# More with Less: Lowering User Burden in Mobile Crowdsourcing through Compressive Sensing

**Liwen Xu[†], Xiaohong Hao[†], Nicholas D. Lane[‡], Xin Liu[*], Thomas Moscibroda[⋈]**

[†]Tsinghua University, [‡]Bell Labs, [*]UC Davis, [⋈]Microsoft Research

## ABSTRACT

Mobile crowdsourcing is a powerful tool for collecting data of various types. The primary bottleneck in such systems is the high burden placed on the user who must manually collect sensor data or respond in-situ to simple queries (e.g., experience sampling studies). In this work, we present Compressive CrowdSensing (CCS) – a framework that enables compressive sensing techniques to be applied to mobile crowdsourcing scenarios. CCS enables each user to provide significantly reduced amounts of manually collected data, while still maintaining acceptable levels of overall accuracy for the target crowd-based system. Naïve applications of compressive sensing do not work well for common types of crowdsourcing data (e.g., user survey responses) because the necessary correlations that are exploited by a sparsifying base are hidden and non-trivial to identify. CCS comprises a series of novel techniques that enable such challenges to be overcome. We evaluate CCS with four representative large-scale datasets and find that it is able to outperform standard uses of compressive sensing, as well as conventional approaches to lowering the quantity of user data needed by crowd systems.

## INTRODUCTION

Mobile crowdsourcing is a promising way to collect large-scale data about ourselves and the urban areas we live in [29, 19, 11, 10, 13, 14, 21, 24]. Target data for such systems is highly variable and can be broken into two primary types: (1) sensor data from the environment (e.g., noise [30], air pollution [43]) and, (2) user-provided information (such as a survey response) which is necessary when sensors struggle to capture the target phenomenon. Examples of this second type includes information related to people – for instance, their wellbeing or mental state [22, 6, 25] – and complex questions about the environment: Is there graffiti along this street? Do you see any empty parking spaces? Have you found rats in your building?

One of the key bottlenecks in crowd systems is the high burden on users, if they decide to participate. Users are required to perform time-consuming data collection steps regularly (even multiple times per day [25, 26]) while they participate. This can act as a barrier which prevents larger numbers of users to opt-in and join the system. It also causes users to stop participating over time [15]. As a result, participation is often closely tied to financial incentives [40, 32]. Thus reducing the need for user contributions can also decrease the cost of running the system. For all of these reasons, approaches that allow mobile crowdsourcing systems to operate with fewer user contributions are acutely needed.

In this paper we examine the suitability for Compressive Sensing [12, 17] (CS) to help address this critical issue. The application of CS in other domains (e.g. vision) has led to significant advances by lowering the amount of data that must be sampled to capture (through reconstruction) complex phenomena. However, to the best of our knowledge, this is the first time CS has been applied to mobile crowdsourcing – especially, when capturing responses to questions directed at users (e.g., experience sampling applications) are considered.

Already known approaches for lowering the required amount of manually collected data include general-purpose statistical methods, such as sub-sampling and interpolation [23]; along with domain-specific techniques from population surveying [31] and geospatial analysis [16] (e.g., Kriging interpolation [35], James-Stein estimators [34]) that leverage temporal patterns, user demographics and geographic characteristics. However, crucially all of these methods presuppose, and then leverage, certain relationships within the collected data. In contrast, CS has the ability to utilize inherent structure that may not be obvious and may not correspond to more "natural" ways of considering the data (e.g., spatial relationships). Consequently, it is possible for CS to identify and exploit a superset of potentially stronger, and otherwise ignored, structure towards reconstructing the underlying phenomena.

The core technical challenge of applying CS to manually collected crowd data exists because it was never designed for data of this type. Conventional CS is designed to deal with 1-D vectors, and other data types that can be easily vectorized. But crowdsourcing datasets, which are investigated in this paper, usually have multiple columns each of which can be considered a separate dimension. The multi-dimensional nature of such data allows them to be vectorized in many ways, this makes it non-trivial to apply CS directly. This forces novel processing steps to be introduced to the conventional CS pipeline that: mine the inherent data correlation in the original data which may be arbitrary and complex; preserve the important ones in the structural conversion; provide sufficient data for base training; and handle missing data.

Toward addressing these challenges, we propose *Compressive CrowdSensing* (CCS) – a framework designed to enable mobile crowdsourced data collection with fewer user contributions by leveraging compressive sensing. Central to CCS is our


*UbiComp '15*, September 7–11, 2015, Osaka, Japan.

DOI: http://dx.doi.org/10.1145/2750858.2807523

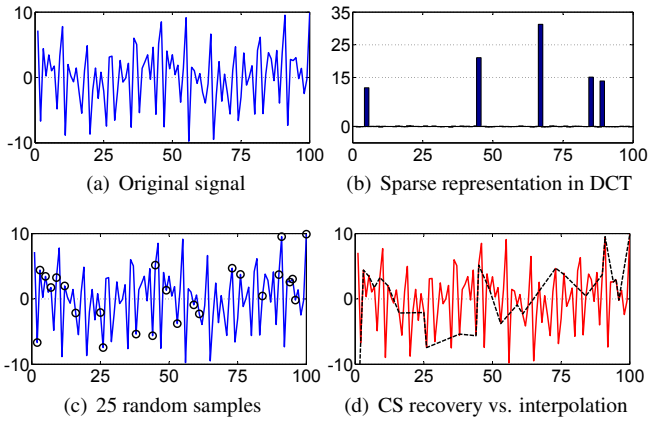



**Figure 1:** Illustrative Compressive Sensing Scenario

*Data Structure Conversion* technique that is able to search a variety of representations of the data in an effort to find one that is then suitable for learning a custom sparsifying base. Our technique is able to mine typical forms of latent inner structure needed by CS – if they exist – for example, temporal and spatial relationships. But more importantly, it is also able to find bases that unexpectedly provide the necessary correlation. For example, a specific CS base for estimating the employee number of small businesses in a city based on crowdsourced user data by exploiting correlations between location and industry type. CCS is then able to reconstruct the target data completely via CS-based recovery algorithms. We are the first to propose a novel and practical mechanism that applies CS theory to reduce the quantity of user contributions necessary for mobile crowdsourcing – especially in the context of multi-dimensional survey data.

The contribution of this work includes:

- We demonstrate the feasibility of applying CS to new types of data that are often collected via users participating in a mobile crowdsourcing system. Prior to our study, it was not known that CS would be applicable for important data domains like large-scale question-based user surveys regarding urban situations.
- We propose CCS, a CS-based framework for efficient data collection via crowdsourcing. To allow this framework to operate we develop novel CS-related techniques – in particular, these enable the use of data that do not have obvious representations with a sparse structure, a fundamental requirement for CS.
- We evaluate CCS by applying it to multiple real-world datasets representative of the data collected within cities by crowd systems. Our findings show CCS is able to maintain far better ratios of user contributed data to overall system accuracy than state-of-the-art baselines.

## COMPRESSIVE SENSING PRIMER

Compressive Sensing (CS), which is an efficient technique of sampling data with an *underlying sparse structure*, is a breakthrough that first appeared in the signal processing field and quickly spread to many application domains (e.g., medical imaging [28]) and computer science communities (e.g., sensor networks [27, 42]). In particular, for data that can be sparsely represented, it shows the possibility to sample at a rate much lower than the Nyquist sampling rate, and to then still accurately reconstruct signals via a linear projection in a specific subspace. A concrete example of this are city-wide traffic speeds that have been demonstrated to have sparse structure [44]; as a result, a dense grid of traffic speeds can be reconstructed from a relatively small vector that roughly approximates the traffic speeds taken at key road network intersections. Because of the ubiquity of inner structured data (e.g. noise [30], soil moisture [38] and traffic [44]), CS is a promising technique to reduce the sampling rate in a variety of data collection scenarios.

