Sparse Sampling for Sensing Temporal Data – Building an Optimized Envelope

Menachem Domb*, Guy Leshem†, Elisheva Bonchek-Dokow‡, Esther David§
Computer Science Department
Ashkelon Academic College
Ashkelon, Israel
Email: *dombmnc@acad.ash-college.ac.il,
†gialsm@acad.ash-college.ac.il,
‡elishevabd@acad.ash-college.ac.il,
§astrdod@acad.ash-college.ac.il

Yuh-Jye Lee
Department of Applied Mathematics
National Chiao Tung University
Hsinchu City, Taiwan
Email: yuhjye@math.nctu.edu.tw

***Abstract*—IoT systems collect vast amounts of data which can be used in order to track and analyze the structure of future recorded data. However, due to limited computational power, bandwith, and storage capabilities, this data cannot be stored as is, but rather must be reduced in such a way so that the abilities to analyze future data, based on past data, will not be compromised. We propose a parameterized method of sampling the data in an optimal way. Our method has three parameters– an averaging method for constructing an average data cycle from past observations, an envelope method for defining an interval around the average data cycle, and an entropy method for comparing new data cycles to the constructed envelope. These parameters can be adjusted according to the nature of the data, in order to find the optimal representation for classifying new cycles as well as for identifying anomalies and predicting future cycle behavior. In this work we concentrate on finding the optimal envelope, given an averaging method and an entropy method. We demonstrate with a case study of meteorological data regarding El Ninio years.**

I. Introduction

On IoT (Internet of things), sensor networks and other applications, we often have sequential data from which we would like to extract useful knowledge. Given the nature of IoT applications, limited computational power and bandwidth resources prohibit us from large-scale data collection. Data may be large in its number of samples, its dimensionality, or both. In general, data aggregation can help to summarize data into a compressed set, where the compressed set preserves most of the critical information from the original data. Working on the compressed data can reduce the complexity of the subsequent knowledge discovery task to a traceable version, without compromising on performance at the task. One special case of data aggregation is sampling, which focuses on selecting only some of the samples from the original data, uniformly or non-uniformly.

In this work, we attempt to find an optimal or close to optimal sampling approach to selecting data for knowledge discovery. In particular, we work on finding the best sampling strategy given sequential data that is generated from IoT applications.

Dictionary learning [1] is known to be useful for extracting patterns hidden in data. We can apply dictionary learning to sequential data, for tasks such as natural language processing, video analysis; as well as to non-sequential data. Given most IoT data that is collected in sequential form, we can find a method that maintains a basis, where we have enough elements in the basis to describe most of the sequential patterns existent in the data. For instance, in an ITS (Intelligent Transportation System), we may use vehicle GPS or other telematics traces to extract driving behavior from drivers. This can help us to collect a set of common sequential patterns from the sequential telematics data. In a smart home system, we may collect a set of most frequent activity trajectories for home members to be used for member authentication.

The problem we discuss here, is finding an optimal sampling method given a set of time series, where we can have the same or similar information before and after the sampling reduction process, as far as the purpose the data serves in the context of the relevant application.

The input given is a set of time series: $D = \{d^{(1)}, d^{(2)}, ..., d^{(n)}\}$, with each time series of the form: $d^{(i)} =< d_1^{(i)}, d_2^{(i)}, ..., d_k^{(i)}, ... >$, and each data point within a series is a pair (time stamp, numeric value): $d_k^{(i)} = (t_k^{(i)}, x_k^{(i)})$. The required output is an optimal set $D_w = \{a_1, a_2, ..., a_m\}$, where a_i can be any sampling artifact, such as a minimal data set, trends, graphs, measurements or rules, which strongly represents and supports the purpose of collecting the original

978-1-5090-5732-0/16/$31.00 ©2016 IEEE

data set D.

How to judge whether D_w indeed represents and supports the purpose of collecting the original data set D, depends on the intended application. There are many known data reduction techniques, which mainly enable restoring the original data set from the reduced one. Among these are compression and compaction routines and dictionary methods. Our purpose is not to be able to restore the original data, but rather to serve best the purpose for which the data was collected.

