Edge mining the internet of things

Gaura, E. , Brusey, J. , Allen, M. , Wilkins, R. , Goldsmith, D. and Rednic, R.

Original citation & hyperlink:
Gaura, E. , Brusey, J. , Allen, M. , Wilkins, R. , Goldsmith, D. and Rednic, R. (2013) Edge mining the internet of things. IEEE Sensors, volume 13 (10): 3816-3825.
http://dx.doi.org/10.1109/JSEN.2013.2266895

Edge mining the Internet of Things

Elena I. Gaura, James Brusey, Michael Allen, Ross Wilkins, Dan Goldsmith, and Ramona Rednic

***Abstract*—This paper examines the benefits of *edge mining*—data mining that takes place on the wireless, battery-powered, smart sensing devices that sit at the edge points of the Internet of Things. Through local data reduction and transformation, edge mining can quantifiably reduce the number of packets that must be sent, reducing energy usage and remote storage requirements. Additionally, edge mining has the potential to reduce the risk to personal privacy through embedding of information requirements at the sensing point, limiting inappropriate use. The benefits of edge mining are examined with respect to three specific algorithms: Linear Spanish Inquisition Protocol (L-SIP), ClassAct, and Bare Necessities (BN), which are all instantiations of General SIP (G-SIP). In general, the benefits provided by edge mining are related to the predictability of data streams and availability of precise information requirements; results show that L-SIP typically reduces packet transmission by around 95% (20-fold), BN reduces packet transmission by 99.98% (5000-fold) and ClassAct reduces packet transmission by 99.6% (250-fold). Although energy reduction is not as radical due to other overheads, minimisation of these overheads can lead to up to a 10-fold battery life extension for L-SIP, for example. These results demonstrate the importance of edge mining to the feasibility of many IoT applications.**

I. Introduction

The ability to collect data about environments, equipment, people and activities has drastically increased over the past two decades, primarily due to advances in low power wireless computing and the ubiquity of smart phones. Pervasive low-power, smart embedded devices and RFID tags enable Internet of Things (IoT) applications, where smart objects sense the environment, react autonomously to physical events and trigger actions with or without human intervention [1]. Many IoT scenarios (smart cities, products, mobility, health and living) [2] are based around Wireless Sensor Networks (WSNs) and are built using resource constrained devices that *sense* and *send* data wirelessly over limited bandwidth connections. Enabling IoT applications at scale hinges on advances in two key areas:

- *Cost*—the energy and infrastructural cost of powering sensors, communicating wirelessly, and storing the associated data; and
- *Analytics*—the provision of automatic interpretation of raw measurement data into information that is relevant, timely and actionable.

We propose that mechanisms for moving intelligence and analytic capabilities into the network will result in better utilisation of the scarce energy resources at IoT nodes, with less transmissions leading to less network load and a higher data yield. We term these mechanisms *edge mining,* or data mining that takes place on the wireless, battery-powered, smart sensing devices that sit at the edge of the IoT. Three edge mining algorithms are presented in this paper: Linear Spanish Inquisition Protocol (L-SIP) [3], ClassAct [4], and Bare Necessities (BN) [5]. These algorithms reduce sensing messages, thus using less energy on-device and requiring less storage on the remote server. L-SIP applies to sensing applications where it is desirable to reconstruct the original signal within some error bound. ClassAct is a human posture recognition approach that just transmits or stores posture and the timing of postural changes but not the original accelerometer signal. Finally, BN discards even timing and is appropriate where only a summary of relative time spent in different states is needed. These algorithms originate in the WSN and Body Sensor Network (BSN) domains but are readily applicable to the IoT: they are conceptually simple to implement, operate on real-time data streams, and do not impose significant on-device processing requirements. These features are fundamental to successful edge mining IoT applications. Furthermore, these algorithms are shown to be instantiations of a general edge mining algorithm that we term General SIP or G-SIP.

The specific contributions of this paper are:

1) The introduction and definition of edge mining as a fundamental approach in the IoT;
2) The demonstration of using edge mining to reduce packet transmission, energy usage and remote storage requirements through three specific algorithms: L-SIP, ClassAct, and BN and a generalised form: G-SIP;
3) Specific analysis around the details of implementation and use of the proposed edge mining techniques in real applications.

The remainder of this paper is organised as follows: Section II presents and evaluates G-SIP, L-SIP, ClassAct and BN, Section III discusses related work and Section IV provides conclusions and future work.

II. Data mining at the edge

We define edge mining (or data mining at the edge) to be *processing of sensory data near or at the point at which it is sensed, in order to convert it from a raw signal to contextually relevant information*. Figure 1 illustrates our view of edge mining as a process that runs on individual sensor nodes. State estimation is a key aspect to this process since it transforms the raw signal into a form that is meaningful in the context of the application. For example, rather than a raw binary beam-break signal indicating someone is entering the room, the application-level state might be in terms of the number of people in the room. State estimation may involve filtering or smoothing or even sensor fusion. Once a state estimate has been formed, event detection decides whether the change in state is significant (or eventful). Generally speaking, event

E. I. Gaura, J. Brusey, M. Allen, R. Wilkins, D. Goldsmith, and R. Rednic are with Cogent Computing, Coventry University

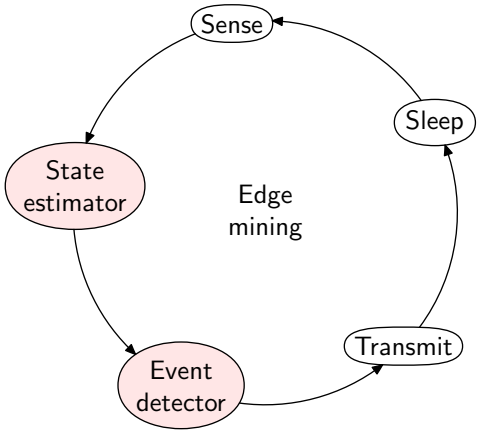

Figure 1. A summary of the edge mining process at the node.

detection should be simple. Complex event detection can often be simplified by changing the state estimator to include more information in the state vector.

