Real-time stream mining: online knowledge extraction using classifier networks

Luca Canzian and Mihaela van der Schaar

I. Introduction

The world is increasingly information-driven. Vast amounts of data are being produced by different sources and in diverse formats including physiological measurements [1], tweets [2], and multimedia files [3]. Many businesses and government institutions are also embracing automation and relying on a variety of sensors and infrastructure to collect, store, and analyze data on a continuous basis. It is becoming critical to endow assessment systems with the ability to process *streaming* information from sensors in real-time in order to better manage physical systems, derive informed decisions, tweak production processes, and optimize logistics choices.

Stream mining refers to the broad class of techniques that can be used in *sense-and-respond* systems that *continuously* receive data streams from multiple sources and employ *analytics* aimed at detecting multiple concepts and turning the data into actionable information. An example of a Stream Mining Application (SMA) is the surveillance application represented in Fig. 1, which will be adopted as a case study throughout the paper. In this application, multiple aerial and ground reconnaissance videos are collected by different cameras and are processed in real-time by a *network of classifiers* that are trained to detect different high-level semantic features. In practice, the classifiers are localized across distributed and interconnected processing nodes. Fig. 2 shows a specific classifier network for a video stream acquired by a single camera, whereas Fig. 3 shows an example of how the classifiers are localized across distributed and interconnected processing nodes. The configuration of each classifier and the network topology (i.e., how classifiers are interconnected) must adapt to both the characteristics of the video streams and to the currently

The authors are with the Department of Electrical Engineering, UCLA, Los Angeles CA 90095, USA.

The material is based upon work funded by the US Air Force Research Laboratory (AFRL). Any opinions, findings, and conclusions or recommendations expressed in this article are those of the authors and do not reflect the views of AFRL.

The work of Luca Canzian and Mihaela van der Schaar was partially supported by the NSF grant CCF 1218136.

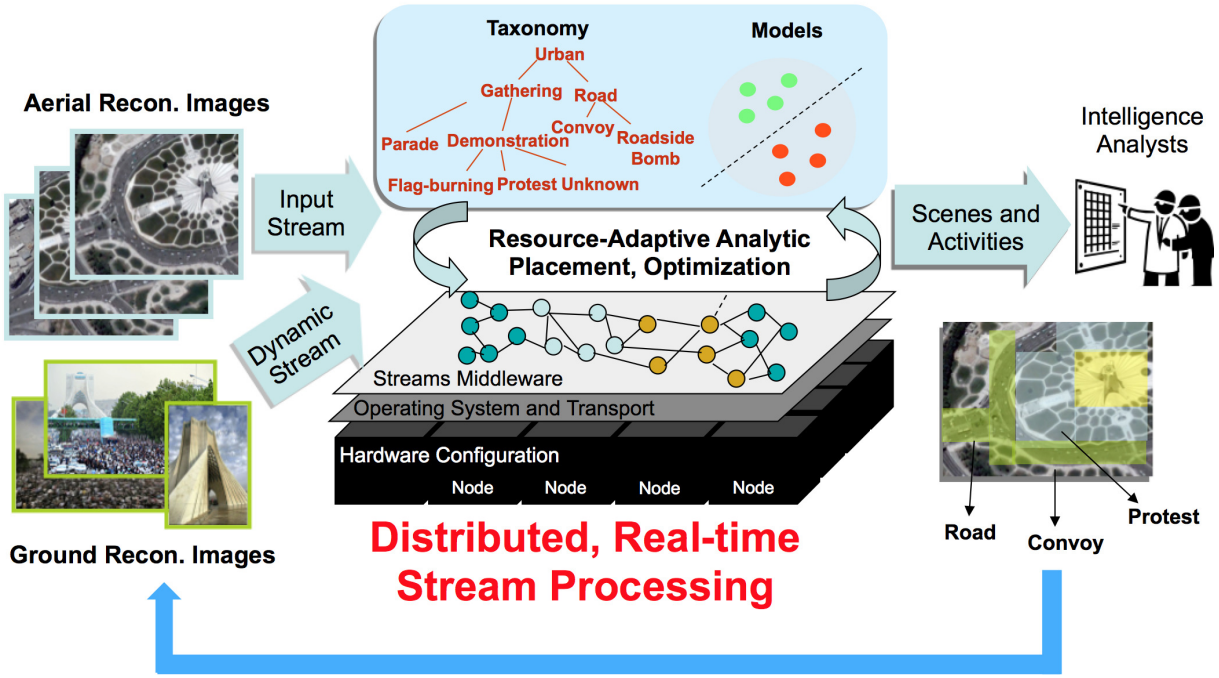

Fig. 1: A surveillance application involving analysis of data streams from multiple sources.

available system resources (e.g., CPU, memory, I/O bandwidth). The results of the processing are finally used to take informed actions, e.g., to turn on or off cameras, to open or close dynamically some roads, and to provide evacuation paths.

