Information Systems 48 (2015) 151–163

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

An analytics appliance for identifying (near) optimal over-the-counter medicine products as health indicators for influenza surveillance

Ruhsary Rexit [a,b,*], Fuchiang (Rich) Tsui [a,*], Jeremy Espino [a], Panos K. Chrysanthis [b,*], Sahawut Wesaratchakit [a], Ye Ye [a]

[a] The RODS Laboratory, Department of Biomedical Informatics, University of Pittsburgh, Pittsburgh, PA 15260, United States
[b] The ADMT Laboratory, Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15260, United States

ARTICLE INFO

Available online 10 June 2014

Keywords:
OTC analytics appliance
Distributed search
Outbreak detection
Time series analysis
Syndromic surveillance

ABSTRACT

In the era of "Big Data", a challenge is how to optimize our use of huge volumes of data. In this paper, we address this challenge in the context of a public health surveillance system which identifies disease outbreaks using individual and population health indicators. Our goal is to automate and improve the accuracy of the selection process of the health indicators, a process which is data-intensive and computationally expensive. The health indicators selection process traditionally has been carried out manually by public health experts in collaboration with health data providers. In particular, we present an approach for identifying sets of over-the-counter (OTC) medicine products whose aggregate sales correlate optimally with aggregate counts of emergency department (ED) visits. Towards this goal, we propose an OTC Analytics Appliance which utilizes a distributed search engine to efficiently generate time series of time-stamped records and supports "plug-and-play" search and correlation functionalities. Using the OTC Analytics Appliance with the Pearson correlation coefficient function, we evaluate Brute-force search, Greedy search, and Knapsack search for their ability to select the optimal or suboptimal set of OTC products automatically. Our results show that greedy search is the most preferable, producing a set of OTC products whose sales that correlate optimally or near optimally to ED visits, while achieving acceptable search times with large datasets. Also, our evaluations show that our approach using the greedy search can be potentially used to efficiently identify different optimal OTC medicine products for detection of different types of disease outbreaks.

1. Introduction

An outbreak detection system is a public health surveillance system used for identifying increases in the incidence rate of a disease. A Syndromic Surveillance system is a type of outbreak detection system that monitors the health status of a community and identifies outbreaks using individual and population health indicators. Those health indicators are available before a confirmed diagnoses or laboratory confirmation [1]. Syndromic Surveillance systems have used various data sources as health indicators, including over-the-counter (OTC) medication sales, emergency department (ED) chief complaints, school absenteeism data, and web search queries [2–4]. Among these, ED data generally serves as the core data source of many Syndromic Surveillance systems such as BioSense [5] and RODS [2]. Researchers have shown that common outbreaks can be

* Corresponding authors.

http://dx.doi.org/10.1016/j.is.2014.05.008
0306-4379/

detected 1–2 weeks earlier with ED data than through conventional disease reporting methods [6].

A common methodology employed in Syndromic Surveillance systems is to aggregate health-related temporal events into time series that are analyzed algorithmically for the detection of outliers. The intuition of using OTC medication sales is that sick individuals typically purchase some OTC medications to treat themselves before seeing a doctor. For example, in an Syndromic Surveillance system using OTC medication sales as an indicator to detect influenza outbreaks, the epidemiologist would analyze a time series of the daily sales of all cough syrup, thermometers, and fever reducers in a specific geographic region. If daily sales of these products exceed some threshold (e.g., three times the standard deviation from a baseline value), that could indicate a disease outbreak.

The effectiveness of a Syndromic Surveillance system depends on three factors:

1. The availability of health related data from providers e. g, hospitals, food and drug retail industry.
2. The selection of good health indicators (such as specific medications sold) to be used for the detection of a disease outbreak by the Syndromic Surveillance systems.
3. The ability of the Syndromic Surveillance system to provide query processing and data analyses *on the fly*, as the data arrives at the system.

Both the bootstrapping, i.e., the selection of health indicators, and the outbreak detection activities of a Syndromic Surveillance system during its deployment involve data filtering and spatio-temporal aggregation over time series. The key difference is that in the former case of the selection of health indicators the query processing and analysis is carried out on historical data as opposed to the latter case of detection which is carried out on current data.

Despite this difference, the increasing volume of monitored OTC product sales in United States is a challenge for Syndromic Surveillance systems, we must be able to identify an optimal set of health indicators for a given syndrome as well as meet the near-real-time requirements of detection.

Initially, we employed and explored data warehouses and externally-implemented continuous queries, but we soon discovered that the performance of the datawarehouse- based online analytical processing (OLAP) was not able to support either the selection process or the detection process. The data warehouse approach required both large amounts of storage for storing the fact tables and pre-computed statistics and also incurred large overhead in first storing and then retrieving the data for analysis [7, 8]. For this reason we subsequently explored the use of (1) a data stream management system which efficiently execute continuous queries before storing the data [9–11], to support the detection process and (2) a distributed query processing system where filtering and aggregation take place over collaborating computers, possibly in the cloud, to support the selection process which often requires multi-year worth of baseline data. In this paper, we present the result of our exploration of using a distributed search engine to implement the selection process for identifying the optimal set of health indicators.

Specifically, our work in this paper was motivated by three observations: (1) traditionally, the selection process has been performed manually by public health experts, (2) it was limited by the amount of data used, and (3) traditionally, the analysis was centered in the measuring of the relationship of a manually selected set of health indicators to some survey, such as of sick individuals or hospital visits. These observations formulated our hypothesis that *if we want to accelerate the selection process and make it more accurate, then we need an efficient solution for processing large volume of aggregated data and time series that automatically identify the optimal set of health indicators*.

To support our hypothesis, we developed an *OTC Analytics Appliance* that efficiently generate time series of time-stamped records such as unit sales of certain OTC products and used it to compare different search algorithms to identify a set of thermometer products (such as strip or digital, oral or forehead for babies or adult thermometers) whose sales over time optimally, or close to optimally, correlates with ED visits for symptoms (such as fever) consistent with Constitutional syndrome.

Contributions: The two key contributions of this paper is as follows:

- The development of an OTC Analytics Appliance, which utilizes a distributed search engine, called ElasticSearch [12], to efficiently generate time series of time-stamped records. The OTC Analytics Appliance provides an Optimal OTC Identifier module with “plug-and-play” search and correlation functionalities and a GUI to display time series graphs.
- An evaluation of three search algorithms, *brute-force search*, *greedy search*, and *dynamic programming* (knapsack search) for their ability to select the minimum set of OTC products automatically. Our results using the Pearson correlation coefficient function show that greedy search is competitive to the brute-force search, producing a set of OTC products whose sales optimally correlate to ED visits, while at the same time maintaining scalability with large datasets. The knapsack search exhibits the worst performance. Also, our evaluations show that our approach using the greedy search can be used to efficiently identify different optimal OTC medicine products for detection of different types of disease outbreaks.

Roadmap: Section 2 introduces the OTC Analytics Appliance. Section 3 presents our experimental datasets and methods. Section 4 presents the experimental results for the optimality of the three search algorithms using data collected during one year period. Section 5 evaluates the robustness of the three search algorithms whereas Section 6 presents their scalability evaluation with an extended dataset of four years. Section 7 briefly reviews related studies and Section 8 concludes with future work.