The algorithms presented in this paper sit within this model. G-SIP provides a general form for algorithms that combine state estimation with consideration of what the sink already knows. L-SIP adopts a linear model of the measured value, which is sufficient for many phenomena. ClassAct estimates state in terms of human posture and event detection is simply based on when that posture changes. BN extends the state estimator to summarise it over time in terms of a histogram. Event detection here is when the distribution changes significantly.

Traditionally, work towards the generic objective of efficient energy resource utilisation has focused on exploiting inter-node communication, in-network intelligence building, hardware optimisation and energy management. Our experience, which is reflected in the literature [6], is that collaborative WSNs are unusual in real-life systems, so this paper assumes that a WSN is a *network of individual smart sensing devices* that are *discrete sources of data and information*, and that *transmit over a shared ad-hoc wireless network*. This means sensor nodes do not collaborate with each other at an *application level* to fulfil the goals of the application or achieve resource usage reduction. Since edge mining is applied at application level, the concepts put forward are not dependent on any particular MAC or multi-hop data transport layer. Thus the edge mining is generic and widely applicable.

Central to the concept of edge mining is the observation that the better the information requirements of a given application are understood, the greater the potential to reduce the raw signal. This allows less data to be sent, thus using less energy and taking up less storage on a remote server. For example, a mobility application may require only summaries of raw data: GPS traces can be reduced into mobility states, or classified by location, i.e. “at work”, “at home” or “on route”. Personal activity data from a worn accelerometer, say, can be summarised into posture or activity, such as “sitting”, “walking”, “running” and so forth.

Edge mining focuses on saving packets rather than bits. Even when considering a single sensor, there is often much information redundancy in the original signal; the signal has low entropy and is thus highly compressible in information terms. Nonetheless, ordinary compression schemes (e.g. Huffman encoding) only reduce the number of bits to be transmitted and still require a constant or at least regular emission of data packets from the sensory source. Each data packet involves some overhead: the packet itself has headers and checksums; typically some energy is consumed powering up and powering down the radio; packet acknowledgement is usually required; and other nodes in the network will expend power in forwarding the message. Clearly, saving packets is more important than saving bits.

Optimising energy, network usage and data storage are relevant concerns in an end-to-end context in the IoT because individual devices must be long-lived and must be careful to minimise the amount of traffic they introduce to the network. At the remote server level, care must be taken to store only the data and information that are relevant so that the volume of data for analysis is minimised. A valuable side-effect of edge-mining with well-defined information requirements for an application is that privacy is improved since signals are transformed at the point of acquisition, making it difficult to further mine them to determine unintended information.

In the remainder of this section, an overarching algorithm template for edge mining, termed General SIP (G-SIP), is presented. This then leads to three instantiations: Linear SIP (L-SIP), ClassAct and BN. These are presented and evaluated with respect to message reduction and their respective benefits.

A. Sending only unexpected information (General SIP)

The first approach that we shall examine is called the Spanish Inquisition Protocol (SIP) [3] after the Monty Python skit. Simply put, under SIP, a node only transmits that which the receiver *does not expect*. More verbosely, each individual wireless sensor sends data packets when the signal changes in a way that the data sink (or central database server) could not be expected to accurately predict from the packets that it had previously received. In this work, we provide a generalised form of the algorithm. General SIP (G-SIP) is agnostic on prediction mechanism (or model) other than requiring that both node and sink share the same one. For many sensor signals, a piecewise linear approximation is sufficient to bring about a substantial reduction in the number of data packets that need to be sent and this observation leads to Linear SIP. The G-SIP algorithm assumes that:

- The point-in-time signal can be converted into a state estimate (initially on its own and then subsequently with reference to the past state estimate).
- The state estimate is sufficient to support short term prediction. (It is not strictly assumed that the state has the Markov property, although it is generally helpful if it does.)
- The sink (or central database) stores enough information in order to predict the current state.
- If a sent packet is acknowledged then it was successfully received and stored at the sink. (This implies end-to-end acknowledgement rather than acknowledgement from a relay.)

Algorithm 1 General Spanish Inquisition Protocol (G-SIP) (node algorithm). Note that this is a generalised form of the previously published SIP algorithm [3]

1) $\mathbf{z} \leftarrow$ obtain vector of sensor readings
2) $t \leftarrow$ current time
3) $\mathbf{x}_{\text{new}} \leftarrow$ estimate new state $(\mathbf{z}, \mathbf{x}_{\text{old}}, t_{\text{old}})$
4) $\mathbf{y}_s \leftarrow$ predict sink state $(\mathbf{y}_{\text{sink}}, t_{\text{sink}}, t)$
5) $\mathbf{y}_{\text{new}} \leftarrow$ simplify $(\mathbf{x}_{\text{new}})$
6) if eventful $(\mathbf{y}_{\text{new}}, \mathbf{y}_s)$ or $t - t_{\text{sink}} \geq t_{\text{heartbeat}}$
   (if the state is eventful or if time since the last transmission exceeds a threshold)
   a) transmit $(\mathbf{y}_{\text{new}}, n, t)$
   b) $n \leftarrow n + 1$ (increment sequence number)
   c) when acknowledgement received:
      i) $\mathbf{y}_{\text{sink}} \leftarrow \mathbf{y}_{\text{new}}$
      ii) $t_{\text{sink}} \leftarrow t$
7) $\mathbf{x}_{\text{old}} \leftarrow \mathbf{x}_{\text{new}}$
8) $t_{\text{old}} \leftarrow t$

The G-SIP algorithm template at the node is shown as Algorithm 1. This algorithm operates once per sensing cycle by *sensing* (1, 2), *filtering* (3), *detecting events* (4, 5, 6), conditionally *transmitting* (6a–c), and updating the "old" state (7, 8). The sensor readings $\mathbf{z}$ and the state $\mathbf{x}$ may be quite different in nature. For example, the sensor reading might be the voltage level for a light beam detector while the state may be a count of the number of occupants in the room. Furthermore, multiple sensors may be involved, such as a PIR alongside the light beam detector. The state should contain sufficient information to both support appropriate filtering (3) and allow prediction (4). For example, for a diurnal signal, the state might encode a smoothed version of the last 24 hours of data. Since the state vector $\mathbf{x}$ may contain much more than needs to be transmitted, it is often useful to generate a simplified form $\mathbf{y}$.