The development of efficient stream mining mechanisms to extract the relevant knowledge from the data streams and make timely decisions based on this knowledge will benefit from the confluence of two major scientific and technological shifts: **1)** advances in basic techniques in learning, data mining, and real-time data analytics, and **2)** advances in building, adapting, and managing networks of classifiers. Currently, the focus of the learning and data mining communities has been on the former research agenda, but with the emergence of applications relying on run-time decisions based on high-rate data streams that are analyzed multiple times (in multiple locations and with different analytics), the latter research area is becoming increasingly important.

This magazine paper 1) surveys the works dealing with building, adapting, and managing networks of classifiers, 2) describes the challenges and the limitations of the current approaches, 3) discusses possible directions to deal with these limitations, and 4) presents some open research questions that need to be investigated.

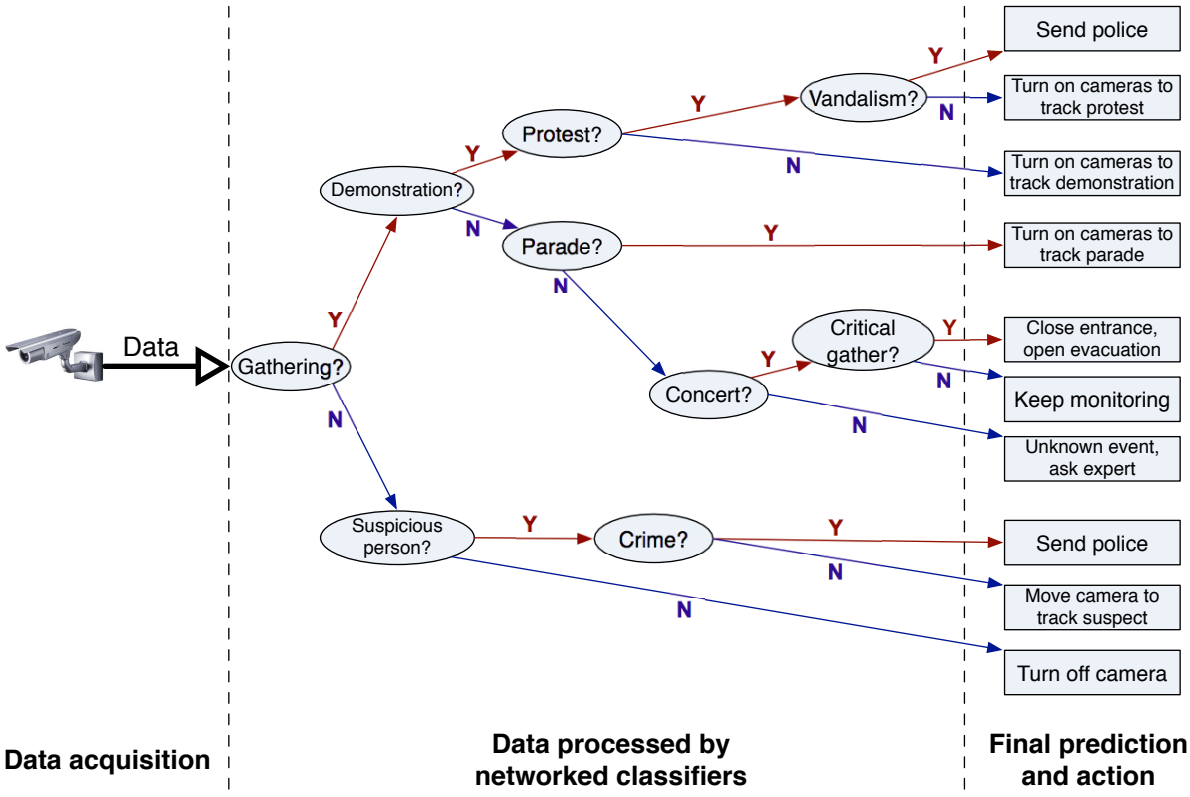

Fig. 2: A hierarchical classifier system for surveillance applications. The acquired data is analyzed by multiple classifiers trained to detect different high-level semantic features. The order at which classifiers are invoked depends on the network topology and on the intermediate processing results.

II. Network of Classifiers

A SMA can be viewed as a processing pipeline that analyzes streaming data from a set of raw data sources to extract valuable information in real-time. Due to the naturally distributed set of data sources and classification tasks, as well as the high computational complexity of the analytics, distributed stream mining systems have been recently developed [4]. These systems leverage computational resources from a set of heterogeneous and distributed processing nodes (ranging from personal computers and residential gateways to large-scale data-centers), and provide the framework to deploy and run different SMAs on various network topologies. In such systems, complex jobs are decomposed into a network of operators in order to enhance scalability, reliability, and allow cost-performance tradeoffs.