2. System architecture

This section gives an overview of OTC Analytics Appliance, then explains each module of the system such as

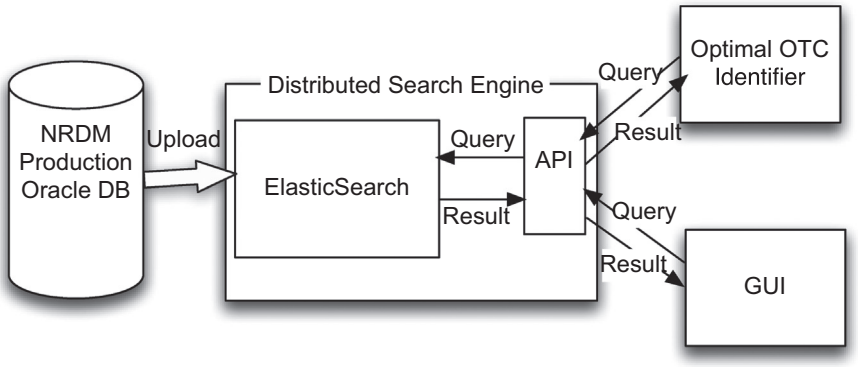

Fig. 1. The OTC Analytics Appliance architecture.

distributed search engine, Optimal OTC Identifier and *graphical user interface* (Fig. 1).

2.1. System overview

As with all "Big Data" analytics, the two fundamental challenges that we need to address are the inter-related problems of data storage and data processing. In the context of our work, the latter, i.e., the data processing, is the biggest challenge of the two. One solution to these problems was to adopt one of the emerging MapReduce-based systems, such as Apache Hadoop [13] that supports distributed processing and provides a distributed file system. However, writing MapReduce jobs is not exactly trivial. It requires sufficient programming skill to rephrase problems into an algorithmic form suited for MapReduce execution. Given the ease of creating SQL queries over equivalent MapReduce jobs manually (especially for complex queries), we explored an alternative that offers users the ability to phrase queries in a declarative syntax. Although Hive [13] was proposed as a front-end to Hadoop to provide an SQL-like syntax, we selected to use the Elastic Search [12,14], a RESTful, distributed search engine (also based on Apache) because it provides both our needed functionality as well as a straight forward data path from our current National Retail Data Monitor (NRDM) production database server to a web-based graphical interface to display time series graphs.

Fig. 1 shows the overall architecture of our system which runs on top of a cluster or a network of computers. Data collected from the various health-related providers are stored in an Oracle Database Server. Data needed for a given analysis are extracted using SQL queries from our production Oracle database and saved to individual comma delimited files. The SQL query can support any time period, however, given the popularity of retrievals of annual data, the system provides pre-compiled queries that accepts one year time period for convenience and speed. A loader job parses the retrieved data, transforms each record into a JSON string and sends the JSON strings via HTTP to the cluster/network using the batch index functions of Elastic Search. The API provides an interface for the distributed search engine, where a submitted query is parsed and passed to the ElasticSearch and the query results from ElasticSearch are returned to the query invoker, e.g., Optimal OTC Identifier and GUI.

2.2. Distributed search engine

A distributed search engine is a system wherein data records are stored over a network of computers (or nodes) which act collaboratively to answer queries as well as to balance the workload among them automatically and transparently. These records are indexed locally within each node, which means there is no global catalog (hash table) of data distribution but each node has partial catalog. Thus, data retrieval topologically is not a star but rather a star-chain as shown in Fig. 2 of a simple configuration with three nodes. When a query is issued to the network (distributed system), the query is directed to the most lightly loaded node, Node 3 (Step 1 in Fig. 2). Based on its local catalog, Node 3 identifies which records stored locally meet the query parameters and which nodes store records that might meet the query parameters (Step 2). Then, the query is forwarded to all the identified nodes with records which might be a part of the query result, Node 1 in our example (Step 3). When Node 1 receives the request from Node 3, it carries out the same steps as Node 3, identifying and forwarding the query to Node 2 (Step 4). When a node, such as Node 2, receives a query and is not aware of any other node in the chain to further forward the query, it returns the locally stored records as a part of the query result back to Node 1 (Step 5). In turn, Node 1 appends its records which are part of the query result to the ones received from Node 2 and sends them to Node 3 (Step 6), which in turn, sends them to the Client (Step 7).

We constructed the distributed search engine of the OTC Analytics Appliance using Elastic Search. Elastic Search is a distributed search engine built on top of a text search engine called Lucene [12,14]. Elastic Search stores a document (records in our case) in shards located on different nodes. It decides which shard to put the document in by computing a hash tag of the documents primary keys, which by default is a tuple of _index, _type and _id of a document. In our case of records, the hash tag was generated based on date of sale, store zip code, Universal Product Code and promotion status. It then uses the hash tag *modulo* the number of shards (nodes) in the system to pick a shard to store the document. As mentioned above, we chose Elastic Search primarily for its ease of deployment and built in query functions for

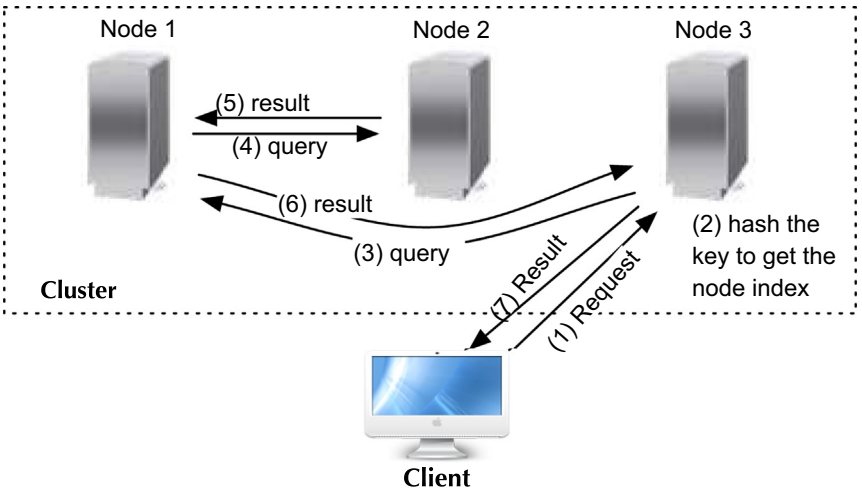

Fig. 2. Distributed computing Elastic Search scheme.

grouping data over time periods and feature values suitable for time series analysis (i.e., faceting). Also, since it is open-source, our OTC Analytics Appliance can be easily replicated and used by the community.

We constructed an API using Java to facilitate the creation of Elastic Search queries for the search engine. Elastic Search queries are written using JSON. For the current purpose of the OTC Analytics Appliance, the API supports two different queries: `getTimeSeriesWithDistribution` and `getTimeSeries`. These JSON queries provide many parameters, such as state, county, zip code, vendor, store, product category, and product IDs, etc., that allow querying with multiple conditions by setting list of values for each of them. They return time series objects for a set of product IDs aggregated over days, weeks, months or any time period, that are not weighted average. Their full specifications are as follows:

```
getTimeSeriesWithDistribution (String indexType, Date
    startDate, Date endDate,
  Boolean promotion, String[] stateList, String[]
    zipFilter, String[] fipsFilter,
  String[] catFilter, String[] vendorFilter, String[]
    gtinFilter, String[] storeFilter,
  String interval, Integer numCat, Integer numVendor,
    Integer numZipcode, Integer numFips,
  Integer numState, Integer numGtin, Integer numStore,
    String breakdownOf,
  Boolean timeSeries, Boolean distribution);
getTimeSeries (String indexType, Date startDate, Date
    endDate, Boolean promotion,
  String[] stateFilter, String[] zipFilter, String[]
    fipsFilter, String[] catFilter,
  String[] vendorFilter, String[] gtinFilter, String
    [] storeFilter, String interval);
```

2.3. Graphical user interface

In order to help public health experts to better understand of the OTC data and experiment with query results, we developed a web-based graphical user interface (GUI) to display time series graphs of the OTC data, shown in Fig. 3.

