Towards In Time Music Mood-Mapping for Drivers: A Novel Approach

Arun Sai Krishnan[1,2], Xiping Hu[2], Jun-qi Deng[3], Li Zhou[4], Edith C.-H. Ngai[5], Xitong Li[6], Victor C.M. Leung[2], Yu-kwong Kwok[3]

[1] National Institute of Technology, India | [2] The University of British Columbia, Canada | [3] The University of Hong Kong, Hong Kong | [4] National University of Defense Technology, China | [5] Uppsala University, Sweden | [6] HEC Paris, France
arunsaikrish@gmail.com, {xipingh, vleung}@ece.ubc.ca, {jqdeng, ykwok}@eee.hku.hk, edith.ngai@it.uu.se, lix@hec.fr

ABSTRACT
Road safety is a huge concern due to the large number of fatalities and injuries caused by road accidents. Research has shown that fatigue can adversely affect driving performance and increase risk of road accidents. It has been shown that driving performance is enhanced by stress-relieving music which thereby promotes safer driving. Context-aware music delivery systems promote safer driving through intelligent music recommendations based on contextual knowledge. Two key aspects of situation-aware music delivery are effectiveness and efficiency of music recommendation. Efficiency is a critical aspect in real-time context based music recommendation as the music delivery system should quickly sense any change in the situation and deliver suitable music before the sensed context-data becomes obsolete. We focus on the efficiency of situation-aware music delivery systems in this paper. Music mood-mapping is a process which helps in understanding the mood of a song and is hence used in situation-aware music recommendation systems. This process requires a large processing time due to the complex calculations and large sizes of music files involved. Hence, optimizing this process is the key to improving the efficiency of context-aware music delivery systems. Here, we propose a novel cloud and crowd-sensing based approach to considerably optimize the efficiency of situation-aware music delivery systems.

Categories and Subject Descriptors
C.2.4 [**Computer Systems Organization**]: COMPUTER COMMUNICATION NETWORKS – Distributed Systems – Distributed Applications.

General Terms
Design, Performance, Human Factors, Algorithms

Keywords
Crowd-sensing; Cloud; Context-aware; Vehicular Sensor Application; Music Recommendation; Mood-Mapping; Offloading; Music Matching

DIVANet'15, November 02-06, 2015, Cancun, Mexico
© 2015 ACM. ISBN 978-1-4503-3760-1/15/11…$15.00
DOI: http://dx.doi.org/10.1145/2815347.2815352

1. INTRODUCTION
The total number of road traffic deaths worldwide is unacceptably high at 1.2 million per year according to statistics published in the world report on road traffic injury prevention by the World Health Organization (WHO) [1]. In addition to the reported deaths, another 50 million people are either disabled or injured on the roads every year. Analysis on the data on accidents in Europe reveal that around 10-20% of all road accidents are caused by reduced alertness of drivers due to factors like negative emotions or fatigue [2]. Previous research shows that listening to suitable music while driving could ameliorate the driver's mood and result in the driver feeling relaxed [3], which could enhance driving performance substantially.

Context-aware music recommendation for drivers has been achieved in our previous research work [4, 5] by leveraging the computing capabilities of smartphones to coordinate and analyze sensing data from various sources [6] to deliver suitable music to the drivers. This application can be implemented based on Vehicular Ad hoc Networks [7], which are considered as a benign and cooperative environment to support the cloud-based-platform.

There are two key aspects in enhancing driving performance through effective music delivery. Firstly, the right music should be delivered to help in easing fatigue and negative emotions like anger which cause poor driving. For instance, soothing music should be played when the driver is angry and high-energy music should be delivered when the driver is tired. This aspect focuses on the effectiveness of the music delivery and can be gauged by the effect of the recommended song in improving the driver's mood and enhancing driving performance. Secondly, suitable music should be delivered in time to aid in the process of improving driving performance. This aspect focuses on the efficiency of the music recommendation process and it is very important as efficiency is a critical factor in real-time situation-aware music delivery systems as suitable music needs to be delivered in a timely manner before the context which is sensed becomes obsolete. The music recommendation must be timely in order to ensure continuity in delivery of suitable music. SAfeDJ [4], an application which brings context-awareness to music playlists while driving, illustrates the importance of the two aspects mentioned above in effective music recommendation for drivers. Our work in this paper is focused on improving the efficiency of situation-aware music delivery systems through novel techniques.

Music recommendation systems extensively utilize a process called music mood-mapping which helps in understanding the nature and mood of a song. Context-aware music delivery systems determine the context and mood of a driver through various sensors and make use of the mood-mapping process to identify the suitable song to play according to the current context of the driver. Music mood-mapping is a computationally intensive process and mapping the mood of every song in the music library of a driver's smartphone requires an enormous amount of processing time on a smartphone. For example, the music mood-mapping in our former work SAfeDJ [4] takes 78 seconds per song on an average if performed on a smartphone. Extrapolate this to a library of 200 songs and the processing time turns out to be around 4.5 hours. The alternative solution to this would be to perform the music mood-mapping in a cloud system with a higher computing power. However, this solution is not very feasible due to the large network overhead and substantial latency involved in transferring large music files to the cloud. Thus, optimizing the time taken in music mood-mapping is a critical problem to solve in order to ensure timely delivery of suitable music to drivers and improve the efficiency of context-aware music recommendation systems.

In this paper, we present a novel cloud and crowd-sensing based approach which significantly reduces the time involved in the music mood-mapping process and considerably improves the efficiency of situation-aware music delivery systems. In the traditional methods, the music mood-mapping is normally performed on every song in the music library of a driver's smartphone or related account in cloud [8].

