An *In-the-wild & Synthetic* Mobile Notification Dataset Evaluation

Author
School
University
Email

Author
School
University
Email

Author
School
University
Email

***Abstract*—Managing the vast amounts of information being pushed at mobile users is a challenge that is becoming increasingly difficult as the number of connected devices and users continues to expand. In order to overcome this challenge, a Notification Management System (NMS), needs a number of detailed data resources in order to decide what to do with an incoming notification *in-the-wild*. Explicit data contained within the notification and contextual information regarding the user and immediate environment are both necessary in order for a system to accurately infer a user's preferred delivery time for a given notification. Due to the sensitive nature of notifications and contextual data, it is difficult to acquire the explicit notification datasets which sufficiently describe the incoming notifications as well as the current contextual states of the user. This poses a problem for prospective research in the domain of Notification Management as arduous and time-consuming data collection is necessary if a hypothesis depends on unique notification/user features not previously collected. Without a number of rich notification datasets, either experimentation is limited to synthetic, vague or incomplete data, or time must be invested in developing a system to capture the required features.**

In this paper we evaluate a notification dataset previously collected *in-the-wild* and subsequently used in an evaluation of a NMS. We outline the necessary features of the collected dataset as well as its limitations. As a comparison, we then evaluate the process of creating a synthetic notification dataset derived from a mobile usage study carried out by the MIT Media lab. The synthetic dataset is henceforth used to optimize a previous set of knowledge base rules and membership functions used within the Fuzzy Inference System (FIS) of a NMS. The resulting optimized rules can be presented to the user as a means of throttling notifications based on their goals.

***Index Terms*—Mobile applications; Fuzzy systems; Particle swarm optimization; Data acquisition;**

I. INTRODUCTION

Mediating mobile notification interruption is becoming an ever more important challenge in ubiquitous computing. An increasing number of connected devices and application subscriptions results in a torrent of information being pushed at users' throughout the day lacking intelligence or empathy for the user [1]. Current means of throttling the stream of incoming information are blunt and in no way user friendly as they rely on modifying individual application settings. Some apps have been created for the purpose of setting device-wide rules for notification delivery [2], [3] however, these fail to provide adequate functionality which fully captures an individual user's preferences on an ongoing basis. This results in important notifications being potentially missed, which is an unforgivable user experience, or unwanted notifications still reaching and distracting the user.

Recent research into personalized notification management has utilized both incoming notification content and user context to infer a preferred delivery time for incoming notifications [4], [5], [6], [7], [8]. In order to develop systems such as these, rich datasets which relate closely to real world use are necessary for evaluation and training purposes. Few (if any) open datasets exist which hold explicit notification content, current user context and user feedback on preferential notification delivery. This challenge results in researchers having to allocate time to developing systems which log notification and or context attributes *in-the-wild* [4], [9], [10], [11], [12], [13]. In addition, due to the ethical implications of collecting sensitive information about test subjects, the notification content collected is usually incomplete, abstracted or hypothetical. Consequently, NMS's trained using these abstracted datasets fail to differentiate between small nuances present between incoming notifications and a user's context.

Moreover, few NMS's take into account a user's trust in autonomy. In order for the NMS to be of use in the real world, a user must give up a measure of control to a machine. This is a shift in power and one that is difficult for a user to accept. Research has shown that user's are wary of giving up power to NMS and prefer transparency and some degree of input from the user [14]. The majority of current research in the area of Notification/Interruption management is comprised of "black box" systems which don't give the user explicit input into the decision making process. These systems are lacking an intuitive interface for viewing how or why a notification is delivered to them and a means to contribute an element of control. Mehrotra et al. are some of the few now addressing the usability of such systems [15].

In this paper, we evaluate a modest notification dataset [16] captured *in-the-wild* and comprised of:

- Notification content
- User context
- Preferred notification delivery method

In this evaluation we compare features captured in this the with those of other datasets used in Notification/Interruption management and discuss both the limitations involved in collecting the data as well as the potential for scaling.

Subsequently, this paper then discusses the process of creating a large-scale synthetic notification dataset derived from a mobile usage study [17]. The purpose of this is to further our understanding of the explicit information which is being captured on the user and evaluate both its implications regarding user privacy and its potential for throttling notification delivery in the form of user control. Moreover, this paper proposes a novel means of improving a NMS through use of only a synthetic dataset. Particle Swarm Optimization is used with the synthetic dataset to find optimal parameters for a stereotypical user. Newly derived rules are then applied to the NMS and the results are evaluated using real qualitative data.

The structure of the paper is as follows: Section II discusses current trends in mobile notification dataset procurement as well as possible alternatives to harvesting real-world data. Section III evaluates an existing mobile notification dataset captured *in-the-wild*. Section IV details the creation of a synthetic dataset which is proposed as an alternate method of procuring useful notification data for the purposes of improving a NMS's algorithm. Section V evaluates the quality and limitations of the synthetic dataset. Section VI details two experiments performed to ascertain if the synthetic dataset could provide value to improving the NMS algorithm. Finally, Section VII concludes with a summary of findings from each section.

II. RELATED WORK

The majority of explicit data captured to date exploring the area of notification and interruption management is not open to the wider research community due to ethical complications arising when it comes to sharing personal user information. Mobile notifications paired with contextual user data is quite sensitive in nature, particularly at the level of detail required for Notification Management. Hence, researchers are developing ad-hoc software solutions to capture notification and user features. There is no one set list of features and many solutions overlap. Tables I and II contain a brief summary of known features collected and explored by a number of in-depth research studies carried out in this area [4] [13] [18] [19] [9] [12] [20] [8].

Table I contains features captured using an Experience Sampling Method (ESM). This broadly involves sending notifications to users' which contains a number of self-evaluating questions regarding the interruption, their current feelings toward the interruption, their current context and so on. Table II on the other hand involves developing software for installation on the users' devices to capture data regarding notifications and users' context. Both sets of features are useful in describing the context behind the notifications, the users and the relationship between them. Ideally, a NMS would like to have access to all this information so as to make the best possible informed decision. However, much of the information being logged is of a very sensitive nature. The obvious privacy concerns that jump out are of course user *Location*, *Microphone* and *People Nearby*. *Location* is intrusive to the user of the device, but it could be argued that having signed up to a NMS, this could be a small price to pay. However, the *Microphone* and *People Nearby* features implicate not only the user, but also the people they interact with. These people are unaware that their location and engagements are being monitored.