In particular, we focus on two possible purposes. Given the sequential data, one purpose could be classification, and another, time series prediction.

Series Classification: In this problem, we consider the set D_w and the full data set D as containing the same information, if they produce the same classifier [2]. That is, if $f(d) = f_w(d) \in \{-1, 1\}$ for every new data series d, where f is a classifier learned from D and f_w is a classifier based on D_w. For instance, we can judge whether a series of yearly temperatures represent an El-Ninio year or not, or whether a series of sensor data is characteristic of a suspected intrusion or not.

Time Series Prediction: Here we consider two sets D and D_w as containing the same (or similar) information if both are able to predict the future pattern of an initial series d. That is, we can use either D or D_w to predict a future item d_n with similar accuracy [1].

In this work we deal mainly with classification. The rest of this article is organized as follows: in the next section, we discuss related work. In Section III we detail our proposed method for creating D_w from the given D, namely, by creating what we call an envelope around an averaged series. Section IV introduces the data we used and shows experimental results of our proposed method. In Section V we conclude our work and outline future directions.

II. Related Work

Real-world data typically contains repeated and periodic patterns. This suggests that the data can be effectively represented and compressed using only a few coefficients of an appropriate basis. However, distance estimation when the data is represented using different sets of coefficients is still a largely unexplored area. Vlachos et al. [3] formulate the problem of estimating lower/upper distance bounds as an optimization problem and establish the properties of optimal solutions to develop an algorithm which obtains an exact solution to the problem. It is applicable to any sequential or high-dimensional data and any orthogonal data transformation. Sakurada and Yairi [4] use auto-encoders with nonlinear dimensionality reduction for the anomaly detection task. Their work demonstrates the ability to detect subtle anomalies where linear PCA fails. Reeves et al. [5] present a framework to archive and query massive time series streams. They apply multi-scale analysis to decompose time series and to obtain sparse representations in various domains. Chilimbi and Hirzel [6] implement a dynamic pre-fetching scheme that operates in several phases. First, profiling, which gathers a temporal data reference profile from a running program. Next, an algorithm for extracting hot data streams, which are data reference sequences that frequently repeat in the same order. Then, dynamically injecting code at appropriate program points to detect and pre-fetch these hot data streams. Finally, the process enters the hibernation phase where the program continues to execute with the added pre-fetch instructions. At the end, the program is de-optimized to remove the inserted checks and pre-fetch instructions, and control returns to the profiling phase. Lane and Brodley [7] present an approach on the basis of instance-based learning (IBL) techniques. Classification boundaries are selected from an a posteriori characterization of valid user behaviors, coupled with a domain heuristic. An empirical evaluation of the approach on user command data demonstrates that we can accurately differentiate the profiled user from alternative users when the available features encode sufficient information. Furthermore, they demonstrate that the system detects anomalous conditions quickly – an important quality for reducing potential damage by a malicious user. They present several techniques for reducing data storage requirements of the user profile, including instance-selection methods and clustering. An empirical evaluation shows that a new greedy clustering algorithm reduces the size of the user model by 70%, with only a small loss in accuracy. Kasiviswanathan et al. [8] address the problem of identifying emerging topics through the use of dictionary learning. They propose a two stage approach based on detection and clustering of novel user-generated content to derive a scalable approach by using the alternating directions method to solve the resulting optimization problems. Mairal et al. [1] focus on modeling data vectors as sparse linear combinations of basic elements generating a generic dictionary, and then adapt it to specific data. They propose an optimization algorithm for dictionary learning, based on stochastic approximations, which scale up gracefully to large datasets of training samples. Aldroubi et al. [9] prove that given a set of data vectors in a Hilbert space, there exists an optimal collection of subspaces minimizing the sum of the square of the distances between each vector and its closest subspace in the collection. This collection of subspaces gives the best sparse representation for the given data and provides an optimal model for sampling in union of subspaces. Rubinstein et al. [10] survey the various options up to the most recent contributions and structures. Cherian et al. [11] propose learning over-complete dictionary models where the signal is allowed to have both Gaussian and (sparse) Laplacian noise. Dictionary learning in this setting leads to a difficult non-convex optimization problem, which is further exacerbated by large input datasets. Duarte-Carvajalino and Sapiro [12] introduce a framework for the joint design and optimization of the nonparametric dictionary and the sensing matrix. They demonstrate the use of random sensing matrices and those matrices that are optimized independently of the learning of the dictionary. The presentation of the framework and its efficient numerical optimization is complemented with classical image datasets.