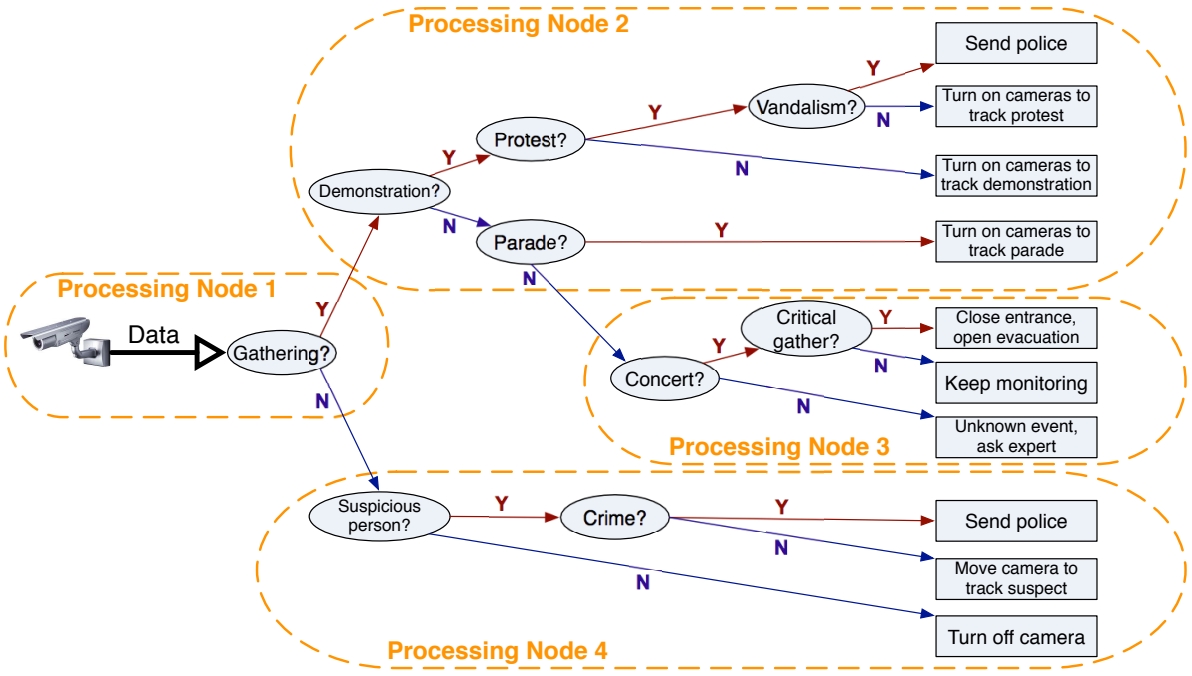

Fig. 3: Classifiers are localized across different autonomous processing nodes.

In this paper, we will focus on stream mining applications that are built using a network topology of low-complexity *binary classifiers*. Binary classifiers are processing elements that partition their input data into two classes of interest. These may be used to filter out irrelevant information or annotate multimedia data with different attributes relevant to the mining task.

Fig. 2 represents a specific network of classifiers that processes the video acquired by a single camera for a surveillance application. Different classifiers are trained to detect different high-level semantic features. For example, the "Gathering" classifier is used to filter the incoming data into two streams, thereby shedding a significant volume of data before passing it to the downstream classifiers: positively identified data is forwarded to the "Demonstration" classifier and the remaining data is forwarded to the "Suspicious person" classifier. Classifiers can be located at the same processing node or across different nodes. For instance, in Fig. 3 the "Gathering" classifier is located in the same node (i.e., Processing Node 1) of the camera, whereas the "Suspicious person" and "Crime" classifiers – that are specialized to detect situations involving a single individual – are located in the Processing Node 4.

Deploying a network of classifiers in this manner enables successive identification of multiple features in the data and provides significant advantages in terms of deployment costs and

processing delays. Indeed, less computing resources are required because data is dynamically filtered through the network and the analysis of multiple data can be carried on in parallel, e.g., meanwhile the current data is analyzed by the "Demonstration" classifier, other data can be acquired and analyzed by the "Gathering" classifier.

This magazine paper will not focus on classifier design, for which many solutions already exist; instead, it will describe how configuring networks of distributed classifiers, while trading off the processing accuracy against the available processing resources or the incurred processing delays. There are two types of configuration choices we will discuss: the ordering of networked classifiers into processing topologies and the local operating points at each classifier. As it will be clear in Section V, this problem can be formulated as a Network Optimization Problem (NOP) [5]. In fact, the classifiers in Fig. 2 act as routers that forward the data to other classifiers. However, differently from traditional NOPs, the classifiers not only route or discard data based on the current network condition, they also need to process the data and to take routing decisions based on both the network conditions and the results of such processing. Importantly, the operating point of one classifier influences the distribution of input data to downstream classifiers; therefore, the configuration of the network and of the operating points of *all* the classifiers are coupled in the optimization problem, making it significantly more complex than traditional NOPs.

III. Challenges for Stream Mining Applications

There is a unique combination of multiple features that distinguishes SMAs from traditional data analysis paradigms. We summarize these features in the following.

Large scale system of autonomous nodes. SMAs need the development of frameworks for knowledge extraction from high volume of distributed data streams. As a consequence, SMAs need decentralized approaches. Indeed, in the surveillance application it would be unfeasible requiring the cameras to acquire the images on a continuous basis with the highest resolution and send them to a central processing unit for complex data analytics; this approach would require huge communication bandwidths and energy consumptions, and long transmission and processing delays. In practice, classifiers needs to be distributed over a set of processing nodes, which can exchange only limited and/or costly messages and can make decisions autonomously, based on their available information. This involves formally defining local objectives and associated inter-node message exchanges that enable the decomposition of the application into a

set of autonomously operating nodes, while ensuring global performance. Moreover, since the autonomous classifiers need to interact to fulfill the demands of an application, the nodes need to learn online the effect of both their experienced dynamics as well as the coupling with the other classifiers with which they interact in meeting the demands of a SMA.

