Natural Language Search of Sensor Data

Keyi Zhang and Alan Marchiori
Department of Computer Science
Bucknell University, Lewisburg, PA
Email: {kz005, amm042}@bucknell.edu

***Abstract*—As sensors become more affordable, sensor networks are increasingly deployed to monitor diverse environments. However, these sensor network deployments often utilize different standards for communication and data storage. As a result, it is challenging to build large-scale pervasive systems able to find, query, and analyze information across a diverse set of sensor networks. Additionally, aggregating sensor data from various sources is difficult because data can be sampled using different levels, units, rates, and resolutions. To address these challenges, we have developed a pervasive sensor network search environment based on natural language processing and the semantic web. By using natural language, we can understand the context of the query and use semantic rules about the data to aggregate and transform data to more useful results. To demonstrate the system, we have deployed the environment in two application domains. In each domain, the system successfully answers domain-specific natural language queries.**

I. Introduction

Wireless Sensor Networks (WSN) are popular in both academia and industry. The number of wireless sensors deployed to monitor the weather, infrastructure, agriculture, and other critical areas continues to grow. As sensor networks are typically designed to produce and consume data within their own application domain, it is challenging to share and merge sensed data across different application domains.

Besides the main challenge in creating a pervasive data environment with the heterogeneities in a shared sensor network [1], for people who have little technical background, it is not easy to retrieve data from complex systems without assistance. Therefore, creating a method to intelligently select and fuse data from different heterogeneous sensor networks at a high level still remains a challenge.

Fortunately, there are several existing technological building blocks from which to start; semantic technologies provide a means to overcome the integration problem [2], where engineering-oriented sensor networks present limits on a large-scale complex system. There are two proposed semantic standards to model sensor devices: Semantic Sensor Network (SSN) from the World Wide Web Consortium (W3C) and Sensor Web Enablement (SWE) from the Open Geospatial Consortium (OGC). While these standards provide useful interfaces for pervasive data access, they do not provide an easy-to-use high-level interface to query and fuse data from heterogeneous networks. For instance, the Resource Description Framework (RDF) in W3C and SPARQL Protocol and RDF Query Language (SPARQL) in SSN both make sensor selection and fusion possible, but using this system can be challenging because of the application-specific query languages used.

However, the advance of Natural Language Processing (NLP) makes it easier to understand the semantic meaning of a text. NLP can help us identify the main objects that the inputs address and what type of objects they refer to. NLP has been applied to many application areas such as speech recognition and question answering. However, in order to perform pervasive sensor data searches, we need a system that will bridge the NLP technology and the sensor systems.

The contribution of this work is the extension and application of NLP to the sensor network domain to answer high-level queries that would traditionally be expressed in an application-specific query language. The natural language query components are linked to the semantic web and are therefore able to accommodate the heterogeneities present in underlying sensor networks, significantly simplifying pervasive sensor data access.

The structure of the paper is as follows: in Section II, we provide an overview of existing approaches; in Section III, we describe the system design and our design choices; in Section V, we evaluate the NLP pipeline implementation and present several applications using the proposed architecture.

II. Related Work

CASSARAM and CA4IOT are designed to select sensors effectively and efficiently [3] [4]. Semantic querying and quantitative reasoning are used to perform context-aware sensor searches and selection. CASSARAM also introduces user-priority-based weighted Euclidean distance comparison in multidimensional space so that indexing and comparison is faster and more efficient. However, users still need to use the slider-based interface and SPARQL search for further data queries.

Martin Molina *et al.* proposed a novel approach to use open geographic data to generate natural language description for hydrological sensor networks [5]. Instead of using NLP and ontology to derive geographical coordinates (latitude and longitude) as well as the type of physical quantity measured by each sensor, they use ontology to generate natural language from template-based semantic reasoning. The metadata information of each sensor in the sensor networks and geographic feature extraction are used in template selection rules to generate textual description for the sensors. Compared to our system, this new approach uses semantics and natural language in a reversed fashion: natural language is the output, rather than input.

