Diverse Relevance Feedback for Time Series with Autoencoder Based Summarizations

Bahaeddin Eravci and Hakan Ferhatosmanoglu

Abstract—We present a relevance feedback based browsing methodology using different representations for time series data. The outperforming representation type, e.g., among dual-tree complex wavelet transformation, Fourier, symbolic aggregate approximation (SAX), is learned based on user annotations of the presented query results with representation feedback. We present the use of autoencoder type neural networks to summarize time series or its representations into sparse vectors, which serves as another representation learned from the data. Experiments on 85 real data sets confirm that diversity in the result set increases precision, representation feedback incorporates item diversity and helps to identify the appropriate representation. The results also illustrate that the autoencoders can enhance the base representations, and achieve comparably accurate results with reduced data sizes.

Index Terms—time series analysis, relevance feedback, autoencoders, diversity

1 Introduction

Processes that record data with respect to time are common in many applications ranging from finance to healthcare. A time series is an array of data elements with such temporal association. Large amounts of time series data need to be browsed and analyzed taking temporal relations into account. Typical analytics tasks over time series include browsing, classification, clustering, and forecasting. These tasks usually begin with identifying a representation that aims to link the end purpose of the application and the properties of the time series. Various representations have been proposed to transform the time series, each with a different perspective to meet the requirements of different applications, user intents, and data properties. A class of representations, such as piecewise aggregate approximation (PAA) and symbolic aggregate approximation (SAX), are used to identify features in the time domain, while others, including discrete Fourier transform (DFT) and discrete wavelet transform (DWT), involve frequency domain properties dealing with the periodic components in the series.

After the time series is transformed, measures of similarity are used for analytics tasks. One of them is browsing, where users seek similar time series items from a database by issuing a representative query, such as searching on-line advertisements with similar view patterns with respect to a specific product, stocks that are related in terms of price relative to the stock of choice, regions with similar earthquake event patterns to the city of interest. A multitude of similarity measures (from L_p norms to dynamic time warping (DTW), etc.) has been proposed ([1], [2], [3]). Indexing methods for similarity queries have also generated extensive interest in the community given the computational load of the algorithms [4].

Even though time series retrieval has been studied widely, there is much less work in utilizing relevance feedback (RF) for time series, which has enjoyed a great deal of attention in information retrieval. In RF, a set of data items is retrieved as a result of an initial query by example, and presented to the user for evaluation. The informal goal is to present the right set of initial results that maximize the information return, as opposed to a theoretically top matching set, to correctly model the user intent for the subsequent retrieval iterations. We have considered time-series retrieval with diversity based relevance feedback in our PVLDB paper [5]. Information retrieval and machine learning concepts are adapted to time series with representations that capture the temporal relations. Given the initial result set, the user annotates the items for the next round and the search process continues to eventually find the particular series of interest.

The user annotation can be further exploited to infer the suitable representations for the current user utility, by populating the result set with matching results using different representations. Based on the feedback, the number of items from better performing representations is increased in the following rounds to converge to the representation(s) which maximizes the user utility. Representation feedback is useful to serve different user groups requiring different representations and in dynamic databases where the properties of the data are changing.

We expand the methodology used in prior work by adopting autoencoder neural networks to extract sparser representations of the time series. The method aims to identify appropriate parts of representation(s) with reduced data sizes. Autoencoders can also be effective in combining multiple representations and selecting relevant features from best performing representations by analyzing the dataset. We assess the potential of autoencoders use for time series data in this retrieval context. One can use similar deep network techniques to learn and identify accurate representations by analyzing large and diverse time series datasets. These general networks can also be trained to specific tasks, e.g., stock analysis where patterns are identified with years

• *B. Eravci is with the Department of Computer Engineering, Bilkent University, Turkey. H. Ferhatosmanoglu is with the Department of Computer Science, University of Warwick, UK and with the Department of Computer Engineering, Bilkent University. Contact e-mail: beravci@gmail.com*

of human expertise.

The contributions of our work are the following:

- We utilize different time series representations for RF to capture a variety of global and local information. The performance of RF is enhanced by using diversity in the result set. A mechanism for on-the-fly representation selection based on exploiting the feedback is presented.
- We utilize autoencoders to decrease the complexity of the overall features and extract useful data aware features. A representation map is learned from the data and can be used in other time series tasks. The presented approach exploits the advantages of RF and diversification, and illustrates a potential use of autoencoder type networks in time series retrieval.
- We perform experiments using 85 real data sets, and provide insights on the performance of RF, autoencoders, and diversity to improve time series retrieval. We discuss the performance of the developed methods under different data properties.

Experimental results show 0.23 point increase in precision averaged over all four representations, with 0.48 point increase in specific cases in the third round of RF. Introducing diversity into RF increases average precision by 6.3% relative to RF with no transformations and 2.5% relative to the RF using the proposed representation. Results also show that the representation feedback method implicitly incorporates item diversity and converges to the better performing representation, confirming it to be an effective way to handle changing data properties and different user preferences.

Experiments with the autoencoders present runtime performance increases of around 6-9x due to reduced total data volume with a mild degradation in the average precision. We have also observed that in some challenging data cases, where the precision is low, the accuracy improves when using autoencoders, which is encouraging to further pursue this approach.

2 Related Work

Significant work on relevance feedback has been performed in the information retrieval community ([6], [7], [8]). RF has also been used in the image and multimedia retrieval applications ([9], [10], [11]). Some studies pose the RF problem as a classification task and propose solutions within this context [12], [13].

Combining relevance and diversity for ranking text documents using Maximal Marginal Relevance (MMR) objective function aims to reduce redundancy in the result set while maintaining relevance to the query [14]. There are recent studies which analyze MMR type diversification and provide efficient algorithms for finding the novel subset [15]. Ambiguity in queries and redundancy in retrieved documents have been studied in the literature also focusing on objective evaluation functions [16]. Expected maximization is used to generate diverse results in web search applications [17]. Scalable diversification methods for large datasets are developed using MapReduce [18]. Xu et al. propose using relevance, diversity and density measures together for ranking documents within an active learning setting [19]. Diverse results have been reported to increase user satisfaction for answering ambiguous web queries [20] and for improving personalized web search accuracy [21]. Graph based diversity measures for multi-dimensional data has been proposed in [22].

Top-k retrieval has been studied also as a machine learning problem to rank documents according to user behavior from analyzing implicit feedbacks like click logs. A Bayesian based method is proposed as an active exploration strategy (instead of naive selection methods) so that user interactions are more useful for training the ranking system [23]. A diverse ranking for documents is suggested to maximize the probability that new users will find at least one relevant document [24]. There is recent interest to address the biases (e.g. presentation bias where initial ranking strongly influences the number of clicks the result receives) in implicit feedbacks using a weighted SVM approach [25]. We also note some studies concentrating on ways to balance diversity and relevance while learning ranks of documents ([26], [27]). One can approach the result set selection problem using active learning where the main aim is to label parts of the dataset as efficiently as possible for classification of any data item in the dataset. There is a variety of techniques each with a different perspective such as minimization of uncertainty concerning output values, model parameters and decision boundaries of the machine learning method [28].

Time series data mining research has immense literature on methods for representations, similarity measures, indexing, and pattern discovery [29]. Besides using geometric distances on coefficients [30], dynamic time warping (DTW) and other elastic measures are used to identify similarities between time series due to non-aligned data ([1], [3], [31]). Contrary to its popularity in the information retrieval, RF and diversification have not attracted enough attention in the time series community. Representation of time series with line segments along with weights associated to the related segments and explicit definition of global distortions have been used in time series relevance feedback [32], [33]. We have addressed representation feedback for time series retrieval with diversification [5].

Autoencoder neural networks formulate an unsupervised learning that uses the input data as the output variable to be learned [34]. The network structure and the training objectives force the outcome to be a sparse representation of the input data. It has attracted a renewed interest lately with deep network approaches generally utilizing restricted Boltzmann machines [35]. There has been recent work done on time series visualization utilizing autoencoder structures [36]. Time series forecasting with neural networks is reported to be advantageous even with relatively small data cases [37].

3 Method

3.1 Problem Definition

We consider a database, $TSDB$, of N time series: $TSDB = \{TS_1, TS_2, ..., TS_N\}$. Each item of $TSDB$: TS_i, is a vector of real numbers which can be of different size, i.e. $TS_i = [TS_i(1), TS_i(2), ..., TS_i(L_i)]$ where L_i is the length of a particular TS_i. Given a query, TS_q (not necessarily

in $TSDB$), the problem is to find a result set (a subset of $TSDB$) of k time series that will satisfy the expectation of the user. Since we formulate the solution in an RF setting, the user is directed for a binary feedback by annotating the items in the result set as relevant or irrelevant.

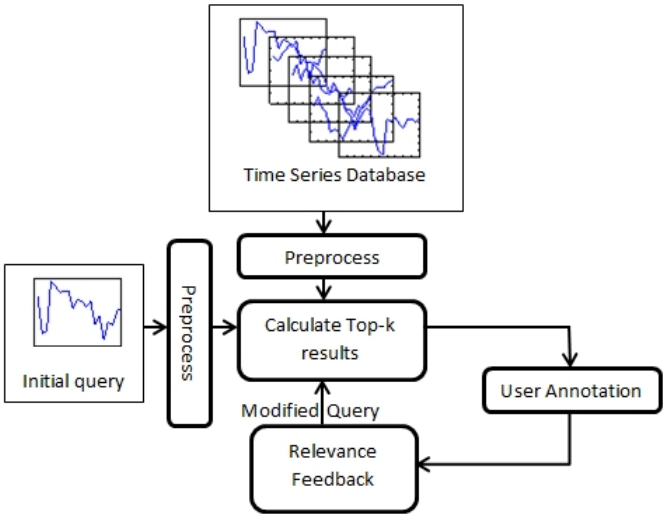

Fig. 1. Time series retrieval with relevance feedback

3.2 Diverse Relevance Feedback for Time Series

Relevance feedback (RF) mechanisms iteratively increase retrieval accuracy and user satisfaction based on user's annotation of the relevance of each round of results. Figure 1 illustrates a basic feedback mechanism where items that are more relevant are presented in the successive rounds. The first step is the transformation of the time series into a representation to capture relevant features, optionally with a normalization procedure such as unit-norm or zero-mean. Given an initial time series query (TS_q), the relevant transformation is applied and a transformed query vector, q, is found. T_i denotes the transformed TS_i according to a transformation ($\mathcal{F}$), i.e., $T_i = \mathcal{F}(TS_i)$.