However, a single song may be present in the music libraries of multiple drivers. This results in a lot of redundant computation as the music mood-mapping for a single song happens individually on multiple smartphones. Our approach eliminates these redundant computations through a novel cloud-based crowd-sensing framework. Every time music mood-mapping is performed on all the songs in the music library of a smartphone, the smartphone queries a centralized database through a cloud-server. The centralized cloud-database maintains a complete list of songs whose music moods have already been mapped by individual smartphones. The cloud-server then responds with the mood-mapping of all the songs in the smartphone's music library whose moods have already been mapped. Subsequently, the smartphone performs music mood-mapping on the rest of the songs in its music library which have not been mapped in the centralized cloud-database and updates the centralized cloud-database with the newly computed mood-mappings of these songs.

Situation-aware music delivery is generally enabled through smartphones. Although modern smartphones are equipped with increasingly powerful processors and very good computing powers, computationally intensive processes such as music-mood-mapping still take a considerably long time on smartphones. An alternative approach is to offload the computationally intensive music mapping process to a much more powerful cloud system which would drastically reduce the time taken in music mood-mapping and battery consumption of the smartphone [9-13]. However, varying network signal strengths in dynamic driving environments, network overheads and latency in transferring large music files to the cloud are the limitations of this alternative approach.

Our approach amalgamates the advantages of both the approaches presented above through the use of an intelligent offloading system which decides whether the computation should be performed locally on the smartphone or offloaded to the cloud for optimal performance of the music-recommendation system. The intelligent offloading system takes into consideration various factors such as signal strength, type of network, size of music file and duration of the song while deciding whether or not to offload the music mood-mapping to the cloud.

There may exist different versions of the same song with slightly varying meta-data such as title, album, artist and duration. However, the fundamental nature and mood of these slightly varying versions are the same. Hence, mood-mapping of slightly different versions of the same song is redundant. In order to eliminate this redundant mapping, we need to tackle the challenge of identifying slightly different versions of a song pre-mapped on the cloud. We address this challenge through a novel meta-data and music feature based similarity detection system which determines similarity between two songs by analyzing the meta-data and music features of the songs which are being compared.

The detailed design of our approach is described in Section 2. The implementation details are described in Section 3 followed by experimental results in Section 4. Related work to the problems addressed in our work is presented in Section 5.

2. DESIGN

In this section, we describe the design of our unique framework which enables substantial time optimization in music mood-mapping and improved efficiency in situation-aware music delivery for drivers by leveraging the power of the cloud and crowd-sensing.

2.1 Architecture

The system architecture of our framework is presented in Figure 1. It consists of the cloud tier and the mobile tier.

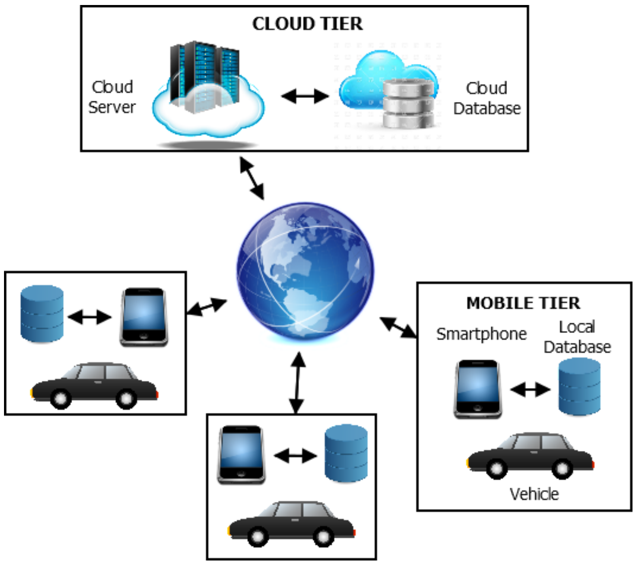


Figure 1. System Architecture

2.1.1 Cloud Tier

The cloud tier consists of a cloud server and a centralized cloud database. It also hosts the meta-data and music feature based song similarity detection system. The cloud server receives requests from smartphones to check for the music mood-mappings of songs. It then queries the cloud database through the song similarity detection system which returns the music mood-mappings of pre-mapped songs to the smartphone which sent the request. The centralized cloud database enables crowd-sourcing by storing the music mood-mappings of songs from various smartphones. Once the music mood-mapping of a particular song is performed on a smartphone, it is updated in the centralized cloud database. Thus, redundant computation of music mood for the same song across multiple smartphones is eliminated.

2.1.2 *Mobile Tier*

The mobile tier consists of a smartphone and a local database. It also hosts the intelligent offloading system which optimally determines whether a new song's music mood-mapping should be performed locally on the smartphone or offloaded to the cloud. Whenever a smartphone has to perform music mood-mapping on all the songs in its music library, it first checks if they are already mapped in its local database. Then, it sends a request to the cloud server with the list of songs that haven't been mood-mapped yet. The cloud server responds with the mood-mappings of all the songs from the list that have already been mood-mapped in the centralized cloud database. Then, based on the decision of the intelligent offloading system, the mood-mapping of the songs which haven't been mood-mapped yet in the centralized cloud database is either performed locally in the smartphone or offloaded to the cloud. The newly calculated mood-mappings are then updated in the local and cloud databases. This process helps in achieving effective and efficient crowd-sensing of music mood-mapping.

We present the block diagram of our novel framework in Figure 2.