If the state is eventful or if the time since the last transmission exceeds some threshold $t_{\text{heartbeat}}$, then the new state $\mathbf{y}_{\text{new}}$ will be transmitted along with a sequence number n. Note that an optional "simplify" transformation can be used here to remove unnecessary information from the state before transmission. The sequence number is useful to identify lost packets. On acknowledgement of this message, the local copy of the sink's state $\mathbf{y}_{\text{sink}}$ will be updated along with the associated timestamp t_{sink}.

G-SIP is tolerant of packet loss if the simplified state vector $\mathbf{y}$ contains sufficient information to allow a prediction on its own (i.e., the Markov property holds). For example, if $\mathbf{y}$ contains a temperature and rate of change of temperature, then any reconstruction error caused by packet loss will not propagate past the subsequent packet. However, if $\mathbf{y}$ only contains the change in temperature since the last message, any packet loss will introduce an error in the reconstruction that will propagate indefinitely.

Algorithm 2 Linear SIP (L-SIP) phrased in terms of G-SIP in Algorithm 1.

estimate new state
dEWMA filtering:

$$\begin{aligned} x_1' &\leftarrow \alpha z + (1-\alpha)(x_1 + x_2 \Delta t) \\ x_2' &\leftarrow \beta\left(x_1' - x_1\right)/\Delta t + (1-\beta)\, x_2 \end{aligned}$$

(Update filtered estimates of value x_1 and rate of change x_2. Δt denotes the time interval between samples.)

predict sink state

$$\mathbf{y}' \leftarrow \begin{pmatrix} 1 & t - t_{\text{sink}} \\ 0 & 1 \end{pmatrix} \mathbf{y}_{\text{sink}} \text{ (linear extrapolation)}$$

simplify
$\mathbf{y} \leftarrow \mathbf{x}$ (no simplification)

eventful?
yes if $|y_1' - y_1| > \varepsilon$
(The measurement is eventful if the value estimate y_1 differs from the prediction y_1' by at least some threshold ε.)

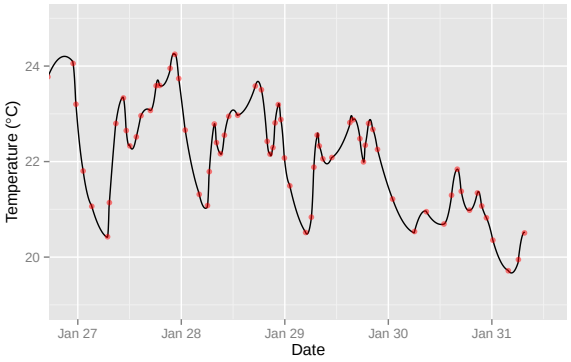

Figure 2. Example reconstructed temperature time series based on L-SIP with a threshold of 0.5 °C. Signal reconstructed with 68 L-SIP packets instead of 1189 raw data packets (6%). Circles are used to mark when transmissions occurred. This graph is derived from a deployed house monitoring system that uses L-SIP.

B. Linear SIP

Linear SIP (L-SIP) is an instantiation of the G-SIP template that encodes the state as a point in time value and rate of change $\mathbf{x} = (x, \dot{x})^T$. A number of methods can be used for state estimation, such as: an Exponentially Weighted Moving Average (EWMA), Normalised Least Mean Squares (NLMS) or a Kalman Filter (KF). EWMA or double EWMA (dEWMA) is sufficiently simple to be implemented efficiently on low powered nodes and often provides good performance. Events are detected when the value component of the state differs by more than some threshold when compared to the predicted value. Algorithm 2 summarises L-SIP in terms of G-SIP.

1) Message reduction performance: An example of a temperature time-series reconstructed from L-SIP packets is shown in Figure 2. The reconstruction uses splines to incorporate the gradient information. Full discussion of the spline reconstruction is outside the scope of this paper but the key features are to match gradients at consecutive points and to incorporate knowledge about the threshold used. For the example in Figure 2, there were 68 L-SIP packets transmitted

Table II
BASELINE MICRO-BENCHMARK ESTIMATES FOR A TELOSB MOTE WITH A FIVE MINUTE SAMPLING CYCLE. CTP SEND TIME IS BASED ON LOGS FROM A 200+ NODE NETWORK AND INCLUDE RETRIES.

	Time (ms)		μA		μAs
Temperature	220	×	458	=	100.76
Humidity	75	×	458	=	34.35
Voltage	0.017	×	536	=	0.009
CTP send	473	×	18,920	=	8,949
LPL check	1,500	×	18,920	=	28,380
Idle	297,712	×	9	=	2,679
Totals	300,000				40,522

Table III
MICRO-BENCHMARK ESTIMATES FOR USING L-SIP ON TELOSB FOR VARIOUS LISTENING OPTIONS. THE ESTIMATE FOR TSMP IS BASED ON REPORTED DUTY CYCLES OF 0.1%. THE ENERGY USE IS RELATIVE TO THE ESTIMATED USE FOR LPL WITHOUT L-SIP AS SHOWN IN TABLE II.

	Energy use relative to LPL
LPL + L-SIP 5%	79%
TSMP 0.1% + L-SIP 5%	27%
No listening + L-SIP 5%	10%

instead of 1189 raw data packets, corresponding to a 94% reduction in packets. Packet reduction has been assessed for a number of data sets previously [3] and is briefly restated in Table I. Filtering is performed using either an Exponentially Weighted Moving Average (EWMA), Normalised Least Mean Squares (NLMS) or a Kalman Filter (KF). The extent to which the transmissions can be reduced is related to sensing frequency, the error threshold, and the nature of the data. The Intel data set [7] is included as a basis for comparison with other reduction algorithms, all of which L-SIP outperforms (see [3]).