The representation selected needs to be compatible both with the data and the user intention. For example, if the user desires to retrieve data with specific periods like weekly patterns, frequency domain approaches like DFT can serve the purpose. Shift invariant and length independent representations are advantageous to handle time offsets between the series and varying sized series. Transforms that help compare local and global properties of time series items would be expected to be functional for diversity based browsing. Following these observations, we focus on representations based on wavelet transform and SAX, in addition to Fast Fourier Transform (FFT), and the raw time series as a baseline in our experiments. FFT is a computationally optimized version of DFT with the same numerical outputs. Experiments with four different representations provide insights on how different types of representation work with RF and how they affect the precision for different datasets.

SAX translates the time series into a string of elements from a fixed alphabet, enabling the use of string manipulation techniques. Subsequently, a post-processing method is utilized which converts the SAX string into a matrix (SAX-bitmap image in the visualization context) by counting the different substrings of various lengths included in the whole string [38]. SAX-bitmap is reported to be useful in extracting sub-patterns in the time series and is a perceptually appropriate representation for humans in visualizing and interpreting time series. In this context, we use it as a transformation of the time series to a vector, effectively counting the number of different local signatures, which is then used with different distance measures for similarity retrieval. The level of the representation (L) corresponds to the length of the local patterns in the SAX representation. The length of the SAX-bitmap vector is M^L, (M is the number of symbols used in the SAX process) and is independent of the time series length (L_i). SAX divides the series into blocks inherently extracting local features of the time series which is useful for diverse retrieval methods. The total number of occurrences gives information about the global features as well.

Wavelet transform is a time-frequency representation used extensively in image and time series processing. Wavelet transformed data (scaleogram) provides frequency content of the signal for different time durations. The transform extracts low-pass features (relatively slow varying components) giving an averaged version of the overall series and high pass features (components relatively fast varying) which are related to jumps and spikes in the series. Down-sampling of the series along the branches of the process allows the transform to extract information from different scales of the data. Representation level (L) in wavelet controls the amount of detail in the low pass and high pass regions. Dual-tree complex wavelet transform (named due to two parallel filter banks in the process) is relatively shift-invariant with respect to other flavors of the algorithm which is a reason behind its selection for this study [39]. As a summary, CWT decomposes the time series into local patterns in both time and frequency with different scales and can help for diversity as different subsets of the information given by the transformation provide different perspectives of the data.

Top-k retrieval identifies k results ranked according to a measure of relevance to q. A traditional assumption in nearest neighbor retrieval is that the distribution of the user interest is around the query decaying with the distance to q. However, there can be data points close to the query in the theoretical sense yet not related because of the distribution of the user interest. RF techniques analyze the distribution around the query point with a limited number of user annotated data items. After each iteration of RF, the user is given the opportunity to evaluate the resultant items presented by the system. A variety of different techniques can be utilized for the feedback mechanism, such as Rocchio's algorithm ([6]) which moves the query point in space closer to the relevant items. It is among the early RF methods, recently considered with different variants [40], [41], [42]. We have adapted a modified version of Rocchio's algorithm.

Equation 1 details the procedure (Algorithm 1 Line 19) where $\mathbf{Rel}$ and $\mathbf{Irrel}$ denote the set of items classified

relevant and irrelevant respectively by the user.

$$q_{new} = \frac{1}{|\mathbf{Rel}|}\sum_{i=1}^{|\mathbf{Rel}|} Rel_i - \frac{1}{|\mathbf{Irrel}|}\sum_{i=1}^{|\mathbf{Irrel}|} Irrel_i \quad (1)$$

The original query affects the results of the newly formed query via Equation 2, since they are combined to calculate the distances for ranking the data items. We have also experimented with a Rocchio algorithm which directly replaces the original query at each iteration and found that the modified version performs better. The RF mechanism forms new queries or modifies the initial query in the next iterations for retrieving the similar time series items from the database.

$$Dist(q_1, q_2, ..., q_r, T_{test}) = \frac{1}{r}\sum_{i=1}^{r} Dist(q_i, T_{test}) \quad (2)$$

where r is the RF iteration number

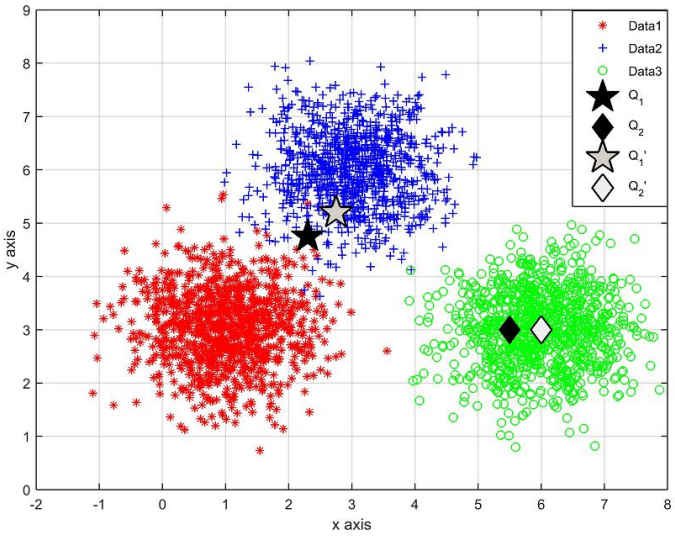

Fig. 2. An example case of data and query movement with Rocchio based algorithm

We present an example to illustrate the mechanics of the query relocation in Figure 2 and to discuss the potential advantages of utilizing diverse results. We have plotted three different classes of data (using three normal distributions with different means, each representing a user's or a group of users' interest) and queries of two extreme cases: Q_1 query on the boundary in terms of user interests and a Q_2 query near to the main relevant set. These queries are moved to revised points (or the effect of using new queries translates to this effect via Equation 2) Q_1' and Q_2' given that Data2 and Data3 are considered relevant by the user respectively. If nearest neighbor (NN) retrieval is used, the result can be a degenerate list with too little variation and limited information about the user intentions since Q_1 and Q_2 are already known. If one provides a larger radius around the query, which samples the region around the query, the displacement of the vectors from their original location will be higher. On the other hand, over-diversification of the results, causing very few relevant items finding a spot in the result set, will hinder and lower the accuracy of the next phase of user annotation. A probabilistic explanation to this intuition is given in Section 3.2.2.

Algorithm 1 High-level algorithm for diverse relevance feedback

1: Initialize k : number of items in result set
2: Initialize $RFRounds$: number of RF iterations
3: Initialize $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_{RFRounds}]$: MMR parameters
4: Initialize $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_{RFRounds}]$: CBD parameters
5: Input q_1 : initial query (transformed if needed)
6: Input $TSDB$: time series database (transformed if needed)
7: **for** $i = 1 \rightarrow RFRounds$ **do**
8: // Find Top-k results
9: **if** Nearest Neighbor **then**
10: $\mathbf{R}$ = Top-K(q_1,...,k,q_i,$TSDB$)
11: **else if** MMR **then**
12: $\mathbf{R}$ = Top-K_MMR(q_1,...,q_i,k,λ_i,$TSDB$)
13: **else if** CBD **then**
14: $\mathbf{R}$ = Top-K_CBD(q_1,...,q_i,k,α_i,$TSDB$)
15: **end if**
16: // User annotation of the result set
17: (**Rel**,**Irrel**) = User Grade($\mathbf{R}$)
18: // Expand query points via relevance feedback
19: q_{i+1} = Relevance_Feedback(**Rel**,**Irrel**)
20: **end for**

To diversify the top-k results, we consider two methods: maximum marginal relevance (MMR) and cluster based diversity. MMR (Algorithm 1 Line 12) merges the distance of the tested data item to both the query and to the other items already in the relevant set as given in Equation 3 and a greedy heuristic is used until a specific number of items is selected from the dataset. $Dist$ can be any distance function of choice. When λ is 1, the $DivDist$ collapses to an NN case and the result becomes a mere top-k set. When λ decreases, the importance of the distance to the initial query decreases giving an end result set of more diverse items which are also related to the query. The second term of the $DivDist$ involves pairwise comparisons of data points in the database which is independent of the query and is performed repetitively for each query. To decrease the running time, we use a look-up table that stores all the possible pairwise distances calculated off-line once at the beginning for the particular database.

$$DivDist(T_q, T_i, \mathbf{R}) = \lambda Dist(T_q, T_i) - \frac{1}{|\mathbf{R}|}(1-\lambda)\sum_{j=1}^{|\mathbf{R}|} Dist(T_i, T_j) \quad (3)$$

Cluster Based Diversity (CBD) method uses a different approach than the optimization criteria as given in Equation 3. The method is inspired by [43] which proposes a clustering based method for finding best representatives of a data set. This method (Algorithm 1 Line 14) retrieves Top-αk elements ($\alpha \geq 1$) with an NN approach and then clusters the αk elements into k clusters. α controls the diversity desired where increasing α increases the diversity of the result set and $\alpha = 1$ case corresponds to NN case.

We implement the k-means algorithm for the clustering phase. The data points nearest to the cluster centers are chosen as the representative points presented to the user. An advantage of CBD is that the tuning parameter α is intuitive and the results are relatively predictable.