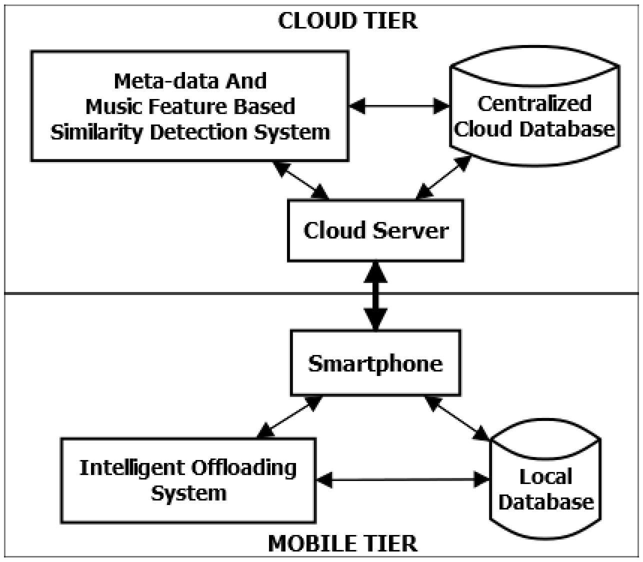

CLOUD TIER
Meta-data And Music Feature Based Similarity Detection System
Centralized Cloud Database
Cloud Server
Smartphone
Intelligent Offloading System
Local Database
MOBILE TIER

Figure 2. Block Diagram

2.2 Subsystems

Our framework consists of two critical subsystems for intelligent offloading of music mood-mapping and meta-data and music feature based similarity detection. These subsystems help in achieving improved efficiency in situation-aware music delivery systems for drivers. A detailed description of these subsystems is presented below.

2.2.1 *Intelligent Offloading*

The role of the intelligent offloading system is to determine whether the music mood-mapping of a new song is to be performed locally on a smartphone or offloaded to the cloud for improved efficiency and optimal performance of a context-aware music recommendation system for drivers. Driving situations dynamically change due to constant motion of the vehicle. Hence, the decision to offload mood-mapping or not is made based on real-time contextual data. The factors to be considered while making this decision include signal strength, type of network connection, size of music file and duration of the song to be mood-mapped.

When the smartphone is not connected to any network, there is no option of offloading the music mood-mapping and hence it should be performed locally. If the smartphone is connected to a network, the network connection could be Wi-Fi, 4G LTE, 3G or 2G. Wi-Fi and 4G LTE networks have very high data transfer speeds and hence transferring music files to the cloud through these networks takes lesser time. 3G networks have relatively slower data transfer speeds than Wi-Fi and 4G LTE networks and hence, transferring music files over a 3G network to the cloud takes a relatively longer time. However, in some cases we can offload over 3G networks as well when the time saved in the computation on the cloud is greater than the time taken to transfer the music file across the 3G network. File transfer to the cloud over 2G networks is infeasible due to very slow data transfer speeds over 2G networks. Signal strength plays an important role in the decision making process as well. If the signal is too weak, data transfer is slow and unreliable even if the device is connected to a Wi-Fi or 4G LTE network.

The size of the music file is also an important factor in the decision making process. Transfer of very large music files to the cloud is infeasible due to the large network overhead and increased time of transfer involved. Another reason why the file size is an important factor is the cost of data transfer to the cloud. Smartphone users generally have limited data plans from their network carriers. If the data usage limit is exceeded, the users are charged additionally by their network carriers and it can get very expensive. Hence, the user's data usage limit and the data usage settings should be considered while deciding if music mood-mapping should be offloaded to the cloud.

Music mood-mapping generally involves analysis of spectral music features which are obtained through computationally intensive processes such as Fast Fourier Transform (FFT). It has been observed that the time taken to perform these computations are proportional to the length of the song. Thus, shorter songs take lesser time to mood-map while longer songs take more time to mood-map. Hence, the music mood-mapping of very short songs can be done locally on the smartphone irrespective of the type of network connection to avoid the network overhead and time-delay in transferring the file to the cloud.

Different context-aware music delivery systems have different methods of music mood-mapping. They may vary widely in the kind of operations performed on the music file and computational intensity of the operations performed. Thus, there is no single solution or approach to decision making in the intelligent offload system which optimally fits in all context-aware music recommendation systems.

The logic for the decision making process to offload mood-mapping or not varies from one situation-aware music delivery system to another and is obtained through extensive experimentation on the particular music recommendation system in question. An example of this is illustrated in Section 3, where the integration of our novel framework with a situation-aware music delivery system named SAfeDJ [4] is discussed.

2.2.2 *Similarity Detection*

The same song might be present in slightly different formats which vary slightly in meta-data like title, album, artist and duration. However, the fundamental nature and mood of these different versions of the same song is the same. Hence, we need to identify if a song is similar to an existing song pre-mapped in the centralized cloud database in order to avoid redundant mood-mapping of slightly different versions of the same song. We use a novel meta-data and music feature based similarity detection system in our approach to solve this problem.

Meta-data of music files include data such as the title, album, artist and duration of a song. This data provides valuable information regarding a song which helps in identification of similar songs and is hence an important component in our system. To improve the accuracy of our system and make it more comprehensive, we also take music features from the music mood-mapping process of a situation-aware music delivery system into consideration in the similarity detection process. Different context-aware music recommendation systems have different music mood-mapping processes. Different music mood-mapping processes extract different features from the song and some of these features help in identifying similar songs and are hence an integral component of our similarity detection system. The various factors to be considered in the similarity detection system are illustrated in Figure 3.