Mendez-Vazquez et al. [21] discuss the difficulties associated with harvesting quality datasets from pervasive spaces. Highlighted in their discussion are the problems regarding inconsistent sets of features found across datasets and the difficulty in recruiting participants willing to have their data monitored. They propose the solution of creating synthesized data using Markov Chains, Poisson processes and probability distributions. One such advantage of synthetic data expressed is the degree of control it gives as the events generated can be tuned to fit a hypothesized pattern. In comparison, this paper similarly discusses a means by which a synthetic dataset is derived from a real-world mobile usage study in order to preserve privacy and maintain control over the behavior of the notification and user context data. The goal is to have a dataset which can be modeled to reflect real-world behavior but does not contain sensitive information pertaining to a real user.

Corno et al. [22] also derive a notification dataset for testing Notification Management algorithms. Using an existing mobile usage dataset [23], they add synthetic data as required to enable the dataset to train various machine learning algorithms used within a NMS. This is an example of both the need in the research community for access to data quickly in order to test and improve hypotheses as well as highlighting the prospect of synthesized data being sufficient for purposes of improving algorithms.

Mehrotra et al. [13] went the alternate route of developing software to capture notification and user context *in-the-wild*. The objective and subjective data collection method used was comprehensive as they implemented both sensory and ESM procedures for gathering data. For privacy reasons the content of notifications were dismissed however. In contrast, Fraser et al. [16] do not completely discard the notification content of their *in-the-wild* notifications but instead enable the user to uplift the sensitive information to an abstract form. If notification content could be generated synthetically while maintaining real-world underlying traits, there would be no need to abstract or discard valuable contextual information such as this.

III. "IN-THE-WILD" DATASET EVALUATION

In a recent study of notification management [16] two notification datasets were harvested from two users'. An *Android* device was used to capture features of incoming mobile notifications via an application developed for the purposes of the study. This application was left to run continuously on both users' mobile devices' over the course of 10 weeks. A total of 3,174 notifications were captured. The notification features logged in the study were as follows:

- Notification title - Used to identify the *Sender*
- Application package name - Used to identify the *App*
- Notification message body - Used to identify the *Subject*

Feature	Description
Availability	Current availability status of the user
Location	Current user location
Conversation	Currently engaged in conversation
Movement	Whether in transit
People nearby	Who is currently with the user
Mood	Current mood, 1-5 Likert scale or free-text
Activity	Current activity & complexity
Sender relationship	Relationship between user and sender
Subject	Notification subject
Disruption	Ranking of sentiment toward notification disruption
Decision	Reason for opening notification
Awareness	Time to notice the notification alert
Placement	Where the user's mobile was located
Content	Rating of notification content usefulness
Recommendation	When/where would like to receive this notification

TABLE I: Features for notification and interruption management collected using ESM.

Feature	Description
Accelerometer	Accelerometer readings
Location	GPS location
Proximity	Proximity sensor readings
Microphone	Microphone readings for noise
Screen events	User taps, opening apps etc.
SMS Events	Incoming SMS messages
Call Event	Incoming call events
Time	Incoming notification time
Response	Notification subject
App	Sending application
Alert Type	Signals used to alert user e.g. sound, lights
Title	Notification title
WiFi	Connection to WiFi
Arrival time	Arrival time of the notification
Removal time	Time notification removed from the taskbar
Response	Whether the notification was clicked
Phone status	Whether the phone was within use in last minute
Ringer mode	Current ringer mode, e.g. vibrate, sound

TABLE II: Features for notification and interruption management captured with sensors.

- Delivery date and time

These features were logged using the *Android NotificationListenerService* which identified when notifications were delivered to the users' status bar. On delivery of the notification, the features listed above were logged and stored in a local *SQLite* database on the user's phone. Both users' *Google Calendar* data was also stored on particular days as was an additional ranked list of terms used for uplifting the explicit notification data to an abstract form. For example, based on the context surrounding the notification, the user might uplift the *Subject* of the notification to "work" and the *Sender* of the notification to "colleague". The notification dataset was evaluated using two data quality metrics [24] pertaining to the research being carried out:

- Believability - *the extent to which the data is regarded true and credible*.
- Completeness - *the extent to which data is not missing and is of sufficient breadth and depth for the task at hand*.

While the data captured in this dataset scored highly on completeness due to few missing data points it can also be argued that, due to mixed results in the management of notifications across multiple users' discussed later in the paper, the notification and user data captured wasn't at a sufficient level of granularity for the task at hand.

Moreover, the believability of the data as viewed by the two users' involved in the study was surprising low. This highlights a problem which is sometimes overlooked due to emphasis put on performance as opposed user experience, and that is users' don't yet fully trust automated systems to make important decisions on their behalf. Black-box NMS's which have no means of input from the user and which make decisions that are not easily transparent and comprehensible are not acceptable for users' who are unwilling to give up complete control over their notifications [14].

To compare features recorded in the notification dataset currently being discussed and those found in Table I and Table II, it is clear that features capturing user interaction with the notification were missed. This is a definite limitation of the dataset as insight into features such as *Removal time Response* and *Screen events* could help in understanding user behavior toward notifications in general but also their bias toward certain types. Similarly, in terms of the number of contextual features available, the opportunity of tapping into a rich user-context store of information was missed. While using only *Google Calendar* information may limit the granularity of the dataset, it can also be argued that it is the cost of preserving the privacy of the user. Other context features such as *Location* (GPS) and *Microphone* are much more intrusive to a user than their calendar. Striving to find the optimal balance between preserving privacy and achieving rich contextual data is a prominent challenge in this domain.

The scalability of the method by which this dataset was captured is also limited as it requires users to uplift their data to standardized term sets. The explicit data captured from the notification would be difficult to uplift accurately in an autonomous manner as the nuances of the notifications are subtle and depend heavily on the contextual circumstances surrounding it. Ethical concerns also arise due to the notification data being captured not only implicating the user involved but also those who engage with them. Time and effort too is a major drawback as it is difficult to ask participants to uplift every notification they receive throughout the day and log every contextual aspect of their current state at the time of the notification. Therefore, other means of capturing the data in an non-intrusive but transparent manner needs to be investigated as well as the possibility of simulating notification data.