The method for diversification can be tailored with respect to the methods used for searching and learning the user feedback. For example, one can utilize a support vector machine (SVM) binary classifier to learn the relevant/irrelevant sets instead of the distance based ranking method. In this case, figuring out the cluster centers as in the CBD method would likely perform worse since SVM classifier tries to learn the boundaries of the classes and would benefit from annotations around these boundaries. Whereas, NN-like distance based models with query movement mechanism would benefit more by learning the centroids of the relevant data with low uncertainty.

3.2.1 Algorithmic Complexity

We present the algorithmic complexity of the retrieval methods in terms of N (the number of time series in database), L (the length of time series or representation), and k (the number of requested items). The NN based retrieval first calculates distances to all items in dataset ($O(NL)$) and finds k nearest items ($O(kN)$) which corresponds to a total complexity of $O(N(L+k)) = O(N)$.

For the MMR case we have two possibilities with respect to Equation 3:

- Without memoization: Distance calculations to all items in the dataset ($O(NL)$), distance calculations for relevant set items ($O(N \cdot L \cdot (k-1) \cdot (k-2)/2) = O(NLk^2)$, finding the minimum distance element k times ($O(kN)$) with an overall complexity of $O(NL(1+k^2)) = O(N)$.
- With memoization: Distance calculations to all items in the dataset ($O(NL)$), distance calculations from lookup table for relevant set items ($O((k-1) \cdot (k-2)/2) = O(k^2)$, finding the minimum distance element k times ($O(kN)$) with an overall complexity of $O(N \cdot (L+k)) = O(N)$ (where $NL \gg k^2$).

For the CBD case, we first find αk nearest neighbors ($O(N(L+\alpha k))$ and cluster the results. K-means clustering (based on Lloyd's which has a limit i for the number of iterations) is considered $O(NkLi) = O(NkL)$ algorithm. The total complexity for CBD case is $O(N(L+\alpha k+Lki)) = O(N)$.

3.2.2 Illustrative Analysis of Diverse Retrieval

We now present an illustration of the intuition behind using diversity in the RF context. Given a query, q, we retrieve a top-k list using NN with the last element d distance away from the query. Figure 3 illustrates the relevant set, $R \sim \mathcal{N}(0, \sigma^2)$ and the irrelevant set, $IR \sim \mathcal{N}(\mu, \sigma^2)$ assuming Gaussian distributions for both.

If there are N relevant and M irrelevant items, we can find the number of relevant ($k1$) and irrelevant items ($k2$) in the top-k list with approximations as:

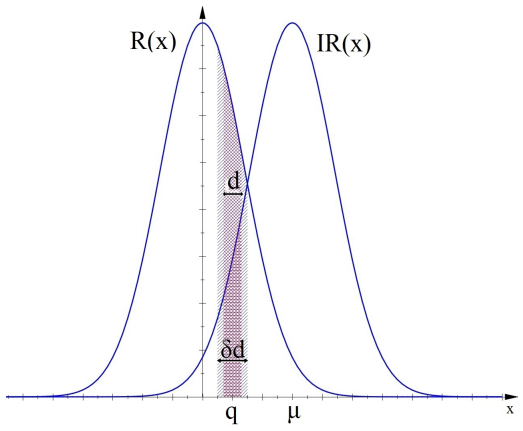

Fig. 3. Data distributions used in analysis

$$
\begin{aligned}
k_1 &= N \cdot \int_{q-d}^{q+d} R(x)\,\mathrm{d}x \approx N \cdot R(q) \cdot 2d \quad if \;\; k_1 \ll N \\
k_2 &= M \cdot \int_{q-d}^{q+d} IR(x)\,\mathrm{d}x \approx M \cdot IR(q) \cdot 2d \quad if \;\; k_2 \ll M \\
k &= k_1 + k_2
\end{aligned}
\tag{4}
$$

We can then define and calculate the precision for the query as:

$$
Prec(q) = \frac{k_1}{k_1 + k_2} = \frac{N \cdot R(q)}{N \cdot R(q) + M \cdot IR(q)} \tag{5}
$$

This formula follows the general fact that if R and IR are separable (μ is very large) or if the query point is near the mean of R precision will be high. We also observe that the performance is dependent on the accuracy of the known model (i.e. the R and IR distributions) itself. We learn the model of the relevant set in the RF setting by modifying the query according to the feedback from the user. Consider a simplified RF model that forms the query for the next iteration (q_2) as the average of all the relevant items, i.e.:

$$
q_2 = \sum_{i=1}^{k_1} R_i = \int_{q-d}^{q+d} x \cdot R(x)\,\mathrm{d}x = \sqrt{\frac{\sigma^2}{2 \cdot \pi}}[\mathrm{e}^{q-d} - \mathrm{e}^{q+d}] \tag{6}
$$

A diverse set of points around q would span a larger distance (δd) around the query, which is also shown in Figure 3. In this case we get a modified q_2' from the relevance feedback as:

$$
\begin{aligned}
q_2' &= \sum_{i=1}^{k_1} R_i = \int_{q-\delta d}^{q+\delta d} x \cdot R(x)\mathrm{d}x \\
&= \sqrt{\frac{\sigma^2}{2 \cdot \pi}}[\mathrm{e}^{q-\delta d} - \mathrm{e}^{q+\delta d}] \quad \delta > 1
\end{aligned}
\tag{7}
$$

Diversity ensures $q_2' < q_2$ which increases our understanding of the relevant data distribution and consequently the query precision via Equation 5. If the precision is already high (i.e., if R and IR are well separated or the query is not near the R and IR boundary), then the precision increase due to diversity will not be significant.

3.3 Representation Feedback

The choice of representation is a fundamental challenge in time series retrieval. Many different approaches for time series representation have been proposed in the research community for different cases. For a fixed goal of analysis over static data, one can perform off-line experiments with different representations and choose the representation according to the given performance criteria. This would not work for dynamic data. Moreover, a single predetermined representation may not be suitable for all the users of the system even for static databases. For example, a group of users may be interested in time domain features while another group may focus on frequency domain properties. A unified time series representation appropriate for all applications and user intentions is hard to reach.

Algorithm 2 High-level algorithm for representation feedback system

1: Initialize r : number of representations
2: Initialize $RFRounds$ and input q_1
3: $TSDB_r$: time series database in representation r
4: Initialize k_i for $i : 1 \dots r$
5: **for** $i = 1 \rightarrow RFRounds$ **do**
6: // Find Top-k results using any alternative method
7: $\mathbf{R} = \emptyset$
8: **for** $j = 1 \rightarrow r$ **do**
9: $\mathbf{R}$ =$\mathbf{R} \cup$ Top-K(q_1^j,...,q_i^j,k_j,$TSDB_j$)
10: **end for**
11: // Let user grade the retrieval results
12: (**Rel**,**Irrel**) = User Grade($\mathbf{R}$)
13: // Expand query points via relevance feedback
14: **for** $j = 1 \rightarrow r$ **do**
15: q_{i+1}^j = Relevance_Feedback(**Rel**,**Irrel**)
16: **end for**
17: // Update representation feedback parameters
18: k_i = UpdateK(k_i,**Rel**,**Irrel**)
19: **end for**

The user feedback can be utilized to select the appropriate representation(s) based on the presented results. For this purpose, we initially retrieve the similar items based on different representations. The result set is partitioned according to each representation's performance with the aim of identifying best performing representation or the best combination of representations. The overview of the representation feedback algorithm is given in Algorithm 2. This method is used in conjunction with query modification in each iteration. The benefit of fusing different time series representations is to reach an aggregate expressive power from each representation, with implicit diversification to improve the RF performance.

The value k (given in Algorithm 2 Line 4) is divided into k_i values (where the sum of k_is add up to k), each regulating the share of different representations in the final result set. An equal distribution can be selected in the first round of RF. Any prior knowledge (e.g. by learning from user logs) about the performance of the representations can be used to estimate the initial k_i values.

The initial set is populated by top matching k_i items from each of the representations, using any of the NN, MMR or CBD retrieval methods (Algorithm 2 Lines 7-10). After the evaluation of the presented set by the user, k_i values are updated (in Algorithm 2 Line 18) according to the accuracy of the related representation. The starting value and update of the k_i partitions are given in Equation 8.

Initialization:

$$k_i = \left[\frac{k}{r}\right] \text{ where } r \text{ is number of representations}$$

Update:

$$k_i = \frac{\text{Number of relevant items from representation } i}{\text{Number of relevant items}}$$
$$\forall i \leq r \tag{8}$$

3.4 Time Series RF using Autoencoders

Autoencoders have been proposed in machine learning as a structure to learn and transform the input data into a set of important parameters from which the original data is synthesized back. In this respect, autoencoders are a good candidate to extract useful data-oriented features which are also low-dimensional. One can utilize autoencoders for two important prospects in time series: choosing and blending different time series representations, and reducing the already extracted set of features to important features. Since the autoencoder model does not need any teacher who dictates the class of the time series, this unsupervised learning method can be applied for RF based time-series retrieval achieving the two goals via an analysis of the dataset.

Autoencoders are implemented using neural networks, defined by the layers of neurons stacked on top of each other which can be connected in different configurations. Each artificial neuron is composed of a weighted summation unit, summing all the signals in its input and an activation function (σ) which is generally chosen as a non-linear function. A class of neural networks called multilayer perceptron (MLP) which has at least three layers (an input and output layer, one or multiple hidden layers) is constructed of fully interconnected neurons. The topology and the number of nodes in the network determine the space of possible learnable functions whereas the weights between nodes after the training phase defines the exact functionality of the network.