The GUI has controls to select the time period of data to visualize (start date, end date) and allows to zoom in and out on the plot by selecting different time frame (1month, 3 month, 6 month, etc.). Users can also specify or filter vendor ID, state, FIPS (county code), zip code, store ID, OTC categories, GTIN (Global Trade Item Number) or UPC (Universal Product Code), date of sale, total unit sales, promotion status (promoted unit sales) by entering the appropriate values to the filter boxes. By being able to set these various query constraints, users can reduce the number of results returned and better visualize a portion of the dataset that the users would like to observe and explore.

In addition, the GUI displays the breakdown of the data based on geographic location, category, store type and OTC product in the form of bar charts or pie charts. Users are able to specify a limit or size of the breakdown categories.

We programmed the GUI in Java and Javascript utilizing the JQuery, High Charts and Play Framework libraries.

2.4. Optimal OTC identifier

The Optimal OTC Identifier is the key module responsible to generate the optimal OTC product set. It is designed to use the ElasticSearch API to query OTC time series based on the specified input parameters and further process them (e.g., aggregates daily time series into weekly and applies filters) as needed before are passed as inputs to the search algorithms. Currently, the Optimal OTC Identifier provides two different filters: *product-level* and *store-product* filtering. The product-level filtering excludes OTC products that have less than a specific number of days of sales over a specified period. The store-product filtering, on top of product filtering, also excludes stores that have less than a specific number of days of sales over the study period.

The input parameters of Optimal OTC Identifier are API name, API URL, the file name that store the list of OTC products to be considered in the analysis, the file name that has the daily ED visits, product level filtering

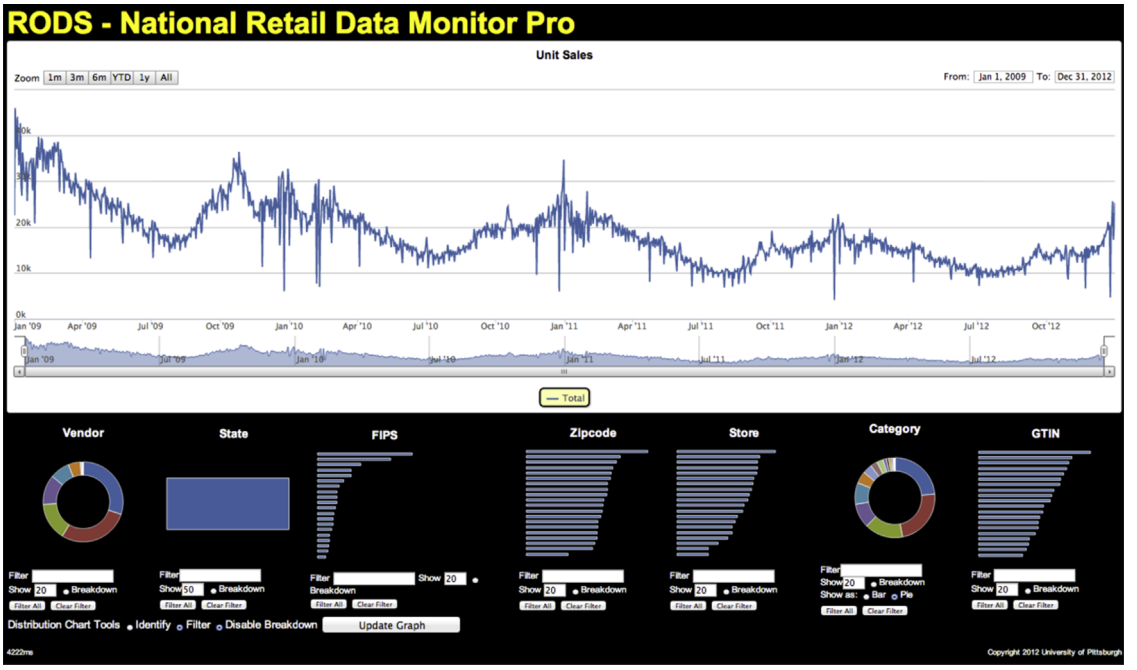

Fig. 3. Distributed search engine GUI.

(specify the number of days of sales for per product as a filter), store level filtering (specify the number of days of sales for per store as a filter), the file name that stores list of store codes, start date (starting date for the query), end date (end date for the query), and type of the search algorithms (specify the algorithm to be used).

The module outputs optimal or suboptimal product set (that has the highest correlation value depends on the search algorithms we choose as an input), correlation value, an image file that has the weekly time series chart of OTC set and ED visits.

Currently, the Optimal OTC Identifier uses the Pearson correlation coefficient function [15] to compute correlation values between the two time series: a set of OTC product weekly sales and weekly ED visits. The Pearson correlation coefficient equation is written as the following:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2\sum_{i=1}^{n}(y_i-\overline{y})^2}} \quad (1)$$

In Eq. (1), r is a measure of the correlation coefficient (linear dependence) between two variables (time series) X and Y written as x_i and y_i (where $i=1, 2, \ldots, n$), n is the sample data size, and $\overline{x}$ and $\overline{y}$ are the mean values of two samples from the data.

We implemented three search algorithms – brute-force, greedy, and dynamic programming (Knapsack search algorithm) – using the Java programming language.

Brute-force search algorithm: The brute-force search or exhaustive search looks at all possible combination of thermometer products that is queried from our (distributed) search engine and computes the correlation value of each set. It retains the set that has the highest correlation value with ED visit data. The advantage of this approach is that it is complete and optimal, i.e., it searches the entire space of available OTC medical product sets. The disadvantage is that its execution time is proportional to the number of candidate solutions and not scalable to large datasets. Specifically, the time complexity of brute-force search is $O(2^N)$ [16] where N is the number of OTC medical products.

Greedy search algorithm: We designed a greedy search algorithm that uses a successor function that removes OTC medical products from the set. It starts the computation from an initial set of all OTC medical products, and gradually eliminates one product from the set that has the least contribution to the correlation at each step. This algorithm is complete (i.e., always finds a solution if one exists), but does not always give the optimal solution. However, it gives the optimal solution most of the time as shown by our experiments in Sections 4 and 5. Compared to the brute-force search, it is efficient and scalable to our large datasets. The time complexity of this type of greedy search is $O(N^2)$.

Knapsack search algorithm: This approach is adopted from algorithm for $0-1$ Knapsack problem. It is a dynamic programming method for solving optimization problems. The Knapsack search computes the solutions to the subproblems once and stores the solutions in a table so that they could be reused later. Specifically, it selects one OTC medical product at each step, and adds it to the knapsack. If adding this OTC product to the subset of OTCs in the knapsack increases the correlation value of the subset with ED visits, then it remains in the knapsack. Otherwise, the OTC product is discarded (does not kept in the knapsack) and moves to the next OTC product in the list. Once an OTC product is eliminated from the knapsack, it does not have another opportunity to be clustered with the subset in the

knapsack. By performing in this way, the Knapsack solution eliminates some of the possible combinations that may actually include the optimal subset. This means that the order of the input set determines the output set. Although there is a way to find out which order of the input set gives an optimal set, we decided not to implement it since it is an expensive operation (i.e., almost behaves like brute-force). Thus, the implemented Knapsack search is not optimal but it is complete. It has much less execution time comparing to the brute-force algorithm. Like the greedy search, it shows a time complexity of $O(N^2)$ [16] and it reduces time complexity at the expense of memory. We decided to use this algorithm as a kind of low bound in our evaluation.

