Data Science for Food, Energy and Water: A Workshop Report

Naoki Abe[1], Yiqun Xie[2], Shashi Shekhar[2], Chid Apte[1], Vipin Kumar[2], Mitch Tuinstra[3], Ranga Raju Vatsavai[4]
[1]IBM Research,[2]University of Minnesota,[3]Purdue University,[4]North Carolina State University
[1]{nabe,apte}@us.ibm.com, [2]{xiexx347,shekhar,kumar001}@umn.edu, [3]mtuinstr@purdue.edu, [4]rrvatsav@ncsu.edu

ABSTRACT

At the 22nd ACM SIGKDD conference on Knowledge and Data Discovery (KDD), a workshop on Data Science for Food, Energy and Water (DSFEW) was held to foster an interdisciplinary community intersecting data science and societally important domains of food, energy and water. The workshop included keynotes, panel discussion, presentations and posters, and introduced the emerging area of DSFEW to ACM SIGKDD audience, and triggered interdisciplinary idea-sharing in DSFEW research. The workshop website is sites.google.com/site/2016dsfew.

Keywords

food, energy and water nexus; data science

1. BACKGROUND

In the coming decades, the world population is projected to grow significantly (Fig. 1). Thus, securing the essential resources of food, energy and water (FEW), is one of the most pressing challenges the world faces today. The challenge is made harder due to climate change, rising economies and interactions among food, water and energy systems.

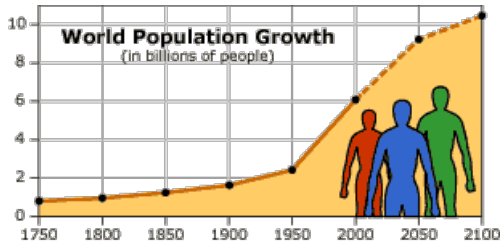


Figure 1: Projected world population growth [7].

It is difficult to consider food, water or energy security in isolation due to their complex interactions. For example, energy production needs water for cooling and may use bio-fuels. Conversely, food production requires energy and water as shown in Fig. 2. Trying to achieve energy security in isolation may lead to unanticipated surprises for food and water security [13]. For example, food prices rose in many parts of the world in 2008 coincident with increased subsidies

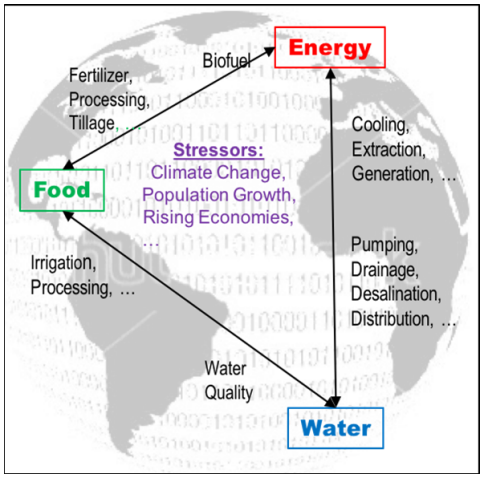


Figure 2: Interactions among Food, Energy, Water Systems (best in color) [2]. For example, food production not only needs water for irrigation and energy for fertilizer but also may degrade water quality due to run-offs.

for biofuels. Similarly, incentives for growing crops have depleted water resources (e.g., Aral Sea, Ogallala aquifer) and affected water quality (e.g., dead zone in Gulf of Mexico). To reduce such unanticipated consequences, the nexus approach jointly considers the interactions among food, energy and water systems [11].

Understanding the FEW nexus is among the highest priorities at the United Nations [8] as well as many countries. In 2011, Stockholm Environment Institute initiated a conference on "The Water, Energy and Food Security Nexus, Solutions for the Green Economy" to better understand the nexus [10]. In 2014, U.K. funded a set of research proposals on FEW (e.g., WEFWEBs at University of Glasgow) [1]. In U.S., a recent National Intelligence Council report identified it among the greatest challenges facing our world in the coming decades [12]. The US National Science Foundation (NSF) has also started a multi-year cross-directorate initiative titled Innovations at the Nexus of Food, Energy and Water Systems (INFEWS) [4]. More international research efforts are in need to address global FEW challenges (e.g., global FEW choke point in China, US and India [3]).

In 2015, US NSF sponsored a set of workshops to engage a diverse set of research communities to identify research chal-

lenges and opportunities. This ACM SIGKDD workshop on DSFEW is motivated by the NSF workshop, "A Workshop to Identify Interdisciplinary Data Science Approaches and Challenges to Enhance Understanding of Interactions of Food Systems with Energy and Water Systems" [2].

The US NSF INFEWS data science workshop [2] identified research needs in five key areas: 1) Integration of data sets and data-driven models of multiple types at many spatial and temporal scales, 2) Predictive and causal modeling of spatial and temporal data, with particular attention to auto-correlation, non-stationarity, and scale of FEW data, 3) Multi-stakeholder decision support, including methods for eliciting and sharing preferences, supporting negotiation and consensus building, 4) FEW nexus life cycle thinking, and 5) FEW data uncertainty, incompleteness, and bias. The workshop also underscored the need for community infrastructure, such as shared data sets, evaluation metrics, models, and tools, and the training of a new generation of scientists with the requisite training in the data sciences and the FEW sciences to facilitate progress at their interface.

The goals of this workshop were: (1) To introduce the emerging area of DSFEW to the KDD community; (2) To invite scientists and practitioners in FEW domains to the KDD community, and interest them in leveraging our technology and expertise; and (3) To innovate new technology, leveraging existing KDD technology where appropriate, to address the challenges in FEW, by bringing together a multidisciplinary audience and enticing them to synergize.

The workshop included two keynote presentations on predictive phenomics of plants and remote sensing in agricultural applications, followed by a set of presentations and posters on phenotyping, object recognition, change detection, prediction, regression and optimization. A panel discussion was held with leading experts in DSFEW on the motivation, problems, current stage, challenges and next steps in DSFEW research.

2. KEYNOTE PRESENTATIONS

Dr. Naoki Abe and Dr. Chid Apte opened the workshop and introduced the keynote speakers. Prof. Patrick Schnable, Director of Plant Sciences Institute and Center for Plant Genomics at Iowa State University, highlighted how field sensors and data science approaches are helping to build automated phenotyping systems and predict crop performance. Crop phenotypes (e.g., yield and drought tolerance) are determined by genotype, environment and their interaction (GxE). Genotyping data have been made accessible for all major crops. However, phenotypic data, a powerful source to statistically model GxE, remains with a limiting volume. Prof. Schnable described a framework to collect phenotypic data using high-throughput and high-resolution field sensors and robots. The dataset is analyzed using data science techniques (e.g., machine learning, correlation analysis) that help model relationships between phenotype properties and performance of the plants. Prof. Schnable has also explored interactions between neighboring plants. This analysis revealed that seed orientation influenced performance of adjacent plants.

The second keynote, Prof. Melba Crawford, associate dean for research in the College of Engineering at Purdue University, summarized the types of data collected through remote sensing and how they are applied to agriculture. With recent developments in remote sensing, data are generated via a variety of platforms (e.g., space-based, airborne, proximal sensing platforms). For space-based platforms, sensors are evolving from complex, multi-purpose to lower cost and measurement specific constellations of small satellites. Hyperspectral satellite images contain rich information for detailed spectral analysis of crops. Advanced data science techniques are necessary to explore the increased volume of data introduced by hyperspectrum, high resolution and high time frequency. For airborne and proximal sensing platforms (e.g., unmanned aerial vehicles, autonomous vehicles), new data sources, such as LiDAR point cloud, are becoming increasingly popular in agricultural applications. Topographical dataset (e.g., digital elevation models) can be derived from LiDAR point cloud and provide additional height information in analysis. These massive, multi-modality datasets from new sensors offer opportunities in agriculture applications from plant-level phenotyping to large-scale crop mapping and plantation monitoring. New algorithms in data science are needed to address challenges of multi-temporal, multi-scale and multi-sensor analysis.