Time series data is used in the autoencoder network both as input and output where the input data is first 'encoded' to a new representation space $\mathbf{z}$ ($\mathbf{z} = \mathcal{H}_{encoder}(TS_i)$) and then decoded using the encoded values to the final output ($TS_i' = \mathcal{H}_{decoder}(\mathbf{z})$). The criteria for a learned model and representation is defined by a loss function of choice based on the data and application which is usually the discrepancy between the data and its generated counterpart. Although replicating the input time series instead of classification or ranking may not be considered a good learning target, the useful and important product of autoencoders is the encoded data, $\mathbf{z}$, which identifies key structures within the data. This process, which can also be considered as a non-linear dimension reduction determining local and global features, encodes the raw or transformed time series data into a sparse vector to be used in the RF based retrieval framework. Autoencoder essentially learns a data aware

representation, and reduces the length of the time series which will decrease the runtime of the retrieval process. It also enables combining of different transformations and identifying important features from different representations if used with multiple representations. The overall algorithm is presented in Algorithm 3.

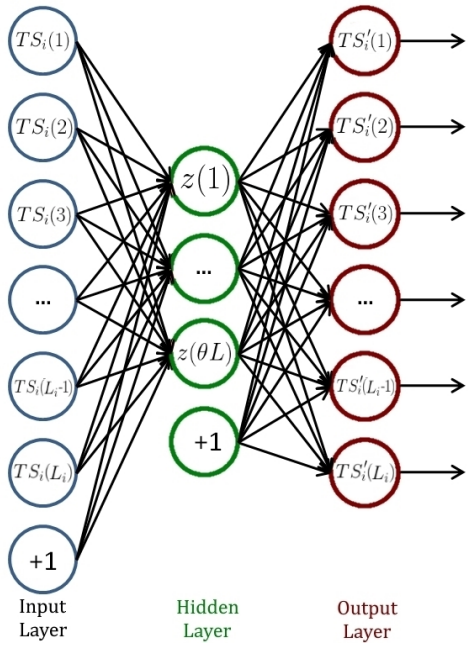

Fig. 4. Autoencoder network structure

We employ an MLP based autoencoder network whose configuration is provided in Figure 4 where TS_i is the input time series and TS'_i is the synthesized series using the encoded values (**z**) in the hidden layer. Constraining the number of neurons in the encoder (hidden) layer to be considerably less than the input layer forces the model to learn a subset of important features within the data. We denote the parameter $\theta < 1$ (defined in Algorithm 3 Line 8) as the ratio of the number of neurons in the encoder to the number of input layer neurons to quantify compression ratio.

We aim to minimize the difference between original and regenerated counterparts ($(TS_i - TS'_i)^2$) or ($\mathcal{F}(TS_i) - \mathcal{F}(TS_i)')^2$)) in the training phase. Training of the autoencoders (Algorithm 3 Line 10-13) is carried out by back-propagation which is a gradient descent based optimization technique used widely in training neural networks. We also include regularization parameters to the cost function used in the optimization process so that each neuron in the hidden layer activates with respect to a group of time series specializing to specific features present in this particular group. The result of the training phase provides us the weight matrix which characterizes the encoder functionality $\mathcal{H}_{encoder}$.

After the training phase, the time series database and queries are encoded into the newly learned representation using the weight matrix (Algorithm 3 Line 15-18) and the retrieval phase with the diversity achieving methods can be executed without any change.

Algorithm 3 Overview of RF system using autoencoders

1: Initialize k : number of items in result set
2: Initialize $RFRounds$: number of RF iterations
3: Initialize $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_{RFRounds}]$: MMR parameters
4: Initialize $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_{RFRounds}]$: CBD parameters
5: Input q_1 : initial query (transformed if needed)
6: Input $TSDB$: time series database (transformed if needed)
7: // Parameters for autoencoder
8: Initialize θ for sparsity level of the autoencoder
9: σ as activation function of ANN
10: *Train Autoencoder*
11: Initialize network with L_i input nodes, $\theta.L_i$ input nodes and L_i output nodes
12: Train the network using back propagation
13: Extract the weight and bias ($\mathbf{W}, \mathbf{b}$) matrices for encoder
14: *Diverse Retrieval System*
15: **for** $i = 1 \rightarrow N$ **do**
16: $\quad TSDB'(i) = \sigma(\mathbf{W}.TS_i + \mathbf{b})$
17: **end for**
18: Transform the query : $q'_1 = \sigma(\mathbf{W}.q_1 + \mathbf{b})$
19: **for** $i = 1 \rightarrow RFRounds$ **do**
20: $\quad$ // Find Top-k results
21: $\quad$ **if** Nearest Neighbor **then**
22: $\quad\quad$ **R** = Top-K(q'_1,...,k,q_i,$TSDB'$)
23: $\quad$ **else if** MMR **then**
24: $\quad\quad$ **R** = Top-K_MMR(q'_1,...,q_i,k,λ_i,$TSDB'$)
25: $\quad$ **else if** CBD **then**
26: $\quad\quad$ **R** = Top-K_CBD(q'_1,...,q_i,k,α_i,$TSDB'$)
27: $\quad$ **end if**
28: $\quad$ // User annotation of the result set
29: $\quad$ (**Rel**,**Irrel**) = User Grade(**R**)
30: $\quad$ // Expand query points via relevance feedback
31: $\quad q_{i+1}$ = Relevance_Feedback(**Rel**,**Irrel**)
32: **end for**

The database ($TSDB$ in Algorithm 3 Line 6) can be constituted of the following: time series (TS), transformation (FFT, CWT, SAX, etc.) of time series, a combination of time series and/or its representation (e.g. TS, FFT, CWT features concatenated). If a combination is used the system can extract different perspectives from multiple representations from the data similar to the representation feedback process in Section 3.3.

The initial size of time series database, $N \cdot L$ is reduced to $N \cdot L \cdot \theta$ after the encoding process which changes the values of the coefficients in the computational complexity given in Section 3.2. This proportional decrease is achieved both in terms of computational load and memory.

4 Experiments

We evaluate the performance of the methods with experiments on 85 real data sets, all data currently available in the UCR time series repository[44]. The data sets used with their respective properties are provided in Table 1. Since we have an unsupervised application, the training and test datasets are combined to increase the size of the data sets.

The numbering of the datasets in this paper is according to the numbering given in the table. The number of classes within the data sets varies from 2 to 60, lengths of the time series (L) in the data sets vary from 24 to 2709, the size (number of time series N) of the data sets vary from 40 to 16,637.

TABLE 1
Datasets used for experiments

No.	Name	Number of classes	Time series length	Dataset size
1	50Words	50	270	905
2	Adiac	37	176	781
3	ArrowHead	3	251	211
4	Beef	5	470	60
5	Beetle Fly	2	512	40
6	Bird Chicken	2	512	40
7	CBF	3	128	930
8	Car	4	577	120
9	Chlorine Concentration	3	166	4307
10	CinC ECG Torso	4	1639	1420
11	Coffee	2	286	56
12	Computers	2	720	500
13	Cricket_X	12	300	780
14	Cricket_Y	12	300	780
15	Cricket_Z	12	300	780
16	Diatom Size Reduction	4	345	322
17	Distal Phalanx Outline Age Group	3	80	539
18	Distal Phalanx Outline Correct	2	80	876
19	Distal Phalanx TW	6	80	539
20	ECG 200	2	96	200
21	ECG 5000	5	140	5000
22	ECG Five Days	2	136	884
23	Earthquakes	2	512	461
24	Electric Devices	7	96	16637
25	Fish	7	463	350
26	Face (all)	14	131	2250
27	Face (four)	4	350	112
28	Faces UCR	14	131	2250
29	Ford A	2	500	4921
30	Ford B	2	500	4446
31	Gun-Point	2	150	200
32	Ham	2	431	214
33	Hand Outlines	2	2709	1370
34	Haptics	5	1092	463
35	Herring	2	512	128
36	Inline Skate	7	1882	650
37	Insect Wingbeat Sound	11	256	2200
38	Italy Power Demand	2	24	1096
39	Large Kitchen Appliances	3	720	750
40	Lightning-2	2	637	121
41	Lightning-7	7	319	143
42	MALLAT	8	1024	2400
43	Meat	3	448	120
44	MedicalImages	10	99	1141
45	Middle Phalanx Outline Age Group	3	80	554
46	Middle Phalanx Outline Correct	2	80	891
47	Middle Phalanx TW	6	80	553
48	Mote Strain	2	84	1272
49	Non-Invasive Fetal ECG Thorax1	42	750	3765
50	Non-Invasive Fetal ECG Thorax2	42	750	3765
51	OliveOil	4	570	60
52	OSU Leaf	6	427	442
53	Phalanges Outlines Correct	2	80	2658
54	Phoneme	39	1024	2110
55	Plane	7	144	210
56	Proximal Phalanx Outline Age Group	3	80	605
57	Proximal Phalanx Outline Correct	2	80	891
58	Proximal Phalanx TW	6	80	605
59	Refrigeration Devices	3	720	750
60	Screen Type	3	720	750
61	Shapelet Sim	2	500	200
62	Shapes All	60	512	1200
63	Small Kitchen Appliances	3	720	750
64	Sony AIBO Robot Surface	2	70	621
65	Sony AIBO Robot SurfaceII	2	65	980
66	Star Light Curves	3	1024	9236
67	Strawberry	2	235	983
68	Swedish Leaf	15	128	1125
69	Symbols	6	398	1020
70	Toe Segmentation1	2	277	268
71	Toe Segmentation2	2	343	166
72	Trace	4	275	200
73	TwoLead ECG	2	82	1162
74	Two Patterns	4	128	5000
75	UWave Gesture Library All	8	945	4478
76	Wine	2	234	111
77	Word Synonyms	25	270	905
78	Worms	5	900	258
79	Worms Two Class	2	900	258
80	Synthetic Control	6	60	600
81	uWave Gesture Library_X	8	315	4478
82	uWave Gesture Library_Y	8	315	4478
83	uWave Gesture Library_Z	8	315	4478
84	Wafer	2	152	7174
85	Yoga	2	426	3300

4.1 Experimental Setting

We first transform all the time series data to CWT, SAX and FFT. SAX parameters are $N = L_i$, $n = \lceil \frac{N}{5} \rceil$ (meaning blocks of length 5), an alphabet of four with SAX-Bitmap level of 4 and CWT with detail level $L = 5$. We performed the same experiments also on the raw time series without any modification (TS) to compare the effectiveness of the representations. The values for L and n can be optimized for different data sets to further increase accuracy. We have experimented with several different values around the vicinity of the given values ($n = \lceil \frac{N}{6} \rceil$, $L \in [3, 4]$) for randomly selected datasets and have seen that the improvement in precision is still evident on similar scales. Since the objective of this study is not to find solutions for specific cases and we aim to enhance RF via diverse results for general cases, we did not fine tune parameters, we used the same set of parameters for all the data sets for an impartial treatment.