The common underlying idea of the reviewed approaches

is the definition of the problem they are aiming to solve. The problem attempted to be solved is, optimizing the size of the collected sampling data so that it keeps the proper balance between the quantity of sampling data and the information extracted from it. Hence, looking for ways to manipulate the sampling data and reduce it to a manageable size and still be able to extract from it the desired information as it is able to do using the original sampling data.

Our problem statement focuses on extracting concepts, methods, rules and measurements so that, at the end of the process, the original sampling data becomes redundant and need no longer be stored. However, to keep improving and adjusting the extracted artifacts to natural changes in the behavior of the sampled mechanism, we incorporate in our approach an ongoing learning process. In addition, in our study we concentrate on time dependent streaming sampling data, divided by fixed periods so that we are able to repeat our analysis process for each period/cycle. Thus, while there are many classification algorithms using time series sampling, our aim here is not to compare the performance of yet another classifier, rather our purpose is to present a flexible method for compactly representing the data, with several parameters which can be chosen and adjusted.

Indeed, some of the reviewed work, such as Reeves et al. [5], can be revisited and be adjusted to our problem statement and so become a valid alternative to the approach we present.

III. Envelope Based Approach

As mentioned, our proposal assumes periodic data sampling and extraction of logical artifacts at period level. In a glance, we analyze sampling data collected over several periods. We divide the period to time units. For example, for a year period, we divide it into daily time units. For each time unit we extract one value representing it. This is done by averaging the samples collected during the time unit. In our example we may calculate the average value of all samples of that day. We may also decide to select one of the samples to represent the day, e.g., the first or last sample. We then calculate the average value for each time unit out of the collected values for the same time unit in all periods, resulting with an average value for a given time unit. We repeat this process for all time units in the period and get a graph representing the average values for an average and common period.

Assuming we have the average graph line for an average period, we now calculate the envelope around this average. The generated envelope represents the standard range of values such that an unanalyzed period can be compared to this envelope. If its graph value is completely within the envelope it means that this period is a standard period. If it is completely out of the envelope then it is purely an out of standard. In case sections of the graph are within the envelope while other sections are out of it, we use an entropy measure to calculate the overall ”distance” of the given period from the standard envelope. Assuming an existing entropy threshold, we are able to decide if the period in a standard period or not.

We apply the same concept at the unit level and decide if a specific time unit in a period is within the standard or not. This specific check is relevant, for example, to anomaly detection of IoT behavior.

In summary, the entire process is based on three key elements: average graph per period, envelope around the average graph, and an entropy value representing the overall distance of a period from the envelope.

Each of these elements– average, envelope and entropy– can be one of several possibilities. For example, as an average measure, the straighforward arithmetic average could be used. But the geometric or harmonic, or any other, could just as well be evaluated and used. For the envelope, a simplistic choice would be min-max values. Alternatively, standard deviation or confidence interval could be employed. For the entropy measure, Shannon’s original definition could be taken, but this has variations as well. These three elements effect each other, and every choich of such a triplet– average, envelope and entropy– will produce a different behaviour of the compressed classifier. Our object is to find the best such triplet, so as to be able to disregard the original data after extracting the representing envelope from it, without compromising on our ability to succesfully analyze future series. In our current work we consistently use the arithmetic average and classical entropy, and focus on finding the best choice of envelope.

A. Finding the Optimal Envelope

We begin with a supervised learning approach, for classification, in which each time series is labeled as one of two classes. To demonstrate, using the data set from our experiments, the time series are year-long recordings of temperature samplings, labeled as positive, if the corresponding year was an El-Ninio year, or negative, otherwise.

We now describe in detail the process of building the classifier, with emphasis on finding the optimal envelope. Optimality here is in the sense of best predictive output, that is, finding the envelope parameters which result in the classifier with the best results.

