Learning Correlation Space for Time Series

Han Qiu
Massachusetts Institute of Technology
Cambridge, MA, USA
hanqiu@mit.edu

Hoang Thanh Lam
IBM Research
Dublin, Ireland
t.l.hoang@ie.ibm.com

Francesco Fusco
IBM Research
Dublin, Ireland
francfus@ie.ibm.com

Mathieu Sinn
IBM Research
Dublin, Ireland
mathsinn@ie.ibm.com

ABSTRACT

We propose an approximation algorithm for efficient correlation search in time series data. In our method, we use Fourier transform and neural network to embed time series into a low-dimensional Euclidean space. The given space is learned such that time series correlation can be effectively approximated from Euclidean distance between corresponding embedded vectors. Therefore, search for correlated time series can be done using an index in the embedding space for efficient nearest neighbor search. Our theoretical analysis illustrates that our method's accuracy can be guaranteed under certain regularity conditions. We further conduct experiments on real-world datasets and the results show that our method indeed outperforms the baseline solution. In particular, for approximation of correlation, our method reduces the approximation loss by a half in most test cases compared to the baseline solution. For top-k highest correlation search, our method improves the precision from 5% to 20% while the query time is similar to the baseline approach query time.

KEYWORDS

Time series, correlation search, Fourier transform, neural network

ACM Reference Format:
Han Qiu, Hoang Thanh Lam, Francesco Fusco, and Mathieu Sinn. 2018. Learning Correlation Space for Time Series. In *Proceedings of Conference on Knowledge Discovery and Data Mining (SIGKDD'18).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Given a massive number of time series, building a compact index of time series for efficient correlated time series search queries is an important research problem [2, 8]. The classic solutions [2, 27, 37] in the literature use Discrete Fourier Transformation (DFT) to transform time series into the frequency domain and approximate the correlation using only the first few coefficients of the frequency vectors. Indeed, people have shown that using only the first 5 coefficients of the DFT is enough to approximate the correlation among stock indices with high accuracy [37].

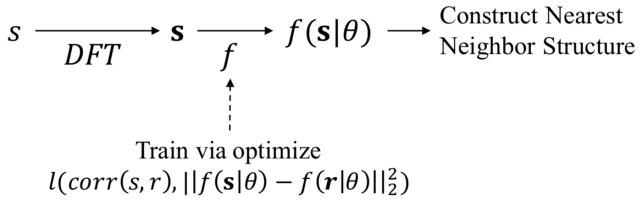

Figure 1: Solution Framework

Approximation of a time series using the first few coefficients of its Fourier transformation can be considered as a dimension-reduction method that maps long time series into a lower dimensional space with minimal loss of information. An advantage of such dimension-reduction approaches is that they are unsupervised methods and are independent from use-cases; therefore, they can be used for many types of search queries simultaneously. However, they might not be ideal for applications on particular use-cases where the index is designed just for a specific type of query.

In practice, depending on situations we may serve different types of search queries such as top-k highest correlation search [13], threshold-based (range) correlation search [2, 27] or even simple approximation of the correlation between any pair of time series. Different objective function might be needed to optimize performance of different query types. For instance, for the first two query types it is important to preserve the order of the correlation in the approximation, while for the third one it is more important to minimize the approximation error.

To overcome such problems, in this paper we propose a general framework to optimize query performance for specific combination of datasets and query types. The key idea behind this framework is sketched in Figure 1. Denote s as a time series in the set S and $\mathbf{s} = DFT(s)$ as the discrete Fourier transformation of s. We use a neural network $f(\cdot|\theta)$ with parameters θ to map $\mathbf{s}$ into a low-dimensional space vector $f(\mathbf{s}|\theta)$. This neural network tries to approximate the correlation between a pair of time series s and r using the Euclidean distance $\|f(\mathbf{s}|\theta) - f(\mathbf{r}|\theta)\|_2$ on the embedding space $f(S)$. We can then utilize classic nearest neighbor search algorithms

SIGKDD'18, August 2018, London, UK

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Notations	Meanings
s, r, u	time series
$\mathbf{s}$	DFT of s
$\bar{s}$	Mean value of s
σ_s	Standard deviation of s
$\hat{s}$	Normalized series of s
M	time series length
m	embedding size
k	selection set size
η	selection threshold
F	top-k selection set
θ	network parameters
$f(s\|\theta)$	neural network with parameter θ
ρ	query precision
δ	query approximation gap
$\|s-r\|_2$	Euclidean distance between s and r
$d_m(\mathbf{s},\mathbf{r})$	Euclidean distance between $\mathbf{s}$ and $\mathbf{r}$ considering only the first m elements

Table 1: Notations

on this embedding space to index time series and improve query efficiency.

We also notice that specific loss function is needed in training the network to optimize performance of specific type of query. In experiments, we will see that using the right loss function for a specific type of query is crucial in achieveving good results. Therefore, in this paper we also propose relevant loss functions for all aforementioned search query types.

In summary, the contributions of this work are listed as follows:

- we propose a framework to approximate time series correlation using a time series embedding method;
- we do theoretical analysis to show non-optimality of the baseline approach from which it motivates this work. We proposed appropriate loss function for each type of query and provide theoretical guarantee for these approaches;
- we conduct experiments on real-world datasets to show the effectiveness of our approach compared to the classic solutions [2]: our approach reduces the approximation error at least by a half in the experiments, and improves the precision for top-k correlation search from 5% to 20%;
- we open the source code for reproducibility of our research[1].

The rest of the paper is organized as follows. We discuss background of the paper in section 2. Three important problems are formulated in section 3. In section 4 we show the non-optimality of the baseline approach from which it motivates this work. We also show theoretical guarantee of proposed loss functions for each problem formulated in section 3. Sections 5 and 6 show how to build and train the embedding neural networks. Experimental results are discussed in section 8 and section 9 concludes the paper.