3. Experimental data and evaluation methods

OTC medication sales: We obtained sales data from the National Retail Data Monitor (NRDM). NRDM is a public health surveillance system that collects and analyzes daily over-the-counter point of sale data to rapidly identify disease outbreaks. NRDM was built by the RODS Laboratory at the University of Pittsburgh in collaboration with the food and drug retail industry, as well as state and local health departments [17].

NRDM collects daily sales data from over 33,304 (30,820 active) stores from 15 (12 active) different retailers across the United States and has been operational since 2003. NRDM has a transactional database of 1.23 billion records for over 9000 medications over a period of 9+ years. As alluded previously (Section 2.3), each record contains vendor ID, state, FIPS (county code), zip code, store ID, OTC category, and GTIN (Global Trade Item Number) or UPC (Universal Product Code), date of sale, total unit sales, promotion status (promoted unit sales). The reason why we keep track of promotion status is that if there is a rise in a category that predominated by a single product and that product is promoted we can infer that it is a false signal.

We loaded four years of transactional OTC data for Pennsylvania (from 1 January 2009 to 31 December 2012) from NRDM onto our OTC Analytics Appliance's distributed search engine deployed on a five node cluster. Each node was allocated 20 GB of RAM, 24 GB of disk space and two CPUs. The data was distributed over five shards with replicas (10 total shards) and comprised 18.5 million records. The data occupied 6.2 GB and it took approximately 35 h to index the data into the cluster.

ED visit data: We retrieved time series of daily ED visits for Constitutional chief complaints in Allegheny County and the entire state of Pennsylvania, for the time period of 2009–2012, from the Pennsylvania Real-time Outbreak and Disease Surveillance (RODS) system. The PA RODS System is a public health surveillance system for the state of Pennsylvania that collects de-identified ED visit data from 166 (111 active) hospitals since 1999. Emergency department visits and daily aggregated number of different syndrome categories were obtained from emergency departments. Hospitals send patient visit data including registered chief complaint to RODS Laboratory from clinical encounters over virtual private networks and leased lines using the Health Level 7 (HL7) message protocol in real time. CoCo (Complaint Coder) automatically classifies the registration chief complaint from the visit into one of the seven syndrome categories (Respiratory, Botulinic, Gastrointestinal, Neurologic, Rash, Constitutional, Hemorrhagic) using Bayesian classifiers [2,18].

Evaluation: We evaluated the three search algorithms by comparing their search results in terms of (1) *optimality*, measured by the correlation coefficient values (CCVs) computed by the Pearson correlation coefficient function (provided in our OTC Analytics Appliance), (2) *robustness* in terms of consistently identifying optimal product set over different periods of time, and (3) *scalability*, measured by their runtime.

In our evaluation, we generated time series of thermometer sales for Allegheny County, Pennsylvania. Although the NRDM offers 23 OTC categories, we chose the thermometer sales category as our indicator for influenza outbreaks because researchers found a strong correlation (the correlation value is 0.91) between patients with Constitutional syndrome visiting emergency departments (EDs) and OTC thermometer sales in Pennsylvania in past influenza seasons [19]. Villamarin et al. also demonstrated high correlation (the correlation value is 0.89) between actual and predicted ED visits using thermometer sales data [20]. There were 596 OTC thermometers. For our experiments, we selected daily aggregated ED visits for the Constitutional category because it generalizes complaints such as fever, chills, or malaise. To reduce the impact of noisy data, we use in our search queries both filtering processes, namely product-level filtering and store-product filtering, provided by the Optimal OTC Identifier of the OTC Analytics Appliance.

We conducted our evaluations on an Apple iMac computer (3.06 GHz Intel dual cores CPU, 4 GB RAM). The iMac computer served as a client computer that queried the distributed system described in Section 2.

4. Optimality evaluation

An optimal solution gives an OTC product set that has the highest correlation value with ED visits and optimality can be guaranteed with a brute-force algorithm. Since the brute-force algorithm takes a long time to produce an optimal solution, an alternative algorithm that gives a suboptimal or close to optimal solution to the problem is often more useful. In this section we evaluate brute-force search, greedy search, and knapsack search for their ability to select the set of OTC thermometer products that optimally correlate with ED visits. In this optimality evaluation, we use only one year of data, 2009, which were aggregated to weekly time series. As a reference (baseline) we used the correlation value between ED visits and all 596 thermometer products — 0.9077. We conducted two optimality experiments: (1) with product-level filtering and (2) with store-product filtering.

4.1. Product level filtering

We created multiple datasets by applying varying levels of product-level filtering. By setting a product sales threshold of 10 days, we generated a dataset (Dataset 1) that

Table 1
Comparison of search algorithms with different OTC product sets after product level filtering (CCV⋆: Pearson correlation coefficient value; Size†: number of OTCs in the set).

Datasets	Filter criteria per product	Filter result	Brute-force search		Greedy search		Knapsack search	
		Size†	CCV⋆	Size†	CCV⋆	Size†	CCV⋆	Size†
Dataset 1	≥ 10	28	0.9592	9	0.9592	9	0.9586	11
Dataset 2	≥ 20	26	0.9592	9	0.9592	9	0.9583	10
Dataset 3	≥ 30	25	0.9589	7	0.9589	7	0.9285	15
Dataset 4	≥ 50	23	0.9589	7	0.9589	7	0.9567	11
Dataset 5	≥ 70	20	0.9586	6	0.9586	6	0.9565	9

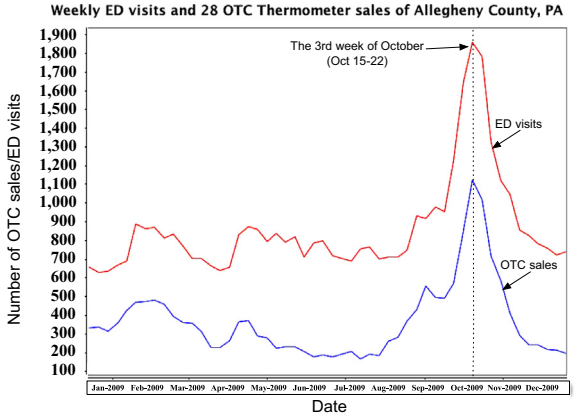

Fig. 4. 28 OTC thermometer sales and ED visits.

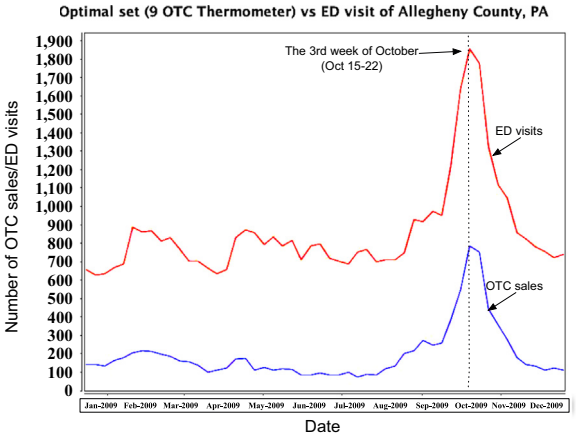

Fig. 5. Nine OTC thermometer sales and ED visits.

included 28 OTC (out of 596) thermometer products. Then, by varying the threshold from 10 to 70 days, we generated additional datasets with 26 OTC thermometer products (Dataset 2 with threshold 20 days), 25 OTC thermometer products (Dataset 3 with threshold 30 days), 23 OTC thermometer products (Dataset 4 with threshold 50 days), and 20 OTC thermometer products (Dataset 5 with threshold 70 days). Incidentally we found that using a filtering threshold of 40 and 60 days generated exactly the same set of products found in Dataset 3 and Dataset 4, respectively. In all cases our datasets have the property that a dataset with a smaller set of products has a subset of products found in datasets with a larger set of products.