Timing issues. For many SMAs, high-volume data streams (e.g., video data streams) must be processed in real-time. For example, a surveillance application needs to detect with low latency a crime in order to inform the police and catch the criminal before he escapes, and it must foresee situations such as a high and increasing density of people in order to open evacuation paths before the density level becomes critical. SMAs need to meet the desired delay requirements while minimizing misclassification and other costs. This is all the more challenging in a distributed environment, where the synchronization among nodes may not be possible or may lead to sub-optimal designs, as various nodes may experience different environmental dynamics and demands.

Resource constraints. A key challenge in distributed real-time stream mining systems arises from the need to cope effectively with system overload, due to large data volumes and limited system resources (e.g., CPU, memory, I/O bandwidth). In fact, the computational cost incurred by a classifier limits the amount of input data that the classifier can handle. For example, detecting suspicious actions in crowded places may require a processing time that is longer than the acquisition time, this means that the acquired video must be filtered by other low-complex classifiers before invoking the classifier that is responsible to detect suspicious actions.

Learning and adaptation to dynamics. SMAs naturally evolve over time due to (i) heterogeneous and dynamic data stream characteristics, (ii) classifier dependencies, (iii) congestion at shared processing nodes and (iv) communication delays between processing nodes. For example, the detection of criminal activities during exceptional events (e.g., concerts) may be more complex – both in terms of accuracy and resource requirements – than during a normal day. Additionally, several different queries may need to be satisfied by the system, requiring reconfiguration as queries change dynamically. Hence, the proposed solutions must employ online learning algorithms to learn how to adapt the classifier network topology and the classifier operating points in order to cope with all of these experienced dynamics.

IV. Current approaches for stream mining

In this section, we describe the traditional approaches that have been considered to deal with stream mining systems, and we discuss some limitations associated to such approaches. We divide the literature into two parts: we first review the works dealing with the construction of the classifier topology, then we review the works dealing with the adaptation of stream mining systems to resource constraints.

A. Building a network of classifiers

Stream mining systems build upon distributed mining systems and query optimization research [6]. In such systems, stream processing jobs are built as topologies of distributed operators performing feature extraction, classification, aggregation, and correlation [4], [7].

The topology construction problem has been studied as part of the broader pipeline-ordering problem [8], [9] as well as the classical set-cover problem [10] in query optimization. To minimize the end-to-end processing time, these approaches design low-complexity algorithms to order a set of filters applied to streaming data. They use centralized greedy optimization techniques to derive the appropriate order, and develop bounds on performance, in terms of approximation factors to the utility of the optimal solution.

However, the focus of the above works has been on simple and deterministic operators. For instance, they only consider perfect classifiers, i.e., classifiers that make no mistakes. The extension of such approaches to real classifiers – that are subject to mistakes – is not trivial for a number of reasons. First, a classifier mistake can have repercussion on the rest of the network, causing either an increase in the end-to-end processing time or mistakes in the final results of the stream mining system. As a consequence, the system accuracy and the end-to-end processing time must be jointly optimized. Second, classifiers can operate in different configurations in order to tradeoff among different types of errors, and the performance of an individual classifier is coupled to the performance of the other classifiers it is connected to. As a consequence, the optimal topology depends also on the configurations of all the classifiers, and the optimal system design involves a joint optimization over the topology and the classifier configurations.

Additionally, the above approaches do not support dynamic adaptation to environment characteristics and neglect computational and network resource constraints (e.g., CPU, memory, I/O bandwidth). Indeed, they are based on centralized algorithms, which require information about

each classifier operating point to be available at one node, and for that node to manage the entire classifier network. Centralized approaches are limited in their scalability and adaptivity to dynamics, are subject to a single point of failure, and require exchange of information among the nodes of the distributed stream mining system. A key research challenge in distributed stream mining systems is the management of limited network resources, while providing desired application performance.

B. Adaptation to resource constraints

A significant body of work exists which discusses how to adapt stream mining systems to resource constraints [6], [11], [12]. The majority of these approaches are based on load-shedding algorithms, which determine when, where, what, and how much data to discard given the observed data characteristics, quality of service requirements, available resources, and delay constraints.

Naive load shedding performs well for simple data management jobs such as windowing or aggregation, for which the quality of job results depends only on the sample size. However, such simple relationships between quality and sample size do not hold for more complex data classification tasks, where results depend on dynamic statistical properties of the data, semantic relationships between the concepts of interest, underlying features extracted, and selected classification algorithms.

There is also work on intelligent load shedding [13], where a load shedder attempts to maximize a quality of decision measures based on the predicted distribution of temporally-correlated feature values. However, the approach in [13] considers information pertaining to only a single classifier. Without jointly considering the resource constraints at possibly multiple downstream classifiers in the network, the joint classification performance can be highly suboptimal, and the end-to-end processing delay for a cascade of classifiers can become intolerable for real-time applications.