**Sparse Structure.** The sparse structure of a dataset is the key to applying CS theory. Figure 1 illustrates this concept with a simple example. In Figure 1(a) an artificial signal is shown that was generated with sparse structure, specifically the signal is composed of a number of correlated signals of varying offsets. This is similar to phenomena like temperature and rainfall that are influenced by the past weather at different timescales (i.e., the weather of the prior hour, prior day, prior month all influence the current temperature and rainfall). Figure 1(b) shows the signal projected into a discrete cosine transform (DCT) base [8] – a linear vector commonly used in CS to represent signals. We can see the signal is sparsely represented by this base because only a few DCT coefficients are necessary to capture the signal. Importantly, because of this fact the seemingly complex signal seen in Figure 1(a) can be reconstructed faithfully (Figure 1(d)) with only 25 values.

To more formally understand sparse structure, let $\mathbf{y} \in \mathbb{R}^n$ be a signal of interest. Let $\mathbf{y}$ be decomposed under a certain base $\mathbf{\Psi}$, i.e. $\mathbf{y} = \mathbf{\Psi x}$ where $\mathbf{x}$ is the coefficient vector. $\mathbf{x}$ is called $k$-*sparse* if it has only $k$ non-zero entries, and $\mathbf{y}$ is called *compressible* when $k$ is small. In other words, in some dimension (transformed by $\mathbf{\Psi}$ etc.), $\mathbf{y}$ can be represented as a sparse signal with a few non-zero entries. As an example, the signal in Figure 1 is $5$-*sparse*. Similarly, temperature and depth data in oceans have been shown to be $40$-*sparse* [27] (using a Fourier base and resulting in a 98% accuracy level).

**Random Sampling.** In practice, *random sampling* is one of the most popular CS sampling methods because it is easy to implement. Figure 1(c) shows the artificial signal being randomly sampled through time. Under this method a randomly selected $m$ samples out of the $n$ ground-truth entries is used to capture a signal $\mathbf{y}$, where $m \ll n$. Typically, if $m \geq k \log n$, the signal reconstruction is guaranteed by CS theory for a $k$-sparse signal vector $\mathbf{y}$. In the context of the signal shown in Figure 1(a), this means with the 25 random readings CS guarantees a signal normally described with 100 readings can be reconstructed.

Mathematically, this sampling method is equivalent to randomly picking several rows from $\mathbf{\Psi x}$. CS theory shows the sampled signal ($z$) is:

$$\mathbf{z} = \mathbf{\Phi y} = \mathbf{\Phi \Psi x} \tag{1}$$

with $\mathbf{\Phi}$ a linear encoder and so is a $m \times n$ random partial identity matrix. As the sampling is under our control, the values of $\mathbf{\Phi}$ are always known.

**Data Reconstruction.** Figure 1(d) compares the reconstruction of the initial artificial signal (Figure 1(a)) using CS (seen in red) and standard linear interpolation (seen in black). Even though both approaches use 25 samples each, clearly reconstruction using CS is much more accurate.

More formally, CS reconstruction begins with a sampled signal $\mathbf{z}$ and a target original signal $\mathbf{y}$ to reproduce. Because $\mathbf{y} = \mathbf{\Psi x}$, finding $\mathbf{x}$ – which is a vector with no-more-than $k$ non-zero entries – is equivalent to finding $\mathbf{y}$. Typically, $l_1$-norm minimization is used to find $x$ due to its polynomial complexity. This is equivalent to finding an $x$ with the smallest $\ell_1$-norm; i.e.,

$$\underset{\mathbf{x}\in\mathbb{R}^n}{\operatorname{argmin}} \|\hat{\mathbf{x}}\|_{l_1}, \text{subject to } \mathbf{\Phi\Psi}\hat{\mathbf{x}} = \mathbf{z}. \quad (2)$$

Many naturally occurring phenomena are suitable for CS even thought they do not perfectly fit the requirements – they are termed *approximately compressible*. In such cases, the phenomena may have a few large non-zero entries but also additional small – but not completely zero – entries. In practical settings, these issues are ignored and the reconstructed signal is an approximation of the original.

**Base Learning.** The base $\mathbf{\Psi}$ plays a critical role in transforming the signal of interest $\mathbf{y}$ to a sparse signal $\mathbf{x}$ where $\mathbf{y} = \mathbf{\Psi x}$ (a process called sparsifying). In some cases, a standard base, such as a Fourier base, can be used. In others (including this work) a good base needs to be trained using historical data.

Consider historical data $\mathbf{Y} = \{\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_N}\}$ ($\mathbf{y_i}$ is column vector representing a signal of interest, just like $\mathbf{y}$). Base learning seeks to find a base $\mathbf{\Psi}$ that minimizes the total error between the given historical data and its sparse representation:

$$\underset{\mathbf{\Psi},\mathbf{x_i}}{\operatorname{argmin}} \sum \|\mathbf{y_i} - \mathbf{\Psi x_i}\|_{l_2}. \quad (3)$$

As an example, in Figure 1 as already mentioned the DCT base was employed. However, if another base was used significantly worse results may have resulted. For instance, while under DCT the signal is 5-*sparse* but under a wavelet base it becomes 10-*sparse*, and would require twice as many samples to maintain the same reconstruction accuracy as the DCT base.

**Stages of CS.** We describe the steps taken when applying CS within an illustrative scenario of performing a survey that collects 1-D data. Under this scenario, we survey the severity of a rat infestation and count rat sightings in different areas of a city within a particular period of time.

1. We use a set of historical data $\{\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_N}\}$ which contains past rat sighting counts of the targeting areas to train a base $\mathbf{\Psi}$ according to Eq. (3). $\mathbf{\Psi}$ captures the sparse structure within the rat infestation phenomenon.
2. Instead of conducting a complete survey of all city locations, we select a small random sample of areas in which a survey of rat sighting is carried out. Thus we obtain a small set of samples $\mathbf{z}$ of the data of interest.
3. Finally we use $\mathbf{z}$ to recover the data of interest $\mathbf{y}$ based on Eq. (2) and using the trained base $\mathbf{\Psi}$.

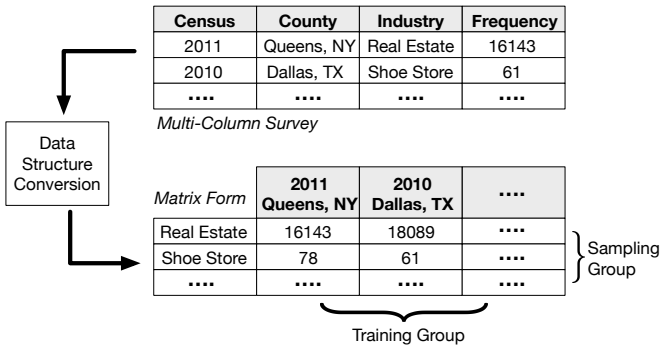



**Figure 2:** Data Structure Conversion

## COMPRESSIVE CROWDSENSING

Mobile crowdsensing leverages participating users to collect various forms of data from the real world using a mobile device. At times, users collect data like noise pollution or air quality through sensors embedded in the device. At other times, this data is manually entered by the user such as, ranking a restaurant they visited, personal demographic data, or reporting a sighting of a rat infestation. Typically, the data collected across many participating users is then aggregated to give a global view of a phenomenon – such as, a "heatmap" of the city-wide spatial intensity of noise or rat sightings.