Rui Yang and Lingfeng Wang developed a multi-agent based intelligent system for building energy management [6]. The system is able to accomplish complicated tasks by dividing

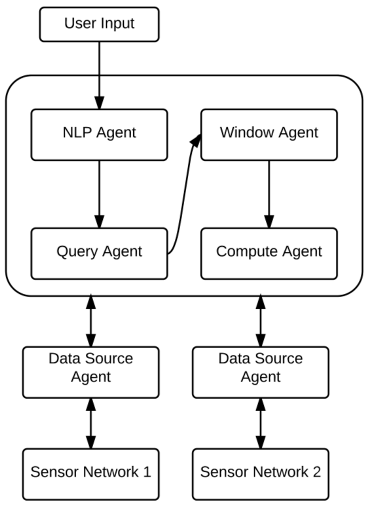

Figure 1. System diagram of the pervasive data environment.

and separating the overall goal into several local agents. For instance, personal agents provide the personal preferences of occupants to local agents and the central agents continuously interact with the local agents in the ever-changing environment. Therefore, the system provides an open architecture where agents can be easily configured and new agents can be added without interfering with the whole system. Although our system is similar to this system, the multi-agent architecture is applied to the more general problem of sensor selection and query instead of building energy management.

Context Query Language (CQL) [7] provides high level context query processing in the pervasive computing domain. CQL aims to translate a high level query, for instance, "is user is at home" into constraints on raw context data such as GPS position. Based on XML, CQL is able to present each context values such as position and environment by carefully defining a *context scope*. A Context Broker (CB) is also introduced into the system to provide a centralized server for sensor information. Multiple context providers can be plugged in and registered in the system because the CB acts as an aggregator of all related context raw data. However, CQL is not designed to provide a user-friendly interface.

III. Design

A pervasive data environment is designed using an agent-based model where tasks are distributed to multiple agents which work together to achieve the desired end goal. There are several reasons why we choose agent-based model. First, in distributed sensing, the agent-based system models a natural way to describe the environment. Each sensor networks can be thought of as a data agent. Within the environment, other agents exist, potentially executing on different hosts, that persist, analyze, aggregate, or deliver data to the user. Second, an agent-based system is self organizing and enables the implementation of privacy controls over each sensor network or database. The agent can apply different privacy policies based on the identity of the requester. Third, the agent-based architecture allows the system scale up by simply adding more agents to the system.

We define three core types of agents: data, compute, and query. Typically, one data agent would act as a gateway between each sensor network and the environment. The data agent must be customized to accommodate the heterogeneous nature of the underlying sensor network. Compute agents consist of smaller types of agents, handling all computation on raw data such as windowing, downsampling, and other mathematical functions. The query agent processes user queries, collects, and returns the results from other agents to the user.

An illustration of the system is shown in Figure 1. The natural language query is first sent to the NLP agent to extract query information. The extracted query information is then sent to the query agent, which sends the query to all registered data agents. Data agents independently determine if they can satisfy the query. Data agents that can satisfy the query then send their results back to the query agent. When query results are received by the query agent they are either sent back to the user or to a compute agent (the window agent in 1 is a specific type of compute agent) for further aggregation.

Sensor discovery is possible using two different levels of abstraction, namely, network-level and ontology-level. Network-level discovery uses network identifiers to identify sensors. Ontology-level discovery allows a sensor network to describe it's own ontology or use an existing network ontology. New ontologies are analyzed and measurement types exposed, this process is explained in Section IV. If a sensor network uses an existing ontology, no additional action needs to be taken.

To handle high-level data fusion with aggregation and even more complex data processing pipelines, we introduce "virtual sensors". A virtual sensor is a unit that does not produce observation results but takes other sensor's measurements and produces an intermediate result back to the system. A virtual sensor can be created during the system setup and treated as a part of the source data processing, or it can also be created on demand in response to a user request. Section III-B further describes this process.

A final issue to consider is privacy. As mentioned by Haowen Chan and Adrian Perrig, sensor networks aggravate issues of privacy because they make large volumes of information easily available through remote access [8]. To minimize the risk of exposing sensitive data to a malicious user, we enforce a configurable privacy policy. After connecting to the system, each agent is assigned a unique identifier, which is used as a part of their topic of interest. An example of the usage can be found in Section V. Each client must register before they can retrieve any sensor data. The system imposes a privacy policy on each client. These policies could simply exclude certain data or otherwise limit access by filtering or mutating the returned data.