Optimal sets with product level filtering: We ran the brute-force search algorithm against all the Datasets 1–5. Table 1 (Column 4) shows the number of OTC products in the identified *optimal* set (i.e., output search result) and their corresponding aggregated correlation value with the ED values, for each dataset. In all cases, the aggregated correlation value of the optimal set is larger than the correlation value 0.9077 of the total 596-product set as well as of the correlation value of 0.9079 of the largest, 28 products Dataset 1. In all cases, the aggregated time series of the optimal sets and ED visits for Constitutional syndrome are peaked at the third week of October (October 15–22), 2009. Figs. 4 and 5 show this for Dataset 1 and its optimal nine OTC product sets, respectively. It is interesting to note that the correlation values of the individual OTC products in an optimal set are always smaller than that of the correlation value of their aggregated time series. For example, the individual correlation value of the nine OTC products with ED visits ranges from 0.0967 to 0.9414 compared to the 0.9592 of the correlation value of their aggregated time series. Also, note that the Datasets 1 and 2 produced the same optimal sets with the same highest correlation values.

Comparison between three search algorithms: We ran the greedy and knapsack search algorithms against all the Datasets 1–5. Table 1 summarizes the experiment results of all three search algorithms. Again, in all cases, the aggregated time series of the optimal sets and ED visits are peaked at the third week of October (October 15–22), 2009. In all cases, the brute-force and greedy search algorithms found the same optimal OTC thermometer product sets whereas the knapsack consistently found larger sets (i.e., greater number of OTC products). It is interesting to note that the set of products found in smaller more optimal sets found in this search may not necessarily be a subset of the larger less optimal sets.

4.2. Store-product filtering

We applied store-product filtering i.e., store-level filtering combined with product-level filtering. By setting product sales threshold at 10 days and store sales threshold at 60 days, we generated a dataset (Dataset 6) that included 26 OTC (out of 596) thermometer products. By varying the product threshold from 10 to 50 days and store sales threshold from 60 to 100, we generated additional datasets with 25

Table 2
Comparison of search algorithms with different OTC product sets using store-product filtering (CCV⋆: Pearson correlation coefficient value; Size†: number of OTCs in the set).

Datasets	Filter criteria		Filter result	Brute-force search		Greedy search		Knapsack search	
	Per store	Per product	Size†	CCV⋆	Size†	CCV⋆	Size†	CCV⋆	Size†
Dataset 6	≥ 60	≥ 10	26	0.9612	8	0.9612	8	0.9522	12
Dataset 7	≥ 70	≥ 20	25	0.9595	10	0.9595	10	0.9581	11
Dataset 8	≥ 80	≥ 30	22	0.9569	9	0.9569	9	0.9569	9
Dataset 9	≥ 90	≥ 50	21	0.9561	7	0.9561	7	0.9501	7
Dataset 10	≥ 100	≥ 70	17	0.9480	8	0.9480	8	0.9433	7

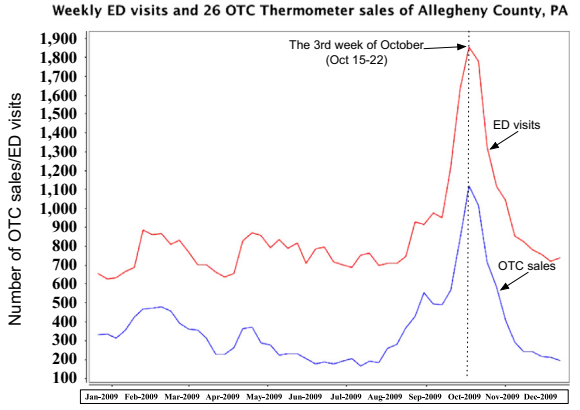

Fig. 6. 26 OTC thermometer sales and ED visits.

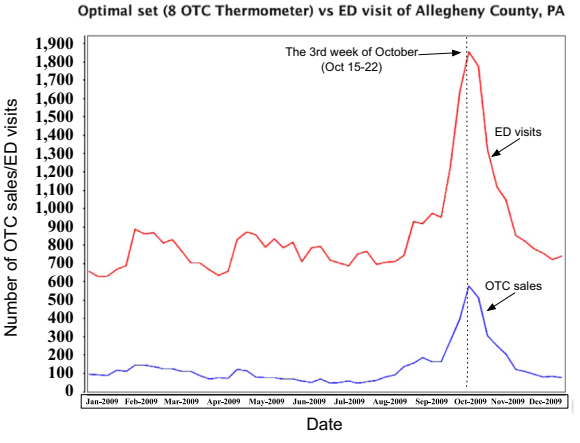

Fig. 7. Eight OTC thermometer sales and ED visits.

OTC thermometer products (Dataset 7 with store threshold 70 days and product threshold 20 days), 22 OTC thermometer products (Dataset 8 with store threshold 80 days and product threshold 30 days), 21 OTC thermometer products (Dataset 9 with store threshold 90 days and product threshold 40 days), and 17 OTC thermometer products (Dataset 10 with store threshold 100 days and product threshold 50 days). As is the case with product-filtering, datasets with a smaller set of products are a subset of datasets with a larger set of products.

Optimal set with store-product filtering: As in the preceding experiment, we ran the brute-force search algorithm against all the Datasets 6–10. Table 2 (Column 4) shows the number of OTC products in the identified *optimal* set and their corresponding aggregated correlation value with ED visits, for each dataset. Again, in all cases, the aggregated correlation value of optimal set is larger than the correlation value 0.9077 of total 596-product set as well as of the correlation value of 0.9126 of the largest 26 products Dataset 6. In all cases, the aggregated time series of the optimal sets and ED visits for Constitutional syndrome are peaked at the third week of October (October 15–22), 2009. Figs. 6 and 7 show this for Dataset 6 and its optimal 8 OTC product set, respectively. Again, the correlation values of the individual OTC products in an optimal set are always smaller than the values of aggregated time series. For example, the individual correlation value of the eight OTC products with ED visits ranges from 0.1316 to 0.9425 compared to 0.9612 for their aggregated time series. In this experiment, Dataset 6 produced the second smaller optimal set with eight products and the highest correlation value (0.9612) as compared to Dataset 9 with a 7-product optimal set and the second lowest correlation value (0.9571).

Table 3
Significant difference test between two individual correlation values.

Product sets	Sample size (weeks)	Correlation values	Significance
28 OTC (product filtering)	52	0.9070	$p \leq 0.035$
9 OTC (optimal set)	52	0.9592	

Comparison among the three search algorithms: We ran the greedy and knapsack search algorithms against all the Datasets 6–10. Table 2 summarizes the experiment results of all three search algorithms. Again, in all cases, the aggregated time series of the optimal sets and ED visits peak at the third week of October (October 15–22), 2009. The brute-force and greedy search algorithms found the same optimal OTC thermometer product sets whereas the knapsack consistently found larger sets (i.e., greater number of OTC products). The smaller optimal sets may not necessarily be a subset of the larger ones.

4.3. Significance tests

We performed significance tests on correlation coefficient values: (1) total OTC product set after filtering and

Table 4
All products without filtering.