1) Collect several cycles of data from a given sensor. For example, daily temperature collected over a year is considered one cycle. Each record in the cycle contains a time stamp and a numeric value. These cycles are labeled positive or negative, e.g. ”El-Ninio Year” and ”Non-El-Ninio Year”.
2) Veryify that the collected data truly represents the sensor’s normal and abnormal behavior.
3) Calculate the average cycle for each class, using a chosen average function, by averaging at every time stamp over all the cycles in the respective class.
4) Calculate the envelope around the average, for each class, using different envelope functions– Min-Max, Standard Deviation, and Confidence Interval– by taking at every time stamp, the calculated interval above and below the average.

The process of constructing the best envelope is described in Figure 1. We begin with raw data collected during N peri-

ods generating corresponding data cycles, where each record corresponds to a value related to a specific time unit. For example, a cycle is a year and a time unit is a day. These cycles have already been classified positive or negative according to some classification criteria. These classified cycles will later be used to determine the best envelope among several proposed envelopes. This data is filtered and cleansed by removing defected records and merging records to get one record per time unit. After this preliminary data preparation, we commence with the process.

The process is divided into 4 stages. In stage 1 we use a selected average method and calculate the average graph line representing the N given cycles. This is done horizontally by calculating the average of the values related to the same time unit across all N cycles. For example, we calculate the average of the values for 01-Jan across the various years. Doing so for all time units will generate the average graph line. In stage 2 we select several distance calculation methods and for each method we construct its associated envelope. This is done by calculating the distance value for each distance method, e.g. Min/Max difference, Standard Deviation, and Confidence Interval. Taking the distance value we add and subtract it from the average line to get the envelope around the average. We repeat this process for all distance methods. At this stage we have constructed several envelopes around the average line. Our goal now is to select the envelope which is most effective in classifying unclassified cycles. This is done in Stages 3 and 4. In Stage 3 we use one entropy method with an associated threshold value. An unclassified cycle with an entropy value lower than the threshold will be classified positive, and negative otherwise. For each envelope we calculate the entropy of the given classified cycles. The result is a set of entropy values, where some are below the threshold while other are above it. In Stage 4 we calculate the prediction power for each envelope and select the one with the highest prediction power. This is done by summing, for each envelope, the number of cases its prediction was right, and calculating the average entropy of these correctly classified cycles. We do the same for wrong predictions.

We repeat this for all classified cycles. We then sum up the number of correct predictions and their total entropies. We do the same for wrong predictions. We then subtract the total wrong numbers from the correct numbers. To get the final measure, we multiply these two numbers and get a measure representing the prediction power of the corresponding envelope. We repeat this process for all the constructed envelopes. We'll select the envelope having the highest prediction power.

B. Calculating the Entropy

The entropy of a period, given an envelope, is calculated as follows: Mark for every time stamp whether the cycle's value at that timestamp is below, within, or above the envelope. Calculate the frequency of each of these three possibilities– below (p_1), within (p_2) and above (p_3)– and use these as a ternary probability distribution, which entropy is calculated according to the formula

$$\sum_{i=1}^{3} p_i log(p_i)$$

.

The entropy measure is expected to return its minimum value– i.e., 0– at the two extreme cases: when the cycle graph is entirely contained within the envelope, and when the cycle graph lies entirely outside of the envelope. The first case fits in with our model, since a low entropy value would lead to the correct conclusion that the cycle is of the type which the envelope represents. However, the second case would seemingly cause a false positive classification. We are not bothered by this, since we don't expect such a sharp exclusion from the envelope. Rather, all cycles are expected to fall mostly within the envelope, and those which diverge enough from the envelope, will have a high entropy value which will lead to the right conclusion.

Calculation of the entropy threshold is an outcome of a sensitive analysis we conduct on the entropy value.