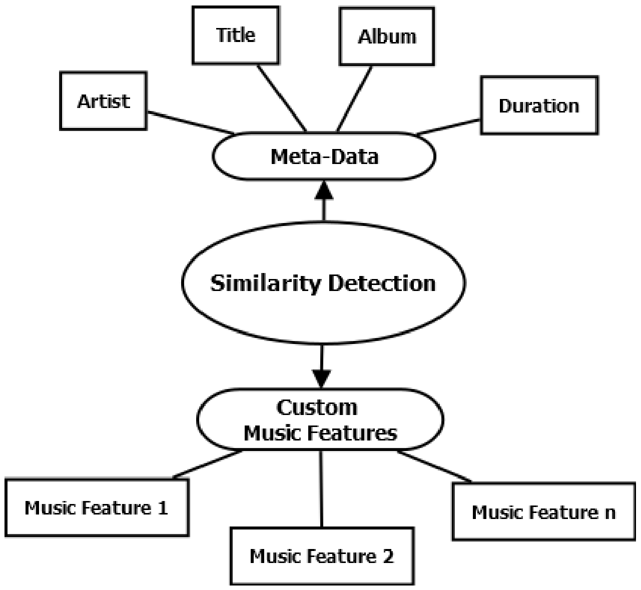


Figure 3. Factors to be Included in Similarity Detection

We use a novel weighted scoring system to determine similarity between songs in our approach. The different factors mentioned above are assigned weights according to their level of importance in determining similarity between two slightly different versions of the same song. The factors can be either strings (title, album and artist) or numeric values (duration and custom music features). As illustrated in Figure 4, strings are compared for similarity using substring detection and sequence matching algorithms while numeric values are compared using the absolute difference between the values of the factors of the songs being compared.

Substring detection and sequence matching are the two techniques we employ to gauge the similarity of two string valued factors. We learn from our studies and observations on meta-data of slightly different versions of the same song that the meta-data of these slightly varying versions are similar to each other and only differ because of small differences in spelling or additional words and characters in the meta-data. Hence, substring detection and sequence matching are good indicators of similarity between the string data in this case. Another technique that can be used is the Levenshtein distance between two strings which measures the difference between two strings by calculating the minimum number of single-character edits required to convert one string to the other.

In our scoring system, half the weightage of the string valued factor being compared is assigned to the substring detection and the other half is assigned to sequence matching. If one of the two strings being compared is a substring of the other, it is a strong indicator of similarity. Hence, a score equal to half the weightage of the string valued factor being compared is awarded if one of the strings being compared is a substring of the other.

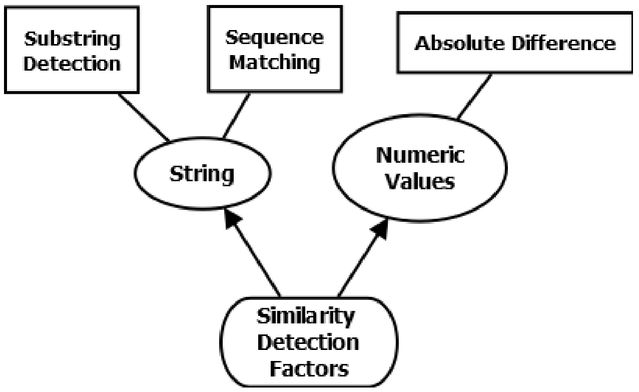


Figure 4. Methods to Measure Similarity

We perform sequence matching using the Ratcliff and Obershelp algorithm for pattern recognition [14]. This is also known as the gestalt approach to pattern matching. Here, the similarity between two strings is calculated as the ratio of number of matching characters to the total number of characters in both the strings. The matching character are those characters present in the longest common subsequence. The same idea is then recursively applied to the parts of the sequence on either side of the matching subsequence. This method yields good results which help us in identifying similarity between string valued factors accurately. In our scoring system, half the weightage of the string valued factor being compared is assigned to the result of this sequence matching process. The ratio obtained from applying the Ratcliff and Obershelp algorithm on the two strings being compared is multiplied with half the weightage of the factor being compared in order to obtain the contribution of the sequence matching process to the overall similarity score for the two songs being compared.

Numerically measured values such as duration are observed to be strong indicators of similarity between songs. For instance, song files of the same song obtained from different sources will have almost the same duration if not exactly the same. Factors such as duration which are measured by numeric values are compared using the absolute difference between the values of the factors of the two songs being compared. If the absolute difference is zero, then they exactly match and hence, it contributes the entire weightage of the factor being compared to the final similarity score. We divide the results into bands based on the absolute difference. For instance, if the absolute difference is less than ten percent of the value of the song which has already been mapped, it could contribute three-fourths of the weightage of the factor being compared to the overall similarity score and if it is between ten and twenty percent, it can contribute half the weightage of the factor being compared to the overall similarity score. The limit values of these bands and their respective contribution to the score can be varied according to the