The objective of the CCS framework is to enable mobile crowdsensing systems to gather information on a variety of large-scale phenomena but do so with fewer data samples contributed by users. CCS is also designed to be easily adapted so that it can work as part of a wide range of crowd systems. In the remainder of this section, and the one that follows, we describe the design and operation of the CCS framework.

We begin this description by first introducing a running example representative of the types of crowd applications we target. Consider a mobile crowd system aimed at surveying how many businesses within different industries fit into the business statistics category of being "non-employers" – in other words, they do not have any paid staff besides the business owners. Typically this data is collected only annually by a large manual survey [3]. If this could be performed by a crowd system it could be done potentially much more frequently and at a lower cost. To perform this task a crowd system would leverage participating users to complete a brief set of questions. As an example, the survey that performs this task aggregates survey responses into a table as shown in Figure 2.

In any data collection performed with CCS a particular subset of survey columns is focused upon for reconstruction using CS. This is called *target data*, which in this case is the number of businesses. The final goal is to accurately recover the target data while collecting fewer total data samples from users participating in the system.

### Challenges to Applying CS

A core motivation for our work is that, in many crowdsourcing applications, the collected data has underlying structure and relationships. For example, it is intuitive that nonemployment numbers are related to business types and years. Therefore, a

natural question is whether CS techniques can reduce the data sampling rate while maintaining high reconstruction accuracy.

The key challenge is that it is not clear how to apply CS to this type of crowd collected data. CS is originally designed to deal with 1-D vectors, and data types that can be easily vectorized. For example, images and physical signals like temperature in a 2-D space (such as temperature represented spatially). However, crowd survey responses, which is studied in this paper, usually has multiple columns, and each column is effectively a dimension. A large variety of potential vectorizations exist due to the multi-column nature of survey data. This makes it non-trivial to apply CS directly, and thus the critical step is to generate a suitable and efficient data structure that CS can use.

Such processing is challenging because we have to: extract the inherent correlation in the original data which may be arbitrary and complex; select and preserve the key correlations during structural conversion; and finally provide sufficient data for base training while also handling missing data issues. Addressing these challenges, and thereby enabling CS to be applied to multi-dimensional crowd-collected survey data are core contributions of CCS. To the best of our knowledge, CCS is the first mechanism for applying CS theory to reduce the sampling burden to users for this type of crowd survey data.

### CCS Framework

Instead of directly reducing the signal dimension, we start by observing the correlations between the columns in the survey as seen in historical data previously collected. The correlation between the columns and the target data is the key for the data structure conversion needed to apply CS. In particular, CCS begins by converting multi-column historical survey data, e.g., as shown in Figure 2. In the matrix form, we have a *sampling dimension* and a *training dimension*. The sampling dimension is the vectorized signal dimension of the original signal ($\mathbf{y}$), and the training dimension is the horizontal dimension of $\mathbf{Y}$.

The intuition for this data structure conversion is as follows. The matrix form of the historical data is the format needed in the base training step of CS (i.e., $\mathbf{Y}$), as discussed earlier. Therefore, this matrix form allows us to train a suitable base (detailed later), critical for later reconstruction of the target data (i.e., $\mathbf{y}$). This data structure instructs us how to sample future data as the sampling dimension is the vector signal dimension that we desire. For example, next year, when we want to collect the non-employer data and determine the number of non-employer businesses in every industry field. Instead of conducting a full survey of all industry fields, we only need to *randomly survey* of a small fraction of all industries from users, with the remaining industries reconstructed via CCS.

Ideally the sampling dimension should be highly correlated to the target data. In contrast, the training dimension should be uncorrelated to the target data. This motivates the design of the dimension selection algorithm, presented in detail in the next section. In the non-employer example shown in Figure 2, the sampling dimension is the business type (e.g., real estate); and training dimension is the combination of year and county.

After selecting the dimensions, we next re-organize the multi-column data. This is illustrated in Figure 2 that shows this transformation of multi-column survey data into matrix form. The matrix form, generated from the historical data, is then used to train a CS base that extracts the correlation we need (details discussed in the next section).

Note, when data types from columns are categorical we quantize such values using the classical vector quantization approach [20]. This generates a code-book that maps categorical values into real values after which other operations may occur.

Overall, CCS is comprised of the following high-level phases.

**Stage 1. Data Structure Conversion.** This is the critical step of CCS, which enables us to apply CS techniques to multi-dimensional crowd responses. Specifically, we start with a set of historical survey data as shown in Figure 2 and convert it into a matrix form. We note that a good conversion is critical to the success of CCS. As shown in the evaluation section, reconstruction accuracy is highly sensitive to the data conversion. This step is conducted off-line, prior to data collection from crowd users.

**Stage 2. Base Training.** Given the converted matrix form, as shown in Figure 2, we use an existing base learning algorithm to train a set of potential bases off-line. Note, because of the complex structure typically existing in survey data the most common used bases, such as a Fourier base, do not work well. Therefore, this step is also critical.

**Stage 3. Sampling.** The selection of the sampling dimension has a secondary role in that it guides how sampling is conducted within participating users. For instance, within the example of the non-employer survey, the sampling dimension determines that random sampling is applied on a industry type basis during reconstruction. Sampling is always an on-line procedure.

**Stage 4. Reconstruction.** After collecting the sample dataset, we can use a CS reconstruction algorithm to recover the signal of interests; for instance, the number of non-employer businesses in every industry field in our running example. This step is done once data has been collected from users.

Like other applications of CS, manual framework tuning by operators of the crowd system is necessary. This is especially true for a number of key parameters used within our framework. Our approach to setting these values is purely experimental, and performed by using historical data. Later in our evaluation, we demonstrate this method is able to yield more than adequate results compared to state-of-the-art baselines.

## ALGORITHM DESIGN

We now detail the algorithmic foundations of CCS.

### Data Structure Conversion

This first stage – the *Data Structure Conversion* algorithm – is the most critical (and novel) process in the design of CCS. It is responsible for data selection and representation within CCS and applied against raw data collected by crowds.

As discussed earlier, data correlation plays a key role in the data structure conversion. In some cases, there is obvious spatial and temporal correlation that CS can exploit. For example,

**Algorithm 1** Data Structure Conversion

**INPUT :** User contributed data, presented as columns and rows – with one column selected as the *target*
**OUTPUT:** Matrix of data with columns assigned into *sampling* and *training* columns
**STEP 1:** Calculate correlation matrix $C = \{c(i,j)\}$, where $c(i,j)$ is the correlation of column $i$ and column $j$.
**STEP 2:** Select column $i$ that is most correlated with column $target$, i.e. $|c(i, target)|$ is maximized. If all the dimension selected, go to step 5.
**STEP 3:**
- If $|c(i, target)| \geq c_{tar}$, where $c_{tar}$ is a threshold, put column $i$ into the sampling group.
- Else put column $i$ into training group. Go to step 2.
**STEP 4:** Find column $j$ such that $|c(i,j)| \geq c_{can}$, where $c_{can}$ is another threshold. Put all such column $j$ into the sampling group. Go to step 2.
**STEP 5:** Aggregate data into corresponding matrix cells spanned by columns within sampling and training groups.

one can expect rat sighting reports in a city to have spatial correlations (an example closely examined later in the evaluation). In other cases, data correlation can be much more complex and may not be obvious at the first sight. For example, in the non-employer data, the correlations between the field of industry and the total of non-employer category businesses is much more obscure. Therefore, CCS must carefully explore all potential relationships in order to successfully apply CS.