IV. CREATION OF A SYNTHETIC DATASET

As discussed in previous sections, the availability of open source notification datasets is quite low. The majority of datasets collected are fit-for-purpose and can't be used openly by researchers for ethical reasons. The lack of data available makes it difficult to train a NMS without first developing software to capture both notification and user data. Aside from the added time it takes to develop the mechanism to

capture the necessary notification features, the process also requires finding a group of participants willing to have their notifications monitored on an ongoing basis. Paired with this process are ethical dilemmas. For instance, a participant may be willing to share their explicit notification data with researchers for the purposes of a mobile study, but they are not the sole contributors to their own notifications. The sender of the notification is also captured in this process. Is it necessary therefore, to alert all the participant's contacts also that the information they send to this participant is being monitored or is the participants acceptance alone ethically sufficient? These questions make it difficult to acquire open source, granular, notification datasets which accurately model real-world situations.

As a possible solution to these data-acquisition hurdles, this paper discusses the creation of a synthetic notification dataset derived from a real mobile usage study. The argument is that the notifications and users in the dataset need not be real. They must only resemble underlying real-world behaviors and situations for the dataset to be useful to a NMS for training. The first step in creating a truly synthetic dataset is identifying the necessary features which must be included. The previous section discussed a range of notification and user features which should be available for NMS's to make accurate context-driven decisions. In the scope of this study however, we limit the necessary features to those used in an existing NMS [16] as the dataset can then be tested for validity. The features necessary for testing with the NMS are:

1) Notification data - Sender, subject, application, date/time.
2) Activity data - Events/activities the user is engaged with at the time of notification delivery.
3) User preference data - A general importance a user gives to possible terms of the notification features. For example, on a scale of 1 - 10, how important is the subject *work* to this user or how important is the sender *family member*.

The Friends & Family (F&F) dataset [17] was collected for a mobile-phone-based social and behavioral study undertaken by the MIT Media Lab. The data was collected in 2011 within a residential community with ties to a nearby university. The data was used to investigate social involvements in decision making processes and to explore how those decisions could be improved. The data was collected *in-the-wild* through the *Android* mobile phone devices of the participants. The data includes quantitative sensory data such as accelerometer readings and proximity networks, as well as qualitative data in the form of weekly and monthly surveys. Table III lists the individual datasets which make up the complete F&F dataset and illustrates which ones were chosen for building a synthetic notification/user dataset to be used with the NMS. It also highlights the features of the NMS-required dataset that it will contribute to. For example, *Call Log* and *SMS Log* datasets were both used to infer the time and date of incoming notifications. It was argued that these could be thought of as notifications generated by messenger applications, hence the dataset is modeling real world behavior.

Friends & Family dataset	NMS dataset feature
Accelerometer readings	Not used
Apps installed	Notification app
Apps running	Not used
Bluetooth proximity	Not used
Battery Information	Not used
Call Log	Notification time/date
GPS	Not used
WiFi access points nearby	Not used
SMS Log	Notification time/date
Big 5 personality answers	Not used
Relationships (Couples & Friendship)	Sender ranking
Survey monthly, weekly and daily	Subject, Sender, App, App ranking, User events/activities

TABLE III: Friends & Family datasets used for inferring NMS dataset features.

A. Inference Process

As the F&F dataset does not contain all the explicit features required for the NMS, some assumptions and inferences are necessary to build the synthetic dataset. This section details these assumptions and inferences.

1) Notification date/time: - The time and date of the incoming notification was inferred using the *Call Log* and *SMS Log* datasets. By using the frequency of incoming calls and texts, real-world notification frequency can be synthesized.

2) Notification Sender & Ranking: - The NMS requires a notification to have a sender and for the user to have a relationship ranking for that sender based on their importance to the user. In this case, as the notifications are being generated using *Call Log* and *SMS Log* data, the sender of the notification is provided in the form of an anonymous string id. Using this id, the relationship between the receiver and sender of the notification can be found using either the *Couples* or *Friendship* datasets. The *Friendship* data is comprised of relationship ratings given by each user to other users. The *Couples* data is comprised of a mapping of one user id to another to signify both users are a couple. The assumptions made while inferring the sender are as follows:

- If a sender cannot be found within the *Friendship* or *Couple* datasets, the sender is a stranger and is assigned a user importance rating of 0.
- If a sender is found in the *Friendship* dataset and has a rating greater than 0, the sender is classified randomly as either a *colleague* or a *friend*. This was done simply to add context to the notification and aid in inferring the subject (also discussed in this section).
- If a sender is found in the *Couples* dataset then the sender is classified as family and the user rating is 10. This user rating is assumed and may not always reflect a real-world situation.

3) Notification Subject & Ranking: - Each notification must have a subject and an associated ranking of importance provided by the user for that subject. As no information

on the subject was recorded in the *Call Log* or *SMS Log* datasets, the subject had to be inferred by other means. A fixed set of subjects were therefore assumed and the sender of the notification was used to infer which subject of the set was chosen. The subject set is comprised of {family, work, social, interest}. This set of subjects was not chosen randomly but based on an evaluation of the *in-the-wild* notification datasets previously used with the NMS and also based on the *Survey* dataset which groups activities by these terms enabling the mapping of inferred subjects to ground-truth data. The assumptions are as follows:

- If the sender of the notification is a family member, the subject is family.
- If the sender of the notification is a colleague, the subject is work.
- If the sender of the notification is a friend, the subject is social.
- If the sender of the notification is a stranger, the subject is interest.

These assumptions are quite restrictive such that a family member could just as easily send a social or work notification depending on the situation. However, for the purposes of this study, the notifications are created using straightforward assumptions and do not attempt to model intricate nuances in behavior. This will be a recommendation for future work. It must also be noted that the subject ranking was not derived from the F&F dataset. It may have been possible to assign a ranking from each user to each subject term of *work*, *social*, *interest* and *family* by making assumptions based on their personality or frequency of incoming notification senders (e.g. many notifications from family members might signify a higher ranking), however it was decided that the subject ranking could be used alternatively as a dynamic value which could be tuned depending on the type of user wishing to be modeled. For instance, modeling a work oriented user would require ranking work above social and interest.