Other proposed solutions include using an ensemble of classifiers with adjustable operating points (e.g., detection thresholds) to enable joint optimization of the quality of classification and resource requirements [14]. However, [14] imposes subset relationships between filters, which may not hold when the classification task aims to find data in the intersection of different high level semantic features, for which no subset relationship exists.

Finally, as for the literature dealing with the topology construction, the majority of approaches for adapting the classifier operating points to resource constraints are based on centralized algorithms, and suffer from the drawbacks discussed in Subsection IV-A.

V. Advanced techniques for stream mining

In this section, we describe frameworks to deal with the SMAs features discussed in Section III and we review relevant works that consider and analyze similar frameworks. Specifically, our focus will be on constructing, managing, and adapting topologies of distributed classifiers deployed on a set of resource constrained and heterogeneous processing nodes. The final goal is on developing learning and adaptive schemes which guarantee the quality of the decisions for the queries, have low delay, are able to adapt to the stream and application dynamics, and gracefully scale to the number of queries and processing nodes.

A. Building the classifier topology: a network optimization problem

SMAs pose queries on data that require multiple concepts to be identified. More specifically, a query is answered as a conjunction of a set of N classifiers $\mathcal{C} = \{C_1, \ldots, C_N\}$, each associated with a concept to be identified. For example, the 10 different queries represented in Fig. 2 (e.g., protests with vandalism acts, civil protests, parades, etc.) are answered as a conjunction of 9 classifiers specialized to detect concepts ranging from a group of individuals that are gathering in a place, to a single individual that is committing a criminal action. By partitioning the problem into this ensemble of classifiers and filtering data successively, the amount of resources consumed by each classifier in the ensemble can be controlled. This justifies using a network of classifiers, where the output of one classifier is passed to subsequent classifiers. For simplicity of exposition, in this subsection we focus on chains of classifiers – see Fig. 4 – but the discussion is equally applicable to a general network topology.

Each binary classifier C_i labels input data into two classes $\mathcal{H}_i$ and $\overline{\mathcal{H}_i}$. The class $\mathcal{H}_i$ is considered, without loss of generality, as the class of interest. Data labeled as belonging to $\mathcal{H}_i$ is forwarded, while data labeled as belonging to $\overline{\mathcal{H}_i}$ is dropped. For example, in the classifier chain represented in Fig. 4, whose goal is to detect whether a protest with vandalism acts is going on, the classifier C_1 has the role to detect whether people are gathering in a place. If C_1 labels the data as belonging to $\overline{\mathcal{H}_1}$ (i.e., people are not gathering), then we can conclude that there is no

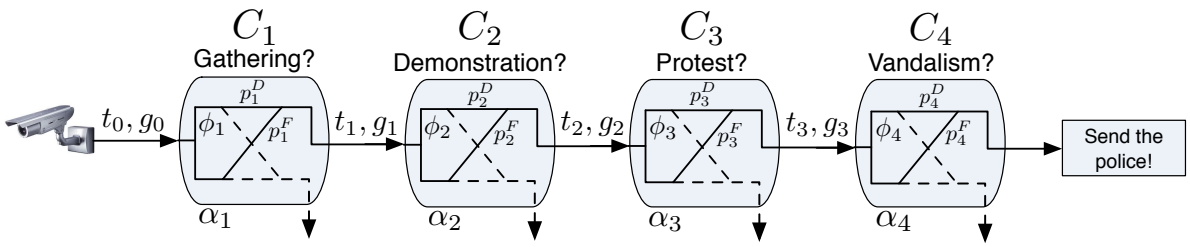

Fig. 4: A chain of classifiers to detect vandalism acts during protests.

protest and, to avoid the unnecessary waste of bandwidth and computational resources, data can be immediately dropped.

Let denote by X the input data of a generic classifier. Each classifier has an *a priori conditional selectivity* $\phi_i = \Pr(X \in \mathcal{H}_i | X \in \bigcap_{k=1}^{i-1} \mathcal{H}_k)$ corresponding to the conditional probability of data belonging to classifier C_i's class of interest, given that it belongs to the class of interest of the previous $i-1$ classifiers. For example, with reference to Fig. 4, ϕ_2 is the probability that a group of people gathered in one place is a demonstration. Similarly, we define the *negative a priori* selectivity as $\overline{\phi_i} = \Pr(X \in \mathcal{H}_i | X \notin \bigcap_{k=1}^{i-1} \mathcal{H}_k)$. Note that these a priori selectivities are inherent to the data features and to the relationships between concepts; thus, they do not depend on the operating points of individual classifiers.

The *detection probability* p_i^D of classifier C_i is the probability that a data entering classifier C_i and belonging to class $\mathcal{H}_i$ is labeled correctly by classifier C_i. For example, with reference to Fig. 4, p_2^D is the probability that a demonstration that is correctly labeled by C_1 is correctly labeled by C_2 as well. Similarly, the *false alarm probability* p_i^F is the probability that a data entering C_i and belonging to class $\overline{\mathcal{H}_1}$ is labeled (erroneously) as belonging to class $\mathcal{H}_1$.