While exploring the inner structure of crowd gathered survey data, columns should be classified into two groups – *sampling* and *training* – based on their correlation with the *target* data (a single column to be reconstructed from samples). The sampling group columns are expected to present the inherent correlation with the target column, thus naturally, the sampling group should capture the hidden correlation in a clear mathematical form. On the other hand, the training group should be only loosely correlated to the target column because the correlation between the target column and the ones in training group is ignored during base learning. Each of these intuitions underpin the design of this stage.

Algorithm 1 formally details the Data Structure Conversion process, which we now describe. **STEP 1** computes the correlation between each candidate column and the target column. Different correlation methods, such as linear correlation and mutual information can be applied depending on the actual data type in the columns. We apply linear correlation during experiments reported in the evaluation due to its simplicity and practicality. In **STEP 2, 3**, columns with higher correlation to the sampling group and columns with lower correlation are assigned to the training group. In particular, $c_{tar}$ is a correlation threshold, above which column $i$ and column $target$ are considered to be sufficiently correlated for column $i$ to be added to the sampling group. The thresholds $c_{tar}$ and $c_{can}$ are selected based on their performance against historical data.

**STEP 4** is designed to address a practical issue – reducing the number of missing values in the final data matrix produced by the algorithm. This step ensures the columns with high correlation to column $i$ are moved to sampling dimension as well, so that the columns in the training group are relatively independent of the columns in sampling group. In some cases, when the number of missing data in this way cannot be lowered in this way, this step seeks to concentrate empty cells (i.e., missing values) in the same range of rows and columns. This allows this region of the matrix to be then removed without also removing too many cells containing actual data. In addition, the values of the thresholds are also adjusted (manually, if found to be necessary) to ensure sufficient amounts of data are in both the sampling and training dimensions so that a good base can be learned.

**STEP 5** is the *data mixture* procedure that applies to the groups of columns in the sampling and training groups. Consider a group of columns $\mathbf{C} = \{C_1, C_2, \cdots, C_n\}$. Each column $C_i$ is consisted of $|C_i|$ distinct elements. The mixed dimension $\mathbf{D} = \{C_1 \times C_2 \times \cdots \times C_n\}$, where $\times$ is the Cartesian product. For example in the non-employer survey case, the field of industry is used to form the sampling groups because of its high correlation with the target while the other two columns are used to form training dimension. Later during the evaluation, assuming actual non-employer survey data, the individual data contribution of users is aggregated into a matrix consisting of $3140$ (counties) $\times$ $5$ (years) resulting in $15700$ column vectors of the length of $475$ (industries).

Finally, although all columns are flagged in the initial converted dataset as either "sampling" or "training", the algorithm does not have to use the full combination of these columns to mix the dimensions. In other words, at times only a subset of columns will be used to mix the sampling (training) dimension. For example, suppose the number of surveys in 2011 is very small, then potentially 2011 can ignored.

#### Base Training

Given the matrix form obtained from the historical data, one can find a proper base that sparsifies the target data. Base training is a stage that mathematically uncovers the inherent correlation in the data structure. In other words, it sparsifies the signals of interests on a best effort basis. Not surprisingly, the standard bases – such as the fourier or DCT base (as illustrated in Figure 3) – that are widely utilized in the applications dealing with natural signals (such as temperature, humidity) are poor choices for the statistical data of surveys that describe activities of society and individuals. Thus CCS conducts *a base learning process* to identify a suitable base.

**K-SVD algorithm.** To perform base training CCS adopts a classic algorithm commonly used for this task – K-SVD [7]. In theory, one can find an optimal base by solving the optimization problem formulated in Eq. (3) – i.e., under which the historical data have the best sparse representations. However, this problem is computationally hard due to its non-convexity.

In practice, K-SVD operates as follows. First, provided a set of historical data $\mathbf{Y} = \{\mathbf{y_1}, \mathbf{y_2}, \cdots, \mathbf{y_N}\}$ ($\mathbf{y_i}$ is column vector representing a contributed user data sample), K-SVD finds a base $\mathbf{\Psi}$ that:

1. Represents each $\mathbf{y_i} = \mathbf{\Psi x_i}$, where the sparsity of $\mathbf{x_i}$ is constrained to a given sparsity level $k$ (i.e., no more than $k$ non-zero entries).
2. Minimizes the total error between the given historical data and its sparse representation:

$$\underset{\mathbf{\Psi}, \mathbf{x_i}}{\operatorname{argmin}} \sum \|\mathbf{y_i} - \mathbf{\Psi x_i}\|_{l_2}, \text{s.t.} \|\mathbf{x_i}\|_{l_0} \leq k.$$

Note that the difference with the above and Eq. (3), is that $\mathbf{\Psi}, \mathbf{x_i}$ are optimized jointly. In contrast here, in K-SVD, they are optimized iteratively and so more computationally feasible. The reader is referred to [7] for more details on K-SVD.

Empirically, the size of historical data needed by the base training is relatively large. It usually requires the training dimension to be multiple times of the size of the sampling dimension, which is a constraint that we need to consider in Algorithm 1 by choosing the appropriate parameters.

**Multiple Base Training.** Typically, we do not know the best value of $k$ a priori and thus CCS needs to train multiple bases – each for a different value of $k$ – for the following reasons. Under K-SVD, $k$ is fixed and given as an input, and the resulting base only guarantees the best $k$-term representation for the given $k$. On the other hand, given a sampling rate $m$, we typically choose $k$ such that $m = k \log n$ (and its corresponding base) as a guideline for good recovery accuracy[1]. However in practice, because sampling rates may change, the optimal value of $k$ also changes, and so does the corresponding base.

Therefore, CCS performs multiple base-training processes. In each process, a different target sparsity value ($k$) is set and the corresponding base is trained. Then in the reconstruction stage, the optimal base is used depending on the target sparsity.

For many crowd scenarios historical data is available from legacy approaches to the same type of data collection (e.g., census data, phone surveys). If such data is not available, we can also use the traditional methods (that sample all data points) for the first few rounds of system operation, which incurs the same cost, and then transition to the CCS framework for later sampling. Note that CCS applies to more stationary situations, where the learned CS base applies for a period of time. When underlying data correlations are gradually changing, CCS may need to be repeated periodically. CCS is not expected to perform well under conditions where data is likely to undergo sudden structural change.

### Sampling

Collecting data from crowd users can begin once Data Structure Conversion has been completed. Sampling itself can occur in two ways – both of which are in keeping with CS theory.

**Passively.** Users provide data (e.g., respond to questions or collect data) when they wish. Contributed data is grouped based on the columns within the sampling and training groups. However, when CCS performs reconstruction sampling group values are randomly selected and the target data provided by these individuals within the selected sample groups are used during reconstruction. For example, in the non-employer data, when the sampling dimension is *field of industry* and the target is *number of business*, CCS randomly choose some industries as samples and counts the number of business in these industries. Only data from users who have information about these industries are used during reconstruction.

**Pro-actively.** CCS selects randomly within the range of sampling dimension values for users from which to solicit data. Random selection within values of the sampling dimension, just as above, occurs (e.g., a certain industry). However, because collection is pro-active; users who have opt-ed into the crowd system with those characteristics are then directly asked to provide data, such as answers to questions or to manually collect sensor data themselves. If such characteristics about the user are not known then all users can be asked.

The choice of which sampling method is selected depends on the deployment specifics and the user and incentives model adopted. However, the requirements of CCS are simple to meet and only require random selection to be employed.

### Reconstruction

After collecting data from contributing users within sampled groups, reconstruction of the target begins by arranging a matrix representation according to the training and sampling groups of columns (decided by Data Structure Conversion). This matrix can be projected into the trained base to recover a sparse representation of the target. Then the missing target values (that are not sampled) can be recovered by just multiplying the base with the recovered sparse representation.