4) Notification Application & Ranking: - The NMS requires a notification to have an associated app through which it is delivered and to have that app assigned an importance value provided by the user. In this case, the app for a particular notification is chosen based on the inferred subject of the notification (which was chosen based on the sender). A manual mapping of subject to app category was assumed and used to infer which app category was chosen for a given notification IV. Once the app category was chosen, a users' known apps were retrieved from the *Running Apps* and *Survey* datasets. Apps matching the chosen category were added to a pool from which one app was drawn randomly to be the app for the given notification. The user ranking of importance was inferred using the *Survey* dataset which asked users to rank their most used and favorite apps.

5) User Events/Activities: - The NMS also requires contextual user data in order to make notification delivery inferences. There is no explicit information regarding a user's current activity at the time of all incoming calls and SMS messages in the F&F dataset hence, a number of assumptions have to be made regarding users' activities. The *Survey* datasets gather information regarding some events and activities that a user partakes in during the weeks of the study. For example, it records visits to the cinema, restaurants and nights in front of the TV. It also records which groups of people the activity or event is associated with. For example, user's specify if the event is attended with family members, colleague, friends or alone. These are used to create a sparse calendar for users. Additionally, the *Survey* dataset records whether or not the user is a student. Based on this a number of assumed fixed events are added to a users calendar to flesh out the contextual information available. The event assumptions are as follows:

- If the user is a student - they have college lectures from 9am to 5pm each weekday with a break of 1 hour for lunch at 12pm.
- If the user is not a student they are assumed to have work from 9am to 5pm each weekday with a break of 1 hour for lunch at 12pm.
- For events found in the *Survey* dataset, the day of the week is chosen randomly and the time and duration are chosen based on the type of event (Table V).

This concludes the creation of the synthetic dataset derived from the F&F dataset. The majority of the synthetic dataset is grounded in real-world user behavior and yet has elements which can be controlled based on the necessary use for the dataset. For example, the frequency of events can be increased or decreased to model users who have busy or sparse schedules. Notification frequency can be increased or notification subject values can be switched depending on the type of user wishing to be modeled. This synthetic dataset was purposely built to match a dataset which could be used by an existing NMS, however, many more inferences could have been made to build up a richer dataset involving notifications and their associated users. For example, personality traits, bluetooth proximity, location and accelerometer data were all

App Category	Subjects
Games	Family, Interest
Lifestyle	Family, Interest
Shopping	Family, Interest
Communication	Family, Interest, Social, Work
Entertainment	Interest, Social
Phone Personalization	Interest
Productivity	Work
Social	Social
Other	Interst

TABLE IV: Mapping of app categories to notification subjects.

Event Type	Time
Most look forward to	5pm - 8pm
Most enjoyed	5pm - 7pm
Cinema	8pm-9pm
Movie	9pm-10pm
TV	5pm-8pm
Restaurant	7pm-9pm

TABLE V: Mapping of event types to assumed relevant times.

recorded in the F&F dataset but not used in this creation of the dataset.

V. EVALUATING A SYNTHETIC DATASET WITH AN EXISTING NMS

The newly created synthetic dataset has a number of limitations making it difficult to evaluate. The most important feature the dataset is lacking is a preferred delivery time for each notification. Without this feature, a NMS is unable to identify if a recommended delivery is correct or not. Hence, it has no criteria from which it can learn and improve. This is a possible drawback of creating a synthetic dataset as the number of opportunities for it to positively contribute to improving a NMS are limited without qualitative feedback from the user regarding notification delivery.

It would be insufficient to simply assign a random preferred delivery to each notification as this would lack consistency and fail to capture the nuances of human behavior the NMS is trying to cater for. For example, identical scenarios could be assigned different preferred deliveries times thus creating a contradiction. In practice this could be a social notification of low importance, sent from an acquaintance of low importance, being delivered a number of times during different work events throughout a week. If these identical scenarios are each given a different random preferred delivery time the NMS will naturally have to keep learning a new set of rules regarding this scenario, meaning the NMS will never converge to a consistent base model of user behavior.

In this paper therefore, the means of using the synthetic dataset for improving the NMS involves making a generic assumption of user behavior and attempting to model deliveries based on this assumption. The assumption made is that users' prefer to receive most work-related notifications straight away if they arrive within working hours and prefer to receive most social-related notifications later during these same hours (a work-play-split). By giving each user in the synthetic dataset this stereotype the NMS can be trained over a diverse set of scenarios to converge to recommending deliveries which match the work-play-split assumption.

The ensuing "rules" learned by the NMS as a result of processing the synthetic dataset are applied to the NMS and used to process the two real-world datasets evaluated in Section III. The two real-world datasets contain preferred delivery times as noted by the respective users' receiving the notifications hence the performance of the learned rules can be ascertained and with it the performance of the synthetic dataset. This of course is not an ideal method of optimizing the NMS algorithm as it is to some extent generalizing as opposed to personalizing. It may become useful however, in cases where an insufficient amounts of personal data is available for the NMS to learn appropriate rules - the cold start problem is an obvious example. While the learned rules are based on a generalization, they are also being derived using a dataset which was created with real-world user and mobile data, hence there is consistent user behavior behind them.