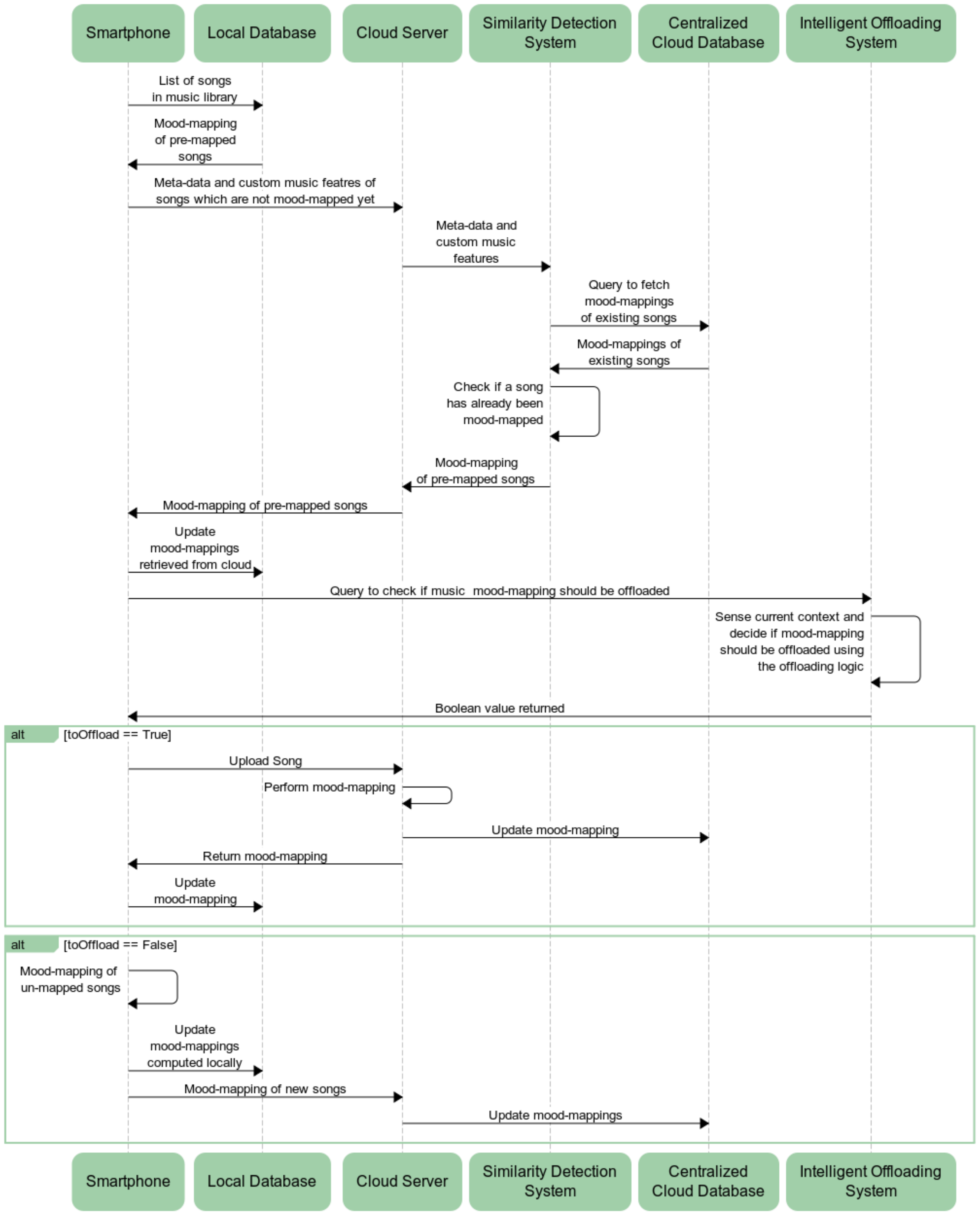

Smartphone
Local Database
Cloud Server
Similarity Detection System
Centralized Cloud Database
Intelligent Offloading System
List of songs in music library
Mood-mapping of pre-mapped songs
Meta-data and custom music featres of songs which are not mood-mapped yet
Meta-data and custom music features
Query to fetch mood-mappings of existing songs
Mood-mappings of existing songs
Check if a song has already been mood-mapped
Mood-mapping of pre-mapped songs
Mood-mapping of pre-mapped songs
Update mood-mappings retrieved from cloud
Query to check if music mood-mapping should be offloaded
Sense current context and decide if mood-mapping should be offloaded using the offloading logic
Boolean value returned
alt
[toOffload == True]
Upload Song
Perform mood-mapping
Update mood-mapping
Return mood-mapping
Update mood-mapping
alt
[toOffload == False]
Mood-mapping of un-mapped songs
Update mood-mappings computed locally
Mood-mapping of new songs
Update mood-mappings
Smartphone
Local Database
Cloud Server
Similarity Detection System
Centralized Cloud Database
Intelligent Offloading System

Figure 5. UML Sequence Diagram

factor being compared and the scenario of usage based on extensive experimentation by varying values to find the optimal limits.

An artist can have multiple albums and one album can have many songs. The designer of the situation-aware music delivery system should take into consideration such facts while assigning weights to the various factors. Hence, the title of the song should have higher weightage than the album and the album should have more weightage than the artist. The system designer may choose to include any number of custom features from the music mood-mapping process being used in the similarity detection process by assigning suitable weights to the same.

The contributions obtained from comparing each of the factors are summed up to obtain the total similarity score. We observe based on experimentation that the similarity scores obtained through this method for different versions of the same song are considerably higher than non-similar songs. After extensive experimentation, the system designer can define a threshold similarity score value. If the total similarity score of two songs is greater than the threshold value, it can be safely concluded that the two songs under comparison are slightly different versions of the same song and hence, there is no need to perform mood-mapping again as their fundamental nature and mood are the same.