Under CCS, the precise stages of this process are performed by an existing CS recovery algorithm – Orthogonal Matching Pursuit (OMP) [36]. CCS uses OMP instead of standard $l_1$-norm minimization due to its high efficiency in comparison (OMP – $O(kmn)$ vs. $l_1$ – $O(n^3)$). The desired sparsity of the target data, $k$, is set to the largest value based on how many samples (i.e., user contributions of data) are provided.

## EVALUATION

In this section, we evaluate CCS under a series of mobile crowdsensing scenarios using diverse large-scale datasets.

### Methodology

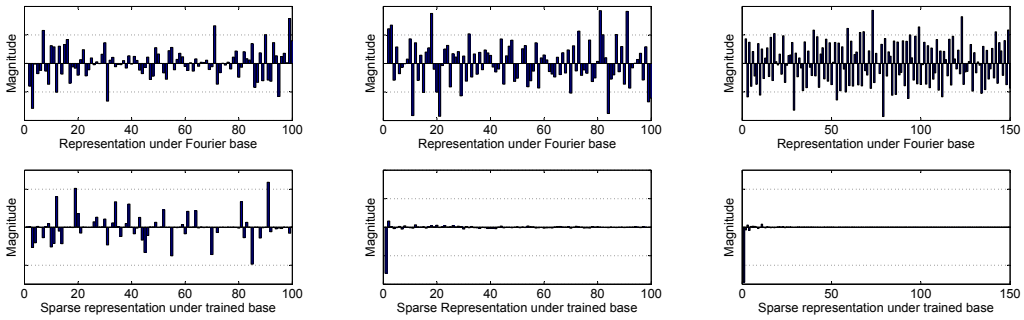
We consider a diverse group of real life datasets, namely rat infestations, noise-complaints, non-employer statistics, and housing attribute reports. These datasets have a variety of dimensions (ranging from 2 to 43). Such diverse datasets allows us to evaluate the general applicability of CCS.

We perform the same random sampling (pro-active version) in these experiments as described in the prior section. Experimentally, this means we select from the underlying dataset based on randomly selected values of the sampling group of columns (e.g., selecting samples that belong to a randomly selected set of industries in the Non-Employer dataset).

**Evaluation Metric.** In CCS, the data we would like to reconstruct is in the format of a vector. For example, in the non-employer survey, it is the vector of the number of non-employer businesses in each field of industry. In all evaluations, we use accuracy ratio as the measurement of recovery accuracy, which counts the percentage of accurate entries in the reconstructed vector signal. Specifically, let $\hat{s}$ be the recovery of a data entry $s$. We consider it accurate if $|\hat{s} - s|/|s| < \tau$, and inaccurate otherwise. In other words, the accuracy ratio is defined as:

$$Acc = \frac{\text{\# of accurate entries}}{\text{total \# of entries}}.$$

[1] to be more accurate, if the signal is $k$-sparse, the sampling rate of $m = k \log n$ can guarantee a lossless recovery under certain technical constraints




(a) Rat Infestation Reports (b) Noise Complaints Heat-map (c) Non-Employer Statistics

**Figure 3:** Comparison of base generated by CCS and a conventional Fourier base

The value of $\tau$ can be set based on the application scenarios, typically in the range of 0.1-0.5.

**Comparison Baselines.** Four baselines are compared to CCS. All baselines reconstruct the original signal (i.e., target data) via a set of random samples, just like CCS.

*Conventional CS.* This approach relies upon the exact same CS stages as CCS, however only a standard Fourier base is used.

*Linear Interpolation.* A method of curve fitting using linear polynomials.

*Spline Interpolation.* Aagin curve fitting is performed but this time using piecewise cubic splines.

*Kriging Interpolation.* A well-known method for geographical interpolation that is popular in the GIS community [35]. It performs well when spatial correlation is high. In our evaluation, this method is used only in datasets with clear spatial components.

*Sampling Only.* This acts as a lower bound for performance, here no reconstruction occurs after sampling. In other words, $x\%$ of sampling results in $x\%$ of accurate entries.

### Base Training Comparison

As detailed in earlier sections, being able to sparsify a crowd collected dataset is a key requirement in being able to apply CS. In particular, CS performs well when the target signal (i.e., a column within a survey) can be represented using only a small number of large coefficients under a certain base. While standard bases, such as Fourier and DCT bases, are widely used, they do not perform well in the multi-column crowdsourcing datasets in general; primarily this because of the inherent complex structure in such datasets. For example, it is hard to even conceptualize what *"frequency"* means with respect to rat sightings or reports about house attributes.

In this first experiment, we compare the base that CCS is able to train compared to a standard Fourier base. This experiment quantifies the improvement in being able to sparsify three of our datasets using CCS. Figure 3(a) $\sim$ 3(c) show the first three datasets represented under the standard Fourier base and under the base trained by CCS. Detailed descriptions of each dataset tested are described in the subsections that follow.

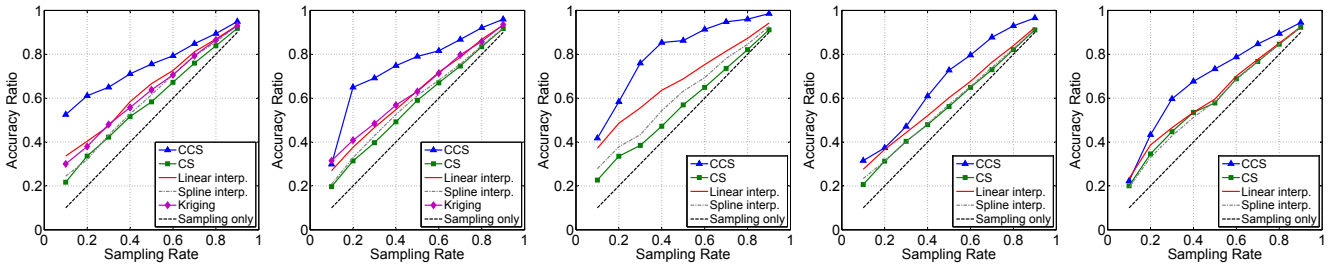
The following comparison shows the empirical necessity of our base training. The top row of figures in Figure 3 clearly show that under the standard Fourier base, a large number of "heavyweight" base coefficients are widely scattered, which indicates none of the datasets can be sparsely represented using the Fourier base. Similar results are observed using other standard bases. In contrast, the row of figures in the bottom show the effectiveness of CCS in revealing inherent sparse structure within our datasets. From the figures we can see, for all datasets, the numbers of "heavyweight" coefficients are much smaller, i.e. sparse, under the trained base.

This difference between the two bases can be quantified by the spread of signal across multiple coefficients. For example, under the rat sightings dataset across 20 coefficients 90% of the signal is spread under CCS – very focused. In comparison, the same dataset under a Fourier base spreads across 37 coefficients to achieve the same 90% output of the signal. We also note that in the rat sighting example, the number of relatively large coefficients is higher than the other cases. This indicates the rat sighting example is hard to sparsify, and not as suited to CCS as other datasets. This does affect CCS performance in a subtle manner, as discussed later.