In the experiments, we explored 5 different methods of top-k retrieval:nearest neighbor (NN), MMR with $\lambda = [0.5, 1, 1]$ (MMR_1), MMR with $\lambda = [0.5, 0.75, 1]$ (MMR_2), CBD with $\alpha = [3, 1, 1]$ (CBD_1) and CBD with $\alpha = [3, 2, 1]$ (CBD_2). In the stated configuration, we explore the effects of diversification on the accuracy by varying the level of diversification in different iterations. We note that MMR_2 and CBD_2 cases decrease the diversity in a more graceful way whereas MMR_1 and CBD_1 go directly to NN case after the initial iteration. We did not try to optimize the parameters (λ and α) of the diversification schemes and the values present themselves as mere intuitive estimates. We implemented a unit normalization method for each dataset and used cosine distance for all the experiments.

We implemented the method given in [33] to compare the performance of our algorithms. This method uses a piecewise linear approximation (PLA-RF) for time series and associates a weight for each part of the series when calculating the distances to query. These weights are modified in each iteration of feedback according to the user feedback.

In the experiments, we model the user to seek similar time series from the same class in the dataset. Under this model, the class of the series is used to generate relevant/irrelevant user feedback after each RF iteration. Items in the result set which are of the same class as the query are considered relevant and vice versa. The experiments were performed on a leave-one-out basis such that we use each and every time series in the database as a query and RF is executed with the related parameters using the database excluding the query itself. Accuracy is defined by precision value based on the classes of the retrieved top-k set. Precision for the query is calculated using the resultant top-k list and the averaged precision over all the time series in the database is considered as the final performance criteria which are defined below:

$$\text{Query Precision}(T_q) = \tfrac{1}{10} \textstyle\sum_{i=1}^{10} \delta(i)$$
$$\text{Average Precision} = \tfrac{1}{N} \textstyle\sum_{\forall T_q \in TSDB} \text{Query Precision}(T_q)$$

$$\text{where } \delta(i) = \begin{cases} 1 & \text{if class of } T_q \text{ is equal to class of } R_i \\ 0 & \text{otherwise} \end{cases}$$

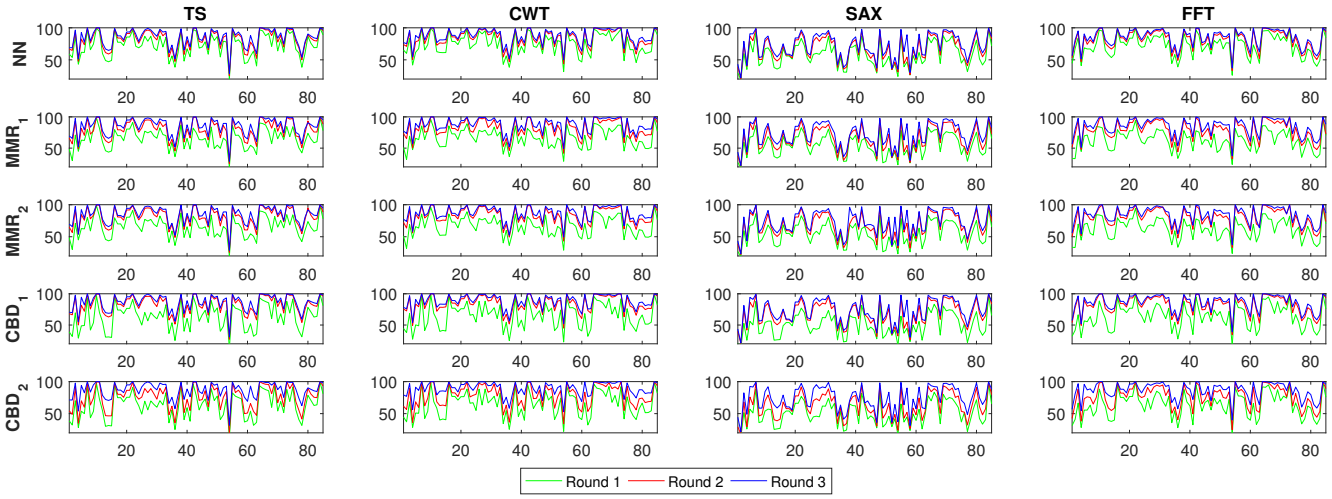

Fig. 5. Performance for three rounds of RF for all the datasets (precision scaled to 100 in y-axis versus dataset number in x-axis)

4.2 Experimental Results and Discussions

4.2.1 Results for Diversity

The experimental results for diversity in the result set are given in Figure 5 for all the data sets. Each row in the figure corresponds to one of five retrieval methods and each column corresponds to the representation (TS, CWT, SAX, and FFT) used. In each individual graph, the average precision in different RF iterations is plotted with the data set number given in x-axis. We present an aggregate result here to summarize the results.

TABLE 2
Average Increase (absolute) in Precision

RF Round	2	3
NN	9.08	12.73
$MMR(\lambda_1)$	14.19	19.98
$MMR(\lambda_2)$	15.75	20.01
$CBD(\alpha_1)$	18.88	22.98
$CBD(\alpha_2)$	12.60	23.44
PLA-RF	3.7	4.4

We calculated the precision (scaled to 100) difference between different rounds and the first round of RF for a particular representation, method and data set (4 representations x 5 methods x 85 data sets = total 5100 cases). Histogram of the resulting improvements is provided in Figure 6. Differences in precision (averaged over all cases) are provided in Table 2 to quantify the performance increase with the use of diverse RF. We also performed a t-test between the average values given in the table and a zero mean distribution to verify the statistical significance of the improvement. The p-values, in the range of 10^{-110}, are notably smaller than 0.05 which is considered as a threshold for significance. RF with the configurations given in this study improves accuracy in all cases without any dependence of data type or data representation and it provides significant benefits with 0.50 point precision increases in some cases. We also note that the proposed methods outperform the state of the art.

Since the experiments produced large amount of results (given the number of time series data types, representations,

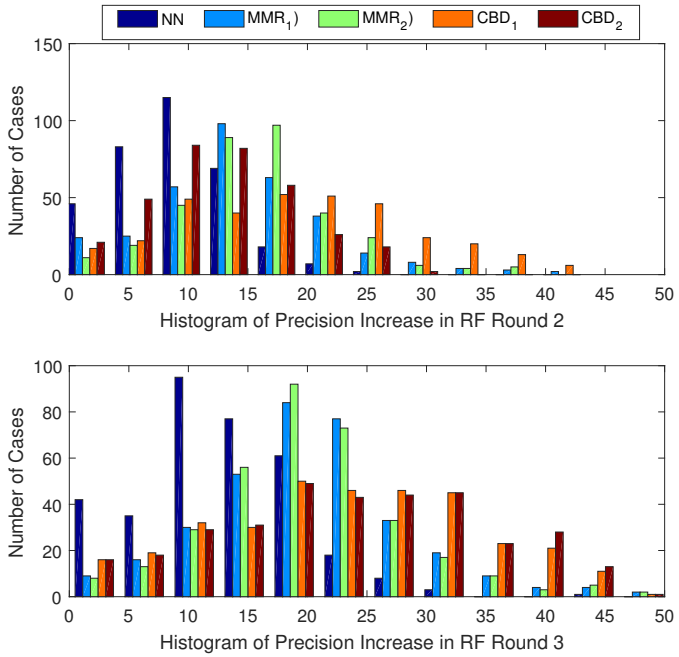

Fig. 6. Histogram of increase in precision with different RF settings feedback

top-k retrieval methods), for illustrative purposes, we consider a reference case where the time series without any transformation and NN only method is used. Accordingly, for each RF round and each data set, the accuracy results are normalized to a total 100 with respect to the base case for that particular data set and RF round. Figure 7 shows the normalized results averaged over all the experimental cases. CWT based representation outperformed FFT, SAX and the time series without any transformation (TS) in nearly all cases. We note that representation parameters are not optimized and different results may be achieved by further optimizing transformation parameters. We did not perform such rigorous testing since it would divert us from the main focus of the study. However, CWT performed

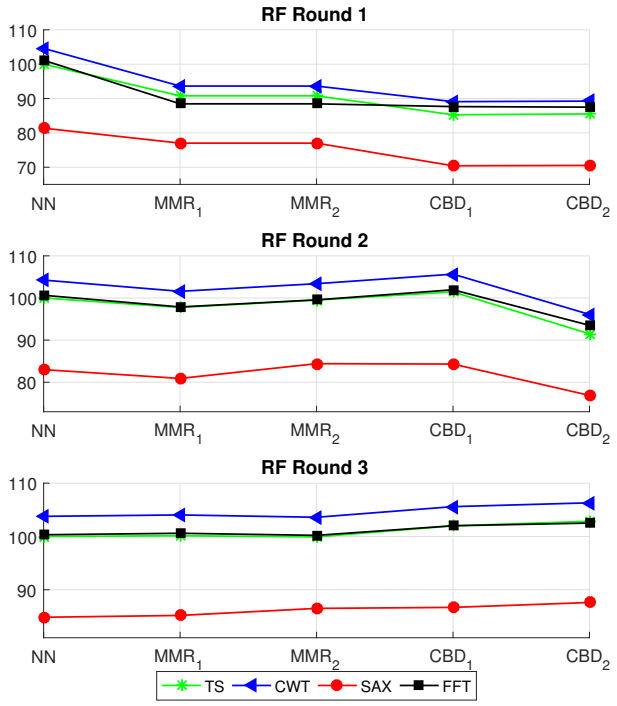

Fig. 7. Normalized performances of different methods and representations

better consistently with no need of parameter optimization.

