.9	1.6	0.9	0.7	
		Lead-50	3.5	1.9	2.1	0.9	1.4	0.3	1.2	1.7	1.9	1.4	0.3	0.6	1.1	0.8	1.1	
	Z Norm	Lead-1	1.7	1.7	1.9	1.5	2.1	0.2	1.5	1.8	1.4	1.2	0.1	0.7	1.4	0.9	0.8	
		Lead-5	0.9	1.8	1.6	1.3	1.6	0.2	1.5	1.8	1.3	1.7	0.1	1.0	1.3	0.9	1.1	
		Lead-10	1.3	1.9	1.6	1.0	2.5	0.2	1.3	1.8	1.6	1.5	0.1	0.6	1.3	0.9	0.7	
		Lead-50	1.8	2.5	2.0	0.9	1.4	0.1	1.3	1.6	1.9	1.4	0.1	0.5	1.0	0.8	1.1	
			10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	
EEG-O1	No Norm	Lead-1	13.5	14.3	14.7	14.6	13.6	13.6	15.1	15.4	15.4	14.1	1.4	1.9	2.4	2.5	2.5	
		Lead-5	14.1	14.6	14.9	15.0	14.8	14.1	15.6	15.9	16.1	15.6	0.9	1.3	1.6	1.9	16.3	
		Lead-10	13.2	14.3	14.7	14.8	14.7	13.8	15.3	15.5	15.5	15.1	0.9	1.4	1.8	1.9	16.3	
	Z Norm	Lead-1	13.4	14.0	14.4	14.6	13.6	13.6	15.0	15.4	15.4	14.1	1.4	1.9	2.4	2.5	2.5	
		Lead-5	14.1	14.6	15.1	15.0	14.8	14.1	15.5	16.1	16.1	15.6	0.8	1.2	1.7	1.9	16.3	
		Lead-10	13.4	14.5	14.8	14.8	14.7	13.9	15.1	15.7	15.5	15.1	1.0	1.5	1.8	1.9	16.3	
			10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	10%	50%	90%	100%	100%	
EEG-O2	No Norm	Lead-1	14.5	15.5	15.6	15.6	15.0	14.6	16.2	16.6	16.7	16.4	1.4	1.9	2.4	2.6	2.5	
		Lead-5	15.2	15.4	15.9	16.0	16.0	15.3	16.6	17.1	17.3	17.1	0.8	1.4	1.7	1.9	17.4	
		Lead-10	14.6	15.3	15.6	15.7	15.3	14.9	16.4	16.8	17.0	16.9	1.0	1.4	1.8	2.0	17.4	
	Z Norm	Lead-1	14.6	15.5	15.8	15.6	15.0	14.6	16.1	16.9	16.7	16.4	1.4	1.9	2.3	2.6	2.5	
		Lead-5	15.0	15.3	15.9	16.0	16.0	15.2	16.5	17.3	17.3	17.1	0.8	1.3	1.8	1.9	17.4	
		Lead-10	14.6	15.4	15.6	15.7	15.3	14.8	16.3	16.8	17.0	16.9	1.0	1.4	1.7	2.0	17.4	
			2%	10%	50%	90%	100%	100%	2%	10%	50%	90%	100%	100%	2%	10%	50%	90%	100%	100%
Building Energy	No Norm	Lead-1	37.3	38.7	48.4	55.1	57.9	50.2	34.5	38.8	50.1	54.2	55.6	51.7	6.4	10.0	33.7	60.3	80.0	62.9
		Lead-5	33.1	41.9	56.6	62.4	71.0	49.7	40.2	44.5	58.5	69.1	74.2	55.8	6.9	11.8	39.2	55.1	60.3	92.7
		Lead-10	35.3	37.5	53.2	65.7	73.8	59.5	34.9	41.4	56.0	63.0	73.2	67.9	6.7	12.0	34.7	46.7	56.4	91.6
	Z Norm	Lead-1	36.5	37.2	51.7	59.4	57.9	50.2	35.3	35.7	51.0	58.1	55.6	51.7	4.8	9.5	35.3	57.5	80.0	62.9
		Lead-5	34.2	40.2	49.1	59.3	71.0	49.7	39.5	40.7	57.7	66.8	74.2	55.8	6.1	11.8	35.8	51.9	60.3	92.7
		Lead-10	36.0	39.4	49.7	68.7	73.8	59.5	39.0	40.3	56.6	70.3	73.2	67.9	7.0	10.9	36.1	53.6	56.4	91.6
			5%	10%	50%	90%	100%	100%	5%	10%	50%	90%	100%	100%	5%	10%	50%	90%	100%	100%
Fuel Cons.	No Norm	Lead-1	189.7	202.5	220.7	203.1	221.3	172.1	201.7	212.0	259.3	312.7	276.8	273.7	41.1	65.8	165.7	237.6	232.5	201.6
		Lead-5	188.4	197.9	241.1	216.0	214.1	213.1	203.8	224.9	249.6	294.2	288.5	270.0	63.2	72.1	181.7	244.0	234.7	223.5
		Lead-10	193.8	202.1	230.7	226.4	227.0	208.9	215.1	230.0	256.9	315.3	335.6	321.5	56.5	75.2	149.7	239.4	261.9	399.2
		Lead-50	196.6	201.5	258.4	264.6	268.5	435.2	205.0	225.2	267.8	289.3	290.0	956.9	37.8	44.6	100.0	175.4	211.0	1008.2
	Z Norm	Lead-1	173.1	186.7	186.3	211.1	221.3	172.1	183.7	197.9	215.7	301.1	276.8	273.7	35.3	49.4	145.4	203.4	232.5	201.6
		Lead-5	181.1	201.7	214.3	211.5	214.1	213.1	195.0	195.9	242.9	301.2	288.5	270.0	33.3	55.5	176.7	232.1	234.7	223.5
		Lead-10	188.1	201.5	218.8	225.4	227.0	208.9	195.2	233.2	261.8	312.6	335.6	321.5	40.5	51.2	146.3	223.8	261.9	399.2
		Lead-50	189.5	194.1	218.8	257.7	268.5	435.2	195.3	214.5	266.6	287.7	290.0	956.9	35.8	45.3	117.6	172.8	211.0	1008.2