Year	2009	2010	2011	2012	2009–2010	2009–2011	2009–2012
CCV	0.9077	−0.4579	0.8008	0.5429	0.8569	0.689	0.6014
Number of products	45	35	27	22	45	48	48

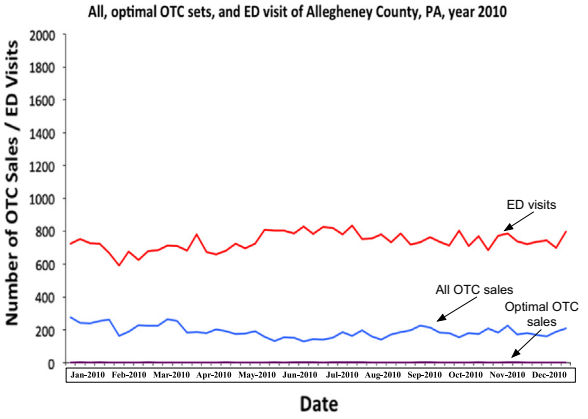

Fig. 8. Optimal OTC set and ED visit time series for 2010.

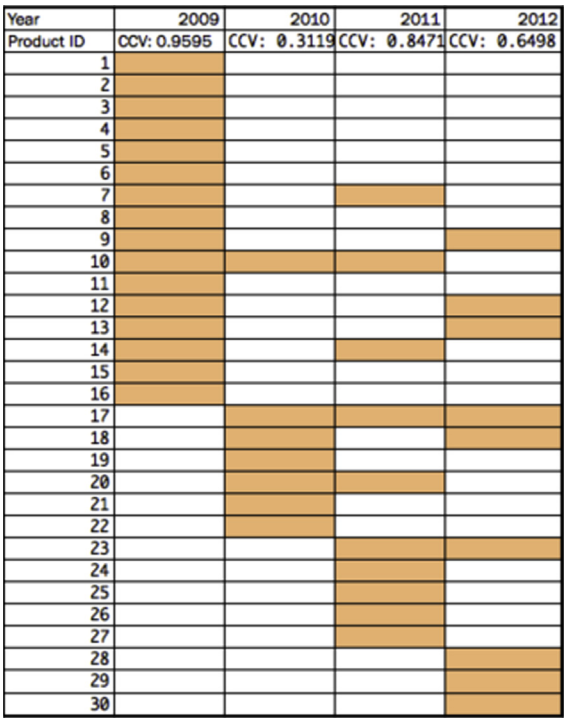

Fig. 9. Best products of individual years. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

(2) the optimal set found by brute-force search. Table 3 shows the significance test results for Dataset 1 with the 28 OTC products vs. an optimal set with nine OTC products. The last column shows that there is a significant difference between their two correlation values ($p <= 0.0352$). This result also holds for other datasets

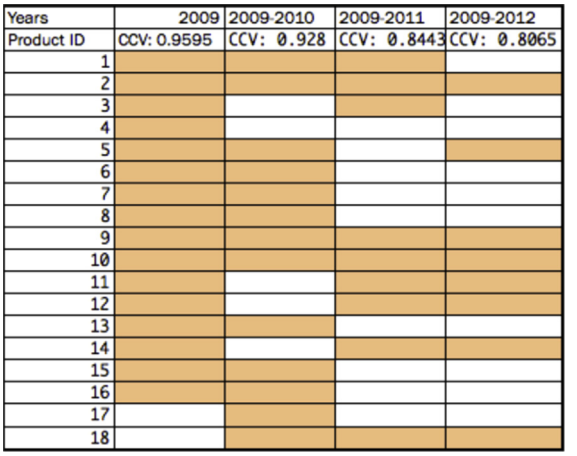

Fig. 10. Best products of multiple years.

in our study. This demonstrates that product sets determined using our method better correlates with ED visits than sets that include all products, and that it is significantly better when used as a health indicator in determining an outbreaking of influenza.

5. Robustness evaluation

Robustness characterizes the effectiveness of an algorithm to consistently identify optimal product sets over different periods of time and different datasets. In order to evaluate how robust and consistent the identified product set was over time, we extended our evaluation to utilize datasets encompassing 4 years (1 January 2009 to 31 December 2012). These extended datasets excluded OTC products with no sales during the entire time period.

We repeated the first experiment with and without product filtering in Section 4 with these extended datasets. We found the best thermometer product set for each year and for multiple years using the greedy search algorithm and checked whether the majority of the best correlated products of the different time periods were the same.

5.1. Product selection without filtering

Table 4 lists the correlation values of all thermometer product sales with ED visits of each individual year 2009, 2010, 2011, and 2012; and multiple years such as 2009–2010, 2009–2011, and 2009–2012. It also shows the number of active products (at least one sales during the entire time period). We note that in 2010, the Pearson correlation value between time series of all OTC thermometers and ED visits is −0.4579, which suggests low correlation and probably opposite tread direction between

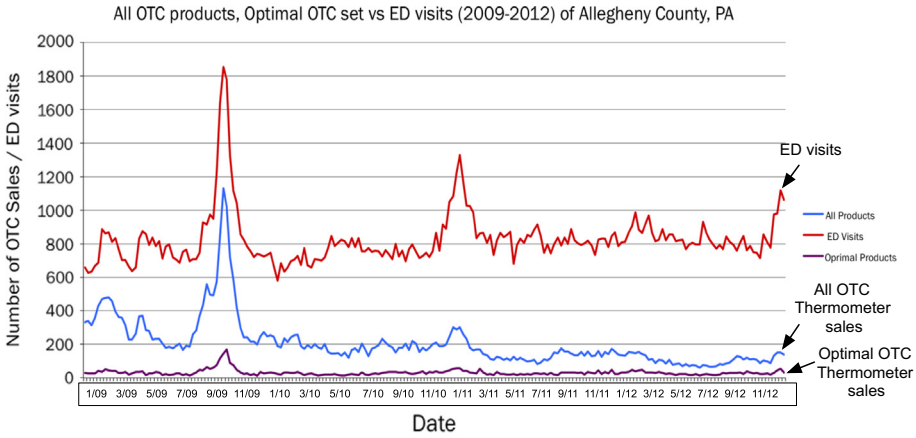

Fig. 11. Optimal OTC set (2009–2012) and ED visit time series.

Year	2009	2010	2011	2012
Product ID	CCV:0.9592	CCV:0.1818	CCV:0.8454	CCV:0.6424
1	■			
4	■			
6	■			
7	■		■	
8	■			
9	■			■
12	■			■
13	■			■
16	■			
17		■	■	
18				■
23			■	■
25			■	
26			■	
27			■	
29				■
30				■

Fig. 12. Best products of individual years with filtering.

Years	2009	2009-2010	2009-2011	2009-2012
Product ID	CCV:0.9592	CCV:0.9279	CCV:0.8443	CCV:0.8056
1	■	■	■	
4	■	■		
6	■	■		
7	■	■		
8	■	■		
9	■	■	■	■
12	■		■	■
13	■	■		
16	■	■		
17		■		
18			■	■

Fig. 13. Best products of multiple years with filtering.

two time series. Part of the reasons may be due to no outbreak peak found in ED visits in 2010. It can be also seen by visual inspection from both Figs. 8 and 11.

The table in Fig. 9 shows the best product sets found from each individual year 2009, 2010, and 2011, 2012. The colored rows in each column represent the optimal set of products for a corresponding year, and the second row in the table has the correlation value for that set to ED visits of the same year. Products sets over different years show few products in common. We found that the 2010 OTC thermometer sales is not very well correlated with ED visit data, the correlation value is 0.3119.