3. PANEL DISCUSSION

Four major questions were posed in the panel discussion: (1) Context: How are food, energy and water communities exploiting data and data-science? (2) Gap Analysis: What are the key pain-points in leveraging data and data-science for food, energy and water? What are the data and data-science knowledge gaps in context of food, energy and water? (3) Nexus: Past approaches to solving Food problems (e.g., via fertilizers) had unanticipated negative impacts to water (e.g., water quality degradation). How may data and data-science help improve understanding of the interactions among food, energy, and water systems? How may they reduce unintended consequences ? (4) Engagement, Community Building: Why should data scientists engage with food, energy and water? How may we build and sustain a community of food, energy and water data science?

Dr. Sivan Aldor-Noiman, director of data science at The Climate Corporation, discussed the complex decision-making challenge faced by food growers and pointed out a major flaw in general machine learning methods when applied to FEW. Machine learning methods broadly have a well-known assumption on their variables that they are independent and identically distributed (i.i.d.). I.i.d assumption does not well-fit FEW datasets since the variables (e.g., soil property, water quality) are spatially autocorrelated under the first law of geography. Considering autocorrelation and brining it into machine learning models poses a great challenge in DSFEW research. Prof. Ronald Turco, the director of the Indiana Water Resources Research Center at Purdue University, discussed the importance of visiting farms to understand farmers' vocabularies and concerns. For example, farmers care about heavy rainfalls and droughts which are worse due to climate change. Dr. David Lapen, research scientist at Agriculture and Agri-Food Canana, shared his experience on water management in agricultural fields. Dr. Charlie Messina, senior scientist at Dupont Pioneer, described his perspectives on DSFEW-related research from his domain expertise. He emphasized the importance of incorporating expert knowledge and on-site experience into data science instead of finding solutions by modeling webpage click-statistics as is often done by some industry companies. Prof. Shashi Shekhar, McKnight Distinguished Pro-

fessor of Computer Science at University of Minnesota, reviewed how food-water-energy relate to the United Nations' 2030 goals for sustainable development. He then outlined the importance of spatial computing in these endeavors [9] and noted research challenges that lie ahead. He concluded by sharing successful stories in DSFEW (e.g., GeoGlam [6]).

In audience question section, Dr. Ramasamy Uthurusamy, a founder of ACM SIGKDD, challenged the audience to identify the moonshots of DSFEW if extensive funding (e.g., one US billion dollars) is available. Examples given by panel speakers include solving world hunger and clear drinking water to everyone. One interesting industry perspective is to crowd-source ideas and solutions from a series of hackathons without huge investment.

Dr. Ramasamy Uthurusamy also encouraged holistic thinking on the life cycle of DSFEW. The workshop focuses mostly on model selection in DSFEW. However, model selection accounts for only a limited portion (e.g., 10%) of the DSFEW life cycle. Data collection, data cleaning, business modeling, policy making, etc., all play critical roles in DSFEW life cycle. Research efforts on all stages of the life cycle require a commitment to community building and interdisciplinary collaboration.

Dr. Nikunj Oza, group lead of Data Sciences in Intelligent Systems Division at NASA, asked about the use of secondary information in FEW domains. Panelists mentioned value of such information (e.g., tweets) in detecting food-borne illnesses and food contamination.

Other discussions include applying advances in precision agriculture to small-farm owners, privacy and security protection with data sharing which are of farmers' concerns, and solutions to climate change under political challenges.

4. PRESENTATIONS AND POSTERS

The presentations and posters are grouped into three tiers: (1) **Domain application**: food, energy and water, as well as their nexus; (2) **Data analytics**: machine learning, data mining, regression and remote sensing techniques; and (3) **Infrastructure**: computation platform and system build. Majority of the work presented in the workshop belong to domain application. A summary of publications is shown in Table 1. In the table, data collection and data management belong to the category "infrastructure".

Food, energy and water: In food-related applications, interet was shown in automated phenotyping of crop plants and pest monitoring. Phenotyping, the process of characterizing properties and traits of plants, can be used to predict the status and performance of plants. Different types of image data are used to compute the phenotypes, including aerial imagery/LiDAR collected by UAV, ground images captured by field cameras and high contrast plant images taken in the lab. Unlike images captured in the field and labs, aerial imagery collected by wide-angle lens cameras on UAVs needs to be preprocessed into ortophotos before phenotyping. In the phenotyping phase, LiDAR point cloud can be used to generate digital elevation models to further incorporate height property of plants. Computer vision techniques (e.g., feature point extraction, pattern recognition) are applied to identify plant structures (e.g., leaves, stems, tassels), measure geometric properties and compute phenotypes. 3D images taken at different angles are also used to generate new phenotype characteristics by comparing morphological metrics among multiple angles. For pest monitoring, a convolution neural network approach is used to find soybean Cyst Nematode eggs with different rotations, shapes and scales. A network-based approach is applied to model the dynamics of invasive plant pests.

In water and energy, applications presented focus on water conservation and hydro-based power generation modeling. For water conservation, an estimation model was created to predict how much water can be saved through turf removal in California urban landscape to deal with the ongoing severe drought. Hydro-based power generation was modeled based on hydro-lake river inflows. A regularized linear regression Lasso method is applied on large scale oceanic-climatic predictors with high-dimensional data but small sample sizes. The forecasted stream flow information is used to estimate electricity production of hydro power stations in Waitaki catchment in New Zealand, which yields about 40% electricity of the country.

Data Analytics: Data analytics approaches were proposed to analyze remote sensing datasets to monitor land cover, identify land cover changes and optimize future spatial allocations of land covers. As a social concern, ethical issues in data science were also discussed.

To assist land cover monitoring, an automatic plantation mapping approach with ensemble learning and hidden Markov model was proposed to estimate palm oil cultivation in southeast Asia and enforce sustainability standard.

For land cover change detection, a support vector machine and convolutional neural network based method was constructed to classify land covers and locate urbanization (e.g., agriculture to residential) in West Bengal, India. Additionally, a multi-instance and multi-view learning framework was proposed to identify spatiotemporal change footprints where lake and river shrinkage happens.

Planning ahead with land cover allocation, a geodesign optimization tool was introduced to facilitate redesign of landscape in agricultural watersheds in the mid-western US. The nexus goal of geodesign is to improve water quality while still providing enough food under economic budget.

Infrastructure: Two infrastructure building efforts on data collection and management were presented. The first introduced an integrated knowledge graph for FEW, built on semantic web technologies and statistical relational learning. The goal is to harmonize diverse FEW data sources to perform ontology analysis. The second system, SmartFarm, combines sensor technologies and cloud computing platform to assist growers making decisions in precision farming with local farm statistics and a variety of external data inputs (e.g., weather predictions, satellite imagery).

5. CONCLUSION & NEXT STEPS

The ACM SIGKDD workshop on Data Science for Food, Energy and Water (DSFEW) introduced the emerging area of DSFEW to KDD data science community and inspired interdisciplinary idea-sharing [5]. Recent research results in FEW domain applications, data science approaches and system infrastructures were presented through keynotes, presentations and posters. Critical research questions in DSFEW were discussed in the panel discussion. The workshop participants are looking forward to growing the DSFEW community through publications (e.g., conference with special interest group, journal special issue) and competitions (e.g., KDD DSFEW challenge).