when aiming forecasting with large leads) attention can actually reduce accuracy; in contrast, variate selection is significantly more effective in eliminating noise and unnecessary data and, thus, consistently provides significantly large reductions in MAE. In the experiments, the only noticeable exception is for NASDAQ-APPLE data set with lead times ≥ 5, using LSTM model with very tight (10%) variate selection – but even for that data set and lead times, RNN and CNN both provide significant accuracy gains using only 10% selected variates.

2) Selego vs. Other Variate Selection Strategies: In Table VI, we present the average DCG scores for the four variate selection algorithms and three data normalization strategies (for a total of 12 alternatives). The DCG scores included in the table are averages of DCG values for all data sets and all variate selection rates reported in Table III.

As we see in this table, under all data normalization strategies, the proposed Selego variate selection strategy provides good results, indicating its robustness to the shape of the data – the best overall DCG result is obtained with Selego under Z-normalized data. In fact, the second best DCG is also provided by Selego under the original, non-normalized data: since Selego ignores the shapes of the patterns, but relies only on the co-occurrence/alignment of key events in the time series, it is inherently robust and does not require normalization to return accurate predictions.

In contrast, similarity/distance based measures (DTW and PCA) perform poorly under all normalization strategies: in fact the worst 6 configurations (among all 12 configurations) are obtained using DTW or PCA, confirming our argument that variates that have high predictive power do not necessarily look similar to the target variate. Note that, while Inf-FS is somewhat competitive against Selego on non-normalized and min-max data, its best overall DCG value, 4.6, is significantly lower than the best DCG value, 5.0, achieved by Selego.

In Table VII, we take a more detailed look at the DCG scores. In particular, we present average DCG scores separately for each normalization strategy. As we see here, when

TABLE VI: Average DCG scores for the four variate selection algorithms and three data normalization strategies (total 12 alternatives) – the DCG scores are averages of DCG values for all data sets and all variate selection rates reported in Table III

Norm Type		Avg. DCG Score (higher the better)	Rank
No Norm	DTW	4.0	8
	PCA	3.7	12
	Inf-FS	4.6	3
	Selego	4.7	2
Min-Max Norm	DTW	4.0	9
	PCA	3.9	10
	Inf-FS	4.5	4
	Selego	4.4	5
Z Norm	DTW	4.1	7
	PCA	3.8	11
	Inf-FS	4.2	6
	Selego	5.0	1

TABLE VII: Average DCG scores for the four variate selection algorithms under two data normalization strategies for different data sets – the presented DCG scores are averages of DCG values for all variate selection rates reported in Table III