2) Energy reduction performance: It has been shown that L-SIP reduces the number of packets transmitted and this is expected to reduce energy consumption. However, the relationship between packet transmission reduction and energy reduction is affected by the underlying network stack [9], specifically the multi-hop transport layer and Media Access Control (MAC) layer. For example, the Collection Tree Protocol (CTP) [10] generates extra network traffic by periodically beaconing to evaluate link quality and maintain an up-to-date routing table, and by forwarding messages from neighbours. In addition, the impact on energy usage of listening for and transmitting messages at the MAC layer is highly variable. Asynchronous interval-listening approaches such as Box-MAC [11] (a.k.a. Low Power Listening (LPL)) can substantially reduce the overhead associated with maintaining a mesh network but still consume significant energy [12]. Synchronous interval listening approaches such as TSMP significantly outperform asynchronous interval-listening protocols. TSMP, for example, has a measured radio duty cycle between 0.01% and 0.3% [13] thus significantly enhancing the potential lifetime with L-SIP.

To model the energy consumption for MAC and CTP, microbenchmarking of an indicative hardware platform running L-SIP is used. Table II shows energy consumption estimates on a TelosB mote for a periodic 5 minute sensing cycle, sampling and sending temperature, humidity and battery voltage, with CTP as the multi-hop transport protocol and LPL at the MAC layer. The CTP send time is estimated using logs from a long-term deployment of over 200 nodes [14] and includes retries. Using this as a baseline, Table III shows the estimated relative improvement of using L-SIP (and CTP) with different MAC approaches, based on the assumption that it reduces the packet count to 5% (a 95% reduction). (This assumption for the packet reduction may be conservative but seems reasonable given Table I.) If LPL is used alongside L-SIP, then the benefit of L-SIP is slight (energy is reduced to 79% of the baseline). However, if a more efficient interval listening scheme is used (e.g. TSMP) then the energy reduction is roughly 4-fold. For leaf-nodes that are not required to listen to neighbours or forward packets, the energy reduction is 10-fold. These results show that although message reduction is an important factor, careful consideration of the lower protocol layers is needed to obtain the maximum energy savings.

3) Network and database effects: In general, reducing packet counts has the potential to reduce congestion and collisions on networks with large amounts of traffic. Reducing the number of transmissions from the node also reduces the number of acknowledgements and retries needed. Furthermore, in a multi-hop network, less forwarding is needed.

Disk storage associated with a sensor network can be substantially reduced if reconstruction is performed on demand rather than prior to storage. Furthermore, producing a graph of data during reconstruction will involve less I/O since fewer records will need to be retrieved from the database. In applications where the node needs to store data locally, L-SIP enables a longer time period to be stored. For example, the 1 Mb flash memory in a TelosB mote, could store an estimated 2.9 years of data for five parameters, assuming 5 minute sampling, 95% message reduction and 70 bytes per record.

4) Discussion: L-SIP shows a dramatic reduction in packet transmission compared to sending all sensed data. Significant reductions in energy usage are also possible but, for multi-hop networks, the energy reduction may be hampered by the overheads of maintaining the network.

G-SIP's ability to reduce the number of packets strongly depends on the model used to convert the raw sensory signal into a representation of state. For many environmental sensors, such as temperature, humidity, gas concentration, and light level, a simple model (such as piecewise constant or linear) produces significant reduction. High frequency signals such as audio or acceleration are not predicted well by such models. Nevertheless, the following section (Section II-C) deals with the problem of converting such a set of signals to a state representation.

A potential concern with the significant packet reduction produced by G-SIP instantiations is to what extent packet loss affects the accuracy of the reconstructed signal. Fortunately, G-SIP performs comparably with "sense and send" (SS) when packet loss occurs. If a single packet is lost, both SS and G-SIP can identify the loss (the sequence count is used for this) and the reconstruction can therefore account for it. Multiple packet loss also causes similar information loss in SS and G-SIP since G-SIP will tend to retransmit on the next sensing cycle until an acknowledgement is received.

Table I
SUMMARY OF L-SIP PERFORMANCE FOR VARIOUS DATA SETS. AVERAGE TRANSMISSION REDUCTION IS 95.5% (4.5% PACKETS SENT).

Data-set	Error Threshold (ε)	Filter	Period (s)	RMSE	Transmitted (%)
HomeREACT Temperature [8] (sensor 1)	0.5 °C	EWMA	300	0.24 °C	4.1
	0.5 °C	NLMS	300	0.75 °C	4.0
	0.5 °C	KF	300	0.25 °C	3.9
HomeREACT Humidity [8]	0.5 %RH	EWMA	300	0.46 %RH	13.3
	0.5 %RH	NLMS	300	2.2 %RH	12.7
	0.5 %RH	KF	300	0.58 %RH	11.3
HomeREACT Light [8]	5 lux	–	300	2.2 lux	4.4
	9 lux	–	300	2.5 lux	2.4
	5 lux	EWMA	300	2.7 lux	1.4
	9 lux	EWMA	300	5.8 lux	0.37
Intel (Node 13) Temperature [7]	0.5 °C	EWMA	30	0.24 °C	1.0
	0.5 °C	NLMS	30	0.41 °C	1.1
	0.5 °C	KF	30	0.26 °C	1.4
	0.05 °C	EWMA	30	0.06 °C	5.3
TelosB Temperature Deployment	0.5 °C	EWMA	300	0.22 °C	1.7

L-SIP is useful for data transmission reduction where the application requires that the raw data stream be reconstructable in the future. The remaining approaches, ClassAct and BN, are G-SIP instantiations that allow for greater message reduction through more precisely defined information requirements in the motivating application.

C. Filtered state classification (ClassAct)

ClassAct is a human posture / activity classifier based on decision trees. ClassAct takes the signals from a set of worn accelerometer sensors and estimates the current posture (such as, sitting, standing, or walking) [15], [16], [4]. As with L-SIP, it transforms a raw signal into a representation of the state before deciding whether the new state is eventful. In comparison with L-SIP, however, the transformation is destructive—it is not possible to reconstruct the original signal. In terms of the G-SIP algorithm, ClassAct estimates the state through decision-tree recognition of posture. As discussed below, a voting filter is required and so the state is stored as a probability distribution over the set of postures. This long form of the state is simplified prior to transmission to be just the index of the most likely posture. The prediction mechanism assumes that the posture will tend to stay the same.