### Rat Infestation Reports

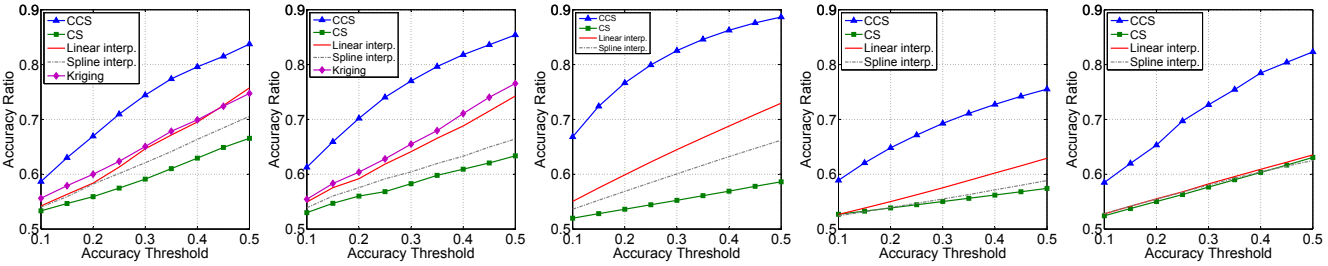
Under our first crowdsensing scenario, the objective is to track the regions where rats are seen in a city. Participants in a crowd system, for example, manually indicate the rat sightings using a mobile app. We assume time and location are automatically recorded by the device.

The objective of CCS in this setting is to accurately estimate the city-wide distribution of a rat infestation (i.e., the reported rat sightings across quantized city regions) while participants provide as few data samples (i.e., manual rat sightings) as possible. To evaluate this scenario, we use a dataset collected by the city of New York [5]. It contains 40,000 actual reported rat sightings over a four year span (January 1st, 2010 – March 12, 2014). This dataset simply contains two columns of information: $\{$*sighting date, sighting location*$\}$. For this experiment, we divide Manhattan, Bronx, Brooklyn, and Queens into 100




(a) Rat Infestation Reports (b) Noise Complaints (c) Non-Employer Statistics: $\{year\}$ in sampling (d) Non-Employer Statistics: $\{year\}$ in training (e) Housing Attribute Survey

**Figure 4:** Accuracy of *target* Reconstruction




(a) Rat Infestation Reports (b) Noise Complaints (c) Non-Employer Statistics: $\{year\}$ in sampling (d) Non-Employer Statistics: $\{year\}$ in training (e) Housing Attribute Survey

**Figure 5:** Sensitivity of Accuracy Threshold ($\tau$)

equally sized square regions (approximately $1 \times 1.4$ sq. miles each). The total monthly reported rat observations within each region is then used as the ground truth of the target value. We use the first two years of the dataset as historical data for base training, and the remaining 15 months for evaluation. CCS selects $\{sighting\ location\}$ as the sampling dimension and $\{sighting\ date\}$ as the training dimension.

As shown in Figure 4(a), CCS achieves a higher level of accuracy compared to all of the other baselines ($\tau = 0.3$). From the figure, we observe that CCS achieves 60% of accuracy using only 20% of samples. In other words, under CCS 60% of all monthly reports for each location (1500 estimates) are within the margin for error ($\tau$ value) using just 20% of original set of rat sightings (i.e., a sampling rate of 0.2). Furthermore, CCS is able to outperform the second best performing baselines, linear and Kriging, by as much as 70%.

CCS is able to perform better than others because it selects the most important coefficients in the reconstruction for a given sampling rate. As the sampling rate increases, so does the number of coefficients selected (recall that given $m$ samples, we can pick $k = m/\log n$ coefficients for recovery), and thus better accuracy results.

As noted in Figure 3(a), the coefficients under the trained base for rat sightings are not so sparse as in the other datasets. As a result, energy is more equally spread among the coefficients. Thus when the sampling rate grows, the accuracy grows more linearly. In the other datasets, as we will see, we will observe a much bigger jump in the beginning as the more important (larger) coefficients are picked up.

We note that in the rat dataset, the spatial correlation is not consistent and only exists in certain directions. We believe that this is the reason why Kriging does not perform as well as in cases with more uniform and reliable spatial and temporal correlations (such as earthquake data). This is similar in the next noise heat-map example.

Figure 5(a) compares the accuracy results under different values of $\tau$, where 50% of samples are used (i.e. sampling rate is 0.5) . One important and interesting observation here is that as the error margin increases (i.e., $\tau$ increases), the performance improvement of CCS is much larger than that of all other schemes. For example, as $\tau$ increases from 0.1 to 0.5, the accuracy of CCS increases from 0.59 to 0.84, while the second best (Kriging) increases from 0.56 to 0.75. This indicates that CCS produces reconstructions that are much closer to the actual values compared to others, and thus the faster improvement. This observation holds for all cases that we evaluated.

### Noise Complaints Heat-map

Now we study a common crowdsensing application – monitoring excessive noise within a city (e.g., [30]). Similar to the prior scenario, participants of a crowd system would manually tag a situation whenever they encounter excessive noise using their mobile devices; or alternatively participants could contribute with microphone samples that are then processed to extract the audio volume. Again, time and location of the data collection can be gathered automatically. The objective is to construct a heat-map of the spatial distribution of city noise with few manually generated reports from participants.

Experiments use the NYC noise complaints dataset [2] that records all noise complaints in NYC from January 1, 2010 to March 14, 2014. More than 240,000 complaints are captured. Each record contains two simple columns: {*reporting date, reporting location*}. Similar to the case in the rat sighting scenario, the dataset provides ground-truth for the evaluation; the city is divided into 100 regions that each receives monthly estimates of the number of noise complaints (i.e., CCS target). Data from 2010 to 2012 is treated as historical data to bootstrap CCS, with the remaining 15 months used for evaluation.

Figure 4(b) presents similar findings to that of the rat sighting scenario ($\tau = 0.3$). Again, we find CCS is better able to more accurately reconstruct monthly noise complaints for 100 spatial regions while assuming just a fraction (from 10% to 90%) of actual reports are collected . However in contrast to the rat sighting scenario, CCS experiences a rapid increase in accuracy as user participation initially increases; and a much slower increase later on. As discussed earlier, this is because the noise map data shows a highly sparse structure in the trained base (i.e., a small number of large coefficients and a large number of smaller coefficients.) Therefore, as the sampling rate increases, the large coefficients are picked up first and thus the speedy improvement. Later on, as only smaller coefficients are left, the accuracy improvement slows.

Last, in Figure 7, we see a clear decrease in the total error as the value of $k$ (the number of coefficients used in recovery) increases. It shows that the sensing data could be represented more accurately with more coefficients and their corresponding bases. Similar results can be observed in the other scenarios reported in this section, and thus omitted here.

#### Non-Employer Statistics

We next examine a crowdsensing goal of collecting information about the number of non-employer businesses in the U.S. (again, these businesses are those without paid employees). The target for this scenario is to count the number of "non-employer" businesses within a randomly selected 100 industry groups. As always, here the underlying goal of CCS is to estimate the actual value for all industries with as few user data contributions as possible (i.e., a low sampling rate).

To study this scenario, we use a U.S. based nation-wide dataset containing one million records from 3140 counties and 475 different industries [3]. We select 70% of records randomly to act as historical data (i.e., training data) in this experiment. The remainder of records act as a validation/test dataset. A single record consists of {*census year, location (county), field of industry, the number of businesses*} (see Figure 2 as an example). CCS selects {*census year*} and {*field of industry*} in the sampling dimension and {*location*} in training dimension.