2 BACKGROUND

Let $s = [s_1, s_2, \cdots, s_M], r = [r_1, r_2, \cdots, r_M]$ be two time series. The Pearson correlation between r and s is defined as:

$$corr(s,r) = \frac{\frac{1}{M}\sum_{j=1}^{M} s_j r_j - \bar{s}\bar{r}}{\sigma_s \sigma_r}, \tag{1}$$

where $\bar{s} = \sum_{j=1}^{M} s_j/M$ and $\bar{r} = \sum_{j=1}^{M} r_j/M$ are the mean values of s and r, while $\sigma_s = \sqrt{\sum_{j=1}^{M}(s_j-\bar{s})^2/M}$ and $\sigma_r = \sqrt{\sum_{j=1}^{M}(r_j-\bar{r})^2/M}$ are the standard deviations. If we define the l_2-normalized vector of the time series s, $\hat{s} = [\hat{s}_1, \hat{s}_2, \cdots, \hat{s}_M]$, as:

$$\hat{s}_j = \frac{1}{\sqrt{M}}\frac{s_j - \bar{s}}{\sigma_s} = \frac{s_j - \bar{s}}{\sum_{j=1}^{M}(s_j-\bar{s})^2}, \tag{2}$$

we can reduce $corr(s,r)$ as $\hat{s}^T\hat{r} = 1 - \|\hat{s}-\hat{r}\|_2^2/2$. Therefore, in following discussion we always assume a time series s to be equal to its l_2-normalized version $\hat{s}$.

The (scaled) discrete Fourier transformation of the time series s, $\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_M]$, is defined as:

$$\mathbf{s}_j = \frac{1}{\sqrt{M}}\sum_{l=0}^{M-1} s_l e^{\frac{-2jl\pi i}{M}} \tag{3}$$

where $i^2 = -1$. By the Parseval's theorem [2, 29], we have $s^T r = \mathbf{s}^T\mathbf{r}^*$ and $\|s\|_2 = \|\mathbf{s}\|_2$. We can then recover the following property in the literature [27, 37]:

- if $corr(s,r) = s^T r > 1 - \varepsilon^2/2$ then $d_m(\mathbf{s},\mathbf{r}) < \varepsilon$, where d_m is the Euclidean distance mapped on the first m dimensions with $m < \frac{M}{2}$

$$d_m^2([\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_M], [\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_M]) = \sum_{j=1}^{m} \|\mathbf{s}_j - \mathbf{r}_j\|_2^2. \tag{4}$$

Based on this property, authors in [2] develop a method to search for highest correlated time series in a database utilizing $d_m^2(\mathbf{s},\mathbf{r}) \approx 2 - 2\cdot corr(s,r)$. This method is considered as the baseline solution in this paper.

3 PROBLEM FORMULATION

Assume we are given a dataset of (regular) time series $S \in R^M$, where $M \in N^+$ is a large number. Some correlation search problems can be formulated as follows:

Problem 1 (Top-k Correlation Search)**.** *Find a method F that for any given time series $s \in S$ and count $k \in N^+$, $F(s,k|S)$ provides a subset of S that includes the k highest correlated time series with respect to s. More formally,*

$$\begin{aligned} F(s,k|S) &= \arg\max_{\{S'|S'\subset S, |S'|=k\}} \sum_{r\in S'} corr(s,r) \\ &= \arg\min_{\{S'|S'\subset S, |S'|=k\}} \sum_{r\in S'} \|s-r\|_2^2 \end{aligned} \tag{5}$$

Problem 2 (Threshold Correlation Search)**.** *Find a method G that for any given time series $s \in S$ and threshold value $\eta \in (0,1)$, $G(s,\eta|S)$ provides a subset of S that consists of all time series r with $corr(s,r) \geq \eta$. More formally,*

$$G(s,\delta|S) = \{r \in S | corr(s,r) \geq \eta\} \tag{6}$$

[1] See the KDD 2018 publication of this paper for the link to the source codes.

Problem 3 (Correlation Approximation). *Find an embedding function $f : R^M \to R^m$ that minimizes the expected approximation error in correlations*

$$f(\cdot|S) = \arg\min_f E_{s,r\in S}|\|f(s) - f(r)\|_2^2 - 2(1 - corr(s,r))| \quad (7)$$

Solutions of these problems are strongly correlated. First, Problem 1 and Problem 2 are almost equivalent. For instance, if the set $\{corr(s,r)|r \in S\}$ contains no identical elements, and if we let $\eta(s,k|S)$ to be the kth largest element in this set, we will have $G(s,\eta(s,k|S)|S) = F(s,k|S)$. Therefore, we only need to discuss one of them; in this paper we will focus on Problem 1.

Second, solutions of Problem 3 can lead to good approximation for Problem 1. If there is an f such that the objective value of Problem 3 is close to 0, or

$$\|f(s) - f(r)\|_2^2 \approx 2(1 - corr(s,r)),$$

we can consider the following approximation of Problem 1

$$\hat{F}(s,k|f,S) = \arg\min_{\{S'|S'\subset S,|S'|=k\}} \sum_{r\in S'} \|f(s) - f(r)\|_2^2 \quad (8)$$

combined with top-k nearest neighbor search structures on the embedding space $f(S)$. If the dimension m of the embedded space is small, we can simply use k-d tree [4]. With a balanced k-d tree, the complexity of searching the top k nearest neighbors among n candidates reduces to $O(k \log n)$ [17].

All problems defined above have several important applications. In time series forecasting, one can search for strongly correlated time series in historical database and then use them to predict the target time series. In data storage, when the whole time series dataset is very large and does not fit available memory, one can compress the time series significantly using the embedding with small trade-off for loss due to correlation approximation.