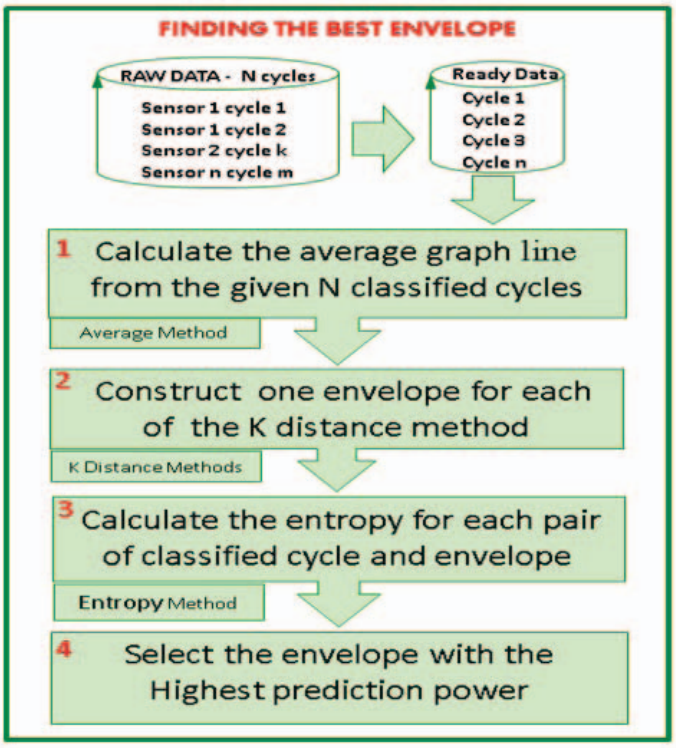

Fig. 1. Process of Finding the Optimal Envelope

C. Classifying a Cycle/Period

Figure 2 describes the process of classifying new, unlabeled data cycles, as we now elaborate:

1) Apply the given data cycle to the envelope, matching up according to time stamps.

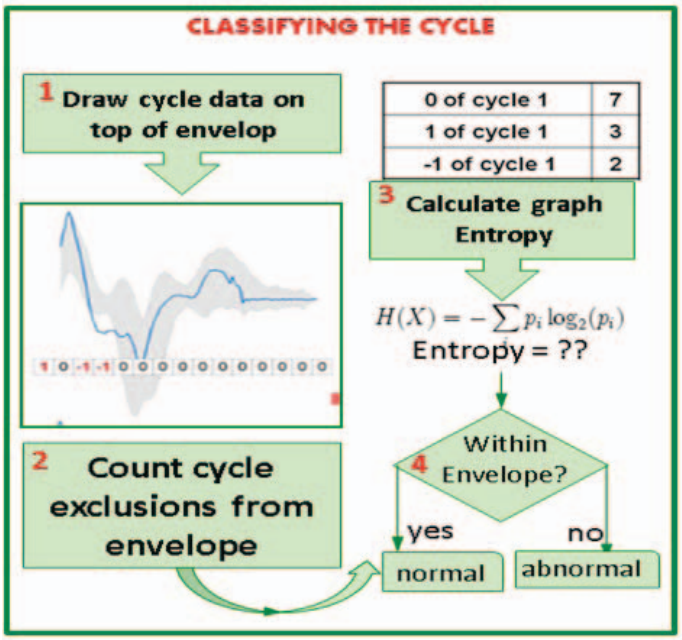

Fig. 2. Classifying a Cycle

2) Note for each time stamp whether the data point is below the envelope, within it, or above it. Mark these cases respectively as -1, 0, +1.
3) Calculate the frequencies of each of the three vaules: -1 (p_1), 0 (p_2), +1 (p_3).
4) Calculate the entropy of the distribution defined by p_1, p_2, p_3.
5) Classify as belonging to the envelope, if the entropy is below the threshold determined in the learning phase.

D. Using the Envelope for Diagnosis and Prognosis of New Ongoing Cycles

In addtion to its use for classification, our proposed method can also be used for times series prediction.

E. Advantages of Proposed Technique

Our proposed technique for defining an envelope as a sparse representation of the data, has several advantages over other existing methods.

Firstly, our technique is actually a family of sampling methods, where each member of this family is defined by the three parameters described above: every choice of average calculation, envelope definition, and entropy formula defines a method which can be applied to the data. It is reasonable to expect that different data sets will require different parameters for the best sampling. Different combinations can be tested and evaluated, before being applied to real-time data, in order to ensure optimal treatment of the data. It is also possible to dissect the data cycles into different time intervals, and use different parameters for each section, for data which requires such temporal differentiation. The technique we propose is therefore flexible and adjustable, made to fit for every given data set.