Once a new song has been identified to be similar to an existing song in the centralized cloud database, the meta-data and custom music features of this new incoming song is stored in the centralized cloud database and is associated with the existing pre-mapped song. Hence, each mood-mapped song in the cloud database has a set of meta-data and custom music features of similar songs associated with it. As the number of songs being compared increases over time, the size of these associated met-data and custom music feature sets grow in size. When a new song is being compared for similarity, all the meta-data and custom music features from the associated set are compared as well. This process improves the accuracy of similarity detection. The similarity detection system becomes comprehensive and increasingly accurate with increased usage over time.

A detailed explanation of the implementation of this novel approach for similarity detection in slightly different versions of the same song is provided in Section 3, where the integration of our novel framework with a situation-aware music delivery system for drivers named SAfeDJ [4] is discussed.

The entire workflow of our novel framework is presented as a UML sequence diagram in Figure 5.

3. IMPLEMENTATION

We've integrated our framework with the SAfeDJ platform [4], which is a context-aware music recommendation system for safe driving as a part of our implementation. As a result, the time taken for music mood-mapping in SAfeDJ [4] has been reduced drastically and its efficiency has been improved considerably.

In our implementation, we used Flask: a Python based web framework to develop our cloud server and RESTful APIs. The mobile tier code was developed in native Java for Android smartphones. The local database in the smartphone was developed using SQLite and the centralized cloud database was developed using MySQL. The smartphone communicates with the cloud server over the internet through RESTful APIs developed in the cloud server.

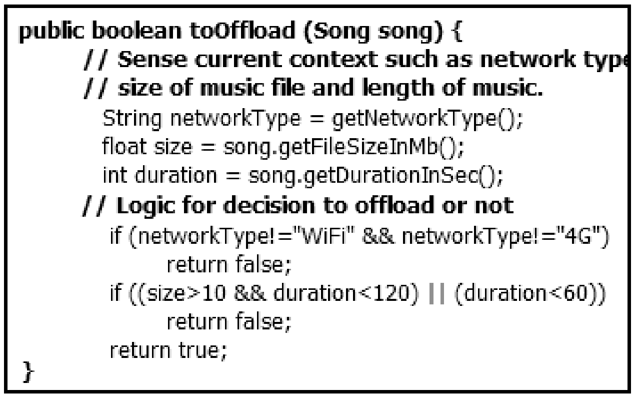
The specific implementations of the two critical subsystems of our novel approach is described in the following subsections.

3.1 Intelligent Offloading

In our implementation of the intelligent offloading system for SAfeDJ [4], we take into consideration, the type of network, music file size and duration of music while deciding if the music mood-mapping of an un-mapped song should be offloaded to the cloud. From experiments, we observed that the time of music mood-mapping in SAfeDJ is proportional to the duration of the song and not the size of the music file. Hence, we inferred that it is better to perform music mood-mapping locally for songs of short duration even if their file sizes are large. Songs of longer duration with small and average file sizes are offloaded to the cloud for mood-mapping in order to leverage the higher computing power of the cloud with small to average network overheads. Music files of large sizes are mood-mapped locally in order to avoid incurring excessive network overheads and increased data transfer time. We have also implemented an option where the drivers can completely disable offloading when they don't have an active data plan on their smartphones.

Another observation we made from our experiments was that offloading to the cloud reduced the overall mood-mapping time only when the smartphone is connected to WiFi or 4G LTE networks. This is due to the high data transfer speeds in these networks which enables quick transfer of music files to the cloud. Hence, we consider offloading mood-mapping in this case. In 3G networks, the music mood-mapping process took similar times when offloaded and computed locally. We don't offload in this case in order to avoid incurring the network overheads associated with offloading.

The sample code for our implementation of the intelligent offloading system in SAfeDJ [4] is shown in Figure 6.



```
public boolean toOffload (Song song) {
        // Sense current context such as network typ
        // size of music file and length of music.
          String networkType = getNetworkType();
          float size = song.getFileSizeInMb();
          int duration = song.getDurationInSec();
        // Logic for decision to offload or not
          if (networkType!="WiFi" && networkType!="4G")
                return false;
          if ((size>10 && duration<120) || (duration<60))
                return false;
          return true;
}
```

Figure 6. Intelligent Offload Code Snippet from SAfeDJ

3.2 Similarity Detection

For similarity detection between songs in the implementation of our framework for SAfeDJ [4], we take into consideration title, album, artist and duration as the meta-data factors. The music mood-mapping process in SAfeDJ [4] extracts two temporal features called zero-crossing rate and unit power to aid in identifying the mood of the song. These temporal features are not computationally intensive to calculate and can be computed with a short processing time of 15 seconds on an average based on our experiments. These two features are good indicators of similarity as slightly different versions of the same song tend to have similar values for these features. Hence, we use zero-crossing rate and unit power as the two custom music features in our implementation of similarity detection for SAfeDJ [4].

Our scoring system varies from scores of zero to one. If the overall similarity score is one, the same song files being compared are exactly the same. The weightage for the different factors and the entire scoring split-up is illustrated in Table 1. After extensive experimentation and testing, we arrived at a threshold value of 0.4. If the overall similarity score between two songs is greater than 0.4, the two songs being compared are very similar and hence mood-mapping need not be performed again. This system has yielded accurate comparison results in our experimentation and testing and has reduced redundant mood-mappings of slightly different versions of the same song to a large extent.