4 THEORETICAL ANALYSIS

This section discusses the theoretical analysis to support our methods. We first show that DFT is not an optimal solution for Problem 1 and Problem 3 in general. This result motivates our work as there is room for improvement over the baseline if data has specific structure that our learning algorithm can leverage. Next we show that in order to solve Problem 1, it suffices to approximate the l_2 norm on S with l_2 norm on $f(S)$. We then propose an appropriate loss function to solve Problem 1 effectively.

4.1 Fourier transform's non-optimality

In the literature [37], it has been shown that using only the first 5 DFT coefficients, it is enough to approximate solutions for Problem 1 on stocks data with high accuracy. However, there is still much room for improvement over the methods based on DFT as shown in the experimental section. This is because DFT approximations blindly assumes that most of the energy (information) in the time series is in the low-frequency components. In the two following examples, we illustrate that the naive DFT method might neither accurately approximate the correlation function nor extract the most important information explaining dataset variability, if the information in the high-frequency components of the time series is non-negligible. Use-case-specific dimension reduction methods such as neural network can avoid such bias and therefore are more preferable.

Example 1. *Consider a set S of real-valued time series s, where the corresponding DFT vector $\mathbf{s}$ of each s has following property*

$$\mathbf{s}_i = \mathbf{s}_{M/4+i},\ \forall i = 1,\cdots,M/4.$$

Recall that for DFT vector $\mathbf{s}$ of real-valued time series s, we have conjugacy $\mathbf{s}_i = \mathbf{s}^*_{M+1-i}$ for $i = 1,\cdots,M$. That implies $4d_m^2(\mathbf{s},\mathbf{r}) = \|\mathbf{s}-\mathbf{r}\|_2^2 = 2 - 2corr(s,r)$ for $m = M/4$. The optimal embedding with size m for the correlation function approximation should therefore be $f(\mathbf{s}) = [2\mathbf{s}_1, 2\mathbf{s}_2, \cdots, 2\mathbf{s}_m]$ which differs from the embedding using DFT a constant factor of 2.

Example 2. *Consider a set S of similar time series s, where the corresponding DFT vector $\mathbf{s}$ of each s is generated as follows:*

$$\mathbf{s}_i = \mu_i^0 + z_i\sigma_i^0,\ \forall i = 1,\cdots,M$$
$$z_i \sim \begin{cases} N(0,\varepsilon) & \forall i = 1,\cdots,m \\ N(0,1) & \forall i = m+1,\cdots,M \end{cases} \quad (9)$$

where $\varepsilon \ll 1$, $\sigma_i^0 \neq 0$, and z_i are sampled independently. It is obvious that for Problem 1, DFT method has almost same precision as random guess.

4.2 Approximating top-k correlated series

In this subsection, we analyze the theoretical performance guarantee of approximation (8). Using notation $F(s,k|S)$ and $\hat{F}(s,k|f,S)$, Problem 1 is equivalent to maximize the precision ρ on f given s and k

$$\rho(s,k,f|S) = \frac{1}{k}|\hat{F}(s,k|f,S) \bigcap F(s,k|S)|, \quad (10)$$

or to minimize the approximation gap δ on f

$$\delta(s,k,f|S) = \frac{1}{k}[\sum_{r\in\hat{F}(s,k|f,S)} \|s-r\|_2^2 - \sum_{r\in F(s,k|S)} \|s-r\|_2^2]. \quad (11)$$

If the mapping f well preserves the metric, that is, the gap between $\|s-r\|_2^2$ and $\|f(s)-f(r)\|_2^2$ is uniformly small, we can prove that the approximation gap δ is also quite small. Formally, we have:

Theorem 4.1. *If for a function $f : R^M \to R^m$, we have*

$$|\|f(s) - f(r)\|_2^2 - \|s-r\|_2^2| \le \varepsilon,\ \forall s,r \in S,$$

where $\varepsilon \in R^+$, then we also have

$$\delta(s,k,f|S) \le 2\varepsilon,\ \forall s \in S, k \in N^+.$$

Proof. In fact, we have

$$\begin{aligned} &\sum_{r\in\hat{F}(s,k|f,S)} \|s-r\|_2^2 \\ \le &\sum_{r\in\hat{F}(s,k|f,S)} \|f(s)-f(r)\|_2^2 + k\varepsilon \\ \le &\sum_{r\in F(s,k|S)} \|f(s)-f(r)\|_2^2 + k\varepsilon \\ \le &\sum_{r\in F(s,k|S)} \|s-r\|_2^2 + 2k\varepsilon \end{aligned} \quad (12)$$

holds for every $s \in S$ and $k \in N^+$. The second inequality above holds since $\hat{F}(s,k|f,S)$ solves the corresponding minimization problem (8). The conclusion then follows trivially. □

Theorem 4.1 provides us a sanity check that approximation (8) becomes more accurate (in terms of δ) as function approximation f gets better. We can therefore design the following loss function on parametric mapping $f(\cdot|\theta)$ for optimization

$$L_\delta(\theta) = E_{s,r\in S}|\|f(s|\theta) - f(r|\theta)\|_2^2 - \|s-r\|_2^2|. \tag{13}$$

Theorem 4.1 can also be extended for more general distance $d(\cdot,\cdot)$ on $R^m \times R^m$.