The table in Fig. 10 shows the experiment result that we found as best product sets after extending the datasets by adding one year each time such as 2009, 2009–2010, 2009–2011, and 2009–2012. For example, we had a product set with 16 products that has the highest correlation value of 0.9595 with ED visit in 2009. By adding 2010 data to that, we had a set with 13 products that has the best correlation value of 0.928 with the same filtering criteria. In this new set there are two new products added and five products are eliminated comparing to the best product set we had in 2009. With 2009–2011 dataset, we had a set with nine products that has a correlation value of 0.8443. Comparing to the previous set, four products are contained again that are selected in 2009 best products set, but they were eliminated in 2009–2010 best set. The last column in the table shows that the best product set with eight products obtained from the dataset of four years, 2009–2012, has the correlation value of 0.8065.

Fig. 11 shows the time series of all OTC thermometer sales, optimal OTC thermometer sales and ED visits of 4 years (2009–2012). The correlation value of the optimal set is 0.8056, which is larger than the correlation value 0.6014 of the all 596-product set.

5.2. Product selection with filtering

Fig. 12 lists the best product sets we found for each individual year 2009, 2010, 2011, and 2012, by applying product level filtering with the threshold of 10 days of sales per year. Compared to the previous table in Fig. 9 without filtering, the number of the products in the best set is decreased each year. The correlation values are similar to those without filtering except for runs using 2010 data. In the year of 2010 there is only one product in the best set in this experiment, which further validates our statement that there was less thermometer sales in this year and sales were not well correlated with ED visit data.

Fig. 13 shows the results for best products (with filtering) for individual years and over multiple years, 2009–2010, 2009–2011, and 2009–2012. In general, we

Table 5
Runtime experiment with extended datasets (with filtering).

Datasets		Runtime	
Year	Number of products	Brute-force search (s)	Greedy search (ms)
2009	28	164	7.3
2009–2010	28	289	12.7
2009–2011	28	422	19.3
2009–2012	27	256	30

Table 6
Runtime experiment with extended datasets (without filtering).

Datasets		Runtime	
Year	Number of products	Brute-force search (estimated)	Greedy search (ms)
2009	45	199 days	21.2
2009–2010	48	90 months	18
2009–2011	48	130 months	26
2009–2012	48	180 months	44

saw that successive individual years had different optimal product sets. The addition of additional years of data resulted increasingly different optimal product sets.

From above two experiments, it is clear that the optimal set identified for a specific year might not be sufficient to detect outbreaks during the subsequent years. One reason could be that the Pearson's correlation may not be a good scoring metric since it does not consider the temporal nature of the data and treats it as independent data points. Another could be that the optimal set need to be recomputed and adjusted at different time periods and periodically during a year. In our future work, we plan to investigate these two reasons considering the correlation of the time series values.

6. Scalability evaluation

In order to measure scalability of our system, we set up two experiments to compare execution time of brute-force and greedy search algorithms to find the optimal product set. The time cost of each search algorithm was measured by excluding the querying and filtering process at the distributed search engine since the excluded cost is common across all three search algorithms. In these experiments we used four datasets with time periods of 2009, 2009–2010, 2009–2011 and 2009–2012 after filtering out the products with no sales during the specific time periods. The reported experiment results are average runtime value taken by executing each of the experiments 10 times.

Experiment 1: In this experiment, we repeated the optimality evaluation with product-level filtering as in Section 4. We set the filtering threshold 10 days of sales for 2009 data (i.e., Dataset 1), 20 days for 2009–2010 data, 30 days for 2009–2011 data and 40 days for 2009–2012 data. The column 2 in Table 5 has the number of remaining products in each dataset after applying those filtering criteria. The third column has the runtime result of brute-force and greedy search algorithms to finish all the computations. For Dataset 1 (2009), brute-force took 164 s to identify the optimal set of OTC products while greedy took 7.3 ms to identify the same optimal set of OTC products. For reference, we also measured the runtime cost of the knapsack search to process Dataset 1 and it was 9.7 ms. Evaluating the largest dataset of 2009–2012 year with filtering, brute-force took 256 s, while greedy took an average of 30 ms.

Experiment 2: To further assert greedy's scalability, we repeated the preceding experiment with the same data sets but without the product-level filtering. The results are shown in Table 6. Brute-force search did not finish even for a moderately larger set of products than the 28 products of Dataset 1. For this reason, we estimated and reported the run time of the brute-force search algorithm based on the computation speed of our computer. Specifically, the estimated run time for processing 45 products for 2009 would take at least 199 days and to compute all four years worth of data would take about 180 months. Because the greedy search algorithms took less than 1 s to find the optimal product set with extended datasets greedy search is a good choice from both the functional and the practical point of view.

7. Related work

In the area of processing large volumes of data, variable or feature selection has many benefits such as facilitating data visualization and understanding, reducing the measurement and storage requirements, reducing training and utilization times, etc. The paper [21] introduced a wide range of aspects of such problems: providing a better definition of the objective function, feature construction, feature ranking, multivariate feature selection, efficient search methods, and feature validity assessment methods. However, out of all these aspects the constructing and selecting subsets of variable are the most relevant to our work. In our approach, once the set of OTC products are selected, then we form a time series of this set of OTC product sales that drives the selection of the subset of the OTC products that constitutes the optimal set of health indicators. The selection is based on the Pearson correlation coefficient function that evaluates how the set of OTC product sales correlates to the time series of ED visits. As mentioned above, the Pearson correlation coefficient function considers data as independent data points and hence its results might not be the best possible. As part of our future work, we plan to consider other functions discussed in [21], especially those that consider the temporal nature of the data, i.e., the correlation of the time series values.

In biosurveillance, finding an optimal set of surrogate health indicators to monitor a disease or syndrome is a general problem. Traditionally, the selection of health indicators has been performed manually by public health experts and is limited by the amount of data available. Researchers have studied OTC products to identify various disease outbreaks, however, they have not formally

evaluated whether different sets of OTC products best correlate with outbreaks of certain diseases. One of the methods that has been studied for measuring the relationship of OTC product sales and diseases is to survey of individuals in the population or survey of sick individuals [4]. Another method is measuring the relationship of OTC product sales and hospital visits that have outbreaks at the same time in the same region, and study OTC products that best correspond to the epidemic curve of the outbreak. Traditionally, both of those methods have been performed manually. Our solution adopted the second method, and automated the process of identifying optimal OTC products which was the goal of our proposed OTC Analytics Appliance.

Google Flu Trend is a surveillance system that has been created to monitor indirect signals of influenza activity using real-time influenza related web search queries. Such a system has an advantage over systems that rely on weekly sentinel physician reporting. In Google Flu Trend, a list of the highest scoring 45 search queries for influenza was sorted by mean Z-transformed correlation. Using those ILI related queries as the explanatory variable, a linear model was fit to weekly ILI percentages between 2003 and 2007. The final model was tested on 2007–2008 data resulting in a mean correlation value of 0.97 with CDC-observed ILI percentage [22]. A drawback of Google Flu Trend is that it depends on user search behavior and search activity may disproportionally increase during periods of more severe outbreaks i.e., people who are not ill are performing searches. Such behavior may have occurred in the 2012–2013 flu season when Google estimates were twice that of the CDC observed activity [23].

8. Conclusions and future work

A Syndromic Surveillance system is a type of outbreak detection system that monitors the health status of a community, and identifies disease outbreaks or health events using various individual and population health related temporal data. The selection of good health care indicators is a major factor in the effectiveness of a Syndromic Surveillance system when detecting an outbreak. Traditionally, this selection has been done manually and was limited by the speed at which queries could be executed and human ability to review the results of these queries. The goal of this paper was to automate and improve the accuracy of the selection process of the health indicators, taking advantage of distributed computing methods and search algorithms.