A. Experiment outlines and assumptions

The work-play-split is used to train two separate features of the existing NMS. Previously expressed as some of the limitations regarding recommended delivery performance [16], these features are:

- **Fuzzy Knowledge Base (FKB)** - This refers to the heuristic rule base found within the fuzzy inference system of the NMS. Essentially this component of the system contains the common-sense rules through which notification delivery is dictated. For example, if the sender is important and the subject is important, deliver the notification now. Or alternatively, if the sender is not important and the subject is moderately important, deliver the notification at the user's next break in activity. Previously, the knowledge base for a user was fixed and didn't change dynamically with the user's current contextual situation. This could be seen as a limitation of the NMS as a user's preferences would generally change throughout the day. For example, a user may not want to be bothered by any notification during working hours with a sender value classified as only *moderately important* but this may not be true for hours outside of work when a user has free time. It could be argued therefore that the resulting recommended delivery for working hours in this case be *later*, but recommended delivery for non-working hours be *now*.
- **Fuzzy Membership Functions (FMF)** - This refers to the membership functions within the fuzzy inference system of the NMS. These govern how each input is mapped to a degree of membership of a linguistic term. For example, a user's relationship with a sender of a notification might have a ranking of 7/10 (rankings will be discussed in the next point). This value will map to both the *Important* linguistic-term class and the *VIP* linguistic-term class to differing degrees. Depending on the degree of membership, rules from the fuzzy knowledge base will activate. Previously, the membership functions in the NMS were static. Similar to the knowledge base, if the output membership functions could dynamically adapt to a user's behavior, recommended delivery of notifications could be improved.

The features outlined above serve as experiments for testing the possibility of improved performance added through use of a synthetic dataset. The FKB and FMF both have input parameters that can be tuned to adapt notification delivery recommendation. The experiments are split as follows:

1) **Experiment 1** - This involves using Particle Swarm Optimization (PSO) to search for the permutation of parameters for the FKB that provide an optimal work-play-split.
2) **Experiment 2** - This involves using PSO to search for the permutation of parameters for the output FMF that provides an optimal work-play-split.

The work-play-split assumption is defined as an ideal percentage split of recommended notification deliveries between

Work

Delivery Method	Now	Break	Period	Later	Much Later
Notification %	40	30	20	5	5

Social

Delivery Method	Now	Break	Period	Later	Much Later
Notification %	5	10	20	30	35

TABLE VI: Work-play-split assumption

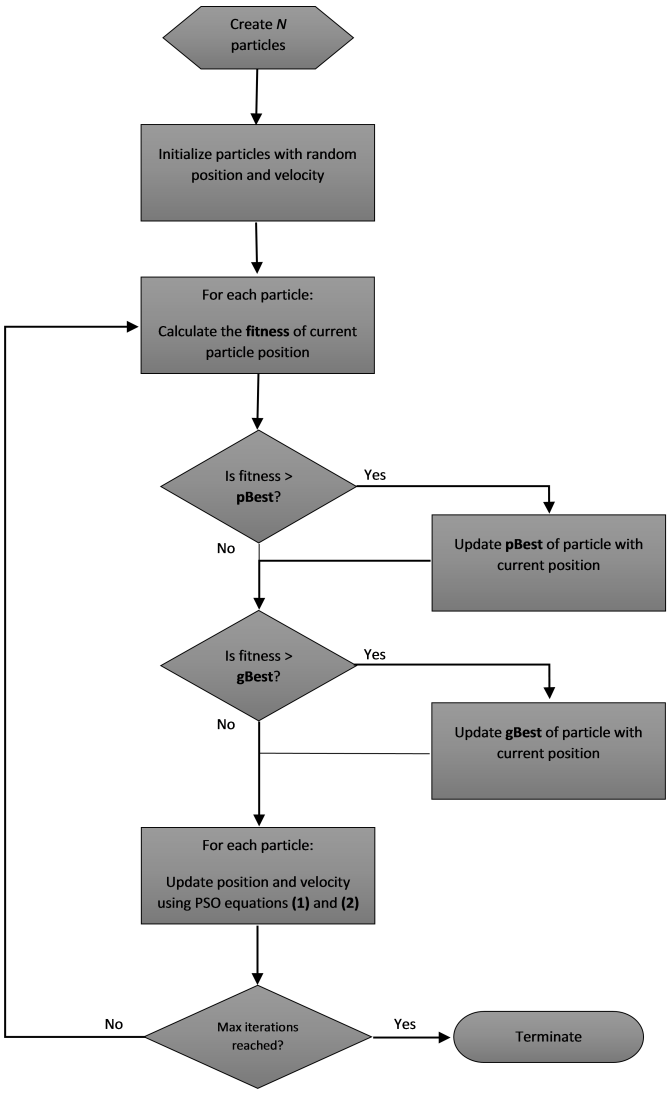

Fig. 1: Generic PSO algorithm.

the hours of 9am and 5pm for notification subjects of *work* and *social* (Table VI).

B. Particle Swarm Optimization

PSO [25] is a population based search algorithm bioinspired by the movement taken by flocks of birds when searching for food. The algorithm consists of initializing a number of particles with a random position (solution) in the search space and a random velocity. Each particle continuously updates its own position and velocity based on its own personal best position (*pBest*) and a global best position (*gBest*) shared between all particles. In this manner, particles fly through the search space and converge towards areas of the search space with better solutions.

PSO is used in Experiments 1 and 2 to search for permutations of input parameters which, when applied to the NMS during processing of the synthetic data-set, yields a split of recommended deliveries closest to the work-play-split. PSO was chosen due to its popularity and success in a wide range of applications, including that of neural network training and fuzzy logic control [26].

Each experiment implements the generic PSO algorithm which is illustrated in Fig.1 PSO equations (1) and (2) referenced in Fig.1 refer to the two update equations used in the algorithm. Equation (1) updates a particles velocity, the rate at which a particle moves towards the *pBest* and *gBest* solution - rand() is a uniform random number between 0 and 1, c1 and c2 are learning factors. Equation (2) then updates the particles current position using the updated velocity.

$$v[] = v[] + c1 * rand() * (pBest[] - present[]) \\ +c2 * rand()"(gBest[] - present[]) \quad (1)$$

$$present[] = present[] + v[] \quad (2)$$

The algorithm uses a fitness function to evaluate the performance of each particles current position. Based on this value, *pBest* and *gBest* are updated. There is no generic fitness function to apply in this instance as it is unique to the problem at hand. The fitness function implemented in the following experiments attempts to minimize the error between the work-play-split generated by the NMS and the synthetic dataset when using a particles solution and the assumed ideal work-play-split (Table VI). Hence the fitness of each particle is an array of error values and *pBest* and *gBest* are updated based on solutions with reduced error.