Although NN achieves the best performance in the first iterations of RF as expected, introducing diversity in the first iteration leads to a jump in RF performance exceeding NN in nearly all the cases. In RF round 3, CBD_2, the best performing method, adds 6.3% (p-value < 0.05) improvement over the reference case and 2.5% (p-value < 0.05) over the case which uses NN method with CWT. Diversity increases its effect further in the third rounds where NN is outperformed even in more cases with similar performance advancements. We also note that CBD_1 and CBD_2) perform best in second and third iterations respectively. This also underlines the enhancement in performance due to increased diversity if the number of iterations increases.

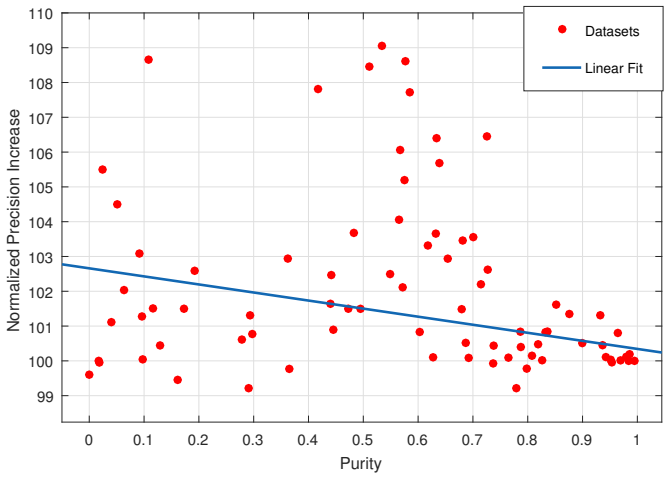

Fig. 8. Normalized performances of different datasets versus purity of dataset

We also investigated the relation between cluster separability within the dataset and the improvements due to diversity. For this purpose, we calculated a separability score for each data set by using a k-means classifier and the average accuracy of the classifier is considered as the separability of the data set. This score, which is in the range $0-1$, essentially quantifies the separability of the classes where a score closer to 1 means easily classifiable datasets. We plot the normalized precision described in the previous paragraphs against the purity of the related dataset with the corresponding linear fit in Figure 8. Effect of diversity is not significant where classes are already separable (datasets with purity in the range $[0.75, 1]$) which is inline with our expectations. Positive effect of diverse retrieval increases when the classes are more interleaved which is the harder case in terms of system performance.

4.2.2 *Performance of Representation Feedback Method*

We have experimented with the proposed representation feedback method and we summarize the results for its use in conjunction with item diversity. Results of normalized performance with respect to the baseline (NN method in the first round of RF) averaged over all the data sets are given in Figure 9. We note that, in contrary to our item-only diverse RF method results provided in the previous section, pure NN retrieval achieves similar performance when top-k partitioning representation feedback is used. We associate this difference in results with the observation that data items retrieved from different representations implicitly provide a diverse result set which improves the performance of RF without the need for further item diversity.

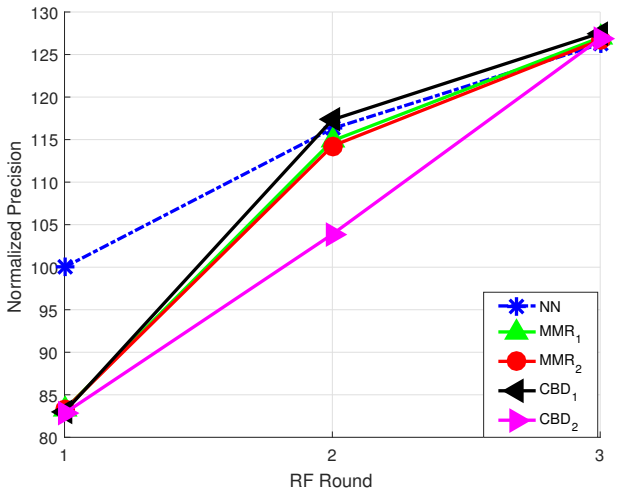

Fig. 9. Normalized performances of top-k partitioning representation feedback methods

We also compare top-k partitioning representation feedback with the best performing method found in the item diversity experiments in Figure 10. The figure illustrates that our principle aim is achieved by the method and as the RF process evolves with subsequent iterations the system converges to the best performing representation. Pure NN retrieval is used in this comparison, since it performed similarly to the other diverse retrieval methods when used in combination with representation feedback.

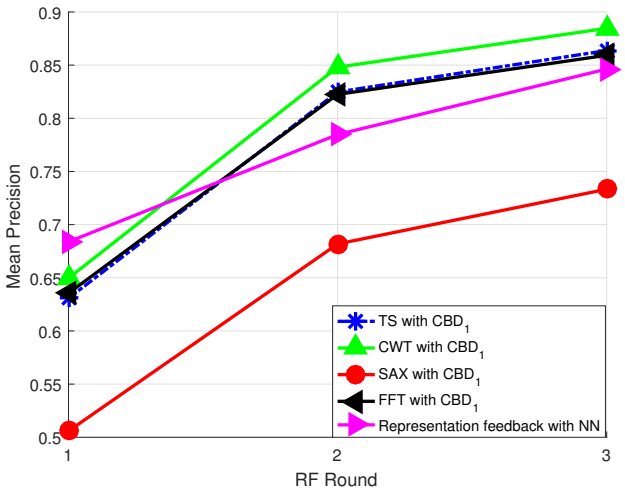

Fig. 10. Accuracy comparison of top-k partitioning representation feedback with item-only diversity

4.2.3 Results for Diverse RF Using Autoencoder

We execute the same experimental setup, using TS, CWT, SAX representations and NN, MMR_1, MMR_2, CBD_1, CBD_2 methods to evaluate the RF system with autoencoders. Additionally, we also experimented on the combination of all the representations (TOTAL) as the input to the system.

We have varied the sparsity index $\theta \in [3, 6, 9]$ to observe the effects of compression on the results. Autoencoder hidden layer node numbers are selected as $[\frac{L}{\theta}]$ and are trained using MATLAB neural network toolbox with default values except for sparsity regularization term which is selected as 4 instead of default 1 to emphasis sparsity in the encoder.

We have experimented on the full 85 data sets, containing time series data of very different properties, to assess the generality of the autoencoder based method. We denote the different cases with the respective input representation and sparsity index, e.g. CWT^3 denotes the outputs of a trained autoencoder with $\theta = 3$ (length of data is reduced to a third of the original length) and CWT transformed time series as input.

We use the normalized precision to quantify the performance, such that precision in the first round of RF with NN is normalized to 100 and all the other precision values are scaled respectively. The normalization is performed for each dataset and transformation (each θ case is also considered a new transformation since the data input for the RF system changes) separately to illustrate the effect of RF more discretely. The results provided in Table 3 demonstrate that diverse retrieval methods achieve similar accuracy improvements with the encoded data. We also observe that even though we reduce the data into a much reduced form in the $\theta = 9$ case the diverse RF system is still working with graceful degradations instead of a sudden breakdown in performance. Precision improvements for the $TOTAL$ representation are also on a similar scale.

The average precision levels (scaled to 100) over all of the datasets are provided in Table 4 with respective methods for the third RF round. We can see that autoencoded features

TABLE 3
Normalized precision improvements with varying autoencoders for third round of RF

	NN	MMR_1	MMR_2	CBD_1	CBD_2
TS	119.7	120.0	119.7	122.3	123.5
TS^3	120.3	121.4	121.5	123.2	124.3
TS^6	119.7	120.4	120.2	122.5	123.7
TS^9	119.0	120.0	120.0	122.4	123.6
CWT	119.2	119.5	119.0	121.4	122.3
CWT^3	117.7	118.6	118.3	120.1	120.7
CWT^6	117.6	118.5	118.4	119.8	120.3
CWT^9	117.9	118.8	118.5	120.3	121.0
SAX	126.8	127.4	130.3	129.6	131.2
SAX^3	121.4	119.8	121.8	123.7	124.7
SAX^6	121.0	119.2	122.3	123.3	124.7
SAX^9	119.8	118.2	121.1	122.4	123.9
FFT	119.5	119.9	119.4	121.7	122.3
FFT^3	120.6	121.1	120.6	122.7	123.3
FFT^6	119.5	120.3	119.9	121.8	122.6
FFT^9	119.3	119.7	119.6	121.8	122.2
TOTAL	119.1	119.7	119.1	121.1	121.8
$TOTAL^3$	118.5	119.3	118.7	120.4	121.0
$TOTAL^6$	118.8	119.4	119.0	120.7	121.2
$TOTAL^9$	118.5	119.3	118.9	120.4	121.1

TABLE 4
Average precision levels for diverse RF with varying configurations

	TS	TS^3	TS^6	TS^9
NN	85.0	84.8	83.9	82.3
MMR_2	84.8	85.4	84.1	82.8
CBD_2	86.9	86.9	86.0	84.8
	CWT	**CWT^3**	**CWT^6**	**CWT^9**
NN	87.1	85.9	85.2	84.2
MMR_2	86.9	86.2	85.6	84.6
CBD_2	88.9	87.6	86.8	86.0
	SAX	**SAX^3**	**SAX^6**	**SAX^9**
NN	71.9	65.5	64.5	63.4
MMR_2	73.3	65.7	64.9	63.9
CBD_2	74.1	67.2	66.4	65.4
	FFT	**FFT^3**	**FFT^6**	**FFT^9**
NN	84.7	84.0	82.3	80.9
MMR_2	84.4	84.0	82.5	81.0
CBD_2	86.3	85.6	84.1	82.5
	TOTAL	**$TOTAL^3$**	**$TOTAL^6$**	**$TOTAL^9$**
NN	86.9	88.7	88.3	88.2
MMR_2	86.8	88.8	88.4	88.4
CBD_2	88.5	90.2	89.8	89.9

are performing without significant losses until $\theta = 6$ value except for SAX-Bitmap case which is considered important since the data is reduced to 16.7% of its original size. The relatively close results are mainly due to the averaging of the significant number of high performing databases in the dataset which are evident in Figure 5.