The performance of the classifier C_i is characterized by its Detection Error Tradeoff (DET) curve that represents the tradeoffs between the detection probability and the false alarm probability. The DET curve can be represented as a function $p_i^D = f_i(p_i^F)$ that is increasing and concave [8]. As a consequence, an operating point on this curve is parameterized uniquely by its false alarm probability p_i^F.

The data that classifier C_i forwards to classifier C_{i+1} consists of correctly labeled data from class $\mathcal{H}_i$ as well as false alarms from class $\overline{\mathcal{H}_i}$. We refer to this quantity as the *throughput* t_i of classifier i. For example, in Fig. 4 the throughput t_2 is the amount of demonstrations that are correctly classified by both C_1 and C_2, plus the amount of non-demonstrations that are classified

(either correctly or erroneously) by C_1 as "people are gathering" and classified (erroneously) by C_2 as "demonstrations". We also define the *goodput* g_i of classifier i as the portion of data correctly labeled. Throughput and goodput are linked to each other, and given the DET curve of each classifier they can be derived recursively [8].

Another important metric is the average time α_i needed for classifier C_i to process a stream tuple. The order of magnitude of α_i depends on the data characteristics, as well as the classification algorithm, and can vary from microseconds (screening text) to multiple seconds (complex image or video classification).

The global utility function of the stream mining system can be expressed as a function of misclassification and delay cost, under resource constraints. Indeed, in practice the N classifiers are instantiated on M processing nodes, each of which has limited resources (bandwidth, memory, and computational resources). The resources consumed by each classifier are proportional to the throughput t_{i-1}, i.e., to the quantity of information it needs to process. Given a topology, the resource-constrained optimization problem can be formulated as a NOP [5]. In this context, instead of deciding *what fraction* of the data to process, as in load-shedding based approaches, it is fundamental to develop approaches to determine *how* the available data should be processed. Specifically, each individual classifier in the ensemble must select its operating point in order to maximize the global end-to-end performance while meeting system resource constraints. This simultaneous selection of the network topology and of the classifier configurations makes the problem significantly more complex than traditional NOPs.

Formulations similar to the approach described above have been analyzed in [3], [8], [15]. [8] proposes algorithms to optimally configure networks of classifiers given system processing resource constraints. It first formally defines a global performance metric for classifier networks by trading off the end-to-end probabilities of detection and false alarm, then it designs centralized and distributed algorithms to provide efficient and fair resource allocation among several classifier networks competing for system resources. In addition to constructing and adapting the networks of classifiers, [15] and [3] analyze also how to configure the individual classifiers in order to tradeoff the accuracy of feature identification with the filtering delay. [15] first develops centralized algorithms for joint ordering and individual classifier operating point selection, then it proposes a decentralized learning approach to design a dynamic routing based order selection strategy. Different learning strategies that lead to rapid convergence, while requiring minimum

coordination and message exchange, are investigated. Exploiting queuing theoretic models, [3] proposes a utility metric that captures both the performance and the delay of a binary filtering classifier system. Moreover, it introduces a low-complexity framework for estimating the system utility by observing, estimating, and/or exchanging parameters between the interrelated classifiers deployed across the system, it provides distributed algorithms to reconfigure the system, and it analyzes these algorithms based upon their convergence properties, optimality, information exchange overhead, and rate of adaptation to non-stationary data sources.

B. Online learning of the classifier operating points

In Subsection V-A we discussed a framework to build an optimal network of classifiers given the a priori conditional selectivities and the DET curve of each classifier. However, in many SMAs these values are not known beforehand and/or they can change in time because of the non-stationary environment. In these scenarios the relevant parameters must be *learned online*.

In the *supervised online learning* framework an entity, which we refer to as *learner*, needs to receive feedbacks about its actions in order to learn which is the optimal way to behave. Such feedbacks can be received from a human expert (e.g., a technician that analyzes a subset of the data offline) or be incorporated automatically in the system loop with delay (e.g., if the task is to predict the trend of the stock market in the next hour, the feedback is automatically received with one hour delay, after the real trend is observed).

The learning task of a classifier which is part of a classifier network is more challenging than in traditional online learning framework for two reasons: 1) the feedback represents a global information about the behavior of the system, it does not provide direct information about the contribution of each individual classifier to the system performance, and 2) the contribution of each classifier is correlated with the network topology and the operating points of the other classifiers. To learn the optimal configuration, the stream mining system must explore multiple configurations – different network topologies and classifier operating points. Whenever a feedback is received, a comparison between the answer provided by the current configuration and the feedback gives an estimate of the performance of the current configuration. This estimate can be exploited to decide whether to explore a new configuration and, in this case, how to compute the new configuration.

To improve the speed of learning of the system, it is possible to exploit any a priori information

about the correlation among the classifiers in the current network topology, and *parallelize* the learning. For example, consider the network topology represented in Fig. 2 and assume that the current stream of data passes through the "Demonstration" classifier. In this case, the final answer of the system is not affected by a change in the operating points of the "Suspicious person" and "Crime" classifiers and by a swap in the positions of these classifiers. This means that a single feedback can simultaneously provide information about multiple network topologies and classifier operating points. By exploitation these considerations the speed of learning can be considerably increased.