Theorem 4.2. *Consider distance $d(\cdot,\cdot)$ on $R^m \times R^m$ and corresponding set approximation*

$$\hat{F}_d(s,k|f,S) = \arg\min_{\{S'|S'\subset S,|S'|=k\}} \sum_{r\in S'} d(f(s),f(r)). \tag{14}$$

If for a function $f: R^M \to R^m$, we have

$$|d(f(s),f(r)) - \|s-r\|_2^2| \le \varepsilon,\ \forall s,r \in S,$$

where $\varepsilon \in R^+$, then we also have

$$\delta_d(s,k,f|S) \le 2\varepsilon,\ \forall s \in S, k \in N^+,$$

where δ_d is defined as

$$\delta_d(s,k,f|S) = \frac{1}{k}[\sum_{r\in\hat{F}_d(s,k|f,S)} \|s-r\|_2^2 - \sum_{r\in F(s,k|S)} \|s-r\|_2^2]. \tag{15}$$

Proof. This theorem can be proved by proceeding with the same argument as in the proof of Theorem 4.1. □

Although we have confidence that a good metric-approximating f can lead to low approximation gap δ, there is no much to say about the precision ρ. For instance, assume we are given a group of time series S and a target series s, and for each $r \in S$ the correlation $corr(s,r)$ is close to others. Or, we can assume $|corr(s,r) - c_0| < \varepsilon$ where c_0, ε are fixed values. In this case, an f with low approximation error $|\|f(s)-f(r)\|_2^2 - \|s-r\|_2^2|$, and therefore low gap δ, might still confuse the relative order of the correlation $corr(s,r)$ and distance $\|s-r\|_2^2|$. This will result in a low precision ρ. To improve practical approximation performance on ρ for Problem 1, we propose a loss function that tries to approximate correlation order

$$\begin{aligned} L_\rho(\theta) = E_{s,r,u\in S}|(\|f(r)-f(s)\|_2^2 - \|f(r)-f(u)\|_2^2) \\ - (\|r-s\|_2^2 - \|r-u\|_2^2)|. \end{aligned} \tag{16}$$

The following corollary from Theorem 4.1 shows that we can also obtain performance guarantee on δ for network optimizing L_ρ.

Corollary 4.3. *If for a function $f: R^M \to R^m$, we have*

$$\begin{aligned} |(\|f(r)-f(s)\|_2^2 - \|f(r)-f(u)\|_2^2) \\ -(\|r-s\|_2^2 - \|r-u\|_2^2)| \le \varepsilon,\ \forall r,s,u \in S, \end{aligned}$$

where $\varepsilon \in R^+$, then we also have

$$\delta(s,k,f|S) \le 2\varepsilon,\ \forall s \in S, k \in N^+.$$

Proof. This reduce to Theorem 4.1 if we set $r = u$. □

At the end of this section, we notice that the uniform property on $\epsilon_{s,r} = \|s-r\|_2^2 - d(f(s),f(r))$ might be too restrictive, while in general neural network can only attain some probabilistic bounds. Since we have

$$\begin{aligned} \delta_d(s,k,f|S) &\le \frac{1}{k}[\sum_{r\in\hat{F}_d(s,k|f,S)} \epsilon_{s,r} - \sum_{r\in F(s,k|S)} \epsilon_{s,r}] \\ &\le \frac{1}{k}|\sum_{r\in\hat{F}_d(s,k|f,S)} \epsilon_{s,r}| + \frac{1}{k}|\sum_{r\in F(s,k|S)} \epsilon_{s,r}|, \end{aligned} \tag{17}$$

we can apply concentration inequalities to obtain following probabilistic bound

Theorem 4.4. *Assume for any given s, $\epsilon_{s,r}$ are independently and identically distributed zero-mean random variables on R. If we further assume*

$$E\epsilon_{s,r}^2 \le \varepsilon, |\epsilon_{s,r}| \le M\ a.s.,\ \forall r \in S,$$

we have

$$Pr(\delta(s,k,f|S) \ge 2c\varepsilon) \le 2\exp(\frac{-kc^2\varepsilon}{2+\frac{2}{3}Mc}),\ \forall c \ge 1.$$

Proof. We can simply apply the Bernstein inequality on each part of the right hand side of (17). □

Such probabilistic bound is less ideal than the deterministic bound in Theorem 4.1. It is more effective for accuracy of threshold-based queries since the corresponding k will be much greater than in the top-k queries and the bound will be much tighter.

5 DESIGN OF EMBEDDING STRUCTURE

In last section, we show that if we have an accurate embedding function f, the approximation of our correlation search will also be accurate. Nevertheless, it is not easy to find such an f. In this section we will discuss several potential neural network structures that we considered.

5.1 Embedding on the time-domain

For direct embedding of time series, a natural selection is the Recurrent neural network (RNN). RNN is a suitable model for sequential data, and recently it has been widely applied in sequence-related tasks such as machine translation [3] and video processing [15]. Specifically, given a sequence $[X_1,\cdots,X_M]$, a (single-layer) RNN tries to encode first m observations $[X_1,\cdots,X_m]$ into state h_m with the following dynamic processes

$$h_{m+1} = g(X_{m+1}, h_m) \tag{18}$$

where g is the same function across m and is called a "unit". Common choices of RNN units include Long-short term memory (LSTM) [19] and Gate recurrent unit (GRU) [11]. We can then take the last state h_M as the final embedding vector

$$f(s) = h_M. \tag{19}$$

This approach has a disadvantage that the training algorithm is very slow when the time series is long because of the recurrent computations. Moreover, in the experiments we observed that this method does not give good results for long time series.

5.2 Embedding on the frequency-domain

To avoid the time-consuming and ineffective recurrent computations, we first use DFT to transform a time series s into a frequency domain $\mathbf{s} = DFT(s)$. Next, we simply use a multi-layer fully connected (dense) network with ReLU activation functions

$$\text{ReLU}(x) = \max\{x, 0\}$$

to transform the frequency vector $\mathbf{s}$ into an embedded vector with a desirable size. At last, we applied l_2 normalization to the embedding vector to project it on a unit sphere. This normalization step realign the scale of the Euclidean distances between embedding vectors with the correlation we aim to approximate.

6 TRAINING EMBEDDING NEURAL NETWORKS

In this section we describe how we train the embedding neural networks for different objective functions.

6.1 Approximation of correlation

For Problem 3, we sample a random batch of pair of time series (s, r) and minimize the following loss function at each training iteration:

$$L_{approximate} = |\|f(\mathbf{s}|\theta) - f(\mathbf{r}|\theta)\|_2^2 - 2(1 - corr(s, r))|.$$

We call this method *Chronos*[2] *Approximation* algorithm in the experiments.