VI. Experiment 1 & 2

These experiments involve tuning the Fuzzy Inference System (FIS), which is made up of Fuzzy Membership Functions (FMF) and the Fuzzy Knowledge Base (FKB), of the NMS toward delivering notifications so they match the work-play-split. PSO algorithms have already shown to be successful in improving the performance FIS's [27] [28] [29].

A. Procedure

The FKB in the NMS contains rules which define how a notification is to be delivered based on the sender, subject and app importance. The following are examples of rules in the FKB:

If SenderInput is NIP and SubjectInput is NIP and AppInput is NIP then AlertOutput is MUCHLATER.

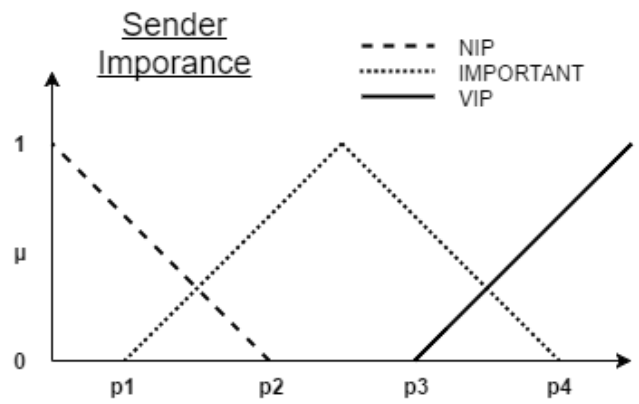

Fig. 2: Fuzzy membership function of Sender importance.

If SenderInput is IMPORTANT and SubjectInput is IMPORTANT and AppInput is NIP then AlertOutput is VERYSOON.

In experiment 1 the parameters to be tuned by the PSO algorithm are the linguistic variables of the *AlertOutput* class comprised of the set {*NOW*, *SOON*, *VERYSOON*, *LATER*, *MUCHLATER*}. By tuning these parameters the NMS is learning which set of rules result in notification deliveries matching the work-play-split.

In experiment 2 the parameters to be tuned are the fuzzy membership functions. These are used to map crisp input values signifying the *contextual importance* of the sender, subject and app to fuzzy linguistic terms of *NIP*, *IMPORTANT*, *VIP* as well as to map resultant fuzzy linguistic terms to crisp outputs. An example of the fuzzy membership function for sender importance is illustrated in Fig. 2 with the possible parameters to be tuned p_n. By tuning the membership function parameters the NMS can learn which values are necessary to apply to achieve notification deliveries in the work-play-split. Previously, these membership functions were created using heuristic knowledge, which was a limitation stated in the paper introducing the NMS [16], as it introduced possible bias into the system. Using PSO to train the membership functions is one such solution to this problem.

B. Results

The PSO algorithm was run across all users in the synthetic dataset who had "work" and "social" notifications. This limited the number of users to 18. The number of particles used for in the PSO algorithm was 15, the number of iterations was 10 and the C1 and C2 learning factors were both set to 2 (the default values). The notification *Subject* term ratings (which were not derived from the F&F dataset) were kept fixed for both experiments with "work" rated 7 and "social" rated 8 (both out of 10) for importance to the user.

Tables VII, VIII and IX illustrate the results of the PSO algorithm for experiment 1. The PSO algorithm was first run over each user in the synthetic dataset and the resulting solutions were recorded. Each solution was then applied to the NMS knowledge base and the original *in-the-wild* datasets were processed by the NMS and a recommended delivery for each notification in the dataset was made. To evaluate the results the Strict Correctness Ratio (SCR) is used (Eq.3). The SCR is simply the percentage of correctly classified notification deliveries. As the *in-the-wild* dataset has the users preferred delivery method, the SCR can be calculated.

$$\frac{correct_{total}}{notifications_{total}} \times 100 \quad (3)$$

The results of applying the PSO generated Fuzzy Knowledge base rules to the NMS are illustrated in Fig. 3.

In experiment 1, two PSO solutions taken from two users of the synthetic dataset were compared. The first solution, *Least Error*, was the parameter solution which provided the closest matching delivery split of notifications to the work-play-split. The second solution, *Best Performance*, was the parameter solution which provided the best performance when applied to the NMS with real *in-the-wild* notification data. As can be seen from Table VII and Table VIII the *Least Error* solution is closer to the work-play-split than the *Best Performance* solution (this is verified in Table IX which has calculated the accumulated percentage difference between the work-play-split and the split of each PSO solution). However, notice in Fig. 3 that the *Least Error* solution also gives a poorer performance than the *Best Performance* solution. This is due to the notification data being used to test the PSO algorithms (the original *in-the-wild* dataset) not conforming to the work-play-split. In fact, the actual split is quite different (illustrated in Tables VII and VIII). The *Best Performance* solution is

	Subject - Work				
	Now	Break	Period	Later	Much Later
Best Performance	39%	0%	21%	40%	0%
Least Error	43%	39%	9%	9%	0%
	Subject - Work				
Work-play split	40%	30%	20%	5%	5%
Actual split	22%	0%	78%	0%	0%

TABLE VII: Exp.1 PSO parameter performance compared with assumed and actual splits for subject "work".

	Subject - Social				
	Now	Break	Period	Later	Much Later
Best Performance	35%	0%	33%	32%	0%
Least Error	40%	14%	12%	30%	4%
	Subject - Social				
Work-play split	5%	10%	20%	30%	35%
Actual split	0%	20%	80%	0%	0%

TABLE VIII: Exp.1 PSO parameter performance compared with assumed and actual splits for subject "social".

	Error work-play-split	Error actual split
Best Performance	162%	248%
Least Error	113%	285%

TABLE IX: Exp.1 total accumulated percentage error associated with both parameter sets.