Online learning schemes to learn the optimal classifier operating points are proposed and analyzed in [16]–[18]. These works consider a network of processing nodes that observe distributed, heterogeneous, and dynamic data sources. These processing nodes pre-process the observed data locally and exchange information to generate a final global prediction. The network topology is not constrained to be a chain or tree of classifiers; hence, each processing node can simultaneously receive multiple correlated inputs – multiple source observations and/or multiple pre-processed outputs from other processing nodes – and need to learn online how to aggregate such information. [16] focuses on regression problems and proposes an online learning scheme in which the amount of information exchanged depends on the estimated correlation among processing nodes, this allows to tradeoff system performance and amount of exchanged information. Instead, [17] focuses on classification problems and proposes a learning scheme with the following features: 1) it requires a minimal exchange of information; 2) the observed data can be discarded after it is processed; and 3) it only requires basic operations such as add, multiply, and compare. These features are extremely important when the available communication bandwidth is small, when the processing nodes have very limited memory and computational capabilities, and when the processing delay plays an important role (e.g., for real-time SMAs). In addition to it, [18] designs a scalable and efficient information dissemination protocol to spread the required information in the network, and it applies the proposed scheme to some well-known real-world data sets showing gains ranging from 20% to 70% with respect to state-of-the-art solutions.

Table I summarizes the main differences between a traditional NOP and the stream mining schemes described in this section.

	Topology / Routing Selection	Classifier Configuration Selection	Online Learning
Traditional NOP [5]	Adapted dynamically based on network state (e.g., congestion experienced by nodes)	No	No
[8]	Adapted dynamically based on network state and on results of local data processing	No	No
[3], [15]	Adapted dynamically based on network state and on results of local data processing	Yes	No
[16]–[18]	Fixed topology, classifiers can communicate with each other	Yes	Yes

TABLE I: Comparison between traditional NOP and the schemes described in Section V.

VI. Open Research Questions

There are several open research problems that need to be investigated for supporting the deployment of large-scale distributed systems for SMAs. We describe these open research problems in the following.

(1) **Adaptation.** There are several interesting topology configuration and adaptation problems that can be used to dynamically modify individual classifiers (e.g., change the model or thresholds for a classifier), automatically reorganize the topology to potentially increase parallelism, or use selectivity of individual operators to reduce workloads on downstream operators. The impact of different levels of adaptation needs to be investigated. Some examples of exploiting these tradeoffs have been considered in [3], [8], [15], but this is a fertile space for future research.

(2) **Speed of learning.** It is fundamental to derived analytics for SMAs that allow to determine both *long-term* and *short-term* bounds on the system performance, with respect to a given metric. For example, there exist preliminary results that characterize the classification accuracy of a data mining scheme in the long term [18]. However, it is vital for real-time stream mining applications to learn timely, achieving a high classification accuracy in the short term as well. Hence, it is important to determine bounds for the *speed of learning*, i.e., the time required by the scheme to guarantee a minimum classification accuracy.

(3) **Optimize resources.** There are several resource optimization problems that relate to taking a given processing topology and mapping it to physical processes that can be instantiated on a distributed computational platform. This requires a multi-objective optimization where communication costs need to be traded off with memory and computational costs. Also, given the

resource requirements, data characteristics may change over time and as a consequence these optimization problems may need to be solved incrementally or periodically. The interaction between these resource optimizations and the system classification accuracy needs to be also formally investigated.

(4) **Robustness to missing and delayed information.** In distributed environments it is difficult to guarantee perfect synchronization and communication among processing nodes. The loss of synchronization may result in information which is received with delay, whereas the communication errors or local failures may result in missing information. The robustness of different learning schemes and network topologies with respect to these issues needs to be investigated.

(5) **Acquisition rate.** In some cases (e.g., sensor networks) the processing nodes can decide the acquisition rate to adopt to collect data. The impact of this additional degree of freedom on the system classification accuracy and on the system resources must be investigated.

(6) **Privacy.** When the data streams include sensitive information, the stream mining system requires control on the privacy of the information, making sure that the analysis respects the desired anonymization, authorization, encryption, and authentication requirements. It is an open research question how this requirements translate into constraints of the network optimization problem and which is their impact on the system performance.

(7) **Conflicting objectives.** There are some scenarios in which the multiple processing nodes belong to different entities that have conflicting objectives. In these scenarios it is not plausible to assume that the nodes cooperate with each others. Non-cooperative game theoretic approaches must be adopted to analyze these distributed environments.

(8) **Unsupervised learning.** While we have posed the problem of distributed learning in a supervised setting (i.e., with feedbacks that are eventually observed), there is also need to build large-scale online algorithms for knowledge discovery in unsupervised settings. Constructing online ensemble methods for clustering, outlier detection, and frequent pattern estimation are all very interesting directions for more research.

Overall, we believe that the space of SMAs is an extremely fertile space for novel research and construction of real-world deployments that have the potential to accelerate our effective use of streaming data.