6.2 Top-k correlation search

For Problem 1, we sample a random batch of tuple of time series (s, r, u) and minimize the following loss function at each training iteration:

$$\begin{aligned} L_{order} =&|(\|f(\mathbf{r}|\theta) - f(\mathbf{s}|\theta)\|_2^2 - \|f(\mathbf{r}|\theta) - f(\mathbf{u}|\theta)\|_2^2) \\ &- 2(corr(r, u) - corr(r, s))| \end{aligned}$$

We call this method *Chronos Order* algorithm in the experiments because it tries to approximate the correlation order. According to Corrolary 4.3, if we optimize L_{order} we also approximate the solutions for Problem 1.

6.3 Symmetry correction

Recall that when s is a real number time series, its DFT $\mathbf{s}$ is symmetric, i.e. $\mathbf{s}_i$ is a complex conjugate of $\mathbf{s}_{M-i}$. Because of this reason, if $M = 2 * m$, we should have $d_M(\mathbf{s}, \mathbf{r}) = 2 * d_m(\mathbf{s}, \mathbf{r}) = 2 - 2corr(s, r)$. Thanks to the symmetry property, people only need to use $m < \frac{M}{2}$ coefficients to approximate the correlation. In such case the approximation of $2 - 2corr(s, r)$ is corrected to $2 * d_m(\mathbf{s}, \mathbf{r})$.

Therefore, in our implementation of the Chronos algorithms we use a corrected version of the loss functions as follows to take into account the symmetry of the embedding:

$$L_{approximate} = |2\|f(\mathbf{s}|\theta) - f(\mathbf{r}|\theta)\|_2^2 - 2(1 - corr(s, r))|.$$

$$\begin{aligned} L_{order} =&|2(\|f(\mathbf{r}|\theta) - f(\mathbf{s}|\theta)\|_2^2 - \|f(\mathbf{r}|\theta) - f(\mathbf{u}|\theta)\|_2^2) \\ &- 2(corr(r, u) - corr(r, s))| \end{aligned}$$

[2]Chronos in Greek means time.

7 RELATED WORKS

Correlation related queries on time series. During the past several decades, people investigated a massive amount of correlation-related query problems on time series datasets in the literature. As our review will be far from exhaustive, here we rather provide several problem categories and some corresponding examples.

One major classification on these problems is whether a specific time series is given in the query. For example, some works [25, 27, 31, 37] are interested in computing correlation among all sequence pairs or providing all one that are correlated (over a certain threshold), while others [16, 28] aims at finding most correlated sequences in the dataset for the given one. There are also some papers addressing both sides [2]. In this paper, we focus on the later category that a specific time series is assumed to be given in the query.

Another major distinction is whether the object they consider is the whole sequence or a subsequence. For problems focusing on whole sequence [2, 27, 28, 37], not only the query time series but also all sequences from the dataset are assume to have (almost) the same length. Our problems also make this assumption. For problems concerning subsequences [16], the query time series is generally much shorter than those in the dataset, which are also assumed to have arbitrary lengths.

Finally, an important distinction is whether the data is provided offline or collected online. On one hand, problems in the online setting [31, 37] typically address the varying nature in data streams and discover stationary submodule in the computation procedure to boost query efficiency. On the other hand, problems in the offline settings [27, 28], including ours, usually deal with the difficulties brought by the massive data size.

Dimension reduction. As far as we know, we are the first to suggest using function approximation, rather than algorithms for feature extraction, for dimension reduction in time series indexing and correlation searching problems. The application of dimension reduction techniques to improve time series correlation search efficiency is first introduced by Agrawal *et al.* [2]. DFT and discrete wavelet transform (DWT) are the two most commonly used methods. The usage of DFT was introduced by Agrawal *et al.* [2], who also constructed R^*-tree for indexing on the transformed Fourier coefficients. Chan and Fu [8], on the other hand, were the first to use wavelet analysis for time series embedding. Regardless of the significant difference between DFT and DWT approaches, it is shown that they have similar performance on query precision in a later paper by Wu *et al.* [34]. Therefore, in this paper we only consider DFT as our baseline method.

We also notice that since Agrawal *et al.* [2] researchers suggested many different dimensionality reduction techniques to provide tighter approximation bound for pruning. To name a few, there are Adaptive Piecewise Constant Approximation (APCA) [21], Piecewise Aggregate Approximation (PAA) [20], multi-scale segment mean (MSM) [26], Chebyshev Polynomials (CP) [7], and Piecewise Linear Approximation (PLA) [9]. However, according to the comparison work by Ding *et al.* [14], none of these significantly outperform DFT and DWT.

Similarity measures. Though simple, Pearson correlation and Euclidean distance (l_2-norm) are still the most commonly used

similarity measure in the literature. We also focus on this measure in this paper. However, Euclidean distance suffers from drawbacks such as fixed time series length and its applications in general are much restricted. Therefore, in many applications, more flexible similarity measure such as Dynamic time warping (DTW) [30] is more appropriate. DTW is an algorithm for similarity computation between general time series and can deal with time series with varying lengths and local time shifting. One of the major drawbacks of DTW is its computation complexity; therefore, most related research on DTW develop and discuss its scalable application in massive datasets [5, 22, 28, 35].

There are also other similarity measures such as Longest Common Subsequence (LCSS) [6, 33], but rather than concerning generalizability, they either focus on a specific type of data or problem related to the l_2-norm and DTW. They therefore did not attract much attention in the past decade. Interested readers are again referred to the comparison work by [14] for more details on similarity measures.

Neural network approximation. With the recent successes in deep learning, neural networks again become the top choice for function approximators. Recent work by Kraska *et al.* [24] further illustrates that neural networks, with proper training, can beat baseline database indexes in terms of speed and space requirement.