Table 1: Summary of workshop publications [5]

Title	Food	Energy	Water	Data collection	Data management	Data Analytics
A Knowledge Ecosystem for the Food, Energy, and Water System	✓	✓	✓		✓	
An end-to-end convolutional selective autoencoder approach to Soybean Cyst Nematode eggs detection	✓					✓
Automated Sorghum Phenotyping and Trait Development Platform	✓			✓		✓
Automated Vegetative Stage Phenotyping Analysis of Maize Plants using Visible Light Images	✓			✓		✓
Predictive Modeling of Sorghum Phenotypes with Airborne Image Features	✓					✓
Fast, automated identification of tassels: Bag-of-features, graph algorithms and high throughput computing	✓			✓		✓
Estimating Phenotypic Traits From UAV Based RGB Imagery	✓			✓		✓
What spins the turbine? Finding spatial climate precursors of hydro-lake inflows: Waitaki catchment, New Zealand		✓	✓			✓
How Much Water Does Turf Removal Save? Applying Bayesian Structural Time-Series to California Residential Water Demand		✓	✓			✓
Satellite Image Analytics, Land Change and Food Security						✓
A Bayesian Network approach to County-Level Corn Yield Prediction using historical data and expert-knowledge	✓					✓
Modeling the Food-Energy-Water Nexus in Critical Biodiverse Landscapes: A Case Study of Tonle Sap, Cambodia and Tulalip Tribe, USA	✓	✓	✓			✓
Plantation Mapping in Southeast Asia	✓					✓
SmartFarm: Improving Agriculture Sustainability Using Modern Information Technology	✓				✓	✓

6. ACKNOWLEDGEMENTS

The workshop was supported by NSF, the Computing Community Consortium and the Midwest Big Data Hub. We thank Jayant Gupta for his workshop notes.

7. REFERENCES

[1] Water energy food: WEFWEBs, research councils UK. http://gtr.rcuk.ac.uk/project/EDD08B5A-61DE-41CA-B967-3554BE85CCBA, 2014.

[2] Computing research association, NSF workshop to identify interdisciplinary data sci. approaches and challenges to enhance understanding of interactions of food sys. with energy and water sys. *Computing Research News*, 27(10), 2015.

[3] KSIConnect, Global choke point: Water-energy-food confrontations in china, us and india. https://www.youtube.com/watch?v=6F2tx903FMI, 2015.

[4] National Science Foundation, Innovations at the nexus of food, energy and water systems. www.nsf.gov/about/budget/fy2016/pdf/37_fy2016.pdf, 2015.

[5] ACM SIGKDD 2016 workshop on data science for food, energy and water. https://sites.google.com/site/2016dsfew, 2016.

[6] GeoGlam (global agricultural monitoring initiative). www.geoglam-crop-monitor.orgf, 2016.

[7] Museum of Paleontology, The ecology of human populations. http://evolution.berkeley.edu, 2016.

[8] United Nations, Transforming our world: the 2030 agenda for sustainable development. sustainabledevelopment.un.org/?menu=1300, 2016.

[9] E. Eftelioglu, Z. Jiang, R. Ali, and S. Shekhar. Spatial computing perspective on food energy and water nexus. *J. of Env. Studies and Sci.*, 6(1):62–76, 2016.

[10] M. Escobar. The nexus of water-energy-food: an introduction to the Inter-ADB sustainability report 2011. *Inter-American development bank sustainability report 2011*, pages 7–9, 2012.

[11] J. Liu, H. Mooney, et al. Systems integration for global sustainability. *Science*, 347(6225):1258832–1–9, 2015.

[12] National Intelligence Council. Global trends 2030: Alternative worlds. globaltrends2030.files.wordpress.com/2012/11/global-trends-2030-november2012.pdf, 2012.

[13] USDOE. The water energy nexus: Challenges and opportunities. http://www.energy.gov/sites/prod/files/2014/06/f16/Water%20Energy%20Nexus%20Report%20June%202014.pdf, 2014.

User Identity Linkage across Online Social Networks: A Review

Kai Shu[†], Suhang Wang[†], Jiliang Tang[‡], Reza Zafarani [♮], and Huan Liu[†]
[†]Computer Science & Engineering, Arizona State University, Tempe, AZ, USA
[‡]Computer Science & Engineering, Michigan State University, East Lansing, MI, USA
[♮]Computer Science & Engineering, Syracuse University, Syracuse, NY, USA
[†]{kai.shu,suhang.wang,huan.liu}@asu.edu,
[‡]tangjili@msu.edu, [♮] rzafaran@syr.edu

ABSTRACT

The increasing popularity and diversity of social media sites has encouraged more and more people to participate on multiple online social networks to enjoy their services. Each user may create a *user identity*, which can includes profile, content, or network information, to represent his or her unique public figure in every social network. Thus, a fundamental question arises – *can we link user identities across online social networks?* User identity linkage across online social networks is an emerging task in social media and has attracted increasing attention in recent years. Advancements in user identity linkage could potentially impact various domains such as recommendation and link prediction. Due to the unique characteristics of social network data, this problem faces tremendous challenges. To tackle these challenges, recent approaches generally consist of (1) extracting features and (2) constructing predictive models from a variety of perspectives. In this paper, we review key achievements of user identity linkage across online social networks including state-of-the-art algorithms, evaluation metrics, and representative datasets. We also discuss related research areas, open problems, and future research directions for user identity linkage across online social networks.

1. INTRODUCTION

In recent years, users have been introduced to many online social networks such as Twitter, Instagram, or LinkedIn. Due to diverse functionalities, different online social network platforms attract users for different purposes such as information seeking/sharing and social connection maintenance. For example, users may use Twitter to publish opinions on political events while adopting Instagram to share their leisure activities [44]. To better take advantage of services provided by each social network, users tend to join multiple online social networks. It has become increasingly popular for users to have accounts (also called user identities) on multiple social networks. As reported by a social media study, by the end of 2013, 42% of online adults are using multiple social media sites at the same time [1]. For example, 93% of Instagram users are involved in Facebook concurrently and 53% Twitter users are using Instagram as well [2].

Implications of Linking User Identities. The increasing popularity of users with accounts on multiple social media sites brings new opportunities and challenges to various mining and learning tasks. First, users with accounts on multiple social media sites give potentials to fully understanding users' interests and provide better recommendations or services [13; 33]. As user uses different online social networks for different purposes, analyzing a user identity on a single social media may not give a comprehensive understanding of his/her personalities and interests. However, if we can link a person's user identities on multiple datasets, collect and analyze his/her data on these social media sites together, we may have a more comprehensive view about the user and provide better services. Second, users with accounts on multiple social media sites allow us to integrate patterns among online social network sites and solve some problems unsolvable by data from only one site such as cold-start and data sparsity problems in many predictive tasks-[18; 62]. For example, a newly founded social media service may not have enough historical data for recommendations to users. If we can identify these users on other well established social media sites, then we can transfer knowledge from the mature social media to the new social media and thus mitigate the data sparsity or cold start problems. Finally, users with accounts on multiple social media sites can also help analyze user migration patterns and guide web developments [33]. Users migrating from one social network to another often reflects the user experience of web development. The linkage of user identities across different social media sites provides a great chance to study use migration behaviors. In addition, linking user identities allows for:

I. Enhancing Friend Recommendation. Online user engagement can increase with better friend recommendations. However, most friend recommendation algorithms recommend (1) non-connected users that (2) share mutual friends, as potential friends. Consider two users u_1 and u_2 that are not connected and are both friends of u_3 on site S_1. Thus, u_1 seems a good candidate for recommendation to u_2 on S_1. u_1 and u_2 are also members of social network S_2 and are also not connected on S_2. Assume that u_1 and u_2 share no mutual friends on S_2. With the information that we have

[1] http://www.pewinternet.org/2015/01/09/social-media-update-2014/

[2] http://www.marketingcharts.com/online/majority-of-twitter-users-also-use-instagram-38941/

from S_1, the recommendation algorithm could recommend u_1 to u_2 on S_2, even though they share no mutual friends on S_2. This type of recommendation is only possible when there is cross-site complementary information.