A. Data Agent

Data agents deliver sensor time-series and metadata upon request. Each data agent listens to a common query request topic and responds if they are able to satisfy the query. Each request typically originates from a query agent and must contain the pertinent query information such as start time, end time, and sensor name or location. For historical data queries, the data source agent returns the full time-series data, if a match exists. However, for a real-time streaming query, the data agent will continuously publish sensor data to the system until the client terminates the query.

Data agents have complete control over the data exposed from their own sensor network to the rest of the system, including sensor time-series data and sensor metadata information. If a sensor network monitors household energy consumption, raw data at the household level or even block level could expose sensitive information. The data agent can either choose not to respond to queries that may have privacy violations, or apply privacy protection algorithms such as privacy-preserving data aggregation (PDA) [9] on the raw data and then return the result.

New sensor networks can be integrated into the system by implementing a customized data agent. The data agent should understand three different types of query: historical, streaming, and metadata. To return the result, the data agent publishes the data with a specific topic name, which will be received by interested clients using the secured topic policy. An example query request can be found in Section IV.

B. Compute Agent

Compute agents process data produced by other agents in the environment such as the query agent. Typical computations include statistical aggregation or mathematical expression evaluations.

The **Window Agent** is a specialized compute agent to address the problem of downsampling sensor data. For example, a sensor may publish sensed data once per minute, but the user may only require hourly average temperatures. The window agent can be used to avoid sending large amounts of unnecessary data to the user. Window agents are able to operate on real-time streaming data and therefore are persistent within the environment.

The **Virtual Sensors** are compute agents that create intermediate results for a data processing pipeline. To configure a virtual sensor, information from two perspectives are needed, namely, the topics of interest and a mathematical expression. The topics of interest are used in the data agent to publish data. The mathematical functions includes statistic-based aggregation and other numeric calculation. The mathematical expression is linked to the sensor measurement type within the ontology so that the system can create the virtual sensor on demand.

C. Query Agent

The query agent is a high-level agent that orchestrates the necessary connections between the data agent and other system agents such as compute agents which are necessary to respond to the user request. When a user query is delivered the first step is to parse the language (Section IV). Then, a data processing pipeline is created by query agent to satisfy the request. If the query needs windowing, the query agent will request the window agent to create a window for the query result. Then, the extracted query parameters are sent to the data agents. It is possible for zero, one, or many data agents to respond to the query. As results are received from the data agents, they are passed through the data processing pipeline and the results are delivered to the user as they become available.

IV. Natural Language Processing with Ontology

The most natural and intuitive way to seek information is to ask questions. For most people, “what is the average temperature in New York city from 2014 to 2015” is quite clear. As mentioned by Kok-Kiong Yap *et al.*, when humans locate objects, they do not do so in terms of absolute coordinates, rather they use identifiable landmarks [10]. There are of course many challenges; however, the natural language based query has the advantage that it is typically vaguely specified. This effectively helps to hide the heterogeneous nature of the underlying sensor networks. Through processing, the system must extract sufficient context to interpret time ranges, select, and fuse the sensor data correctly.

In order to handle sensor networks from different domains in a larger scale and provide a higher level aggregation, we define three aspects of a natural language query: *temporal*, *spatial*, and *characteristic scope* of the sensors. These aspects will be defined in the following sections. An example of a tagged natural language query including these aspects is shown in Figure 2. Based on these three scopes, we designed the NLP query agent to process incoming queries and output extracted information for the next agent using the query processing pipeline.

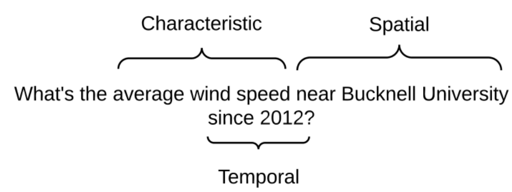

Figure 2. NLP tagging example

The NLP processing pipeline consists of several annotators that tag the text of interest in the query. Each scope has its own annotator to process the tokenized query. The NLP pipeline is shown in Figure 3. After the user submits the natural language query, the temporal annotator will first tag the tokens related to time and convert them to a timestamp range or present time reference. Next, the spatial annotator will extract the tokens specifying location or organization names and generate a SPARQL search to convert them into values such as coordinates. Then the characteristic annotator will tag the tokens related to sensor measurement type or aggregation function using an ontology lookup. Finally the pipeline will publish the result to the designated query topic.