Secondly, this technique can be applied not only for classification but also for time series prediction: data cycles can be diagnosed at real-time, before they are completed and classified, by identifying trends in the cycle graph, as compared to the class envelopes, and by calculating and updating the entropy as the data streams in.

Thirdly, our technique can be used to evaluate reliability of data, online. In cases of high fluctuations or sharp changes in the cycle graph, which don't conform to either of the two class envelopes, it can be suspected that data reliability has been compromised. This can hint to the fact that the sensor is damaged, or that there has been a security breach.

Fourthly, our approach allows self learning and automatic adjustments in case the common behavior changes and a new standard has been established. As mentioned, trends in the cycle graph and envelope can be identified within a cycle. However such trends can be repeated and become the normal case. In such a case, the system can identify the situation based on a pre-defined criteria, and then an adjusted cycle graph may be proposed, such that the number of exceptions is reduced and is manageable.

Fifthly, once in a while, we may run a post mortem and check the system reaction to the actual behavior, and so try to adjust system parameters accordingly.

F. Anomaly Detection for IoT Security

IoT devices generate time related data, i.e. structured records containing a time stamp and one or more numeric values. In many cases we are able to identify recurrent time frames where the system behavior has a repeating format. Hence, IoT data is of a structure to which the envelope approach is highly applicable.

In IoT security we are looking for common data patterns and quantitative measurements. Based on the identified patterns and measurements we are able to extract logical rules that will be executed once an exception is discovered. An exception may be any violation of predefined patterns, measurements and other parameters which represent normal, standard and permitted behavior. In IoT there is an abundance of possible patterns, starting with column level patterns up to a super internet controlling several IoT networks. Our goal is to find the methods and tools to define standard patterns and how they can be identified. Once this is done we can apply the envelope method. In our work we show a two dimension envelope. Using the same concept we can expand it to be a multi dimension envelope. This case is common where there is a dependency among several columns within one record and the same applies for the case where there are dependencies among networks of IoT systems.

IV. Case Study

In the following case study, we do not present a detailed quantitative analysis. Rather, we show only the results which are relevant to this specific data set, since our aim is not to

go into the details, but rather to show in a general way that this is a viable approach. A more rigorous study of the data will be performed in the future. We used meteorological data collected about El-Ninio years (positive class, EN) and Non-El-Ninio years (negative class, NEN), from 1980 to 1998. For the positive envelopes, we took data from the El-Ninio years 1982, '83, '87, '88, '91 and '92. All other years in the range were Non-El-Ninio years. Of all the data recorded, we regarded only the time stamp and the sea-level temperature. We tested three methods for generating envelopes:

1) Minimum over all cycles and Maximum over all cycles.
2) Average cycle $\pm$ standard deviation.
3) Confidence Interval (CI).

Figures 3 and 4 show the envelopes for NEN years. Figure 3 shows two envelopes: the min-max envelope and the average $\pm$ standard deviation envelope. The Y-axis in these graphs is the temperature value, and the X-axis is the time. Within each envelope, the year 1995– a NEN year– is graphed. Its entropy is 0.3631 for the average $\pm$ standard deviation envelope, and 0.2932 for the min-max envelope. Both are below the threshold, which leads to the correct conclusion that it should indeed be classified as NEN.

Figure 4 shows in black, the NEN envelope according to the average $\pm$ standard deviation, and depicts how EN years diverge from this envelope, as compared to the NEN year, 1995. The 1992 and 1988 EN years show clear divergence from the envelope, while 1995, a NEN, is more contained within the envelope. This is nicely captured by the entropy values, which for 1992 was 0.4266 and for 1988 was 0.3857– above the threshold, leading to the conclusion that they are not NEN years– while for 1995 the entropy was 0.3631– significantly lower than those of the EN years, leading to the correct conclusion that 1995 is indeed a NEN year.