We developed an *OTC Analytics Appliance* that measures the relationship between OTC product sales and hospital visits during outbreak periods, and identifies the OTC products that best correspond to the epidemic curve of (an) outbreak(s). Our OTC Analytics Appliance efficiently generates time series of unit sales of OTC products by utilizing a distributed search engine which leverages in-memory search on large datasets across multiple machines (or nodes). Its Optimal OTC Identifier implements an efficient greedy search algorithm that utilizes the Pearson correlation coefficient function to select a minimum of OTC products whose sales over time optimally, or close to optimally, correlates with hospital visits. In order to allow maximum flexibility and experimentation, the Optimal OTC Identifier module is designed with "plug-and-play" search and correlation functionalities. We developed the OTC Analytics Appliance as a user-friendly and easily deployable system that uses the MapReduce computational paradigm.

We used the OTC Analytics Appliance to compare the greedy search algorithm to the brute-force and knapsack search algorithms in identifying the set of thermometer products (such as strip or digital, oral or forehead for babies or adult thermometers) which optimally correlate over time with Emergency Department (ED) visits for symptoms (such as fever) consistent with Constitutional syndrome. Brute-force search served as our gold standard approach on identifying optimal OTC product sets that correlate with ED visits. Since brute-force is computationally intensive, its scalability is limited. The need for a search algorithm which is computationally less expensive and can overcome the scalability challenge led us to the use of a greedy search algorithm. Our greedy search implementation identified the same optimal sets as the brute-force algorithm identified in our experiment. It is efficient in terms of time complexity and scalability to large datasets.

Our implemented basic version of the Knapsack search turned out to be as efficient as the greedy search in terms of run time (< 1 s) and returned good results. However, these results were consistently sub-optimal and worse than those of the greedy search. Although the general Knapsack search guarantees to find the optimal solution (by adding an item into the knapsack either increases or decreases value not depending on the subset in the knapsack), we decided against implementing it since its time complexity is similar to that of the brute-force search and hence, it would not scale to big data sets.

The robustness evaluations reveal that our optimal or suboptimal product set may be different year to year, which suggests the need to periodically update the optimal product set. Adjacent year comparisons of the most optimal sets of products show multiple differences in the sets. If one were to use an optimal set found using our current greedy search implementation one might miss an outbreak in subsequent years. To address the problem, we plan to continuously update the optimal product set on monthly or quarterly basis. We also plan on conducting additional evaluations using different correlation functions that consider the correlations of the time series values as a part of the Optimal OTC Identifier.

Our work was partially motivated by the rapid growth of the available health related data. We are certain that this growth in health related data will accelerate. For this reason, we continue our search for more efficient search algorithms – a search algorithm that is optimal, yet efficient and scalable to big datasets. We plan to experiment with larger regions, such as multiple states, and with multiple categories of OTC products. In addition, we plan to apply our framework to identify optimal product set to build time series modeling for prediction of ED visits.

Acknowledgment

This research was funded by grants from the PA Department of Health (SAP #40000012020) and from the National Retail Data Monitor. It was also partially supported by the NSF Award IIS-105030. We thank Mehmud Abliz for his help in designing the greedy search algorithm in our framework. Also, we thank Leon Lai for maintaining and improving the UPC update procedures where our quarterly renewed OTC product list comes from.

References

[1] Syndromic Surveillance (SS). Available from: ⟨http://www.cdc.gov/ehrmeaningfuluse/syndromic.html⟩ (accessed 14.08.12).
[2] F. Tsui, J.U. Espino, Technical description of RODS: a real-time public health surveillance system, J. Am. Med. Inform. Assoc. 10 (September (5)) (2003) 399–408.
[3] W.R. Hogan, Early detection of pediatric respiratory and diarrheal outbreaks from retail sales of electrolyte products, J. Am. Med. Inform. Assoc. 10 (6) (2003).
[4] R. Li, G.L. Wallstrom, W.R. Hogan, A multivariate procedure for identifying correlations between diagnoses and over-the-counter products from historical datasets, in: AMIA 2005 Symposium Proceedings.
[5] BioSense Fact Sheet. Available from: ⟨http://www.cdc.gov/biosense⟩ (accessed 22.09.12).
[6] L.R. Lazarus, K.P. Kleinman, I. Dashevsky, et al., Using automated medical records for rapid identification of illness syndromes (Syndromic Surveillance): the example of lower respiratory infection, BMC Public Health 1 (2001).
[7] Zhen Liu, Panos K. Chrysanthis, Fu-Chiang Tsui, A comparison of two view materialization approaches for disease surveillance system, in: Proceedings of the ACM Symposium of Applied Computing (SAC'04), January 2004, pp. 754–755.
[8] F.C. Tsui, J. Espino, Key design elements of a data utility for national biosurveillance: event-driven architecture, caching, and web service model, in: Annual Symposium Proceedings/AMIA Symposium, 2005, pp. 739–743.
[9] Mohamed A. Sharaf, Panos K. Chrysanthis, Alexandros Labrinidis, Kirk Pruhs, Algorithms and metrics for processing multiple heterogeneous continuous queries, ACM Trans. Database Syst. (TODS) 33 (March (2)) (2008) 51–544.
[10] Shenoda Guirguis, Mohamed A. Sharaf, Panos K. Chrysanthis, Alexandros Labrinidis, Optimized processing of multiple aggregate continuous queries, in: Proceedings of the Conference on Information and Knowledge Management (CIKM'11), Glasgow, UK, October 2011, pp. 1515–1524.
[11] Shenoda Guirguis, Panos K. Chrysanthis, Alexandros Labrinidis, Mohamed A. Sharaf, Three-level processing of multiple aggregate continuous queries, in: Proceedings of the 28th IEEE International Conference on Data Engineering (ICDE'12), Washington DC, April 2012, pp. 929–940.
[12] Why ElasticSearch?. Available from: ⟨http://www.elasticsearch.com⟩ (accessed 06.05.13).
[13] Apache Hadoop. Available from: ⟨http://hadoop.apache.org⟩ (accessed 08.05.13).
[14] ElasticSearch A Distributed RESTful Search Engine. Available from: ⟨https://github.com/elasticsearch/elasticsearch⟩ (accessed 06.05.13).
[15] B. Rosner, Fundamentals of Biostatistics, 7th ed., 2010.
[16] T.H. Cormen, Introduction to Algorithms, 3rd ed., 2009.
[17] NRDM (National Retail Data Monitor) A Public Health Surveillance Tool. RODS Laboratory, ⟨http://rods.health.pitt.edu/NRDM.htm⟩.
[18] W. Chapman, J. Dowling, M. Wagner, Classification of emergency department chief complaints Into 7 syndromes: a retrospective analysis of 527,228 Patients. Ann. Emerg. Med. November 2005, 46.
[19] J. Que, F. Tsui, Spatial and temporal algorithm evaluation for detecting over-the-counter thermometer sale increases during 2009 H1N1 pandemic, J. Public Health Inf. 4 (1) (2012).
[20] R. Villamarin, G. Cooper, F. Tsui, et al., Estimating the incidence of influenza cases that present to emergency departments, Emerg. Health Threats J. 4 (2011) s57.
[21] Guyon, Isabelle, Andr Elissee, An introduction to variable and feature selection, J. Mach. Learn. Res. 3 (2003) 1157–1182.
[22] J. Ginsberg, M.H. Mohebbi, R.S. Patel, Detecting influenza epidemics using search engine query data, Nature 457.7232 (2008) 1012–1014.
[23] D. Butler, When Google got flu wrong, Nature 494 (February) (2013).