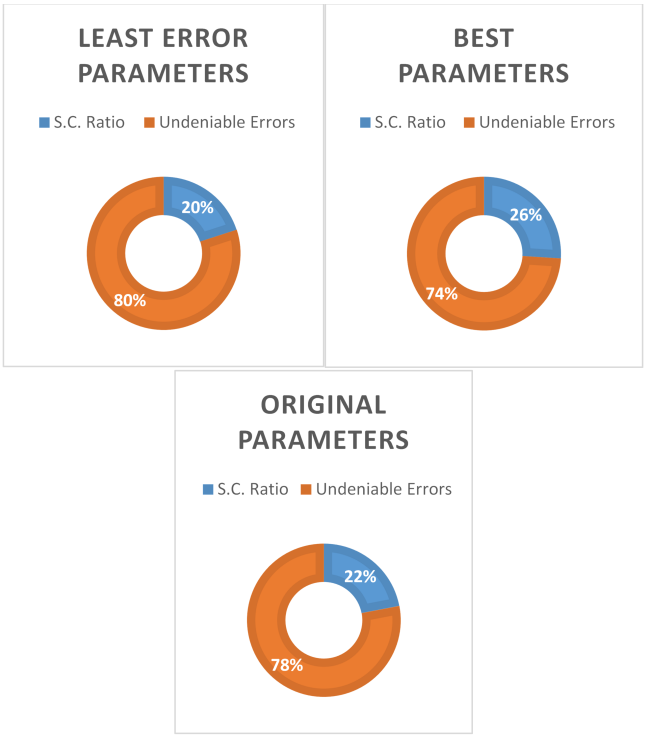

Fig. 3: Exp 1. evaluation of PSO and synthetic dataset performance.

actual closer to this split than the work-play-split which we attempted to reciprocate. Hence it gave a better performance. Note also that by applying a PSO solution closer to the actual notification split, the NMS performed better than it did with its original parameters. The SCR increased by 4%.

The PSO solutions were tested similarly with the second user of the *in-the-wild* notification dataset but the results were less promising. Both solutions resulted in poorer notification deliveries. Hence, it can be concluded that in order for the work-play-split to be effective, it needs to be personalized to the user. However, if there is no data on hand, the general assumption split may provide adequate performance.

Experiment 2 is similar to experiment 1 however the parameters being generated by the PSO algorithm now control the Fuzzy Membership Functions while the Fuzzy Knowledge Base rules remain fixed. By analyzing the results in the same manner as experiment 1, the two methods of tuning the NMS can be compared. In Tables X and XI the *Least Error* and *Best Performance* solutions generated by the PSO algorithm are again compared with the work-play-split and the actual split found from the *in-the-wild* dataset. Table XII illustrates that the *Least Error* solution is closer to the work-play-split (as it has less accumulated percentage split error associated with this split) while the *Best Performance* solution is closer to the actual split. Once again this is reflected in the results as the *Least Error* solution performs poorly against the *Best Performance* solution.The reason for this, as stated in experiment 1, would be due to the actual split better representing the user

	Subject - Work				
	Now	Break	Period	Later	Much Later
Best Performance	34%	0%	13%	0%	53%
Least Error	11%	0%	12%	27%	50%

	Subject - Work				
Work-play split	40%	30%	20%	5%	5%
Actual split	22%	0%	78%	0%	0%

TABLE X: Exp.2 PSO parameter performance compared with assumed and actual splits for subject ”work”.

	Subject - Social				
	Now	Break	Period	Later	Much Later
Best Performance	28%	0%	24%	0%	48%
Least Error	13%	0%	11%	30%	46%

	Subject - Social				
Work-play split	5%	10%	20%	30%	35%
Actual split	0%	20%	80%	0%	0%

TABLE XI: Exp.2 PSO parameter performance compared with assumed and actual splits for subject ”social”.

	Error work-play-split	Error actual split
Best Performance	176%	282%
Least Error	172%	332%

TABLE XII: Exp.2 total accumulated percentage error associated with both parameter sets.

of the *in-the-wild* dataset as opposed to the general work-play-split. Note however, that both solutions had improved SCR's over the original parameters. From these results, it can be concluded that personalizing the Fuzzy Membership Functions with respect to the user can improve the performance of a NMS and also that synthetic data can be used to generate notification usage rules which have the potential to improve performance.

VII. CONCLUSION

This paper explored the limitations of obtaining rich, contextually relevant, mobile notification datasets by examining the ethical implications, the heterogeneous feature space and the added research effort involved in procuring such datasets. An existing *in-the-wild* dataset was henceforth analyzed in terms of its features, ease of obtainment, privacy and quality. An appraisal of the current features being collected within the research domain of Notification and Interruption Management was then performed with a particular focus on the intrusion of privacy to users and the ethical issues it raises. As a possible alternative to mobile notification collection, a means of generating a synthetic dataset derived from real-world mobile usage behavior was expressed. The process of deriving the synthetic dataset and the limitations it imposed on the data was also explored. To evaluate the value of the synthetic dataset, two experiments were performed on an existing *in-the-wild* mobile notification dataset and NMS in order to improve its performance. Particle Swarm Optimization was used with the synthetic dataset to identify NMS input parameters which fit a user's assumed preferred notification delivery breakdown.

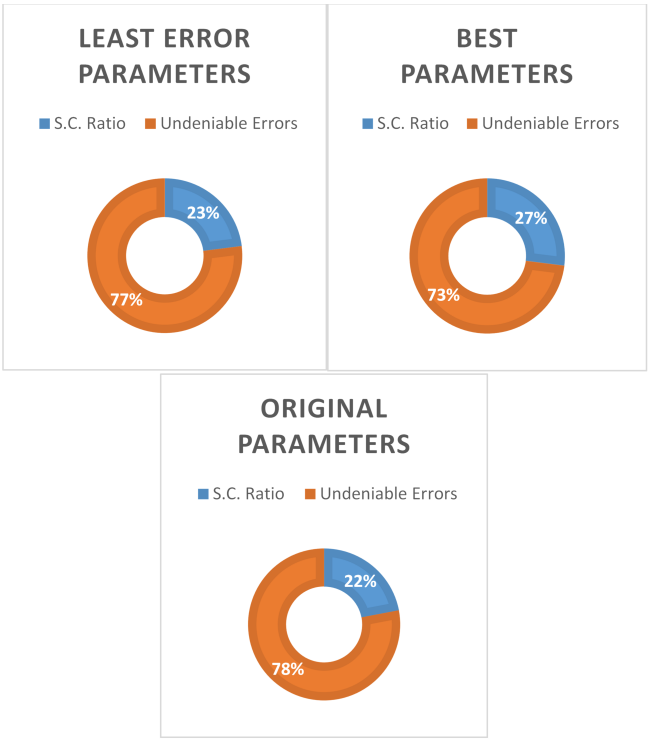

Fig. 4: Exp. 2 evaluation of PSO and synthetic dataset performance.