Table 1. Similarity Detection in SAfeDJ

Factor	Weigh t	Substring Detection	Sequence Matching	Absolute Diff.
Title (string)	0.20	0.10	0.10	-
Album (string)	0.16	0.08	0.08	-
Artist (string)	0.12	0.06	0.06	-
Duration (int)	0.20	-	-	< 3: 0.20 < 7: 0.15 < 10: 0.1
Z.C. Rate (float)	0.16	-	-	<5%: 0.16 <15%: 0.10 <25%:0.05
Unit Power (float)	0.16	-	-	<5%: 0.16 <15%: 0.10 <25%:0.05

4. EXPERIMENTAL RESULTS

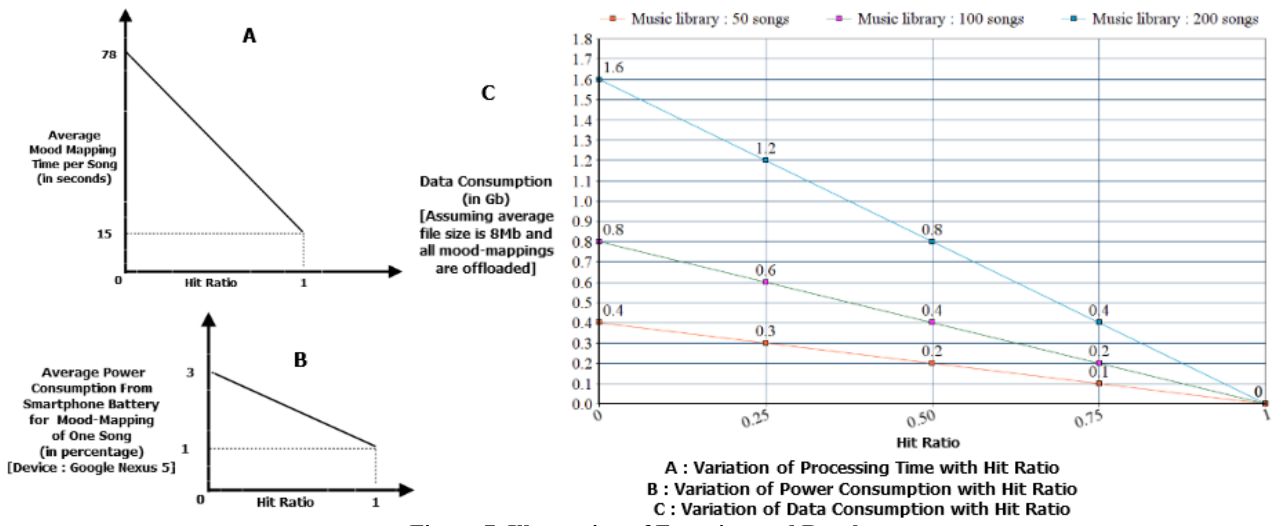
We performed some experiments to compare the time taken with and without the use of our framework in SAfeDJ. Also, we conducted our experiments in the real driving scenario with LTE network. Without the integration of our framework with SAfeDJ, it took 78 seconds on an average on a Nexus 5 smartphone respectively to perform the music mood-mapping of a song. Extrapolating this to a music library of 200 songs, the time taken to mood-map the entire library is close to 4.5 hours. All the experimental results showcased in this section are based on values obtained from tests we performed on a Nexus 5 smartphone.

Then, we tested the SAfeDJ system integrated with our framework. The time taken to query and retrieve the crowd-sensed music mood-mapping of 200 songs from the cloud was 50 minutes and the average time to detect the presence of the mood-mapping of one song and retrieve it is 15 seconds. A majority of this 15 seconds is spent in extracting the custom music features from the song for similarity detection. From these results, we clearly see that our framework is effective in drastically reducing the time taken in the music mood-mapping process.

However, the mood of all the songs in a music library may not be pre-mapped on the cloud. We define hit-ratio as the ratio of number of songs in the library whose mood-mapping is already present on the cloud to the total number of songs in the library. The hit ratio plays a crucial role in testing the performance of our system and is hence used extensively in this section to showcase our results.

Processing time, battery consumption and network overheads are the three main factors which determine the overall efficiency of a situation-aware music delivery system. We performed a number of experiments and tests on our system to study the effects and benefits of using our novel approach with respect to these three critical factors and the results we obtained have been showcased in detail in Figure 7. Part A illustrates the benefits of using our novel approach to considerably reduce mood-mapping time, while part B and part C show the advantages of using our approach in optimizing battery consumption and network overheads respectively.

Initially, the hit ratio will be low as only a few songs are mapped in the centralized cloud database. As more applications and users start using this framework, mood-mappings of more songs will be updated in the centralized cloud database and hence the hit ratio will considerably. Over a period of time the cloud database will consist of the mood-mappings of a large and comprehensive set of songs and hence the hit ratio will be very high. Once this is achieved, the entire music library of a new smartphone can be mapped in a very short span of time as compared to the time taken in music mood-mapping without the use of our novel framework.



A : Variation of Processing Time with Hit Ratio
B : Variation of Power Consumption with Hit Ratio
C : Variation of Data Consumption with Hit Ratio

Figure 7. Illustration of Experimental Results

5. RELATED WORK

Cloud assisted computing for mobile applications on vehicles [15] can enable efficient delivery of music mood-mapping for drivers. Currently, there exist a few situation-aware music delivery systems for drivers such as SAfeDJ [4] which utilize a service centric contextualized cloud [16]. However, these existing systems are not extremely time efficient in the music mood-mapping process and there is no crowd-sensing based approach used in these systems to eliminate redundant music mood-mappings. Hence, our novel approach presented in this paper provides a considerable improvement in time efficiency by leveraging the principle of crowd-sensing and utilizing novel subsystems for intelligent offloading of music mood-mapping to the cloud and similarity detection amongst slightly different versions of the same song.