There are also more neural-network-based methods in time series related problems. Cui *et al.* [12] design an multi-scale convolution neural network for time series classification. Zheng *et al.* [36] apply convolution neural network for distance metric learning and classification. Both works show that the designed structures can achieve baseline performance when data is sufficiently large. However, in these works the networks are supervised to learn the mapping from time series to the given labels, while in this paper the network is developed for unsupervised dimension reduction and feature extraction of dataset structure.

8 EXPERIMENTS

In this section, we discuss the experiment results. The DFT method is considered as the baseline and compared with *Chronos approximation* and *Chronos order* methods in several different types of queries. We first introduce the datasets and experimental settings.

8.1 Datasets

We used four real-world datasets to validate our approaches. One of these datasets is the daily stock indices crawled from the Yahoo finance [1] using public python API. The other datasets are chosen from the UCR time series classification datasets [10]. These datasets were chosen because each time series is long (at least 400 readings) and the number of time series is large enough (at least a few thousands). Overall the characteristics of these datasets are described in Table 2.

Data	# time series	length	size
Stock	19420	1000	400 MB
Yoga	3300	426	26 MB
UWaveGesture	4478	945	71 MB
StarLightCurves	9236	1024	181 MB

Table 2: Datasets used in experiments

8.2 Experiment settings

For each dataset, we randomly split the data into training, validation and test with the ratios 80-10-10%. We use the training set to train the embedding neural networks and validate it on the validation set. The test set is used to estimate the query accuracy. In particular, for the top-k highest correlation queries, each time series in the test set is considered as a target series, and all the other series in the training set is considered as the database from which we look for correlated time series to the target. For correlation approximation queries, we randomly permute the test series and create pairs of time series from the permutation. Since the test set is completely independent from training and validation, the reported results are the proper estimate of the query accuracy in practice.

We fix parameters of the neural network to the default values described in Table 3. Although we didn't fine-tune the network hyper-parameters and explore deeper neural network structures, the obtained results are already significantly better than the baseline algorithms. In practice, automatic network search and tuning can be done via Bayesian optimization [32]. Since search for the optimal network structure and optimal network hyper-parameters is out of the scope of this work, we leave this problem as future work.

Table 3: Parameter settings

parameter	value
optimization algorithm	ADAM [23]
learning rate	0.01
weight initialization	Xavier [18]
number of hidden layers	1
hidden layer size	1024
mini-batch size	256
training iterations	10000 mini-batches

The network was implemented in TensorFlow and run in a Linux system with two Intel cores, one NVDIA Tesla K40 GPU. All the running time is reported in the given system.

8.3 Results on correlation approximation

Figure 2 shows the comparison between the *Chronos approximation* and DFT for approximated solutions to Problem 3. Each subplot in the figure corresponds to a dataset. The vertical axis shows the approximation loss in the test set, while the horizontal axis shows the embedding size m, varying from 2 to 16. As can be observed, our method outperformed DFT in all the test cases with a significant margin. In particular, the approximation error was reduced at least by a half, especially for small embedding size. These empirical results confirm our theoretical analysis on the sub-optimality of DFT for Problem 3.

8.4 Results on top-k correlation search

Figure 2 shows the comparison between the *Chronos order*, the *Chronos approximation* and DFT for approximated solutions to Problem 1. Each subplot in the figure corresponds to a pair of

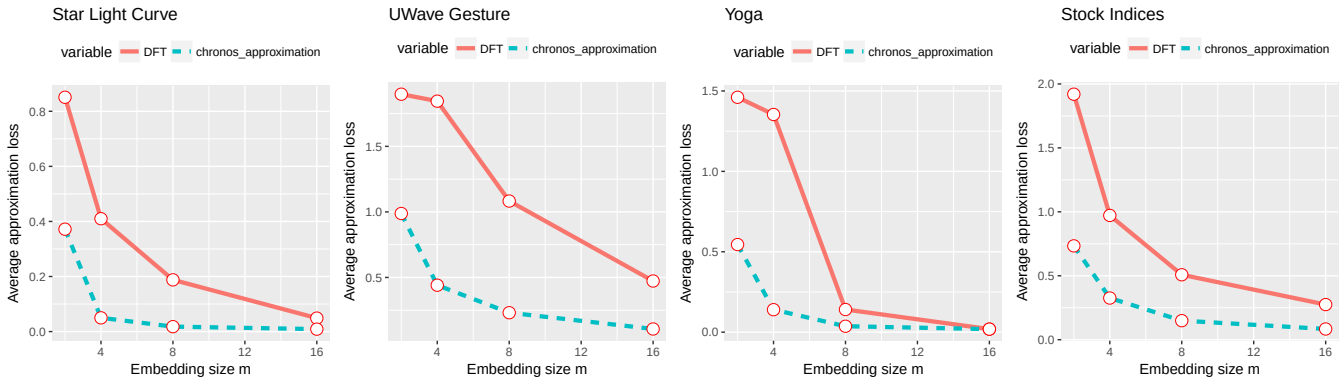

Figure 2: Results on correlation approximation

Data	*Chronos order*
Stock	947
Yoga	914
UWaveGesture	894
StarLightCurves	950

Table 4: Training time (seconds) with 10000 mini-batches

Data	*Chronos order*	DFT
Stock	1.00	1.10
Yoga	0.15	0.16
UWaveGesture	0.24	0.23
StarLightCurves	0.53	0.49

Table 5: Average top-k query time (seconds)

dataset and query size k (varied from 10 to 100). The vertical axis shows the precision at k in the test set, while the horizontal axis shows the embedding size m, varying from 2 to 16.

Although Theorem 4.1 and Corollary 4.3 provide the same theoretically guarantee for the approximation bound of *Chronos order* and *Chronos order* on Problem 1, our results show that the *Chronos order* outperforms the *Chronos order* in terms of the precision. These algorithms differ only in the loss function in training the networks: while *Chronos approximation* directly minimizes the loss of correlation approximation, the *Chronos order* also tries to preserve order information. This result implies that using the right loss function for a given query type is crucial in achieving good results.