We note the performance of the $TOTAL$ representation which increases its performance as it gets sparser and outperforms all the other configurations, supporting the expectation that the autoencoder can remove unnecessary features from the representations and amplify the useful features directly by training on the data. We also looked at the lowest performing 15 datasets (which have precision levels below 70% after 3 rounds RF with the NN method),

i.e., cases that need most improvements. The results for this subset are provided in Figure 12 under different transformations and methods for the third RF iteration. We present the results for NN and CBD_2 with $\theta = 6$ autoencoders to illustrate the general case based on our previous findings. The performance increases are visible for these datasets more explicitly with CBD_2 for $TOTAL^6$ approximately the upper bound and the base case NN with TS is the lower bound for performance. We can see from the figure that the proposed transformations, autoencoder structure and the diverse retrieval methods can increase the accuracy considerably with nearly 0.20 point improvement (around one-third increase relatively).

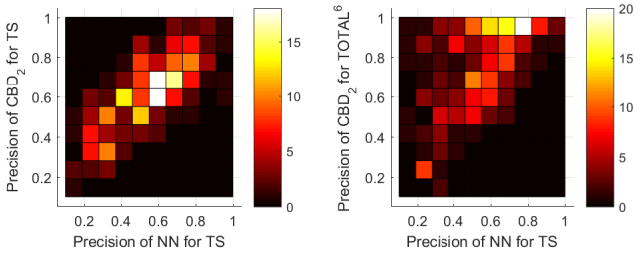

Fig. 11. 2-D histograms (number of queries) of query precision under different methods and transformations in the third iteration of RF for Worms dataset

We illustrate the findings for query precisions and individual queries, over Large Kitchen Appliances and Worms datasets, to get more insights on the problem and the proposed approach. The method returns diverse results in the first RF iteration, which directs the system for subsequent iterations. We see that the queries are performing better under the CWT/FFT transformations which identify more distinguishable features in these cases. We also observe that, encoded $TOTAL^6$ representation can distinguish the better performing transformation and amplifies it to increase the retrieval performance. This is depicted in Figure 11 in which we plotted how the performance of individual queries change with respect to retrieval method (NN vs CBD_2) and transformation (TS vs $TOTAL^6$). If the precision in the first round of RF is very low for the query, diversity RF has a minimal positive effect. The highest gains are seen in the middle range of precision (0.2-0.7) where there is room for improvement and enough information for the RF mechanism to work. It is also evident that choice of representation can change the end result significantly for all query cases.

4.2.4 Runtime Performance

We present the runtime performance of the methods, initially with the times of the transformations into different representations. The computation platform is MATLAB running on a Windows 10 with Intel i7 4720HQ 2.6 GHz processor and 16 GB of RAM. The accumulated runtime for all the 85 datasets is provided in Table 5. SAX-Bitmap transformation is the slowest transform with a significant difference. We think that this is mainly due to very efficient FFT libraries available in MATLAB which is also used extensively in CWT transformation code.

TABLE 5
Total transformation runtime for all the datasets (minutes)

CWT	SAX	FFT
4.28	62.87	0.76

We also examined how the training time of the autoencoder varies with respect to θ parameter and to different transformations. The results for all the datasets summed up is provided in Table 6. We observe that training time increases with the length of encoded data. This is expected since the length of the time series affects the number of nodes and the total number of weights in the network which directly affects the total training times.

TABLE 6
Total training time for autoencoders (minutes)

	$\theta = 3$	$\theta = 6$	$\theta = 9$
TS	127.29	92	79.44
CWT	176.02	119.48	99.84
SAX	63.56	56.78	53.83
FFT	66.69	57.53	53.85
TOTAL	719.17	405.98	308.78

The total experiment runtime (over all the datasets total) is provided in Table 7 with respect to different retrieval methods, sparsity index(θ) and transformation methods. We note the significant reduction in runtime for autoencoded data in which some cases we have 7 fold decrease with respect to full time series data. Each transformation has a different runtime performance which depends on the length of the transformed time series, which is expected, as mentioned in Section 3.2 (Average length for different transformation is as follows: TS: 422.2, CWT: 564.3, SAX: 256.0, FFT: 212.0, $TOTAL$: 1454.5). We also see that the diversificiation methods, MMR and CBD, have comparable runtime performances.

5 Conclusion

Even though combinations of diversity and RF have been explored in the field of information retrieval, they have not attracted enough attention for time series analytics. We have explored the use of diverse relevance feedback over time-series that are summarized using autoencoders. The gains in terms of efficiency by autoencoding and in terms of accuracy by relevance feedback provide a promising solution that could increase user satisfaction in time series retrieval applications. Experimental results from 85 real data sets demonstrate that regardless of the representation, even with a relatively simple RF model, user feedback increases retrieval accuracy, even in just one iteration. Further accuracy improvements are achieved when diversity within the result set is used for RF in many of the cases. The clustering-based diversity method, without any rigorous parameter optimization, performed better in terms of overall precision. Fine tuning of the diversity balance according to the dataset properties and user objectives can further extend the improvements. The analysis of the results provided evidence in favor of result diversification performing better in relatively non-separable data cases which are the challenging cases.

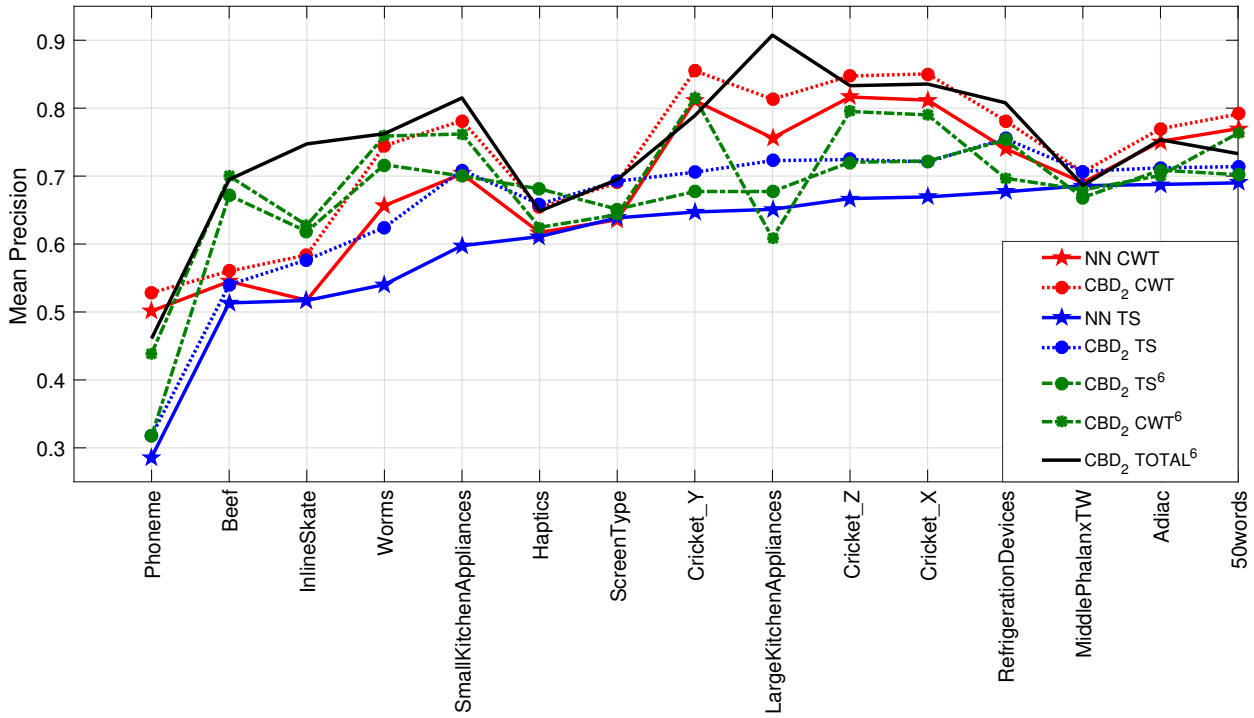

Fig. 12. Performance of RF with various configurations for datasets with low precision

TABLE 7
Total runtime of experiments (minutes)

	NN	MMR_1	MMR_2	CBD_1	CBD_2
TS	119.5	148.7	157.3	136.8	151.8
TS^3	41.6	58.1	66.9	55.4	68.1
TS^6	21.8	35.0	43.9	34.7	46.6
TS^9	15.4	27.6	36.7	28.1	39.6
CWT	125.2	156.1	164.4	142.3	157.8
CWT^3	48.7	66.5	75.9	64.4	77.1
CWT^6	25.2	39.1	49.0	39.8	52.2
CWT^9	18.3	31.3	40.5	31.3	43.4
SAX	65.5	86.9	96.2	80.1	92.9
SAX^3	29.7	44.7	54.3	42.4	54.5
SAX^6	16.5	29.2	38.5	29.0	40.5
SAX^9	11.3	23.1	32.4	23.5	34.8
FFT	61.2	81.1	90.0	76.6	90.0
FFT^3	21.9	35.0	43.9	34.7	46.5
FFT^6	12.0	23.6	32.3	24.6	36.0
FFT^9	8.8	19.9	28.7	21.2	32.7
TOTAL	358.2	426.0	451.3	394.5	439.7
$TOTAL^3$	134.4	166.0	175.1	152.3	168.9
$TOTAL^6$	68.6	90.0	99.0	84.6	98.4
$TOTAL^9$	46.8	64.1	73.1	61.6	74.6

The user feedback can also be used for online selection of the representation by dividing the result list presented to the user and amplifying the suitable representations for the next round. Results show that representation feedback diversifies the result set implicitly because of the use of various representations, which in turn improves the precision even in a case of simple nearest neighbor retrieval.