II. Information diffusion. Information diffusion has been traditionally studied within a single social network. In reality, information and rumors can travel within and across different social networks. Thus, it is interesting to investigate whether information diffuses more within one network or across networks. Moreover, what type of information propagates more within a network and what type propagates more across networks?

III. Analyzing Network Dynamics. Dynamics of single-site social networks are well-studied in the literature. These networks are known to have a power-law degree distribution, a small average path length, and being highly clusterable [63]. However, users belong to multiple sites and these network properties need to be generalized to multiple networks. In particular, it is interesting to determine how close the dynamics of single networks are to that of multi-networks. Recent studies have looked at the types of sites that users join [67] and how degree distributions (i.e., the number of friends) and friends that users have vary across sites [68].

Identity Linking Challenges. Although users with accounts on multiple social media sites bring many opportunities, taking advantage of these opportunities is not a trivial task. It's obvious that all the aforementioned opportunities require us to link users' accounts on multiple online social networks. However, the task of linking users accounts on multiple social media sites, also called user identity linkage, is a challenging task because: (1) user identity information can be rather diverse across different online social network sites for the same person in the real world [49] and (2) online social network data is big, noisy, incomplete and highly unstructured [58]. In particular, user identity data in online social networks has the following unique properties:

Profile Inconsistency: Different online social network sites have different structures and schemes to present user profiles. The user profile attributes can reveal user's basic information such as screen name, real name, age, biography, gender, location, education background, contact information, etc. Online social networks may allow users to selectively show profile attributes publicly and keep some sensitive information (e.g. age or contact information) private. In addition, the same attribute can be filled up with different information, e.g. location, depending on the site and user's purpose. Even in a single platform, a user profile may be deliberately counterfeited similarly to impersonate other users [24], which increases the uncertainty and ambiguity of profile features.

Content Heterogeneity: User's generated content can reflect his/her behavior properties as *when, where* and *what* he/she is posting. The content may involve in various medium types such as text, image, video, check-in, etc. The heterogeneous content information makes it extremely difficult to leverage them simultaneously to accurately link user identities [40]. In addition, online social network platforms may deliberately prohibit users to exchange information with others, resulting in the "Data Isolated Islands" phenomenon.

Network Diversity: Online social network structures for a specific user can be rather diverse on different social media platforms. Each social network structure is constructed for the user's specific objective and only reflects a subset of his/her real world social circle. For example, a PhD student looking for jobs has connections with hiring managers on LinkedIn that does not necessarily mean they are friends on Twitter. In practice, we cannot get the complete network structures for all users as well, due to the large scale and privacy issues maintained by online social network companies. This may prevent us from using graph structure patterns to match user entities as traditional entity resolution tasks [15].

As user identity linkage across online social networks is a very important and challenging problem, it has become a trending research area and attracted more and more research attention. The goal of this article is to provide a comprehensive review of recent studies of user identity linkage methods across online social networks and give a guidance on future research directions. The contributions of this paper are summarized as below:

- The user identity linkage problem has various definitions and formulations. [3] We provide a general and formal definition of the user identity linkage that covers most existing definitions;
- Existing approaches share similar characteristics in the problem-solving process that allows us to present a unified framework for user identity linkage task. The unified framework consists of two phases – feature extraction and model construction. We summarize different aspects of existing feature extraction and model construction techniques;
- Empirical evaluation can quantitatively assess and guide different algorithms. We discuss different datasets and evaluation metrics proposed by existing approaches;
- Linking user identities across online social networks is still an active area and there are many research opportunities. We compare the related research areas and discuss some open issues and possible future research directions.

The rest of this article is organized as follows. In Section 2, we introduce notations and formally define user identity linkage. In Section 3, we present the general framework of user identity linkage approaches. Specifically, we review the details of the feature extraction process in Section 3.1 and illustrate various types of model construction mechanisms in Section 3.2. In Section 3.3, we review the state-of-the-art methods for user identity linkage task. We discuss the datasets and evaluation metrics used by existing methods in Section 4. We briefly introduce the areas related to user identity linkage problem in Section 5. Finally, we discuss the open issues and future directions in Section 6 and conclude this article in Section 7.

2. PROBLEM DEFINITION

In this section, we introduce some basic notations and definitions which are used for user identity linkage task. Without loss of generality, we focus on a single real-world natural

[3] This problem is also known as Social Identity Linkage [40], User Identity Linkage [47], User Identity Resolution [5], Social Network Reconciliation [32], User Account Linkage Inference [54], Profile Linkage [70], Anchor Link Prediction [30] and Detecting me edges [11].

person on two online social network sites. Note that the settings of two online social networks and a single natural person can be easily extended to multiple sites and persons. The basic notations are defined as below,

- Let *User Identity* u refer to the unique social account representation on a social media site for a real natural person $\mathcal{P}$. It can consists of three components: *Profile, Content* and *Network*. Profile $\vec{p_u}$ includes a set of user description features such as username, location, age, among other attributes. Content $\vec{c_u}$ consists of a set of attributes that represent the activities that the user is involved in and includes time, location, text, image, etc. Network $\vec{n_u}$ consists of a set of attributes that describe the user's social connections with other users such as the friends in the ego-network.

- An *Online Social Network* $\mathcal{G}$ is represented as a graph $\mathcal{G}(\mathcal{U}, \mathcal{E})$ where $\mathcal{U} = \{u_1, u_2, ..., u_N\}$ is the set of user identities and $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{U}$ is the set of links in the network.

DEFINITION 1 (USER IDENTITY LINKAGE[4]) *Given two online social networks $\mathcal{G}^s$ (source site) and $\mathcal{G}^t$ (target site), the task of user identity linkage is to predict whether a pair of user identities u^s and u^t chosen from $\mathcal{U}^s$ and $\mathcal{U}^t$ respectively belong to a same real natural person, i.e., $\mathcal{F} : \mathcal{U}^s \times \mathcal{U}^t \to \{0, 1\}$ such that,*

$$\mathcal{F}(u^s, u^t) = \begin{cases} 1, & \text{if } u^s \text{ and } u^t \text{ belong to same person,} \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

where $\mathcal{F}$ is the prediction function we want to learn.

3. A GENERAL FRAMEWORK FOR USER IDENTITY LINKAGE

Many user identity linkage methods are proposed and most of the existing methods can be generalized into a unified framework as shown in Figure 1. This framework is composed of two major phases: i) Feature extraction and ii) Model construction. In the feature extraction phase, for a pair of users, features are extracted from users' profile, content and network structures. The extracted features are then used as inputs for the model construction phase, where a supervised, semi-supervised or unsupervised model is trained according to the availability of labeled pairs. Finally the trained models are used to predict whether two user identities match or not. Next, we will give details of feature extraction and model construction phases.

3.1 Feature Extraction

As previously mentioned, an user identity is composed of profile, content and network components. Next, we will introduce the detail of how to extract features from these components and how they can be represented and leveraged for the model construction phase.

[4]Note that even though it is possible that one person can have more than one user account in each online social network site, most previous work, if not all, assume that one person can only have one user account in one site.

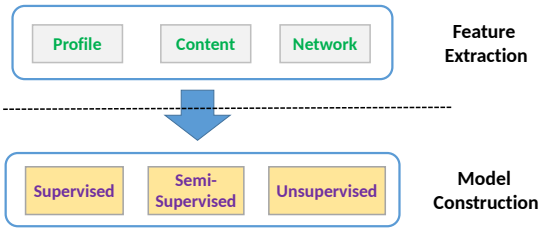


Figure 1: A General Framework for User Identity Linkage.