In all cases, the *Chronos order* algorithm outperforms the baseline algorithm. On the other hand, except for the Stock dataset, *Chronos approximation* algorithm outperforms the baseline algorithm. The overall improvement varies from 5% to 20% depending on the value of k and m.

8.5 Training & Query time

Another important evaluation metric on our methods' performance is the time efficiency in training the embedding network $f(\mathbf{s}|\theta)$ and in subsequent function evaluation for incoming queries. Table 4 shows the training time of the *Chronos Order* method in four datasets when $k = 100, m = 16$. For cases with other values of k and m, the results are similar. We can see that it only takes about 15 minutes to train the network with 10000 mini-batches.

Table 5 shows the query time of different algorithms when $k = 100, m = 16$. Again, for cases with other values of k and m, the results are similar. As can be seen, the queries time is not influenced by the additional tasks for evaluating the function $f(\mathbf{s}|\theta)$. Since the queries time is dominated by the k-d tree traversal, time for evaluating the function $f(\mathbf{s}|\theta)$ is negligible.

9 CONCLUSIONS AND FUTURE WORK

In this paper, we propose a general approximation framework for a variety of correlation search queries in time series datasets. In this framework, we use Fourier transform and neural network to embed time series into low-dimensional Euclidean space, and construct nearest neighbor search structures in the embedding space for time series indexing to speed up queries. Our theoretical analysis illustrates that our method's accuracy can be guaranteed under certain regularity conditions, and our experiment results on real datasets indicate the superiority of our method over the baseline solution.

Several observations in this work are interesting and require more attention. First, the approximation accuracy of our method varies significantly across datasets. Therefore, it is crucial in later real-world applications to more systematically evaluate the applicability of our method and design network structure for specific datasets. Second, the selection of distance function d can be important in improving the performance of our embedding method, since it might be able to balance between the approximation capability of the neural network and the internal similarity within the datasets.

Our work can be further extended in several directions. First, our method can only deal with time series with equal length. However, a large number of real-world time series are sampled with irregular time frequency or during arbitrary time period. Therefore, embedding methods that can deal with general time series should be explored in the future. Second, currently we only consider Pearson correlation as the evaluation metric for similarity. We can investigate the applicability of our framework on other types of

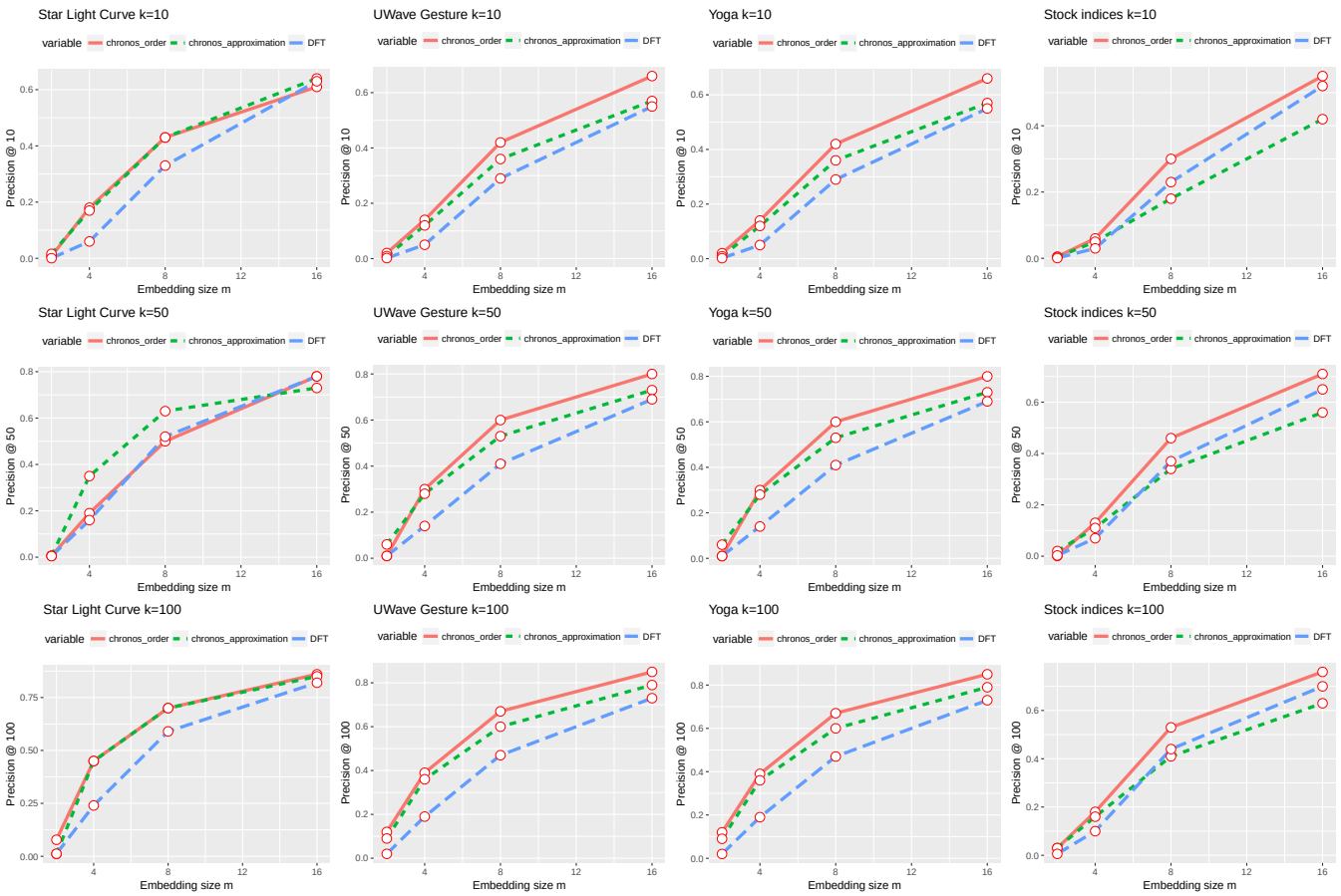

Figure 3: Results on top-k correlation search

similarity measure such as the dynamic time wrapping (DTW) measure. Finally, in some of the aforementioned applications such as time series forecasting, correlation search for the target time series might not provide the most essential information in improving the ultimate objectives. End-to-end systems explicitly connecting correlation search techniques and application needs might therefore be more straightforward and effective.