A. Temporal Scope

As most sensor data are time-series based, we assume for now that every sensor data set has explicit temporal reference information. Data without this information still contains implicit temporal information defined over the set of all data. This implicit information could be used as a temporal reference for search. Therefore, temporal scope is essential to pervasive data access.

The temporal scope describes the time range of interest in the natural language query. We define two classes of temporal query: historical and streaming. The historical scope is used for

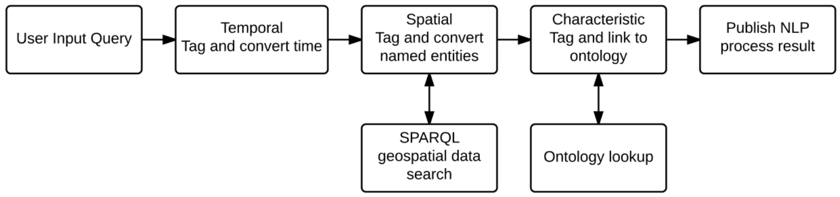

Figure 3. NLP processing pipeline for extracting *temporal*, *spatial*, and *characteristic scope* from a natural language query.

queries between two distinct time instances, occurring in the past, such as: "what was the total rainfall from 2012 through 2014." This query would be converted to the range 2012-01-01 to 2014-12-31. The streaming scope is for current (real-time) conditions, such as: "what is the current outdoor temperature?" It is also possible for a query to span both scopes, such as: "what is today's outdoor temperature?" In this case, the query is split into two queries. One uses the historical scope beginning at midnight through the current time and the second uses the streaming scope to capture future (real-time) data. The streaming request will be deleted automatically when the future time reference expires.

Relative time reasoning imposes another challenge to the temporal information extraction because it can represent both a time-range and a time instance. For example, "yesterday"can be interpreted as a time-range from 00:00:00 to 23:59:59 or exactly 24 hours ago as a time instance. The NLP pipeline will interpret relative times as a time instance if there is a second time instance to pair it with in order to create a range. Otherwise it will be interpreted as the appropriate time-range.

B. Spatial Scope

Spatial scope refers to the physical location of the sensors of interest. When describing locations, people tend to use named entities and relative range rather than numeric latitudes and longitudes, such as "within 20 miles of New York City". Most sensors do not provide named entities in their metadata information. Instead, they typically have coordinates describing the sensor's location. Therefore, we use NLP to extract the named entity from the query and then use geocoding to determine coordinates from the named entity.

Developing a system to obtain named entities is beyond the topic of this paper. Instead, we use common Named-entity recognition (NER) processing software such as Stanford NLP [11].

To enable more general spatial queries such as organization names and building names, we require a richer database of location names and coordinates. The public geocoding dataset has an enormous amount of geographic data [5]. In addition, one can simply link more data into the database if needed, simplifying future development. For example, when querying building performance sensors, where room-level spatial descriptions are required, one can simply populate the linked database with the additional RDF triples of interest. The geocoding data is accessed using a SPARQL query [12].

Although one can construct a SPARQL query to select all the sensors nearby a point of interest in SSN, there are several reasons why we only use the semantic web to query geo-locations. As mentioned in the section III-A, the system allows sensor networks to choose different privacy levels over individual queries. It is difficult to have privacy control over a linked data network. Additionally, many sensor networks do not use SSN standards and it is not easy or efficient to convert them to SSN standard.

C. Characteristic Scope

We define the characteristic scope as any additional information of sensor's metadata besides time references and locations. It can be the observation type, the observation unit, or even the observation event. For instance, "average wind speed" and "maximum energy consumption" are the characteristic scope of a natural language sensor data query. Ontologies used in SSN and SensorML have a rich collection of available observation types and other metadata information. For instance, SWEET [13] from NASA has more than $2,000$ observation types. However, we need to map the natural language tokens to the URI based observation types because these observation types are shared by many different ontologies. Based on the names defined in the ontology, we use rule-based token patterns to map natural language query keywords. As many ontology-based measurement types are well defined and close to natural language, we use a template to link the token and URI. For example "air temperature" is mapped to "air_temperature" and then to http://mmisw.org/ont/cf/parameter/air_temperature.