To provide additional insights into the impact of a column being in either the sampling or training dimension , we repeat the same experiment but manually shift *census year* to the training dimension. Results of the two experiments are shown in Figure 4(c) and 4(d), respectively. In the first scenario, CCS significantly performs others. However in the second scenario, the performance of CCS degrades much although still outperforms other baselines. The reason is that the target

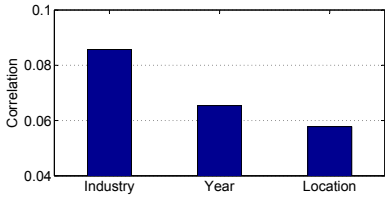



**Figure 6:** Correlation between columns and the target

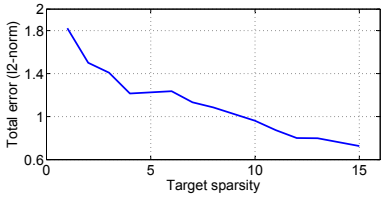



**Figure 7:** Total error decreases as $k$ increases.

data of *the number of businesses* exhibits high temporal correlation, i.e. the column of *year* is highly correlated with the target column as shown in Figure 6, thus making it necessary to include *year* in the sampling dimension. This experiment highlights the importance of appropriately selecting sampling and training dimensions; as well as good CCS performance.

Figures 5(c) and 5(d) show these two scenarios under different values of $\tau$ with a fixed sampling rate of 50%. We do note that as the sampling dimension changes, the absolute values of other baselines change as well effectively because we are trying to recover different signals. For all schemes, the results are better with *census year* in the sampling dimension, with CCS always being the best performing approach.

#### Housing Attribute Survey

In our final scenario, we consider studying housing attributes of privately-owned homes. Users within this crowd system would manually contribute via their mobile device information about either their own home or a home they were familiar with. Typically this information is only collected during mass survey events like the U.S. Census or other smaller scale manual surveys. Instead, with a mobile crowd application, much more timely or focused (i.e., targeting a particular population) data collection can be performed.

We evaluate this scenario using data [1] from a multi-year carefully conducted survey of 300,000 homes in the U.S, performed between 1999 and 2008. This survey provides 43 different attributes (e.g., completion date, parking facility, sale price interval, # of bedrooms, # of bathrooms, and the size) for each home. Again, we use the dataset as the ground truth and, just as in the prior dataset, 70% of data is randomly selected to act as training data.

In this scenario, we use average price-per-square-foot (PSQ) for each home as the objective to estimate across the whole population (i.e., the target data), as PSQ is an important metric in the housing market. Under CCS, the following become sampling dimensions {*sale price interval, primary space heating fuel, primary space heating system*}; and {*number of*

*bedrooms, porch status, construction method*} as training dimensions[2]. Different from all other scenarios we studied in this paper, the number of columns is high in the housing data. To avoid a large number of empty data cells, the remaining attributes are not used in either sampling or training groups.

Figure 4(e) again shows CCS is able to more accurately estimate PSQ than all other alternatives ($\tau = 0.4$). At the same time, Figure 5(e) shows that CCS generates reconstructions that are more in the vincinity of the true values, as before (sampling rate is 50%).

## DISCUSSION

We discuss the generalizability of CCS and its limitations.

**Generalization.** Although our experimental results are based only on a few specific crowd-related datasets and scenarios, we expect CCS will also exhibit similar performance across a wider range of alternatives due to the diversity in our datasets. However, anticipating which crowd-based scenarios are likely to contain the necessary correlations for CCS to excel is difficult. Within broader CS research, how to predict dataset performance is not known. As a result, testing the applicability of CCS must be done experimentally. Our evaluation shows CCS is effective in crowd scenarios where the collected data contains temporal, spatial and demographic related attributes; we believe even if such information is not the core target of the crowd system it is beneficial to collect meta-data of this type for CCS, as it is the foundation for many potential correlations.

**Limitations.** CCS, at this time, has some clear limitations. First, as shown in the rat sighting example, when the data structure has no strongly sparse representation, the results are not as good (although still better than all tested baselines). Clearly, because CS depends on the correlation between the target column with other columns, if the original data shows no such correlation, CS would not apply. Second, because we are combining different columns into sampling and training dimensions, the overall vector space may be very large, and so results in missing values in the converted matrix format. We have taken preliminary steps to address such issues (such as Step 5 in Algorithm 1), further research is needed to better address such issues. Finally, CCS requires historical data of target phenomenon to be available from which a CS base can be trained. Similarly, while being used CCS assumes the structure and relationships within the data and phenomenon are unchanged from what is observed in this historical data. We anticipate on-line base training to be possible under CS but have yet to test this approach.

## RELATED WORK

We survey the most salient related work from crowdsensing and compressive sensing, and also highlight CCS novelty.

**Mobile CrowdSensing.** Due to the rise in prominence of smartphones equipped with sensors, large-scale crowdsensing systems have become much more feasible to deploy. Today a variety of both commercial and research prototype systems target a variety of application domains and monitor, for example: traffic conditions [44], place categories [13], noise pollution [30], WiFi conditions [4]. An important category of crowd systems also attempt to gather information about large populations, for example: happiness [6]. In this work, we have investigated the applicability of a CS framework for both sensor data and survey data forms of crowdsensing.

In an effort to lower the sensitivity of these systems to the number of participants and their level of engagement (i.e., interest level) a variety of approaches have been explored. Incentive mechanisms (e.g., [32, 40]) are powerful way towards increasing the amount and quality of data collected. Similarly, techniques for guiding users towards certain user behavior useful to the crowd system is highly needed (e.g., [33]).

**Compressive Sensing.** Since shifting into mainstream consciousness [12, 17], a steady stream compressive sensing applications have continued to arrive. Within the domain of sensing, a strong body of work has explored the use of compressive sensing in static sensor networks (e.g.,[27] [38]). Recently, mobile CS sensor applications are emerging – for example, human activity sensing using accelerometer data [9, 41]. More closely related to CCS, traffic monitoring using compressive sensing has been explored in studies such as [42] [44] [39]. However, such domain and data specific solutions ignore the important need to collect user generated survey data as well as sensor data.

A core aspect of CCS is its ability to process multiple dimenion data, and understand how to find CS-friendly data representations. However, CCS is not the first to perform this form of processing. Exist work offers several ways to convert high-dimensional signals to 1-D [18, 37]. Among these, [18] is the most related one to our work. By introducing the concept of the Kronecker product, [18] makes it possible to reduce the signal dimension to one. But this product requires that the target signal has sparse expressions on *every dimension*, which is does not hold for the survey data we examine here. In fact, we find this is true of only a few dimensions of the survey data we examine. Applying [18] to our application domain will require each dimension be closely inspected so to determine which have this property.

## CONCLUSION

In this paper we presented CCS, a compressive sensing framework for mobile crowdsensing that recovers large-scale urban information from user contributed data. The main contributions of CCS are two-fold – first, it demonstrates novel crowd-based applications of compressive sensing that lower the amount of data required and thus reduces overall user burden; and second, it develops key new techniques that allow CS to be generically applied to many scenarios.

We evaluate CCS under various representative datasets each of which are useful for managing a city and urban population. Our evaluation shows that CCS outperforms often used existing baseline techniques in all tested scenarios. These results are promising and – given the diversity of the data – suggest our approach may generalize to many other forms of urban-focused mobile crowdsensing.