3.1.1 Profile Features

Profile features $\vec{p_u}$ for a user u are the set of profile fields that describe the user's basic information. Profile features $\vec{p_u}$ can usually be represented by an m dimensional vector i.e., $\vec{p_u} = (p_u^1, p_u^2, ..., p_u^m)$ where each dimension is a profile field. Different online social network sites have different structures and schemes to present user profile features. A list of representative public profile fields are as follows[5],

- **Username**: This refers to the unique identifier that can represent a user in the online social network. A real natural person can choose different usernames on different online social network sites.
- **Screen name**: It is usually formed from the first name and the last name that a user has entered in his profile.
- **Location**: The locations provides information on where the the user lives. Location may come in various forms: detailed addresses, lat-long coordinates, or city names.
- **Biography**: A free-form short text description written by a user as an introduction in online social networks. It often includes the user's occupation, organization, interest, among other attributes.
- **Education**: This refers to the education background for a user and often contains education history such as the names of universities, high schools, middle schools, etc.
- **Avatar**: The thumbnail or image provided by the user to visually present herself.

Other profile fields include gender, age, occupation, email, URL, etc. For two user identities u^s and u^t from source and target online social networks $\mathcal{G}^s$ and $\mathcal{G}^t$, denote their profile features as $\vec{p_{u^s}}$ and $\vec{p_{u^t}}$. Profile features can be utilized in different ways to decide whether u^s and u^t belong to the same person. Existing approaches that use profile features can be categorized into *Distance-based* and *Frequency-based* methods.

Distance-based: The similarity between profile fields of two user identities can be measured by comparing the "distance" between them. For text fields (e.g. username), string similarity schemes such as Jaro-Winkler distance [16], Jaccard similarity, and Levenshtein (Edit) distance are applied [54; 5; 56; 8; 27; 40]. For visual fields (such as the avatar), mean square error, peak signal-to-noise ratio, and Levenshtein distance are utilized [42] to calculate the similarities. After we

[5]Public profile fields are those can be accessed through API without authentication, while private profile fields need authentication.

calculate the distances of each field, we can further compute a weighted similarity score $sim(\vec{p_{u^s}}, \vec{p_{u^t}})$ of u^s and u^t to represent profile features [60].

Frequency-based: Instead of looking at profile attribute values directly and comparing their distances, one can investigate their frequency patterns. For example, profile text fields can be put into a bag-of-word model [47; 46] or the TF-IDF model [76; 52]. Other approaches estimate the uniqueness of profile fields (such as username) via a probabilistic model such as the Markov-chain model [52; 65].

3.1.2 Content Features

Content features $\vec{c_u}$ for user u reveal her activities such as posting, commenting, or replying in online social networks. Each content feature can consists of three types of information: temporal, spatial, and post. In other words, content features are defined as the multi-set of (temporal, spatial, post) bins: $\vec{c_u} = \{(t_1, s_1, p_1), (t_2, s_2, p_2), ...\}$.

- **Temporal**: The temporal information provides the timestamps of user's activities. Temporal information is usually automatically recorded by online social networking sites.
- **Spatial**: Spatial information often comes from the geo-tags attached to user posts, which can be transformed to accurate latitude and longitude values. Without geo-tags, spatial information can also be extracted from the posted texts or images.
- **Post**: Posts come in two forms, namely texts and images. On Twitter, people are more likely to post short texts. While images are preferred by Instagram users.

Content features are often jointly represented to capture special characteristics of user identities. Specifically, existing approaches use content features from the following aspects,

Interest-based: The temporal and post information can collectively reflect the topical interests of user identities. Thus, a long-term topic modeling [50; 40] can be performed to extract the user's core interests.

Style-based: The goal is to use posts to extract the writing style of users. The writing style includes personalized words and emoticons which can help distinguish user identities. Usually an n-gram language model [23] or term-frequency analysis [40; 27; 65; 30] is performed to extract words that distinguish one's identity from all the posts.

Trajectory-based: Trajectory can be extracted from a set of timestamped location data and modeled to capture the unique footprints of users' activities [53].

3.1.3 Network Features

Network features $\vec{n}_u$ for user u refer to the social network interactions with other users in the same online social network. Based on the completeness and connectivity of network topology structures, we can categorize networks into two types: *local network* and *global network*.

- **Local network**: These network features can be built from the *ego-networks* of user identities. The ego-network for each user identity is obtained through the one-hop neighborhoods (e.g. following/followee/friend relationships). In the real world situation, social network API often provides access permission of user's direct friendship if we know the user information. This holds for Facebook API with the appropriate permission set.
- **Global network**: This kind of network often indicates arbitrary merging graph such as a large sample or even complete social networks [5]. In global network, all user identities need to be connected. both immediate (i.e. one-hop) neighborhoods and non-immediate neighborhoods are considered in global networks.

With respect to the two aforementioned different network types, various network features can be constructed. For two user identities u^s and u^t from source and target online social networks $\mathcal{G}^s$ and $\mathcal{G}^t$, their immediate neighbor nodes are denoted as $\Gamma(u^s)$ and $\Gamma(u^t)$.

Neighborhood-based: Based on initial pairs of user identities that match $\mathcal{M}$, neighborhood-based features aim to capture the *match degree* of $\Gamma(u^s)$ and $\Gamma(u^t)$. For example, match degree can be computed using the number of shared identified friends [78; 69; 32], known in/out neighbors and in/out degree [48], and Dice coefficient [5]. Other metrics such as common neighbors, Jaccard's coefficient and Adamic/Adar score are extended to measure the neighborhood similarities as well [76; 30].

Embedding-based: Network embedding techniques can be utilized to learn latent network features that can preserve the original network structure, such as first-order proximity and second-order proximity [57]. In first-order proximity, a pair of nodes u_1 and u_2 in graph $\mathcal{G}$ can be represented as two vectors $\vec{z}_1$ and $\vec{z}_2$ and the probability that an edge is observed is computed by the sigmoid function [43]. In second-order proximity, each node plays two roles, namely the node itself and the "context" of other nodes (such as follower-followee relationship) [39]. Other approaches regard source network $\mathcal{G}^s$ and target network $\mathcal{G}^t$ as an entire network and map it to a hypergraph to learn latent network features [56]. Note that since local network only contains the ego-network structures of user identities, only neighborhood-based network features can be applied. For global network, both neighborhood-based and embedding-based network features can be extracted.

3.1.4 Discussion

We have demonstrated that profile, content and network features can be extracted and represented in different ways. In practical scenarios, these features have their specific characteristics: i) Profile features are relatively easy to obtain since they are usually publicly available; however, different online social network sites may allow users to fill profile fields selectively, which leads to many missing and inconsistent values. Moreover, profile features may be easily impersonated deliberately by other users [24]; ii) Content features can be very sparse for those users who are not active in posting their activities; Thus a continuous process is needed to obtain easy-to-use content features; iii) Network features can also be very noisy because not all edges represent true "friend" relations [40]. In addition, some network features can only be utilized when fully-aligned networks (e.g. global networks) are obtained, which is not practical in real-world scenarios.

3.2 Model Construction

In the previous section, we detail different aspects of feature extraction phase. Here we review the model construction

phase. Following traditional ways of classifying data mining and machine learning models, we summarize existing models into three groups: supervised, semi-supervised and unsupervised models.