In the case study, we compared the envelopes constructed by using the average graph $\pm$ standard deviation and the average graph $\pm$ min-max. For the standard deviation envelope we got a significant entropy value difference, between a classified EN case and a NEN case. In comparison, the min-max envelope resulted in close values of entropy for the EN cycle and the NEN cycle. This means that the ability to differentiate between two extreme situations, using entropy, depends on the parameter used to build the envelope. Hence, in general we have to explore other parameters for defining the distance between the average line to the envelope border lines. For every such calculation we calculate the corresponding entropy for the two extreme values. Then test it with already classified data and see if the calculated entropy allows the clear differentiation between positive and negative cases.

V. Conclusions and Future Work

In this study we dealt with the classification problem of an unclassified cycle of streaming data, which is common in IoT. Classification has recently gained much attention due to rising IoT security issues and threats. We introduced the envelope approach to draw the borders around the standard area representing a specific class. In case of an unclassified

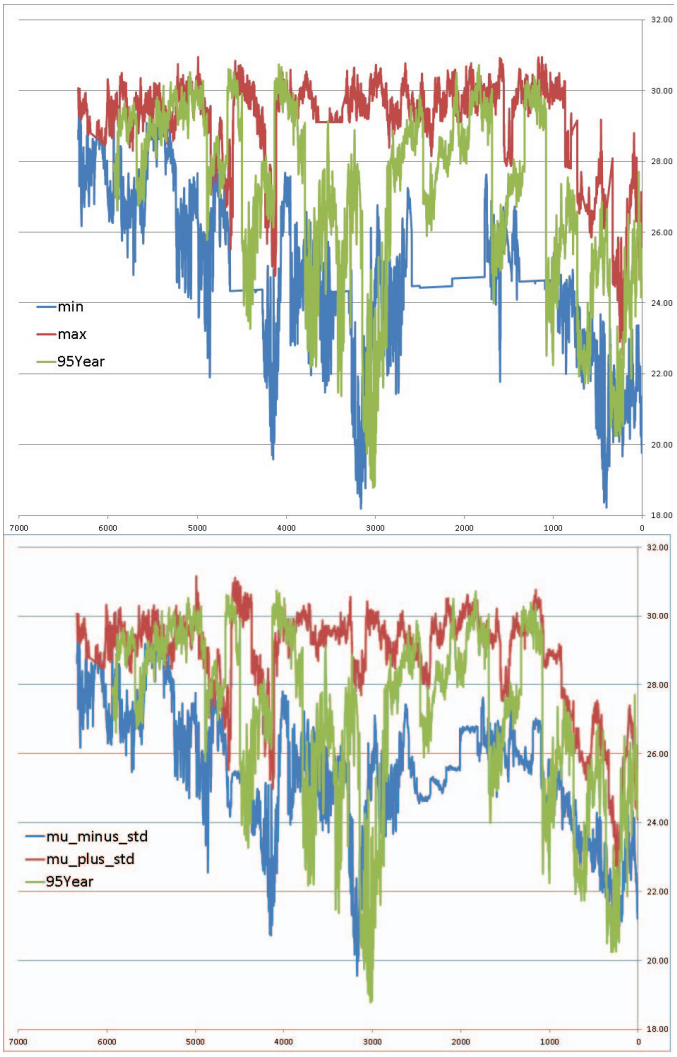

Fig. 3. NEN Envelopes– Min-Max and Standard Deviation

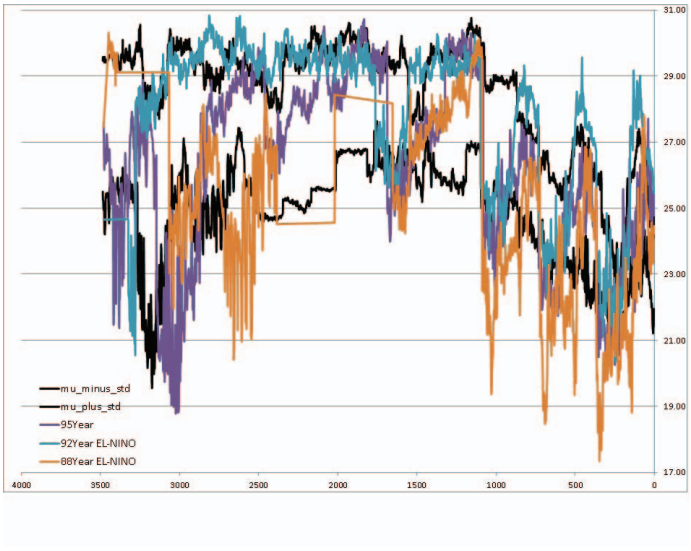

Fig. 4. EN Cycles on NEN Average $\pm$ Standard Deviation Envelope

cycle, we measure its distance from the envelop using an entropy formula. We then compare the result to a predefined threshold. If the entropy value is below the threshold, the cycle is of the same class.