Aggregation functions, if specified, are also part of the characteristic scope. For instance, to satisfy the natural language query "maximum temperature", we first retrieve the raw data from all matched sensor networks for "temperature" and then apply the maximum aggregation function.

V. Implementations, Evaluations and Sample Applications

To implement the system, we applied the system on several real-world data sources from different application domains. For the publish-subscribe message framework, we choose MQTT, a lightweight messaging protocol. It is reported to have lower delay and lower packet loss rate than CoAP [14]. For NLP processing, we integrated the Stanford Named Entity Recognizer [15] (NER) into our own custom NLP pipeline. For geospatial linked data search we also use GeoNames [16] . GeoNames is a public linked data repository containing more than 150 million RDF triples.

A web application was built to demonstrate a search engine like interface for historical and streaming sensor data queries. The user enters a query in natural language and the system will return corresponding data visualized as a time-series plot. The web server acts as a client to the pervasive data environment and sends the query and receives the results. A web socket is used to deliver streaming data.

We evaluated the NLP pipeline using combined objective and subjective evaluation [17]. Due to the deterministic nature of the system, random temporal expressions and locations were chosen from tagged WikiPedia pages [18]. Random characteristic expressions were chosen from CF standard names. Then we generated queries using templates that are defined manually such as "What is the {{TYPE}} in {{LOCATION}} in {{TIME}}". Random confounding words chosen from NLTK corpora were also added into the query at a random position. The query is then input to the NLP pipeline and the tag conversion is used to verify the result. Table I shows the mean precision of 10,000 randomly generated queries. The baseline is tested with default Stanford NER library.

Table I. MEAN NLP SENSOR QUERY TAGGING AND CONVERSION PRECISION ON RANDOM QUERIES FOR BASELINE SCOPES AND QUERIES WITH CONFOUNDING WORDS ADDED.

Query Type	Precision
Baseline location extraction	0.77
Baseline temporal extraction	0.86
Baseline characteristic extraction	0.99
Sensor Query without confounding words	0.86
Sensor Query with 1 confounding word	0.80
Sensor Query with 2 confounding words	0.66
Sensor Query with 3 confounding words	0.52

A. Climatology Network

1) Problem Statements and Requirements: Although climatology data are compatible with geospatial network, the heterogeneous nature of sensor network makes it more challenging. Setting up a weather station is often easier than setting up other sensor networks and many companies and organizations use different standards for their public data. Therefore, the system should be able to: 1) answer the natural language query about temperature, precipitation, and wind speed. 2) fuse sensor data from different sensor networks, 3) and display the sensor metadata information such as geospatial location.

2) Solution: We use the Global Historical Climatology Network Daily [19](GHCN-Daily) and weather station data from a local research weather station to demonstrate fusing data between these heterogeneous sources. GHCN-Daily is an integrated database of daily climate summaries from land surface stations across the globe. These climate summaries contain maximum and minimum temperature, precipitation, wind speed, and snow fall at a daily resolution. We stored the GHCN-Daily data in a local MySQL server. The data from the research station are stored in a local OpenTSDB[1] server, which runs on top of HBase and Hadoop. Each sensor network has its own data source agent connected to the system. We applied the NASA SWEET [13] ontology for both GHCN-Daily metadata and the research station data. However, the sensor metadata for GHCN-Daily are stored in a local RDF database whereas the metadata are stored in a tag-based system in OpenTSDB. Google Maps is used to display the matched sensor information. We generated the token-based rule for NLP from the standard names defined by SWEET using the CF standard names as shown in Equation 1. After adding a URI prefix to the result, we generate the desired measurement type.

[1]http://opentsdb.net/

$$\{\ \{\ /\text{wind}/\ /\text{speed}/\ \}\ \}\Rightarrow\text{“wind\_speed”} \tag{1}$$

3) Evaluation: When the user searches for weather information about a named place, all weather stations within a configurable distance (we use 5 km) are queried. The results are aggregated and returned to the user. As a result, within 5 km of research weather station the results will include data from both the resaerch station and the GHCN-Daily dataset. Outside this distance, only data form the GHCN-Daily dataset is returned.