ClassAct is well suited to embedding on a wireless sensor since it is computationally simple; after machine learning, classification is performed in just a few instructions. Compared to L-SIP, ClassAct is closely tied to the application and thus able to make stronger assumptions about the informational output. For ClassAct, all that is required is to identify which of a small number of postures the subject is in. Furthermore, ClassAct takes account of the fact that a subject is more likely than not to stay in the same posture over time and can thus smooth over the set of postures. This is implemented in ClassAct with a post-processing filter called Exponentially Weighted Voting (EWV) [17] that uses a parameter to control responsiveness to intermittent state changes. Algorithm 3 shows the ClassAct algorithm in terms of G-SIP.

1) Message reduction: In several posture classification trials where ClassAct was deployed on a wireless, on-body accelerometer system, 5182 classification packets were sent across a total of 9.6 hours of trials (343,140 packets of raw data). This represents a 98.5% (or 67-fold) reduction in packets compared to raw data. As shown in prior work [17] and reproduced in Figure 3, a further 4-fold reduction in the number of packets is obtained using EWV (1285 with $\alpha = 0.04$, a 99.6% or 250-fold reduction from the original number of messages). The top half of Figure 3 shows the number of events generated when varying the EWV α parameter between 0.02 and 0.45. The horizontal red bar corresponds to the number of events when no filter is used (5182). In this figure, the reduction of events generated indicates the degree to which α filters intermittent changes in posture. The bottom half of Figure 3 shows the classification accuracy. Optimal classification accuracy is at around $\alpha = 0.04$ and this also corresponds to near optimal event reduction.

Algorithm 3 ClassAct phrased in terms of G-SIP in Algorithm 1.

estimate new state
  $x'_{2\ldots n} \leftarrow x_{1\ldots n-1}$
  $x'_1 \leftarrow$posture estimated using a decision tree($\mathbf{z}$)
predict sink state
  $y' \leftarrow y_{\text{sink}}$ (static posture assumption)
simplify
  $y \leftarrow EWV(\mathbf{x}')$ (voting scheme to estimate posture)
eventful?
  yes if $y' \neq y$
  (if the smoothed posture estimate y has changed)

2) Discussion: ClassAct is an instantiation of G-SIP that compresses a high frequency signal (accelerometer data) by first converting it to an application specific form. In this form, it is relatively sparse and thus highly compressible. The compression thus achieved (99.6% or 250-fold packet reduction) is significantly greater than that achieved by L-SIP at a cost of not being able to meaningfully reconstruct the raw signal. Although ClassAct is tightly coupled to the application, it illustrates the general approach for applying G-SIP to high frequency signals and demonstrates that G-SIP is not restricted in this regard.

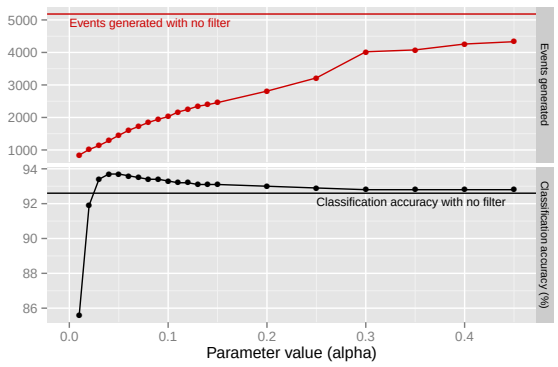

Figure 3. Performance of the EWV event filter based on 9.6 hours of posture trials from 9 accelerometers recording at 10 Hz.

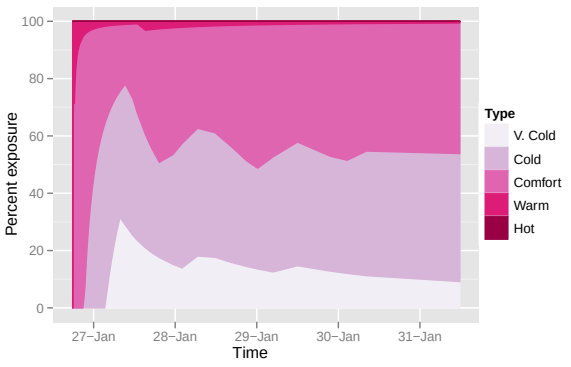

Figure 4. Time-discounted temperature distribution over time for a monitored bathroom.

Algorithm 4 Online time-discounted histogram encoding algorithm for estimating the exposure distribution based on Gaura *et al.* [5] phrased in terms of G-SIP in Algorithm 1.

estimate new state
$x_i \leftarrow \gamma x_i + b\,(i, z)$, *(update bin count)*
for all $i \in \mathcal{B}$.
The predicate function $b\,(i, z)$ gives 1 if the current reading z is in bin i and zero otherwise. The update decays the current count estimate by decay constant γ and then increments the currently active bin. The decay half-life is $t_{1/2} = T \ln 2 / (1 - \gamma)$ where T is the sensing period.
predict sink state
$\mathbf{y}' \leftarrow \mathbf{y}_{\text{sink}}$ (static distribution assumption)
simplify
$y_i \leftarrow x_i / \sum_{i \in \mathcal{B}} x_i$, *(update distribution)*
for all $i \in \mathcal{B}$.
This converts the counts to a distribution that sums to 1.
eventful?
yes if $\exists i \in \mathcal{B} : |y_i - y'_i| > \varepsilon$
The distribution is eventful if at least one component has changed by at least some threshold ε

D. Time-discounted Histogram Encoding (BN)

Bare Necessities (or BN) is used for summarising relative time spent in given states. For some applications, the ability to reconstruct the entire time series is unnecessary and it is only important to know the proportion of time spent in a state or set of states. This is useful, for example, for determining how long is spent in a certain modality (walking, driving, standing) in a given day, or how long temperatures in a room are within acceptable or unsafe bounds on a given day (or other time period). In these applications, most timing and raw signal information can be discarded, thus data transmissions can be even more aggressively reduced.