Norm Type	Var. Select	NASDAQ INDEX	NASDAQ APPLE	EEG-O1	EEG-O2	Building Energy	Fuel Cons.
NoNorm	DTW	5.2	5.1	5.5	5.8	6.3	7.4
	PCA	5.1	5.8	4.7	4.3	7.5	5.9
	Inf-FS	6.1	6.5	5.9	6.1	8.2	8.7
	Selego	6.7	5.7	6.9	6.8	8.8	8.8
Min-Max-Norm	DTW	6.2	5.8	5.6	5.7	7.0	7.2
	PCA	5.1	6.1	4.4	4.6	8.3	6.8
	Inf-FS	5.7	5.0	6.3	6.2	6.4	9.5
	Selego	6.1	6.1	6.7	6.6	9.0	7.2
Z-Norm	DTW	5.8	5.3	5.5	5.4	7.1	8.2
	PCA	4.9	5.5	5.2	4.8	7.3	6.2
	Inf-FS	5.6	5.1	5.8	6.0	7.6	7.1
	Selego	6.7	7.2	6.6	6.9	8.7	9.3

considering *Z-normalized* data, Selego provides the best performance for all data sets/forecasting tasks considered. When considering *non-normalized* data, Selego is superior for 5 out of 6 tasks and only for the "Nasdaq Apple", Inf-FS provides better performance – note, however, Inf-FS performs poorly under *min-max normalization* and *Z-normalization* strategies for this data set. Note that also when considering *min-max non-normalized* data, Selego is superior most of the tasks: DTW is slightly better on NASDAQ index and Inf-FS is better on Fuel Consumption data, but neither consistently outperforms Selego. Instead, Selego proves to be highly robust across data sets and normalization strategies.

3) Impact of Variate Selection on Execution Times: In Tables VIII through X, we see the impact of variate selection on the overall computational complexity (due to space limitations, here we only include results for the building energy data set, the results for the other data sets are similar).

As, we see in Table VIII, as would be expected, variate selection tends to reduce the model training times. The results show that the gains are the most pronounced for the CNN and that Selego provides the highest training time gains. Interestingly, similarity based selection variate strategies (DTW and PCA) hurt the training time under LSTM, which indicates that, if not carried out properly, variate selection can negatively impact training performance.

Table X, then, presents the execution times for the variate selection process that preceeds model training. As we see here, except for DTW, the variate selection times are essentially negligible relative to the model training times reported in Table VIII – DTW takes $\sim$ 2200 seconds to compare 389 source variates to one target variate; i.e, $\sim$ 5.7 second on average per comparing a pair of variates. This indicates that Selego based variate selection, not only provides boosts on accuracy, but achieves this without any penalty on the overall time needed to prepare the data for model training: in fact, when using Selego, the end-to-end training time (including variate selection and model training) shows significant gains.

Finally, Table IX shows that the inference times also slightly improve under variate selection (especially when using Selego, with tight variate budget), but the gains are too slight to be meaningful in the considered application scenarios – though the gains might prove to be significant in other contexts.

V. Conclusions

In this paper, we introduced Selego framework for variate selection to support accurate time series prediction. Selego relies on three key observations: (a) *temporal alignments* among variates can be used to quantify the importance of the recorded variates with respect to a target variate, (b) yet, traditional time series similarity/distance functions, such as DTW, are fundamentally ill-suited for this purpose. Moreover, (c) when coupled with robust variate selection, even simple CNN-based models can potentially be more accurate than complex and costly recurrence based techniques (such as RNN and LSTM). Experiments using LSTM, RNN, and CNN, for different top-X% variates and different forecasting leads on multiple real-world datasets have shown that the proposed framework can offer significant $(90 - 98\%)$ drops in the number of variates and significantly boost the overall prediction accuracies.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Rakesh Agrawal, Christos Faloutsos, and Arun Swami. *Efficient similarity search in sequence databases*. Springer, 1993.

[3] Roy Assaf, Ioana Giurgiu, Frank Bagehorn, and Anika Schumann. Mtex-cnn: Multivariate time series explanations for predictions with convolutional neural networks. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 952–957. IEEE, 2019.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

[5] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *JMLR*, 2012.

TABLE VIII: Model training time (in seconds) for the Building Energy data set (lead time 5 hours)

		Model Training Time as a Function of the ratio of the Selected Variables (in seconds)														
		LSTM					RNN					CNN				
		2%	10%	50%	90%	100%	2%	10%	50%	90%	100%	2%	10%	50%	90%	100%
Building Energy Lead-5	DTW	111.6	112.1	117.3	122.0	109.0	31.0	33.2	35.2	42.4	68.0	8.1	10.5	18.8	23.0	42.1
	PCA	116.6	120.2	121.5	123.6		29.7	34.2	35.7	42.6		8.9	11.1	17.3	23.5	
	Inf-FS	98.7	102.0	103.8	108.2		30.4	33.2	35.7	40.8		9.4	10.0	18.5	23.0	
	Selego	98.5	99.3	104.6	107.9		29.0	35.2	36.3	41.0		8.0	10.2	14.6	23.7	