A query example is shown in Figure 4. With the help of GeoNames there are more than 739000 named places within 5 km of a weather station. The research weather station, for instance, is reachable from 84 different location names.

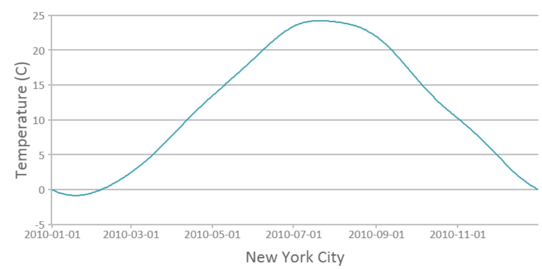

Figure 4. Result for "What is the average temperature in New York during 2010?"

B. Building Energy Monitoring

1) Problem Statements and Requirements: Monitoring Building energy is an important part of any building management system (BMS) and we can apply our system to make it easier for non-expert users to monitor building performance. We have deployed power meters in several residential halls at Bucknell University and logged the data at 1 Hz frequency to an OpenTSDB database. Although the data collection is at room-level resolution, due to concerns about privacy, only floor level or above access is allowed. Therefore, the system should be able to: 1) answer the natural language query about real-time power usage and historical power consumption, 2) compute integral of the energy consumption, 3) and only display floor-level aggregate results to preserve privacy.

2) Solution: Because existing geospatial ontologies do not provide vocabularies for room level information, we defined a simple ontology for room and building information so that token-based rules can be used in NLP. The sensor metadata are stored in RDF-based database and can be approached via SPARQL query. Listing 1 shows how to model a room with a simple ontology. The RDF document describes the room number, floor number, and the building number. We also designed a simple rule-based NLP tagger. An illustration of the token-based rule tagger is shown in Equation 2.

$$\{\ /\text{room}/\ \{\ \text{ner:NUMBER}\ \}\ \}\Rightarrow\text{“ROOM”} \tag{2}$$

The example above will tag tokens such as "room 101" and "room one zero one". After tagging the token, we further extract the room number information and convert it into SPARQL search. We also use virtual sensors and configure a MathML based numeric integral expression for publishing

```
<rdf:Description rdf:resource="http://ex.com/place/room−101">
  <rdfs:label>Room 101</rdfs:label>
  <rdf:type rdf:resource="http://voc.ex.com/rooms#Room" />
  <spacerel:within rdf:resource="http://ex.com/floor=1" />
  <spacerel:within rdf:resource="http://ex.com/building−A" />
<rdf:Description>
```

Listing 1. RDF Description example for a room

data. As the computation needs the integral, we need to take the summation of each second. The aggregation will reset every day. Although the description files contain the room information, the data source agent will not return any data if a single room is requested because of the privacy concerns. However, it will return aggregated data for floor or building level queries.

3) Evaluation: In our system, users can query the historical power data as well as aggregated energy consumption. Real-time streaming data also includes high-level energy consumption, which is handled by a virtual sensor.

Figure 5 shows the visualization after the user input "What is the energy demand at Silbermann Hall floor 1 yesterday." The query successfully combines four different rooms using aggregation (sum) using a compute agent.

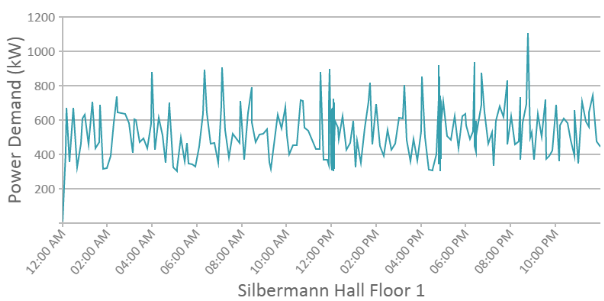

Figure 5. Building energy monitoring search results for "What is the power demand at Silbermann Hall floor 1 yesterday"

VI. Conclusion and Future Work

In this paper, we proposed a new sensor network search engine based on natural language processing and the semantic web. The semantic web provides rich rules and knowledge for NLP token tagging and extraction. The underlying architecture is an agent-based system which divides the query process into many smaller-scale tasks. Such an agent-based design makes the system highly scalable and distributed. For instance, NLP, query, and computing requests are handled by different agents on different machines. Adding a new sensor network to the system is also very simple and only requires that a data source agent to join the system. We recognized the concerns over data privacy, therefore we designed the data source agent in such a way that it has absolute privacy control over the data flow from the underlying sensor network to the system.