[2] The complete definitions of each column can be found in [1]

## REFERENCES


1. Characteristics of new housing - united states census bureau. https://www.census.gov/construction/chars/.
2. Noise complaints heatmap - nyc open data. https://data.cityofnewyork.us/Social-Services/Noise-Complaints-Heatmap/654p-getv.
3. Non-employment statistics - united states census bureau. http://www.census.gov/econ/nonemployer/.
4. Open signal. http://opensignal.com.
5. Rat sighting reports - nyc open data. https://data.cityofnewyork.us/Social-Services/Rat-Sightings/3q43-55fe.
6. Track your happiness. http://www.trackyourhappiness.org.
7. Aharon, M., Elad, M., and Bruckstein, A. -svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on 54*, 11 (2006), 4311–4322.
8. Ahmed, N., Natarajan, T., and Rao, K. R. Discrete cosine transform. *Computers, IEEE Transactions on 100*, 1 (1974), 90–93.
9. Akimura, D., Kawahara, Y., and Asami, T. Compressed sensing method for human activity sensing using mobile phone accelerometers. In *Networked Sensing Systems (INSS), 2012 Ninth International Conference on*, IEEE (2012), 1–4.
10. Arslan, M. Y., Singh, I., Singh, S., Madhyastha, H. V., Sundaresan, K., and Krishnamurthy, S. V. Computing while charging: Building a distributed computing infrastructure using smartphones. In *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, ACM (New York, NY, USA, 2012), 193–204.
11. Campbell, A. T., Eisenman, S. B., Lane, N. D., Miluzzo, E., and Peterson, R. A. People-centric urban sensing. In *Proceedings of the 2Nd Annual International Workshop on Wireless Internet*, WICON '06, ACM (New York, NY, USA, 2006).
12. Candès, E. J. Compressive sampling. In *Proceedings oh the International Congress of Mathematicians: Madrid, August 22-30, 2006: invited lectures* (2006), 1433–1452.
13. Chon, Y., Lane, N. D., Kim, Y., Zhao, F., and Cha, H. Understanding the coverage and scalability of place-centric crowdsensing. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, ACM (New York, NY, USA, 2013), 3–12.
14. Church, K., Cherubini, M., and Oliver, N. A large-scale study of daily information needs captured in situ. *ACM Trans. Comput.-Hum. Interact. 21*, 2 (Feb. 2014), 10:1–10:46.
15. Consolvo, S., McDonald, D. W., Toscos, T., Chen, M. Y., Froehlich, J., Harrison, B., Klasnja, P., LaMarca, A., LeGrand, L., Libby, R., Smith, I., and Landay, J. A. Activity sensing in the wild: A field trial of ubifit garden. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, ACM (New York, NY, USA, 2008), 1797–1806.
16. De Smith, M. J., Goodchild, M. F., and Longley, P. *Geospatial analysis: a comprehensive guide to principles, techniques and software tools*. Troubador Publishing Ltd, 2007.
17. Donoho, D. L. Compressed sensing. *Information Theory, IEEE Transactions on 52*, 4 (2006), 1289–1306.
18. Duarte, M. F., and Baraniuk, R. G. Kronecker compressive sensing. *Image Processing, IEEE Transactions on 21*, 2 (2012), 494–504.
19. Froehlich, J., Chen, M. Y., Consolvo, S., Harrison, B., and Landay, J. A. Myexperience: A system for in situ tracing and capturing of user feedback on mobile phones. In *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, MobiSys '07, ACM (New York, NY, USA, 2007), 57–70.
20. Gersho, A., and Gray, R. M. *Vector quantization and signal compression*, vol. 159. Springer Science & Business Media, 2012.
21. Goncalves, J., Ferreira, D., Hosio, S., Liu, Y., Rogstadius, J., Kukka, H., and Kostakos, V. Crowdsourcing on the spot: Altruistic use of public displays, feasibility, performance, and behaviours. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, ACM (New York, NY, USA, 2013), 753–762.
22. Goncalves, J., Pandab, P., Ferreira, D., Ghahramani, M., Zhao, G., and Kostakos, V. Projective testing of diurnal collective emotion. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, ACM (New York, NY, USA, 2014), 487–497.
23. Howell, D. *Statistical methods for psychology*. Cengage Learning, 2012.
24. Lane, N. D., Xu, Y., Lu, H., Hu, S., Choudhury, T., Campbell, A. T., and Zhao, F. Enabling large-scale human activity inference on smartphones using community similarity networks (csn). In *Proceedings of the 13th International Conference on Ubiquitous Computing*, UbiComp '11, ACM (New York, NY, USA, 2011), 355–364.
25. Lathia, N., Rachuri, K. K., Mascolo, C., and Rentfrow, P. J. Contextual dissonance: Design bias in sensor-based experience sampling methods. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, ACM (New York, NY, USA, 2013), 183–192.

26. LiKamWa, R., Liu, Y., Lane, N. D., and Zhong, L. Moodscope: Building a mood sensor from smartphone usage patterns. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, ACM (New York, NY, USA, 2013), 389–402.

27. Luo, C., Wu, F., Sun, J., and Chen, C. W. Compressive data gathering for large-scale wireless sensor networks. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, ACM (2009), 145–156.

28. Lustig, M., Donoho, D., and Pauly, J. M. Sparse mri: The application of compressed sensing for rapid mr imaging. *Magnetic resonance in medicine 58*, 6 (2007), 1182–1195.

29. Paulos, E., Honicky, R. J., and Hooker, B. Citizen Science: Enabling Participatory Urbanism. *Handbook of Research on Urban Informatics: The Practice and Promise of the Real-Time City* (2009), 414–436.

30. Rana, R., Tung Chou, C., Kanhere, S., Bulusu, N., and Hu, W. Ear-phone: An end-to-end participatory urban noise mapping. In *IPSN '10: Proceedings of the 9th international conference on Information processing in sensor networks*, ACM (New York, NY, USA, 2010).

31. Rea, L. M., and Parker, R. A. *Designing and conducting survey research: A comprehensive guide*. John Wiley & Sons, 2014.

32. Reddy, S., Estrin, D., Hansen, M., and Srivastava, M. Examining micro-payments for participatory sensing data collections. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, UbiComp '10, ACM (New York, NY, USA, 2010), 33–36.

33. Rula, J., and Bustamante, F. E. Crowd (soft) control: Moving beyond the opportunistic. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, HotMobile '12, ACM (New York, NY, USA, 2012), 3:1–3:6.

34. Stein, C. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, University of California Press (Berkeley, Calif., 1956), 197–206.

35. Stein, M. L. *Interpolation of spatial data: some theory for kriging*. Springer, 1999.

36. Tropp, J. A., and Gilbert, A. C. Signal recovery from random measurements via orthogonal matching pursuit. *Information Theory, IEEE Transactions on 53*, 12 (2007), 4655–4666.

37. Wainwright, M. J. Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting. *Information Theory, IEEE Transactions on 55*, 12 (2009), 5728–5741.

38. Wu, X., and Liu, M. In-situ soil moisture sensing: measurement scheduling and estimation using compressive sensing. In *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, ACM (2012), 1–12.

39. Xu, L., Hao, X., Lane, N. D., Liu, X., and Moscibroda, T. Cost-aware compressive sensing for networked sensing systems. In *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*, IPSN '15, ACM (New York, NY, USA, 2015), 130–141.

40. Yang, D., Xue, G., Fang, X., and Tang, J. Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking*, Mobicom '12, ACM (New York, NY, USA, 2012), 173–184.

41. Yang, S., and Gerla, M. Energy-efficient accelerometer data transfer for human body movement studies. In *Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), 2010 IEEE International Conference on* (2010), 304–311.

42. Zhang, Y., Roughan, M., Willinger, W., and Qiu, L. Spatio-temporal compressive sensing and internet traffic matrices. In *ACM SIGCOMM Computer Communication Review*, vol. 39, ACM (2009), 267–278.

43. Zheng, Y., Liu, T., Wang, Y., Zhu, Y., Liu, Y., and Chang, E. Diagnosing new york city's noises with ubiquitous data. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, ACM (New York, NY, USA, 2014), 715–725.

44. Zhu, Y., Li, Z., Zhu, H., Li, M., and Zhang, Q. A compressive sensing approach to urban traffic estimation with probe vehicles. *Mobile Computing, IEEE Transactions on 12*, 11 (2013), 2289–2302.