3.2.1 Supervised Model

For a typical binary classification problem, there are two types of instances: positive instances (matching user identity pairs) and negative instances (non-matching user identity pairs). Suppose $\mathcal{Q} = \{(u^s, u^t), u^s \in \mathcal{U}^s, u^t \in \mathcal{U}^t\}$ denotes all the possible user identity linking pairs and $\mathcal{M} \subset \mathcal{Q}$ represents the positive instances, where u^s and u^t belong to the same natural person. The set of negative instances $\mathcal{N}$ satisfies $\mathcal{N} = \mathcal{Q} - \mathcal{M}$. The positive and negative instances $(\mathcal{M}, \mathcal{N})$ can be divided into the training set $(\mathcal{M}', \mathcal{N}')$ and the test set $(\mathcal{M}'', \mathcal{N}'')$. The goal of a supervised model is to learn a function $\mathcal{F} : \mathcal{U}^s \times \mathcal{U}^t \to \{0, 1\}$ on training set and then evaluation can be performed on test set. Existing supervised approaches fall into the following categories:

Aggregating methods: Aggregating methods combine the similarity scores of different features into a hybrid weighted form.

$$\mathcal{F}(u^s, u^t) = \alpha S_p(\vec{p_{u^s}}, \vec{p_{u^t}}) + \beta S_c(\vec{c_{u^s}}, \vec{c_{u^t}}) + \gamma S_n(\vec{n_{u^s}}, \vec{n_{u^t}}) \quad (2)$$

where α, β, γ are *weight* parameters and S_p, S_c, S_n are *similarity* functions of profile, content and network features [60; 27; 50]. Note that α, β, γ could be 0 if the algorithm excludes the corresponding features.

Probabilistic methods: A probabilistic classifier aims to predict a probability distribution of class labels. The basic assumption is giving a pair of user identities u^s and u^t, the probability of linkage is conditionally dependent on the feature vector $\vec{x}$ extracted from u^s and u^t, training set $(\mathcal{M}', \mathcal{N}')$, and the specific probability model M.

$$\mathcal{F}(\vec{x}) = \arg\max \Pr(y = 1 | \vec{x}, (\mathcal{M}', \mathcal{N}'), M) \quad (3)$$

where $y = 1$ indicates that u^s and u^t are matched. This is also known as *Maximum a posteriori* (MAP) estimate and can be solved based on Bayes theorem [70; 52; 71].

Boosting methods: Boosting methods build a conjunction of several types of weak hypotheses to learn a strong hypothesis [46].

$$\mathcal{F}(\vec{x}) = \text{sign}(\sum_{i=1}^{T} \alpha_i h_i(\vec{x})) \quad (4)$$

where $\vec{x}$ is the feature vector extracted for each pair of user identities (u^s, u^t), h_i is a weak classifier, α_i is the weight assigned to h_i and T is the count of weak classifiers.

Projection methods: Projection methods aim to learn a projection function Φ to map between original feature space (e.g. profile, content and network features) and a latent feature space of user identities in each online social network [43; 47]. Two projection functions Φ_s and Φ_t for source and target online social networks can be learned through the training process. Giving a user identity u^s from source social network $\mathcal{G}^s$, the target matching user identity $\hat{u^t}$ is chosen by following formula,

$$\hat{u^t} = \arg\min_{u^t \in \mathcal{U}^t} \mathbb{D}(\Phi_s(u^s), \Phi_t(u^t)) \quad (5)$$

where $\mathbb{D}$ is function to measure the distance for u^s and u^t in the latent space. Note that projection functions can be achieved simultaneously for the scenario of multiple social networks [47].

Some other supervised approaches try to find a best classifier from multiple traditional classifiers. Usually several popular classifiers are trained and validated such as Naïve Bayes, Decision tree, Logistic regression, KNN and SVM, then the best classifier is selected finally [65; 23; 51; 42; 25; 77].

3.2.2 Semi-Supervised Model

For semi-supervised approaches, both labeled and unlabeled user identities are taken into account and unknown user identity pairs are predicted during the learning process. A set of *seed* matching user identity pairs $\mathcal{M}'$ is obtained beforehand. We also denote $\mathcal{M}'_s = \{u^s | (u^s, u^t) \in \mathcal{M}'\}$ and $\mathcal{M}'_t = \{u^t | (u^s, u^t) \in \mathcal{M}'\}$ as the corresponding user identities in $\mathcal{G}^s$ and $\mathcal{G}^t$. Usually, unknown user identity matching pairs $\mathcal{M}''$ are discovered by utilizing the *topological structure* of the network and the feedback from seed user identity matching pairs $\mathcal{M}'$. Note that for simplicity, we treat transductive learning as a special type of semi-supervised learning [79]

Propagation methods: Propagation methods discover unknown user identity pairs in an iterative way from seed matching user identity pairs. Let $\Gamma(u^s)$ and $\Gamma(u^t)$ denote the neighborhood of user identity u^s and u^t, respectively, the key idea is to define a function Ψ to compute the *match degree* of user identities using *known neighborhood* (i.e. $\Gamma(u^s) \cap \mathcal{M}'_s$ and $\Gamma(u^t) \cap \mathcal{M}'_t$) information and other features (such as profile or content features) extracted from u^s and u^t. In each iteration, the user identity pairs with the highest match degree score (or with score exceeding a threshold) are selected. The propagation process terminates until no more user identity pairs can be found. Usually, two kinds of propagation order are used: i) *Exhaust comparison*, where each candidate matching pair is selected from the remaining unmatched user identities $\mathcal{S}$ and $\mathcal{T}$ [69; 32; 48; 40]; ii) *Local expansion*, where candidate matching pairs are locally expanded from neighbors of existing matched user identities [78; 8; 76; 77].

Embedding methods: Semi-supervised embedding methods usually learn the latent features of user identities collectively in source and target networks $\mathcal{G}^s$ and $\mathcal{G}^t$. To map user identities from the original feature space to a *common embedding space*, the labeled user identities $\mathcal{M}'_s$ and $\mathcal{M}'_t$ of seed matching pairs $\mathcal{M}'$ are constrained to have the same latent representations. In [56], a hypergraph is built and seed matching pairs $\mathcal{M}'$ are projected to a node to ensure the aforementioned constraint. In [39], the following/followee relations are approximated in the latent space with an explicit constraint to ensure that latent feature vectors for $(u^s, u^t) \in \mathcal{M}'$ are equal (i.e. $\vec{z_{u^s}} = \vec{z_{u^t}}$).

3.2.3 Unsupervised Model

Due to the high cost to obtain labeled matching user identity pairs, unsupervised models are performed with only unlabeled data. Only few unsupervised approaches exist since a small set of seed matching user identity pairs can be acquired from user self-posting websites such as Google+ or About.me[6], or through human annotating by comparing profile, content and network features. Existing unsupervised approaches fall into following categories:

[6]https://about.me/

Aligning methods: Aligning methods usually consist of three steps: i) Compute an *affinity score* for every candidate pair of user identities $(u^s, u^t) \in \mathcal{U}^s \times \mathcal{U}^t$ based on profile, content or network features; ii) Build a bipartite graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', w)$, where $\mathcal{V}' = \mathcal{U}^s \cup \mathcal{U}^t$, $\mathcal{E}' = \{(u^s, u^t) | u^s \in \mathcal{U}^s, u^t \in \mathcal{U}^t\}$ and the weight $w(u^s, u^t)$ of edge (u^s, u^t) is given by the aforementioned affinity score; iii) Based on the resulting graph $\mathcal{G}'$, an optimized problem is formalized to achieve one-to-one matching for all user identity pairs. For example, an affinity score is computed by modeling content (i.e. time-location trajectories) similarities and a maximum weight matching scheme is performed to link user identities in [53]. In [34], affinity score is computed from profile and network features (i.e. the username overlap of common friends) and then overlaps are maximized to match all user identities.