We propose a process for constructing the best envelope that most probably classifies to the correct underlying class. The process is based on three measurement methods: average, distance and entropy. For each method there are several alternate formulas we may use. Each combination of these three methods may result in a different envelope and hence different entropy value, for the same unclassified cycle. To select the best envelope we use several already classified cycles and repeat the classification process using a different method triplet combination, each time. We select the combination with the maximum difference between positive and negative values.

In the current study, we have dealt mainly with the problem of classification. However, as mentioned above, the envelope method can be used for time series prediction as well. Even when the device records data which falls within the envelope, trends can be discovered: we may note an inclination of values, such that an extrapolation shows the cycle leaving the envelope in the future. This can be noticed by entropy values which are still low, but creeping upwards toward the threshold. In such a case, appropriate action can be taken ahead of time, to deal with the situation. For example, in the case of our El-Ninio data, temperature recordings along the year might be showing a consistent shift, enabling the prediction that the graph will leave the envelope, and thus the year will be declared as an EN year, before standard methods would be able to do this.

In addition to the initial construction of the class envelopes from the given data, we suggest ongoing improvements of the initial envelopes: with a periodic analysis of recent data, we may recalculate the class averages and their envelopes, so as to refine and revise the envelopes, for improved classification performance.

As mentioned above, this paper presents a general framework, without going into a detailed analysis. In future work we intend to apply the method to other data sets within a more rigorous experimental setup. Doing so will enable us to improve the envelope approach, in several aspects, such as: determining the minimal number of classified cycles required to define the best envelope, expanding the use of the envelope to discover early trends or discover significant changes in behavior and adjusting the envelope accordingly, exploring the possibility of dividing one cycle into several segments and associating a different envelope method to each segment. Classification is only one way to cope with anomaly detection of data streaming. In future work we will expand our approach to generate rules and measurements for discovering additional anomalies.

VI. Acknowledgements

This research was funded by the Israeli Ministry of Science, Technology and Space.

References

[1] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 689–696.

[2] T. G. Dietterich, "Machine learning for sequential data: A review," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2002, pp. 15–30.

[3] M. Vlachos, N. M. Freris, and A. Kyrillidis, "Compressive mining: fast and optimal data mining in the compressed domain," *The VLDB Journal*, vol. 24, no. 1, pp. 1–24, 2015.

[4] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*. ACM, 2014, p. 4.

[5] G. Reeves, J. Liu, S. Nath, and F. Zhao, "Managing massive time series streams with multi-scale compressed trickles," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 97–108, 2009.

[6] T. M. Chilimbi and M. Hirzel, "Dynamic hot data stream prefetching for general-purpose programs," in *ACM SIGPLAN Notices*, vol. 37, no. 5. ACM, 2002, pp. 199–209.

[7] T. Lane and C. E. Brodley, "Temporal sequence learning and data reduction for anomaly detection," *ACM Transactions on Information and System Security (TISSEC)*, vol. 2, no. 3, pp. 295–331, 1999.

[8] S. P. Kasiviswanathan, P. Melville, A. Banerjee, and V. Sindhwani, "Emerging topic detection using dictionary learning," in *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011, pp. 745–754.

[9] A. Aldroubi, C. Cabrelli, and U. Molter, "Optimal nonlinear models for sparsity and sampling," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 793–812, 2008.

[10] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.

[11] A. Cherian, S. Sra, and N. Papanikolopoulos, "Denoising sparse noise via online dictionary learning," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 2060–2063.

[12] J. M. Duarte-Carvajalino and G. Sapiro, "Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization," DTIC Document, Tech. Rep., 2008.