There has been some previous work focusing on offloading modules to the cloud to leverage its computing power and reduce computation time [9-11]. There has also been some work in offloading computation to the cloud in an energy efficient manner [12, 13]. However, an offloading solution specific to music mood-mapping in a driving scenario doesn't exist. Our intelligent offloading system addresses this problem by deciding to offload mood-mapping or not based on real factors such as signal strength and type of network. Factors specific to music files such as duration of the song and size of music file are also considered in our process.

Current fingerprinting based solutions for similarity detection in songs such as Echoprint [17] can be used. However, these solutions utilize complex fingerprinting algorithms which take considerable time to process and compute the similarity between songs. They utilize the core characteristics of the music alone to detect similarity whereas our unique approach detects similarity based on meta-data and some custom music features which are easy to compute. Hence, our score based similarity detection system is quick and accurate in determining similarity between slightly different versions of the same song which in turn eliminates redundant mood-mappings.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a novel cloud and crowd-sensing based approach for music mood-mapping, which can be applied in context-aware music recommendation systems to reduce the time taken in the music mood-mapping process and improve the efficiency of situation-aware music delivery systems considerably. Crowd-sensing of mood-mapping is enabled in our framework through the use of a cloud server and a centralized cloud database. Our intelligent offloading and similarity detection systems further optimize the time taken for music mood-mapping and hence enable us to provide in time music mood-mapping to promote safer driving amongst drivers. The results obtained by integrating our framework with SAfeDJ [4] have been very promising and the time taken to perform mood-mapping on songs has been drastically reduced along with an improved efficiency of music delivery. A key advantage of our crowd-sensing approach is that the performance of the system improves with increased usage over a period of time when more songs are mapped in the centralized cloud database. In the future, we shall explore the possibility of applying machine learning techniques to similarity detection and extending our framework to other media such as images and videos

7. REFERENCES

[1] World report on road traffic injury prevention, 2014, World Health Organization (WHO). Available: http://www.who.int/violence_injury_prevention/publications/road_traffic/world_report/en

[2] A study by Transport - European Commission. Available: http://ec.europa.eu/transport/road_safety/specialist/knowledge/fatique/index_en.htm

[3] M. Zwaag, C. Dijksterhuis, D. Waard, B. L. J. M. Mulder, J. H. D. M. Westerink, and K. A. Brookhuis. The influence of music on mood and performance while driving. Ergonomics. 55, 1 (2012), 12-22.

[4] X. Hu, J. Deng, J. Zhao, W. Hu, E. C.-H. Ngai, R. Wang, J. Shen, M. Liang, X. Li, V. C.M. Leung, and Y. Kwok. SAfeDJ: A Crowd-Cloud Co-design Approach to Situation-aware Music Delivery for Drivers. To appear in *ACM Transactions on Multimedia Computing, Communications and Applications*, 2015.

[5] X. Hu, J. Deng, W. Hu, G. Fotopoulos, E.C.-H. Ngai, Z. Sheng, M. Liang, X. Li, V.C.M. Leung, and S. Fels.2014. SAfeDJ Community: Situation-Aware In-Car Music Delivery for Safe Driving. *In Proc. ACM MobiCom*, 2014.

[6] F. Aalamifar, G. Vijay, P. Khoozani, and M. Ibnkahla. Cognitive wireless sensor networks for highway safety. *In Proc. ACM MSWiM-DIVANet symp,* 2011, pp. 55-60.

[7] F. Silva, A. Boukerche, T. Silva, L. Ruiz, E. Cerqueira, and A. Loureiro. Content replication and delivery in vehicular networks. *In Proc. ACM MSWiM-DIVANet symp,* 2014, pp. 127-132.

[8] S. Nirjon, R. F. Dickerson, Q. Li, P. Asare, J. A. Stankovic, D. Hong, and F. Zhao. Musicalheart: A hearty way of listening to music. *In Proc. ACM SenSys*, 2012, pp. 43-56.

[9] X. Hu, T.H.S. Chu, V.C.M. Leung, E. C.-H. Ngai, P. Kruchten, and H.C.B. Chan. A Survey on Mobile Social Networks: Applications, Platforms, System Architectures, and Future Research Directions. *IEEE Communications Surveys & Tutorials*, vol. 17, 2015.

[10] (2015) X. Hu. Context-aware mobile crowdsensing in mobile social networks. Available: www.mobilesoa.appspot.com

[11] C. Wang and Z. Li. A computation offloading scheme on handheld devices. *J. Parallel Distrib. Comput, vol.* 64, no. 6, June 2004, pp. 740-746.

[12] Liu et al. Energy efficient GPS sensing with cloud offloading. *In Proc. ACM SenSys*. 2012, pp. 85-98.

[13] A. P. Miettinen and J. K. Nurminen. Energy Efficiency of Mobile Clients in Cloud Computing. *In Proc. USENIX HotCloud*, 2010.

[14] (2015) Sequence Matcher [Online]. Available: https://docs.python.org/2/library/difflib.html

[15] Z. Wei, F. Yu, A. Boukerche. Trust based security enhancements for vehicular ad hocnetworks. *In Prof. ACM MSWiM-DIVANet symp,* 2014, pp. 103-109.

[16] X. Hu, L. Wang, Z. Sheng, P. TalebiFard, Li Zhou, J. Liu, and V.C.M. Leung. Towards a service centric contextualized vehicular cloud. *In Proc. ACM MSWiM-DIVANet symp,* 2014, pp. 73-80.

[17] (2015) Echoprint [Online]. Available: http://echoprint.me/