TABLE IX: Inference time (in seconds) for the Building Energy data set (lead time 5 hours)

		Inference Time as a Function of the ratio of the Selected Variables (in seconds)														
		LSTM					RNN					CNN				
		2%	10%	50%	90%	100%	2%	10%	50%	90%	100%	2%	10%	50%	90%	100%
Building Energy Lead-5	DTW	0.21	0.24	0.26	0.26	0.21	0.12	0.12	0.12	0.12	0.14	0.04	0.04	0.04	0.04	0.06
	PCA	0.24	0.21	0.21	0.24		0.14	0.14	0.13	0.13		0.06	0.04	0.04	0.06	
	Inf-FS	0.21	0.24	0.26	0.26		0.13	0.12	0.12	0.12		0.04	0.04	0.04	0.04	
	Selego	0.20	0.21	0.21	0.24		0.12	0.12	0.12	0.12		0.04	0.04	0.04	0.06	

TABLE X: Variate selection times (in seconds) for the Building energy data set (lead time 5 hours)

		Variate Selection Time (in seconds)		
		Event Extraction	Variate Ranking	Total Time
Building Energy	Selego	0.003	0.09	0.09
	PCA	N/A	0.42	0.42
	DTW	N/A	2216.83	2216.83
	Inf-FS	N/A	0.86	0.86

[6] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[7] Vincenzo Bianco, Oronzio Manca, and Sergio Nardini. Electricity consumption forecasting in italy using linear regression models. *Energy*, 2009.

[8] George E.P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. Time series analysis, forecasting and control, 1994.

[9] K Selçuk Candan, Rosaria Rossini, Maria Luisa Sapino, and Xiaolan Wang. sdtw: Computing dtw distances using locally relevant constraints based on salient feature alignments. *VLDB*, 2012.

[10] Lei Chen and Raymond Ng. On the marriage of lp-norms and edit distance. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 792–803. VLDB Endowment, 2004.

[11] François Chollet et al. keras, 2015.

[12] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *ICLR*, 2016.

[13] Harris Drucker, Christopher JC Burges, Linda Kaufman, Alex J Smola, and Vladimir Vapnik. Support vector regression machines. In *NIPS*, 1997.

[14] Zicheng Fang, Peng Wang, and Wei Wang. Efficient learning interpretable shapelets for accurate time series classification. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 497–508. IEEE, 2018.

[15] SM Fernandez-Fraga, MA Aceves-Fernandez, JC Pedraza-Ortega, and S Tovar-Arriaga. Feature extraction of eeg signal upon bci systems based on steady-state visual evoked potentials using the ant colony optimization algorithm. *Discrete Dynamics in Nature and Society*, 2018, 2018.

[16] Yifeng Gao and Jessica Lin. Discovering subdimensional motifs of different lengths in large-scale multivariate time series. *IEEE ICDM*, 2019.

[17] Yash Garg and K Selçuk Candan. Racknet: Robust allocation of convolutional kernels in neural networks for image classification. In *ICMR*, pages 315–323, 2019.

[18] Phil Goodwin, Joyce Dargay, and Mark Hanly. Elasticities of road traffic and fuel consumption with respect to price and income: a review. *Transport reviews*, 24(3):275–292, 2004.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997.

[21] Eamonn Keogh and Chotirat Ann Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and information systems*, 7(3):358–386, 2005.

[22] Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2002.

[23] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[24] Abdullah Mueen and Eamonn Keogh. Extracting optimal performance from dynamic time warping. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD 16, page 21292130, New York, NY, USA, 2016. Association for Computing Machinery.

[25] Karl Pearson. Principal components analysis. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 1901.

[26] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. *IJCAI*, 2017.

[27] Giorgio Roffo, Simone Melzi, and Marco Cristani. Infinite feature selection. In *ICCV*, 2015.

[28] Lior Rokach and Oded Z Maimon. *Data mining with decision trees: theory and applications*, volume 69. World scientific, 2008.

[29] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 1986.

[30] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49, 1978.

[31] Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1983.

[32] Hagit Shatkay and Stanley B Zdonik. Approximate queries and representations for large data sequences. In *Data Engineering, 1996. Proceedings of the Twelfth International Conference on*, pages 536–545. IEEE.

[33] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 1–9, 2015.

[34] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM 06, page 613622, USA, 2006. IEEE Computer Society.

[35] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. Residual attention network for image classification. In *CVPR*, 2017.

[36] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *ICLR*, 2017.