BN, shown as Algorithm 4, is a G-SIP instantiation where the state is encoded as a distribution over bins (e.g. a bin might comprise temperatures between 18 °C and 22 °C). Furthermore, BN weights more recent measurements more highly than older measurements by applying a time discount factor γ. BN makes the following assumptions:

- There is a predicate function $b : \mathcal{B} \times \mathcal{Z} \rightarrow \{0, 1\}$ that yields 1 if measurement $z \in \mathcal{Z}$ is in bin $i \in \mathcal{B}$ where $\mathcal{B}$ is the set of bins and $\mathcal{Z}$ is the set of possible sensor values.
- At least one bin is always applicable and bins do not overlap. Thus, $\sum_i b\,(i, z) = 1$ for all z.
- The sensing frequency is fixed.

As with G-SIP, the algorithm executes on the node during each sensing cycle. The new state is estimated by incrementing the bin x_i if the measurement z is in the range for bin i. The counts x_i are "simplified" by normalising them. The new distribution $\mathbf{y}$ is eventful if any bin has changed (since the last transmitted one $\mathbf{y}_{\text{sink}}$) by more than some threshold ε. Once this transmission has been acknowledged, the "last transmitted" distribution is updated along with the transmission time t_{sink}.

The point of this algorithm is that, for many environments and situations, although the signal may change rapidly, the frequency distribution (or histogram) of that signal changes quite slowly. Furthermore, for many applications, knowing the statistical distribution of the sensed signal is sufficient.

Figure 4 shows an example visualisation of BN data over six days, where a BN-enabled sensor node monitored temperature in a bathroom. In the first day, a higher percentage of time is spent in the comfortable band. As time progresses, the relative times stabilise and the room is generally cold or comfortable, with warm periods (likely when a shower is being used) and very cold ones.

Table IV
COMPARING THE PERFORMANCE OF BN ($t_{1/2} = 1$ MONTH) WITH L-SIP FOR ONE YEAR OF TEMPERATURE DATA [5].

	Transmissions	% of raw	RMSE in band %
Raw	102236	100%	n/a
L-SIP	2900 ± 700	2.8%	0.9 ± 0.2
BN	15 ± 6.5	0.02%	12 ± 4

1) Message reduction: Table IV compares the message reduction between sense and send (SS), L-SIP and BN for one year's worth of temperature data taken from a house monitoring application deployed by the authors [18]. L-SIP reduces the number of transmissions by 97.2% (36-fold) and BN reduces the number of transmissions by 99.98% (5000-fold).

The message reduction achievable with BN is typically significantly greater than that for L-SIP or ClassAct. In fact, given that BN systems might produce only 10 or 20 packets per year, it is vital to include a heartbeat mechanism, to ensure

that dead or failed nodes do not go undetected for an extended period.

2) Discussion: Compared to L-SIP and ClassAct, BN has the most specific application requirement. This enables the majority of data to be discarded and the amount of packets sent by a node to be reduced by well over 99%. Furthermore, since the resultant stored data are statistical summaries, this approach substantially reduces the risk to personal privacy. Privacy issues can be subtle. When monitoring, for example, humidity in a bathroom to assess mould risk, a raw time series might reveal whether, when and for how long showers or baths are used. It is important to place a use limitation [19] on this extra information, which is clearly not part of the original intent of monitoring. Therefore, transforming the data *at the first opportunity* into a form that is difficult to misappropriate aids privacy.

III. Related work

To situate our edge mining algorithms with respect to related work, we refer to the taxonomy of energy saving approaches presented by Anastasi et al. [20]. In this taxonomy, edge mining most resembles the data driven approach to energy conservation algorithms, namely in-network processing and data prediction. SIP is relevant to data compression and prediction, whereas ClassAct and BN are more relevant to in-network processing, specifically on-node processing.

A. Data compression and prediction

Data compression is used to reduce the number of bits required to encode a signal. Data compression algorithms have been developed that operate over distributed, networked nodes and achieve significant energy savings with little or no informational loss [21]. Data compression approaches tend to be agnostic to the type of data being compressed.

Compressed (or compressive) sensing, by comparison, reduces not only the size of data transmitted but also how many samples are taken. This promising technique exploits the statistical characteristics of the monitored phenomena and poses a complex reconstruction task [22], [23], [24].

The data prediction approach to energy conservation requires that both sensor node and sink maintain a model of the parameter being measured and update the model when newly acquired data causes significant change from the model-predicted value, triggering wireless transmissions. L-SIP belongs to a class of these approaches that are based on time-series forecasting, such as PAQ [25], SAF [26], and DBP [9]. When using a linear prediction model, L-SIP is most similar to Derivative-Based Prediction (DBP) with the most noticeable difference being that L-SIP performs data smoothing (thus typically requires floating point calculation) whereas DBP uses absolute and relative thresholds. DBP and L-SIP have similar performance characteristics.

B. In-network processing

In-network processing is another form of data-driven approach to energy reduction, whereby data are transformed or aggregated within the network. In a collaborative wireless sensor network, this could take the form of aggregating data or information across multiple nodes as the packets travel toward the sink. However, the edge mining algorithms presented in this paper do not consider such collaborative behaviour (an extensive summary can be found at [27]) and so only approaches related to on-node processing are discussed. On-node processing algorithms tend to use information or application-specific algorithms to transform or summarise data and provide a reduced stream to the sink. Event detection algorithms are a typical example of this, such as those used to detect animal calls for localisation [28], [29].

Lance, a framework for signal collection, sends summaries of windowed data from sensor nodes that allow a decision at the sink to be made as to how useful the data are and thus whether the high resolution data they describe should be retrieved [30]. BN is similar in that it computes statistics over known time periods, but does not store raw data. BN's histogram-like summaries of time spent in certain states are relevant for a variety of applications that must record relative duration spent in certain fixed states, such as reporting the amount of time spent in different mobility categories within a day. The summaries presented for monitoring mobility patterns using mobile phones [31], [32] are similar in spirit to BN, but are computed from raw data traces at the server level, thus do not explicitly target node-level data reduction.