REFERENCES

[1] A.S. Fialho, F. Cismondi, S.M. Vieira, S.R. Reti, J.M.C. Sousa, and S.N. Finkelstein, "Data mining using clinical physiology at discharge to predict ICU readmissions," *Expert Systems with Applications*, vol. 39, no. 18, 2012, pp. 13158–13165.

[2] J. Lin and D. Ryaboy, "Scaling big data mining infrastructure: the twitter experience," *Proc. ACM SIGKDD*, vol. 14, no. 2, Apr. 2013, pp. 6–19.

[3] B. Foo and M. van der Schaar, "A distributed approach for optimizing cascaded classifier topologies in real–time stream mining systems," *IEEE Trans. Image Process.*, vol. 19, no. 11, 2010, pp. 3035–3048.

[4] M. Cherniack, H. Balakrishnan, M. Balazinska, D. Carney, U. Cetintemel, Y. Xing, and S. B. Zdonik, "Scalable distributed stream processing," *Proc. of CIDR*, vol. 3, Jan. 2003, pp. 257–268.

[5] S. H. Low and D. E. Lapsley, "Optimization flow control–I: basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, 1999, pp. 861–874.

[6] Y. Xing, S. Zdonik, and J-H Hwang, "Dynamic load distribution in the Borealis stream processor," *Proc. of ICDE*, 2005, pp. 791–802.

[7] M. Balazinska, H. Balakrishnan, S. R. Madden, and M. Stonebraker, "Fault-tolerance in the Borealis distributed stream processing system," *ACM Transactions on Database Systems (TODS)*, vol. 33, no. 1, Mar. 2008, pp. 1–44.

[8] F. Fu, D. S. Turaga, O. Verscheure, M. van der Schaar, and L. Amini, "Configuring competing classifier chains in distributed stream mining systems," *IEEE J. Sel. Topics Signal Process.*, vol. 1, no. 4, 2007, pp. 548–563.

[9] A. Condon, A. Deshpande, L. Hellerstein, and N. Wu, "Flow algorithms for two pipelined filter ordering problems," *Proc. of ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, 2006, pp. 193–202.

[10] Z. Liu, S. Parthasarathy, A. Ranganathan, and H. Yang, "Near-optimal algorithms for shared filter evaluation in data stream systems," *Proc. of ACM SIGMOD*, 2008, pp. 133–146.

[11] N. Tatbul, U. Çetintemel, and S. Zdonik, "Staying fit: Efficient load shedding techniques for distributed stream processing," *Proc. of VLDB*, 2007, pp. 159–170.

[12] W.-G. Teng, M.-S. Chen, and S. Y. Philip, "Resource-aware mining with variable granularities in data streams," *Proc. of SDM*, 2004.

[13] Y. Chi, H. Wang, and P. S. Yu, "Loadstar: load shedding in data stream mining," *Proc. of VLDB*, 2005, pp. 1302–1305.

[14] D. S. Turaga, O. Verscheure, U. V. Chaudhari, and L. D. Amini", "Resource management for networked classifiers in distributed stream mining systems," *Proc. of IEEE ICDM*, 2006, pp. 1102–1107.

[15] R. Ducasse, D. S. Turaga, and M. van der Schaar, "Adaptive topologic optimization for large-scale stream mining," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 3, 2010, pp. 620–636.

[16] Y. Zhang, D. Sow, D. S. Turaga, and M. van der Schaar, "A fast online learning algorithm for distributed mining of BigData," *The Big Data Analytics Workshop at SIGMETRICS*, 2013.

[17] L. Canzian, Y. Zhang, and M. van der Schaar, "Ensemble of distributed learners for online classification of dynamic data streams," UCLA, Electrical Engineering Department, Tech. Rep., 2013. Available: http://arxiv.org/abs/1308.5281.

[18] L. Canzian and M. van der Schaar, "A network of cooperative learners for data–driven stream mining," *accepted and to appear in IEEE ICASSP*, 2014.

Luca Canzian [M'13] received the B.Sc., M.Sc., and Ph.D. degrees in Electrical Engineering from the University of Padova, Italy, in 2005, 2007, and 2013, respectively. From 2007 to 2009 he worked in Venice, Italy, as an R&D Engineer at Tecnomare. From September 2011 to March 2012 he was on leave at the University of California, Los Angeles (UCLA). Since January 2013, he has been a Postdoc at the Electrical Engineering Department at UCLA. His research interests include Big Data, online learning, real-time stream mining, and game theory applied to wireless networks.

Mihaela van der Schaar [F'10] is Chancellor's Professor of Electrical Engineering at UCLA. She was Distinguished Lecturer of the Communications Society and Editor in Chief of IEEE Transactions on Multimedia. She received an NSF CAREER Award, 3 IBM Faculty Awards, and several Best Paper Awards. She holds 33 granted US patents. She is also the founding director of the UCLA Center for Engineering Economics, Learning, and Networks. Her research interests include engineering economics and game theory, network science, expert and social networks, online reputation and social media, dynamic multi-user networks and system designs, wireless networks, online and interactive learning, and real-time stream mining.