10 ACKNOWLEDGEMENTS

We would like to thank Dr. Tran Ngoc Minh and Dr. Martin Wistuba for useful discussion during the development of the project. This research was done while the first author was a research intern student at IBM research.

REFERENCES

[1] Yahoo finance. https://finance.yahoo.com/. Accessed: 2017-06-30.
[2] Agrawal, R., Faloutsos, C., and Swami, A. Efficient similarity search in sequence databases. In *International Conference on Foundations of Data Organization and Algorithms* (1993), Springer, pp. 69–84.
[3] Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[4] Bentley, J. L. Multidimensional binary search trees used for associative searching. *Communications of the ACM 18*, 9 (1975), 509–517.
[5] Berndt, D. J., and Clifford, J. Using dynamic time warping to find patterns in time series. In *KDD workshop* (1994), vol. 10, Seattle, WA, pp. 359–370.
[6] Boreczky, J. S., and Rowe, L. A. Comparison of video shot boundary detection techniques. *Journal of Electronic Imaging 5*, 2 (1996), 122–129.
[7] Cai, Y., and Ng, R. Indexing spatio-temporal trajectories with chebyshev polynomials. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data* (2004), ACM, pp. 599–610.
[8] Chan, K.-P., and Fu, A. W.-C. Efficient time series matching by wavelets. In *Data Engineering, 1999. Proceedings., 15th International Conference on* (1999), IEEE, pp. 126–133.
[9] Chen, Q., Chen, L., Lian, X., Liu, Y., and Yu, J. X. Indexable pla for efficient similarity search. In *Proceedings of the 33rd international conference on Very large data bases* (2007), VLDB Endowment, pp. 435–446.
[10] Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., and Batista, G. The ucr time series classification archive, July 2015. https://www.cs.ucr.edu/~eamonn/time_series_data/.
[11] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[12] Cui, Z., Chen, W., and Chen, Y. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995* (2016).
[13] Dallachiesa, M., Palpanas, T., and Ilyas, I. F. Top-k nearest neighbor search in uncertain data series. *Proceedings of the VLDB Endowment 8*, 1 (2014), 13–24.
[14] Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment 1*, 2 (2008), 1542–1552.
[15] Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), pp. 2625–2634.

[16] Faloutsos, C., Ranganathan, M., and Manolopoulos, Y. *Fast subsequence matching in time-series databases*, vol. 23. ACM, 1994.

[17] Friedman, J. H., Bentley, J. L., and Finkel, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS) 3*, 3 (1977), 209–226.

[18] Glorot, X., and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 249–256.

[19] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[20] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems 3*, 3 (2001), 263–286.

[21] Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Sigmod Record 30*, 2 (2001), 151–162.

[22] Keogh, E., and Ratanamahatana, C. A. Exact indexing of dynamic time warping. *Knowledge and information systems 7*, 3 (2005), 358–386.

[23] Kingma, D., and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[24] Kraska, T., Beutel, A., Chi, E. H., Dean, J., and Polyzotis, N. The case for learned index structures. *arXiv preprint arXiv:1712.01208* (2017).

[25] Li, Y., Yiu, M. L., and Gong, Z. Discovering longest-lasting correlation in sequence databases. *Proceedings of the VLDB Endowment 6*, 14 (2013), 1666–1677.

[26] Lian, X., Chen, L., Yu, J. X., Han, J., and Ma, J. Multiscale representations for fast pattern matching in stream time series. *IEEE transactions on knowledge and data engineering 21*, 4 (2009), 568–581.

[27] Mueen, A., Nath, S., and Liu, J. Fast approximate correlation for massive time-series data. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data* (2010), ACM, pp. 171–182.

[28] Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., and Keogh, E. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (2012), ACM, pp. 262–270.

[29] Rudin, W., et al. *Principles of mathematical analysis*, vol. 3. McGraw-hill New York, 1964.

[30] Sakoe, H., and Chiba, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing 26*, 1 (1978), 43–49.

[31] Sakurai, Y., Papadimitriou, S., and Faloutsos, C. Braid: Stream mining through group lag correlations. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data* (2005), ACM, pp. 599–610.

[32] Snoek, J., Larochelle, H., and Adams, R. P. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems* (2012), pp. 2951–2959.

[33] Vlachos, M., Kollios, G., and Gunopulos, D. Discovering similar multidimensional trajectories. In *Data Engineering, 2002. Proceedings. 18th International Conference on* (2002), IEEE, pp. 673–684.

[34] Wu, Y.-L., Agrawal, D., and El Abbadi, A. A comparison of dft and dwt based similarity search in time-series databases. In *Proceedings of the ninth international conference on Information and knowledge management* (2000), ACM, pp. 488–495.

[35] Yi, B.-K., Jagadish, H., and Faloutsos, C. Efficient retrieval of similar time sequences under time warping. In *Data Engineering, 1998. Proceedings., 14th International Conference on* (1998), IEEE, pp. 201–208.

[36] Zheng, Y., Liu, Q., Chen, E., Zhao, J. L., He, L., and Lv, G. Convolutional nonlinear neighbourhood components analysis for time series classification. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2015), Springer, pp. 534–546.

[37] Zhu, Y., and Shasha, D. Statstream: Statistical monitoring of thousands of data streams in real time. In *VLDB'02: Proceedings of the 28th International Conference on Very Large Databases* (2002), Elsevier, pp. 358–369.