The results shown illustrate that the synthetic dataset could contribute to improving the performance of a NMS, particularly in situations where personal mobile notification data was limited.

References

[1] M. Pielot, K. Church, and R. de Oliveira, "An in-situ study of mobile phone notifications," in *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services*, MobileHCI '14, (New York, NY, USA), pp. 233–242, ACM, 2014.

[2] AIO Software Technology, "Notification manager." http://play.google.com/store/apps/details?id=com.imoblife.quietnotification_plugin.

[3] G. Dalakishvili, "Notifications off." http://play.google.com/store/apps/details?id=com.aboutmycode.NotificationsOff&hl=en.

[4] A. Mehrotra, M. Musolesi, R. Hendley, and V. Pejovic, "Designing content-driven intelligent notification mechanisms for mobile applications," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, (New York, NY, USA), pp. 813–824, ACM, 2015.

[5] N. Kern and B. Schiele, "Context-aware notification for wearable computing," in *Proceedings of the 7th IEEE International Symposium on Wearable Computers (ISWC03)*, pp. 223–230.

[6] T. Okoshi, J. Ramos, H. Nozaki, J. Nakazawa, A. K. Dey, and H. Tokuda, "Reducing users' perceived mental effort due to interruptive notifications in multi-device mobile environments," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '15, (New York, NY, USA), pp. 475–486, ACM, 2015.

[7] Y. Qin, T. Bhattacharya, L. Kulik, and J. Bailey, "A context-aware do-not-disturb service for mobile devices," in *Proceedings of the 13th International Conference on Mobile and Ubiquitous Multimedia*, MUM '14, (New York, NY, USA), pp. 236–239, ACM, 2014.

[8] J. Ho and S. S. Intille, "Using context-aware computing to reduce the perceived burden of interruptions from mobile devices," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, (New York, NY, USA), pp. 909–918, ACM, 2005.

[9] V. Pejovic and M. Musolesi, "Interruptme: Designing intelligent prompting mechanisms for pervasive applications," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, (New York, NY, USA), pp. 897–908, ACM, 2014.

[10] J. Ho and S. S. Intille, "Using context-aware computing to reduce the perceived burden of interruptions from mobile devices," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, (New York, NY, USA), pp. 909–918, ACM, 2005.

[11] E. Horvitz, P. Koch, and J. Apacible, "Busybody: Creating and fielding personalized models of the cost of interruption," in *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work*, CSCW '04, (New York, NY, USA), pp. 507–510, ACM, 2004.

[12] J. E. Fischer, C. Greenhalgh, and S. Benford, "Investigating episodes of mobile phone activity as indicators of opportune moments to deliver notifications," in *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, (New York, NY, USA), pp. 181–190, ACM, 2011.

[13] A. Mehrotra, V. Pejovic, J. Vermeulen, R. Hendley, and M. Musolesi, "My phone and me: Understanding peoples receptivity to mobile notifications," 2016.

[14] F. Schulze and G. Groh, "Conversational context helps improve mobile notification management," in *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '16, (New York, NY, USA), pp. 518–528, ACM, 2016.

[15] A. Mehrotra, R. Hendley, and M. Musolesi, "Prefminer: Mining user's preferences for intelligent mobile notification management," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '16, (New York, NY, USA), pp. 1223–1234, ACM, 2016.

[16] K. Fraser, B. Yousuf, and O. Conlan, "A context-aware, info-bead and fuzzy inference approach to notification management," in *2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference sponsored by IEEE (Published by IEEE Xplore) (IEEE UEMCON 2016)*, (New York, USA), Oct. 2016.

[17] N. Aharony, W. Pan, C. Ip, I. Khayal, and A. Pentland, "Social fmri: Investigating and shaping social mechanisms in the real world," *Pervasive Mob. Comput.*, vol. 7, pp. 643–659, Dec. 2011.

[18] N. Lathia, K. K. Rachuri, C. Mascolo, and P. J. Rentfrow, "Contextual dissonance: Design bias in sensor-based experience sampling methods," in *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 183–192, ACM, 2013.

[19] J. Wiese, T. S. Saponas, and A. Brush, "Phoneprioception: enabling mobile phones to infer where they are kept," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2157–2166, ACM, 2013.

[20] G. H. Ter Hofte, "Xensible interruptions from your mobile phone," in *Proceedings of the 9th international conference on Human computer interaction with mobile devices and services*, pp. 178–181, ACM, 2007.

[21] A. Mendez-Vazquez, A. Helal, and D. Cook, "Simulating events to generate synthetic data for pervasive spaces," Citeseer.

[22] F. Corno, L. De Russis, and T. Montanaro, "A context and user aware smart notification system," in *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on*, pp. 645–651, IEEE, 2015.

[23] N. Eagle and A. S. Pentland, "Reality mining: sensing complex social systems," *Personal and ubiquitous computing*, vol. 10, no. 4, pp. 255–268, 2006.

[24] L. L. Pipino, Y. W. Lee, and R. Y. Wang, "Data quality assessment," *Communications of the ACM*, vol. 45, no. 4, pp. 211–218, 2002.

[25] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of machine learning*, pp. 760–766, Springer, 2011.

[26] K. E. Parsopoulos, *Particle swarm optimization and intelligence: advances and applications: advances and applications*. IGI Global, 2010.

[27] H.-M. Feng, "Particle swarm optimization learning fuzzy systems design," in *Third International Conference on Information Technology and Applications (ICITA'05)*, vol. 1, pp. 363–366, IEEE, 2005.

[28] A. A. A. Esmin, "Generating fuzzy rules from examples using the particle swarm optimization algorithm," in *7th International Conference on Hybrid Intelligent Systems (HIS 2007)*, pp. 340–343, IEEE, 2007.

[29] C.-C. Chen, "A pso-based method for extracting fuzzy rules directly from numerical data," *Cybernetics and Systems: An International Journal*, vol. 37, no. 7, pp. 707–723, 2006.