C. Reduction of storage requirements

Data and information acquired and transmitted by wireless sensors in the IoT has to be stored in an appropriate manner on a remote server, such as a database running in the cloud. From here it is likely to be further processed to meet application requirements. Various standard approaches exist to compress data with or without loss. Approaches like de-duplication reduce redundancy in files or block data by making links to a single copy of a piece of data or file. Other standard approaches to remote storage keep only the most recent data and archive the rest to another media.

Many of these approaches work under the assumption that the full dataset must be transmitted and stored; however this "store first analyse later" approach is not suitable for applications that must generate real-time or event-based feedback [33].

D. Improving privacy by pre-processing data

When little is understood about a sensing-oriented application, it is often a necessity to collect complete data streams for offline analysis. However, when the data/information requirements for a given application are well-defined, the extra data generated by recording full raw data streams is at best redundant and at worst a risk for privacy. For example, in an application to determine time spent in a certain modality (driving, walking, standing), GPS data traces of time and location are acquired and stored on a remote server. This represents a privacy issue, as the data could be used to infer where people travel in a given day, or even where their home is.

Langheinrich [19] makes a compelling case for *use limitation* in ubiquitous systems, where the data collected should not be used for any other purposes than the declared aims. When edge mining algorithms are employed to destructively transform data into narrowly-specified information, the number of potential ways this information may be further mined is reduced, thus improving privacy.

IV. Conclusions and future work

This paper has presented *edge mining*, a data-driven approach that transforms data at the point of sensing into a sparse form to reduce packet transmissions, energy use, and storage space. This approach helps address two concerns that may otherwise limit the IoTs scalability: *cost*, in terms of energy, network traffic, maintenance and storage; and *analytics*, in terms of automated interpretation of raw data into actionable information. In this paper, we have presented a general edge mining approach (called G-SIP). To give concrete examples of edge mining, three instantiations of G-SIP have been presented: L-SIP, ClassAct and BN.

Edge mining may also alleviate the threat to privacy that the IoT applications pose. When the question to be answered is well-defined, BN-style edge mining can enforce "use limitation" by transforming the data into a form that cannot be misused.

Although the edge mining algorithms presented originate in the WSN domain, the approaches are applicable to the IoT more generally. Saving energy at the edge of the network is critical to the success of many IoT applications. For example, mobile phones periodically sensing their location (say through GPS) could use L-SIP to substantially reduce the number of packets sent whilst tracking location within a bounded accuracy. Similarly, location could be summarised in terms of dwell regions (or BN bins) such as "at home", "at work", "shopping", allowing the BN algorithm to track a dwell "histogram". This approach would substantially reduce the amount of data stored or transmitted but also provide a useful summary and help identify when significant changes occur. ClassAct illustrates that even complex sensory signals can be successfully compressed and that the fundamental approach is not necessarily limited to slow moving signals.

The G-SIP template provides a general approach to edge mining that is suitable for a broad range of application domains. The key challenges for the application developer are:

1) To be able to identify what information is needed to be delivered by the application;
2) To formulate and formalise the transformation of raw measurements into application-relevant information;
3) To collect the right data and ensure that the data is right to deliver the informational outputs;
4) To produce a top down design that meets application informational requirements and ensure appropriate support for the edge mining approaches.

The application developer is not constrained in terms of platform or programming language—we have already deployed G-SIP edge mining systems on platforms as diverse as: Python on embedded Linux; TinyOS on Telos motes; and Z-Stack on TI 8051.

There is further work to continue to validate the edge mining approach, particularly to identify further applications that can benefit from current edge mining algorithms and develop new edge mining algorithms for new applications. The high-level notion of energy optimisation through packet reduction and application-level processing makes edge mining feasible on many hardware platforms and we believe it will have wide applicability and be a core component of many Internet of Things applications of the future.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128610001568

[2] O. Vermesan, P. Friess, P. Guillemin, S. Gusmeroli, H. Sundmaeker, A. Bassi, I. S. Jubert, M. Mazura, M. Harrison, M. Eisenhauer, and P. Doody, "Internet of things strategic research roadmap," Online: http://www.internet-of-things-research.eu/pdf/IoT_Cluster_Strategic_Research_Agenda_2011.pdf, 2011, cluster of European Research Projects on the Internet of Things.

[3] D. Goldsmith and J. Brusey, "The Spanish Inquisition Protocol: Model based transmission reduction for wireless sensor networks," in *Sensors, 2010 IEEE*, Nov. 2010, pp. 2043–2048, online: http://dx.doi.org/10.1109/ICSENS.2010.5690285. [Online]. Available: http://dx.doi.org/10.1109/ICSENS.2010.5690285

[4] J. Brusey, R. Rednic, E. I. Gaura, and J. Kemp, "Postural activity monitoring for increasing safety in bomb disposal missions," *Meas. Sci. Technol.*, vol. 20, no. 7, pp. 075 204, 11pp, Jul. 2009. [Online]. Available: http://dx.doi.org/10.1088/0957-0233/20/7/075204

[5] E. I. Gaura, J. Brusey, and R. Wilkins, "Bare necessities—knowledge-driven WSN design," in *Proc. 10th IEEE Sensors Conf.* IEEE Press, Oct. 2011, pp. 66–70.

[6] E. Gaura, L. Girod, J. Brusey, M. Allen, and G. Challen, *Wireless Sensor Networks: Deployments and Design Frameworks*. Springer, 2010.

[7] S. Madden. (2010, 7) Intel lab data. [Online]. Available: http://db.lcs.mit.edu/labdata/labdata.html

[8] T. Daniel, E. I. Gaura, and J. Brusey, "Wireless sensor networks to enable the passive house—deployment experiences," in *EuroSSC*, 2009, pp. 177–192.