Progressive methods: Progressive methods try to classify user identity pairs in a progressive way: i) A feature with strong discriminability is used to find *partial* ground truth and ii) all extracted features are considered to perform classification on remaining unlabeled user identities. In [38], n-gram probability is used to automatically acquire training data and then a SVM classifier is utilized to classify remaining unmatched user identity pairs.

3.2.4 Discussion

Here, we discuss some key aspects of aforementioned models for the user identity linkage problem. Since social network data is huge and people always involve in multiple online social networks, we need to consider the *scalability* and *multiplicity* of those methods. i) Scalability: For source and target online social networks $\mathcal{G}^s$ and $\mathcal{G}^s$, the complexity for exhausting comparison is $|\mathcal{U}^s| \times |\mathcal{U}^t|$. Similar to traditional entity resolution problems, some *blocking functions* can be pref-defined to reduce the computation cost [22]. One way is to construct blocking keys from profile, content or network features, so that user identities not matching on the key are not compared [28; 11]. The other way is neighborhood based, where only the neighborhood of known matching user identities are compared [78; 8; 76; 77]. ii) Multiplicity: most existing approaches consider pair-wise user identity linkage problem and the case for multiple platforms are can be extended by integrating pair-wise linkage results in a transitive manner. However, it may suffer the problem of inconsistent results when the order of transitivity changes. A possible solution is to learn the projection function Φ from original feature spaces to latent feature spaces for each online social network individually [43; 47] or simultaneously [56; 39].

3.3 A Summary of User Identity Linkage Algorithms

In the previous subsection, we give an overview about the two important components of algorithms for user identity linkage – feature extraction, and model construction. In this subsection, we further give a summary about representative user identity linkage methods in Table 1. A brief introduction about these algorithms is given below:

3.3.1 Supervised

• MOBIUS [65]: This paper explores the minimum part of profile features (i.e. username) insightfully by modeling user behaviors from human limitation, exogenous factors and endogenous factors, and the Naïve Bayes classifier is used.

• ULink [47]: This paper uses basic profile features and a projection algorithm is proposed. An online version has been developed to handle dynamic scenario of social network datasets.

• Perito'11 [52]: This paper is the first to only use specific profile features, i.e., username, to perform user identity linkage task. The algorithm is a Markov-Chain based probabilistic model.

• LU-Link [23]: This paper utilizes style-based content features and then a logistic regression classifier is applied to predict matching user identity pairs.

• OPL [70]: This approach focuses on linking user identities in cost-sensitive setting. Extensive profile features are extracted and then a probabilistic classifier is applied.

• DCIM [50]: This approach jointly models neighborhood-based network features and interested-based content features by discovering *core interests* of users, which can capture users' characteristics more accurately. An aggregating method is applied for this method.

• Peled'13 [51]: This approach extracts distance-based profile features and neighborhood-based network features. Many popular classifiers are performed in the experiments such as Adaboost, Random Forest, etc.

• Malhotra'12 [42]: This paper utilizes distance-based profile features (also referred to "User footprint") and then the Naïve Bayes classifier is applied. Comprehensive comparison is performed to analyze feature discriminative capacities and username and display name are found to be the most discriminative features.

• Vosecky'09 [60]: This approach uses distance-based profile features and a supervised aggregating method to link user identities. Weights can be learned adaptively for different profile attributes fields.

• Lofciu'11 [27]: This paper focuses on online social tagging systems. It presents a model that use frequency-based profile (i.e. username) features and style-based content (i.e. tags) features, and then a supervised aggregating algorithm is implemented.

• Motoyama'09 [46]: This is a systematic approach for searching and matching user identities on multiple online social networks. It uses frequency-based profile features and then a boosting algorithm is leveraged to classify user identity matching.

• Goga'13 [25]: This method leverages public distance-based profile features then applies several popular classifiers to decide whether user identity pairs are matched.

• PALE [43]: This supervised framework employs embedding-based network features to map social network structures into low dimension space. Based on latent features of user identities, a projection method is applied.

• MNA [30]: This approach extracts style-based content features and neighborhood-based network features. A supervised aggregating algorithm is built on seed matching user identity pairs and weighted maximum matching scheme is utilized to rank all potential user identities.

3.3.2 Semi-supervised

• IONE [39]: This approach extracts embedding-based network features to learn the follower-ship/followee-ship of each user simultaneously. Seed user identity pairs are constrained to transfer the context of social relation network structure. The embedding algorithm is developed to match unknown

Table 1: Comparison of User Identity Linkage Approaches: Algorithms.

Feature \ Model	Supervised	Semi-Supervised	Unsupervised
Profile	[65] [47] [70] [52] [42] [60] [46] [25]		
Content	[23]		[53]
Network	[43]	[78][48] [32] [39]	
Profile, Content	[27]		
Profile, Network	[51]	[76] [54] [5] [8] [56] [77] [11]	[34]
Content, Network	[50] [30]		
Profile, Content, Network		[40]	[38]

user identity pairs.

• COSNET [76]: This paper presents an energy-based model to link user identities by considering both local and global consistency. The local consistency refers to situation of only two social networks, while global consistency is the mapping consistency on multiple networks. COSNET first extracts distance-based profile features and neighborhood-based network features and then use an aggregating algorithm to obtain local consistency.

• HYDRA [40]: This paper proposes a semi-supervised multi-objective framework jointly modeling heterogeneous behaviors and structure consistency. Heterogeneous behaviors including distance-based profile features, style-based content features, trajectory-based content features and neighborhood-based network features are modeled dynamically in a propagation algorithm.

• FRUI [78]: This paper presents a Friend Relationship-Based User identification framework, which first extracts neighborhood-based network features and then develops a semi-supervised propagation algorithm.

• JLA [5]: This approach is based on Conditional Random Fields which jointly models distance-based profile features and neighborhood-based network features. JLA uses propagation method to identify new matching pairs.

• Shen'14 [54]: This approach considers distance-based profile features and neighborhood-based network features, and a propagation method is used to iteratively identify unknown user identity pairs.

• Zhang'16 [77]: This approach leverages distance-based public profile features and neighborhood-based network features to link user identities by a local expansion propagation algorithm.

• User-Matching [32]: This paper proposes an efficient propagation algorithm for online social networks, which uses neighborhood-based network features.

• Bennacer'14 [8]: This paper presents a rule-based propagation algorithm by using network and profile features. It consists two major steps: 1) Selecting candidate user identity pairs based solely on neighborhood-based network features; 2) Determining the exact match by comparing the distance-based profile features.

• MAH [56]: This paper incorporates hypergraph to model the embedding-based network features and proposed a embedding method mapping user identities to lower dimension spaces. Distance-based profile features (e.g. username) are also leveraged to MAH model to achieve better performance.

• NS [48]: This is the first approach to use a graph theoretic model based on neighborhood-based network structure to perform user identity linkage task. NS implements a propagation algorithm.

• DetectMe [11]: This approach combines distance-based profile features and neighborhood-based network features to classify user identity linkages. It first select a candidate list with a threshold to filter similarity score of profile features. Then another threshold is set to filter network structure similarity score.

3.3.3 *Unsupervised*

• Alias-Disamb [38]: This paper focuses on task to decide whether cross-platform user identities with same username belongs to same natural person. First, a feature with strong discriminability is used to find partial ground truth and then all extracted distance-based profile features are considered into a SVM classifier.

• POIS [53]: This approach utilizes trajectory-based content features to link user identities. The model is an aligning algorithm, where affinity score is computed based on time-stamped location data, and then the maximum weighted matching scheme is utilized to find the most likely matching user identities.

• Labitzke'11 [34]: This paper tries to compare neighborhood-based network features (i.e. public available mutual friends) for user identity linkage. The model is a typical aligning algorithm. It first builds a "comparison set" of users' friends and then defines metrics to maximize the overlap and minimize the distance of friends list find the most likely corresponding user identities.