We studied the use of autoencoder type neural networks to enhance the accuracy of time series retrieval and analyzed it within our setting. The data and the computational load on the retrieval engine have been reduced significantly. An autoencoder trained with combinations of representations as input has yielded meaningful performance improvements which shows the potential of this approach.

Use of autoencoder, and even stacked-autoencoders with larger datasets, to extract useful representations is a potential direction for future work. Different structures of autoencoders can be studied in the context of time series retrieval and analytics.

Acknowledgments

This study was funded in part by The Scientific and Technological Research Council of Turkey (TUBITAK) under grant EEEAG 111E217. We thank the data curators of [44] for providing such comprehensive set.

References

[1] D. J. Berndt and J. Clifford, "Using Dynamic Time Warping to Find Patterns in Time Series," in *Knowledge Discovery and Data Mining*, 1994, pp. 359–370.

[2] M. Gavrilov, D. Anguelov, P. Indyk, and R. Motwani, "Mining the stock market: which measure is best?" in *Proc. of ACM SIGKDD*, 2000, pp. 487–496.

[3] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: Experimental comparison of representations and distance measures," *Proc. VLDB Endow.*, vol. 1, no. 2, pp. 1542–1552, Aug. 2008.

[4] A. Camerra, T. Palpanas, J. Shieh, and E. J. Keogh, "isax 2.0: Indexing and mining one billion time series," in *Proc. of IEEE ICDM*, 2010, pp. 58–67.

[5] B. Eravci and H. Ferhatosmanoglu, "Diversity based relevance feedback for time series search," *Proc. of VLDB*, vol. 7, no. 2, pp. 109–120, 2013.

[6] J. Rocchio, *Relevance feedback in information retrieval. In: The SMART Retrieval System Experiments in Automatic Document Processing.*, Englewood Cliffs, N.J.: Prentice Hall, 1971, pp. 313–323.

[7] G. Salton, Ed., *The SMART Retrieval System Experiments in Automatic Document Processing.*, Englewood Cliffs: Prentice-Hall, 1971.

[8] G. Salton and C. Buckley, "Improving retrieval performance by relevance feedback," *Journal of the American Society for Information Science*, vol. 41, pp. 288–297, 1990.

[9] X. S. Zhou and T. S. Huang, "Relevance feedback in image retrieval: A comprehensive review," *Multimedia systems*, vol. 8, no. 6, pp. 536–544, 2003.

[10] M. L. Kherfi, D. Ziou, and A. Bernardi, "Combining positive and negative examples in relevance feedback for content-based image retrieval," *J. Visual Communication and Image Representation*, vol. 14, no. 4, pp. 428–457, 2003.

[11] Y. Rui, T. S. Huang, S. Mehrotra, and M. Ortega, "A relevance feedback architecture for content-based multimedia information retrieval systems," in *Proc. IEEE Workshop on Content-Based Access of Image and Video Libraries*, 1997, pp. 82–89.

[12] Z. Su, H. Zhang, S. Li, and S. Ma, "Relevance feedback in content-based image retrieval: Bayesian framework, feature subspaces, and progressive learning," *Image Processing, IEEE Transactions on*, vol. 12, no. 8, pp. 924–937, 2003.

[13] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *Proc. ACM MULTIMEDIA*, 2001, pp. 107–118.

[14] J. Carbonell and J. Goldstein, "The use of mmr, diversity-based reranking for reordering documents and producing summaries," ser. Proc. of SIGIR '98, 1998, pp. 335–336.

[15] A. Borodin, A. Jain, H. C. Lee, and Y. Ye, "Max-sum diversification, monotone submodular functions, and dynamic updates," *ACM Trans. Algorithms*, vol. 13, no. 3, pp. 41:1–41:25, Jul. 2017.

[16] C. Charles, K. Maheedhar, C. Gordon, V. Olga, A. Azin, B. Stefan, and M. Ian, "Novelty and diversity in information retrieval evaluation," in *Proc. of ACM SIGIR*, 2008, pp. 659–666.

[17] D. Rafiei, K. Bharat, and A. Shukla, "Diversifying web search results," in *Proc. of ACM WWW*, 2010, pp. 781–790.

[18] M. Hasan, A. Mueen, and V. Tsotras, "Distributed diversification of large datasets," in *2014 IEEE International Conference on Cloud Engineering*, March 2014, pp. 67–76.

[19] X. Zuobing, A. Ram, and Z. Yi, "Incorporating diversity and density in active learning for relevance feedback," in *Proc. of European Conference on IR research*, 2007, pp. 246–257.

[20] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong, "Diversifying search results," in *Proc. of ACM WSDM*, 2009, pp. 5–14.

[21] F. Radlinski and S. Dumais, "Improving personalized web search using result diversification," in *Proc. of ACM SIGIR*, 2006, pp. 691–692.

[22] O. Kucuktunc and H. Ferhatosmanoglu, "l-diverse nearest neighbors browsing for multidimensional data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 3, pp. 481–493, 2013.

[23] F. Radlinski and T. Joachims, "Active exploration for learning rankings from clickthrough data," in *Proc. of ACM SIGKDD*, 2007, pp. 570–579.

[24] F. Radlinski, R. Kleinberg, and T. Joachims, "Learning diverse rankings with multi-armed bandits," in *Proc. of ACM ICML*, 2008, pp. 784–791.

[25] T. Joachims, A. Swaminathan, and T. Schnabel, "Unbiased learning-to-rank with biased feedback," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, ser. WSDM '17, 2017, pp. 781–789.

[26] K. Hofmann, S. Whiteson, and M. de Rijke, "Balancing exploration and exploitation in learning to rank online," in *Proc. of European Conference on Advances in Information Retrieval*, 2011, pp. 251–263.

[27] T.-Y. Liu, "Learning to rank for information retrieval," *Found. Trends Inf. Retr.*, vol. 3, no. 3, pp. 225–331, Mar. 2009.

[28] N. Rubens, D. Kaplan, and M. Sugiyama, "Active learning in recommender systems," in *Recommender Systems Handbook*. Springer, 2011, pp. 735–767.

[29] T.-C. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164 – 181, 2011.

[30] R. Agrawal, C. Faloutsos, and A. N. Swami, "Efficient similarity search in sequence databases," in *Proc. of ACM FODO*, 1993, pp. 69–84.

[31] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space." *Intell. Data Anal.*, vol. 11, no. 5, pp. 561–580, 2007.

[32] E. Keogh and M. Pazzani, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback," in *Proc. of KDD*, 1998.

[33] E. Keogh and M. J. Pazzani, "Relevance feedback retrieval of time series data," in *Proc. of ACM SIGIR*, 1999, pp. 183–190.

[34] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," D. E. Rumelhart, J. L. McClelland, and C. PDP Research Group, Eds. Cambridge, MA, USA: MIT Press, 1986, ch. Learning Internal Representations by Error Propagation, pp. 318–362. [Online]. Available: http://dl.acm.org/citation.cfm?id=104279.104293

[35] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[36] N. Gianniotis, S. D. Kgler, P. Tio, and K. L. Polsterer, "Model-coupled autoencoder for time series visualisation," *Neurocomputing*, vol. 192, pp. 139 – 146, 2016.

[37] N. Ahmed, A. Atiya, N. E. Gayar, and H. El-Shishiny, "An Empirical Comparison of Machine Learning Models for Time Series Forecasting," *Econometric Reviews*, vol. 29, no. 5-6, pp. 594–621, 2010.

[38] N. Kumar, N. Lolla, E. Keogh, S. Lonardi, and C. A. Ratanamahatana, "Time-series bitmaps: a practical visualization tool for working with large time series databases," in *SIAM Data Mining Conference*, 2005, pp. 531–535.

[39] I. Selesnick, R. Baraniuk, and N. Kingsbury, "The dual-tree complex wavelet transform," *Signal Processing Magazine, IEEE*, vol. 22, no. 6, pp. 123 – 151, nov. 2005.

[40] H. Zamani, J. Dadashkarimi, A. Shakery, and W. B. Croft, "Pseudo-relevance feedback based on matrix factorization," in *Proc. of ACM CIKM*, 2016, pp. 1483–1492.

[41] J. Miao, J. X. Huang, and Z. Ye, "Proximity-based rocchio's model for pseudo relevance," in *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '12, 2012, pp. 535–544.

[42] I. Ruthven and M. Lalmas, "A survey on the use of relevance feedback for information access systems," *Knowl. Eng. Rev.*, vol. 18, no. 2, pp. 95–145, Jun. 2003.

[43] B. Liu and H. V. Jagadish, "Using trees to depict a forest." *PVLDB*, vol. 2, no. 1, pp. 133–144, 2009.

[44] E. Keogh, Q. Zhu, B. Hu, Y. Hao, X. Xi, L. Wei, and C. A. Ratanamahatana, "The ucr time series classification/clustering homepage," 2011. [Online]. Available: www.cs.ucr.edu/~eamonn/time_series_data/

Bahaeddin Eravci is working towards his Ph.D. degree in computer science at Bilkent University. His research interests are temporal and spatio-temporal data mining and data management.

Hakan Ferhatosmanoglu is a Professor at University of Warwick, UK, and Bilkent University, Turkey. He received the Ph.D. degree in Computer Science from University of California, Santa Barbara in 2001. His current research interests include scalable management and analytics for multi-dimensional data. He received Career awards from the US Department of Energy, US National Science Foundation, Turkish Academy of Sciences, and Alexander von Humboldt Foundation.