We recognize that the current system can only reason using a simple aggregation approach. In the future, we will improve semantic reasoning to answer questions such as "which building had the highest energy usage yesterday" and "where is the coldest location". Additional work will include adding more sensor networks into the system and making the search engine publicly available.

References

[1] A. Kansal, S. Nath, J. Liu, and F. Zhao, "Senseweb: An infrastructure for shared sensing," *IEEE multimedia*, no. 4, pp. 8–13, 2007.

[2] F. Gramegna, S. Ieva, G. Loseto, and A. Pinto, "Semantic-enhanced resource discovery for coap-based sensor networks," in *Proceedings of the 5th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*, June 2013.

[3] C. Perera, A. Zaslavsky, P. Christen, M. Compton, and D. Georgakopoulos, "Context-aware sensor search, selection and ranking model for internet of things middleware," in *Proceedings of the 14th International Conference on Mobile Data Management (MDM)*, vol. 1, June 2013, pp. 314–322.

[4] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Ca4iot: Context awareness for internet of things," in *Proceedings of the IEEE International Conference on Green Computing and Communications (GreenCom)*, Nov 2012.

[5] M. Molina, J. Sanchez-Soriano, and O. Corcho, "Using open geographic data to generate natural language descriptions for hydrological sensor networks," *Sensors*, vol. 15, no. 7, pp. 16 009–16 026, 2015.

[6] R. Yang and L. Wang, "Development of multi-agent system for building energy and comfort management based on occupant behaviors," *Energy and Buildings*, vol. 56, pp. 1 – 7, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378778812005348

[7] C. Fra, M. Valla, and N. Paspallis, "High level context query processing: An experience report," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, March 2011.

[8] H. Chan and A. Perrig, "Security and privacy in sensor networks," *Computer*, vol. 36, no. 10, pp. 103–105, Oct 2003.

[9] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: Privacy-preserving data aggregation in wireless sensor networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM 2007)*. IEEE, 2007, pp. 2045–2053.

[10] K.-K. Yap, V. Srinivasan, and M. Motani, "Max: human-centric search of the physical world," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*. ACM, 2005, pp. 166–179.

[11] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 55–60. [Online]. Available: http://www.aclweb.org/anthology/P/P14/P14-5010

[12] E. PrudHommeaux, A. Seaborne *et al.*, "Sparql query language for rdf," *W3C recommendation*, vol. 15, 2008.

[13] R. G. Raskin and M. J. Pan, "Knowledge representation in the semantic web for earth and environmental terminology (sweet)," *Computers & geosciences*, vol. 31, no. 9, pp. 1119–1125, 2005.

[14] D. Thangavel, X. Ma, A. Valera, H.-X. Tan, and C.-Y. Tan, "Performance evaluation of mqtt and coap via a common middleware," in *Proceedings of the 9th IEEE International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, April 2014.

[15] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005, pp. 363–370.

[16] "Geonames," http://www.geonames.org/, accessed: 2015-08-25.

[17] P. Paroubek, S. Chaudiron, and L. Hirschman, "Principles of evaluation in natural language processing," *Traitement Automatique des Langues*, vol. 48, no. 1, pp. 7–31, 2007.

[18] J. Nothman, N. Ringland, W. Radford, T. Murphy, and J. R. Curran, "Learning multilingual named entity recognition from Wikipedia," *Artificial Intelligence*, vol. 194, pp. 151–175, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.artint.2012.03.006

[19] R. Vose, M. Menne, I. Durre, and B. Gleason, "GHCN Daily: A Global Dataset for Climate Extremes Research," *American Geophysical Union (AGU) Spring Meeting Abstracts*, p. A8, May 2007.