[9] U. Raza, A. Camerra, A. Murphy, T. Palpanas, and G. Picco, "What does model-driven data acquisition really achieve in wireless sensor networks?" in *Pervasive Computing and Communications (PerCom), 2012 IEEE International Conference on*, Mar. 2012, pp. 85–94.

[10] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "Collection tree protocol," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09. New York, NY, USA: ACM, 2009, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/1644038.1644040

[11] D. Moss and P. Levis, "BoX-MACs: Exploiting physical and link layer boundaries in Low-Power networking," Stanford, Tech. Rep. SING-08-00, 2008.

[12] K. Klues, V. Handziski, C. Lu, A. Wolisz, D. Culler, D. Gay, and P. Levis, "Integrating concurrency control and energy management in device drivers," in *Proc. 21st ACM SIGOPS Symp. Operating Systems Principles*. ACM, 2007, pp. 251–264.

[13] K. S. J. Pister and L. Doherty, "Tsmp: Time synchronized mesh protocol," in *In Proceedings of the IASTED International Symposium on Distributed Sensor Networks (DSN08*, 2008.

[14] E. I. Gaura, J. Halloran, J. Brusey, R. Wilkins, and R. Rednic, "Sustainable future? building and life-style assessment," in *International Conference on Intelligent & Advanced Systems 2012*, 2012.

[15] R. Rednic, E. Gaura, and J. Brusey, "Classact: Accelerometer-based real-time activity classifier," in *Proc. 2nd WiSIG Showcase*. Teddington, UK: Sensors KTN, Jul. 2009, pp. 21–25.

[16] R. Rednic, E. Gaura, J. Brusey, and J. Kemp, "Wearable posture recognition systems: factors affecting performance," in *Proc. IEEE-EMBS Intl. Conf. Biomedical and Health Informatics (BHI 2012)*, Shenzhen, China, Jan. 2012, pp. 200–203.

[17] J. Brusey, R. Rednic, and E. Gaura, "Classifying transition behaviour in postural activity monitoring," *Sensors & Transducers J.*, vol. 7, pp. 213–223, Oct. 2009. [Online]. Available: http://www.sensorsportal.com/HTML/DIGEST/P_SI_98.htm

[18] T. Daniel, E. Gaura, and J. Brusey, "Homereact—the real-time environmental monitoring tool suite: Measure, decide, interact," in *Proc. 2nd WiSIG Showcase*. Teddington, UK: Sensors KTN, Jul. 2009, pp. 26–30.

[19] M. Langheinrich, "Privacy by design—principles of privacy-aware ubiquitous systems," in *Proceedings of the 3rd international conference on Ubiquitous Computing*, ser. UbiComp '01. London, UK, UK: Springer-Verlag, 2001, pp. 273–291. [Online]. Available: http://dl.acm.org/citation.cfm?id=647987.741336

[20] G. Anastasi, M. Conti, M. D. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537–568, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870508000954

[21] F. Marcelloni and M. Vecchio, "An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks," *Comput. J.*, vol. 52, no. 8, pp. 969–987, Nov. 2009. [Online]. Available: http://dx.doi.org/10.1093/comjnl/bxp035

[22] H. Mamaghanian, N. Khaled, D. Atienza, and P. Vandergheynst, "Compressed sensing for real-time energy-efficient ecg compression on wireless body sensor nodes," *Biomedical Engineering, IEEE Transactions on*, vol. 58, no. 9, pp. 2456–2466, 2011.

[23] G. Quer, R. Masiero, G. Pillonetto, M. Rossi, and M. Zorzi, "Sensing, compression, and recovery for wsns: Sparse signal modeling and monitoring framework," *Wireless Communications, IEEE Transactions on*, vol. 11, no. 10, pp. 3447–3461, Oct. 2012.

[24] G. Quer, D. Zordan, R. Masiero, M. Zorzi, and M. Rossi, "Wsn-control: Signal reconstruction through compressive sensing in wireless sensor networks," in *Local Computer Networks (LCN), 2010 IEEE 35th Conference on*, Oct. 2010, pp. 921–928.

[25] D. Tulone and S. Madden, "Paq: Time series forecasting for approximate query answering in sensor networks," in *EWSN*, 2006, pp. 21–37.

[26] ——, "An energy-efficient querying framework in sensor networks for detecting node similarities," in *MSWiM*, 2006, pp. 191–300.

[27] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey," *Wireless Communications, IEEE*, vol. 14, no. 2, pp. 70 –87, april 2007.

[28] M. Allen, L. Girod, R. Newton, S. Madden, D. Blumstein, and D. Estrin, "Voxnet: An interactive, rapidly-deployable acoustic monitoring platform," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, april 2008, pp. 371 –382.

[29] A. M. Ali, S. Asgari, T. C. Collier, M. Allen, L. Girod, R. E. Hudson, K. Yao, C. E. Taylor, and D. T. Blumstein, "An empirical study of collaborative acoustic source localization," *Signal Processing Systems*, vol. 57, no. 3, pp. 415–436, 2009.

[30] G. Werner-Allen, S. Dawson-Haggerty, and M. Welsh, "Lance: optimizing high-resolution signal collection in wireless sensor networks," in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 169–182. [Online]. Available: http://doi.acm.org/10.1145/1460412.1460430

[31] J. Ryder, B. Longstaff, S. Reddy, and D. Estrin, "Ambulation: A tool for monitoring mobility patterns over time using mobile phones," in *Computational Science and Engineering, 2009. CSE '09. International Conference on*, vol. 4, aug. 2009, pp. 927 –931.

[32] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava, "Using mobile phones to determine transportation modes," *ACM Trans. Sen. Netw.*, vol. 6, no. 2, pp. 13:1–13:27, Mar. 2010. [Online]. Available: http://doi.acm.org/10.1145/1689239.1689243

[33] Q. Chen, M. Hsu, and H. Zeller, "Experience in continuous analytics as a service (caaas)," in *Proceedings of the 14th International Conference on Extending Database Technology*, ser. EDBT/ICDT '11. New York, NY, USA: ACM, 2011, pp. 509–514. [Online]. Available: http://doi.acm.org/10.1145/1951365.1951426