4. EVALUATION

In this section, we discuss how to assess the performance of algorithms for user identity linkage across online social networks. We focus on the datasets and evaluation metrics for this task. It's worth mentioning that there is no "best" user identity linkage method due to the variety of data sources and application domains.

4.1 Datasets

Real data: Since most online social network sites provide Application Program Interface (API) to grant access to their data, there are lots of datasets available to do *single* social network research. However, there are no agreed benchmark datasets for the user identity linkage task across online social networks. On one hand, it's very difficult to obtain the ground truth of known user identity linkage pairs. On the other hand, most of existing work attempt to tackle this problem from different feature spaces and it is hard to obtain a comprehensive dataset with all kinds of features. We list some publicly available datasets as below, and a more

Table 2: Comparison of User Identity Linkage Approaches: Datasets.

Feature \ Availability	Public	Request	Not public
Profile		[24]	[65] [47] [70] [52] [42] [60] [46] [25]
Content		[23]	[53] [23]
Network			[78][48] [32] [39]
Profile, Content			[27]
Profile, Network	[11] [76]		[51] [54] [5] [8] [56] [77] [34]
Content, Network			[50] [30]
Profile, Content, Network			[40] [38]

comprehensive data set comparison for existing methods is described in Table 2.

- *Goga1315* [7]: This data resource consists of two datasets collected based on google+ sitemap data. The first dataset is used in [23] and contains the data of three social networks, i.e., Twitter, Flickr and Yelp. Both profile and content features are provided in this dataset. The second dataset is used in [24] and public profile features are provided from Twitter, Facebook, LinkedIn and Flickr.
- *Buccafurri12* [8]: This dataset was originally collected in 2012 by the authors in [11] and it was further enriched in 2014 [8]. It contains the data of four online social networks, i.e., LiveJournal, Flickr, Twitter and Youtube. The dataset is composed of 93,169 user identities, 145,580 friendship links and 462 matching user identity links. Profile features such as username and network features such as friendship are explicitly provided in this dataset.
- *Zhang15* [9]: This dataset was collected during 2015 with two kinds of online social networks, namely social network sites and academia network sites [76]. Social network sites include Twitter, LiveJournal, Flickr, Last.fm, MySpace and academia networking sites are ArnetMiner, VideoLecture and LinkedIn. For each online social network, the username and friendship network are provided for user identities. The ground truth of matching user identities are obtained from [52] and the crowdsourcing service on ArnetMiner.

Synthetic data: Some approaches only utilize network features to link user identities. The performance can be evaluated on simulated synthetic networks. According to whether a real social network exists or not, synthetic data sets have two types of sampling strategies:

- *Full synthetic*: Social networks are sampled only based on existing graph generating algorithms and no real social network is provided. In [78; 32], different types of synthetic networks are built such as Erdős-Réyi random graph [20], Watts-Strogatz small-world graph [61], Affiliation Network [35], Barabási Albert preferential attachment model [4] and RMAT [14].
- *Partial synthetic*: A real social network $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is given, and two sub-networks can be constructed based on *node sampling* or *edge sampling* strategies. Node sampling aims to extract two subset of nodes $\mathcal{V}^s$ and $\mathcal{V}^t$ from $\mathcal{V}$ such that $\mathcal{V}^s \bigcup \mathcal{V}^t = \mathcal{V}$ and $\mathcal{V}^s \bigcap \mathcal{V}^t \neq \phi$ [48]. Three common ways for edge sampling [69] are i) Randomly adding edges with a predefined probability; ii) Randomly removing edges with a predefined probability; 3) randomly rewiring edges with a pre-defined probability.

In addition, we summarize the datasets used by representative methods in Table 2 from the feature and availability perspectives. For availability, i) Public means that the dataset can be directly downloaded from the website; ii) Request means that the dataset is provided upon request from the authors; iii) Not public means no explicit sources are provided to acquire dataset.

4.2 Evaluation Metrics

To evaluate the performance of algorithms for the user identity linkage problem, different metrics are proposed. Here, we review some widely used metrics such as prediction metrics and ranking metrics.

Prediction metrics: Most previous approaches consider user identity linkage problem as a binary classification task that, given two user identities u^s and u^t from source and target online social networks $\mathcal{G}^s$ and $\mathcal{G}^t$, determine whether u^s and u^t are matching or not:

- True Positive (**TP**): when predicted matched user identities belong to same natural person;
- True Negative (**TN**): when predicted unmatched user identities belong to different natural persons;
- False Negative (**FN**): when predicted unmatched user identities belong to same natural person;
- False Positive (**FP**): when predicted matched user identities belong to different natural persons.

Based on aforementioned possible classification results, we can define following metrics,

$$Precision = \frac{|TP|}{|TP| + |FP|} \quad (6)$$

$$Recall = \frac{|TP|}{|TP| + |FN|} \quad (7)$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (8)$$

$$Accuracy = \frac{|TP| + |TN|}{|TP| + |TN| + |FP| + |FN|} \quad (9)$$

[7] http://www.mpi-sws.org/~ogoga/data.html
[8] http://www.ursino.unirc.it/pkdd-12.html
[9] http://aminer.org/cosnet

The metrics above are based on pair-wise matching results. In [70], a set-wise metric called *Identity-based Accuracy* (I-Acc) is defined as follows:

$$I\text{-}Acc = \frac{\#\text{ correctly identified user identities}}{\#\text{ground truth user identities}} \quad (10)$$

Note that for $Precision, Recall, F1$ and $I-Acc$, the higher the value, the better the performance.

Ranking metrics: Some approaches may provide a top-k ranking list of potential matching user identities rather than only one. Giving a user identity u^s from source social network $\mathcal{G}^s$, all K candidate identities in target social network $\mathcal{G}^t$ are ranked based on the matching degree with u^s, i.e. $\mathcal{R}_{u^s} =< u_1^t, u_2^t, ..., u_K^t >$. The goal is to rank true matching user identities as top as possible.

The *Receiver Operating Characteristics* (ROC) curve can be drawn by plotting *False Positive Rate* (FPR) and *True Positive Rate* (TPR) as x and y axes respectively. It can compare the performance of different classifiers by changing class distributions via a threshold. TPR and FPR are defined as follows,

$$TPR = \frac{|TP|}{|TP| + |FN|} \quad (11)$$

$$FPR = \frac{|FP|}{|FP| + |TN|} \quad (12)$$

Based on ROC, we can compute the Area Under ROC curve (AUC) value which can measure the overall performance of how well the classifier can rank the positive linking user identity higher than any negative user identity. Based on [26], AUC is defined as below,

$$AUC = \frac{\sum(n_0 + n_1 + 1 - r_i) - n_0(n_0+1)/2}{n_0 n_1} \quad (13)$$

where r_i is the rank of i_{th} positive matching user identities and n_0 (n_1) is the number of positive (negative) user identities.

By assuming only one user identity in $\mathcal{R}_{u^s}$ that can match u^s (i.e. $n_0 = 1$), there is only one positive matching denoted as r_1 and AUC can be computed by,

$$AUC = \frac{n_1 + 1 - r_1}{n_1} \quad (14)$$

A similar measure called $Hit-Precision$ [47] is defined as follows,

$$Hit\text{-}Pricision = \frac{n_1 + 2 - r_1}{n_1 + 1} \quad (15)$$

Other metrics are proposed such as *Mean Reciprocal Rank* (MRR) [27] and *Mean Average Precision* (MAP) [43] which are calculated by the average performance of Reciprocal Rank (RR) and Average Precision (AP) over all user identities that need to be classified. Under the assumption that $n_0 = 1$, Reciprocal Rank and Average Precision are,

$$AP = RR = \frac{1}{r_1} \quad (16)$$

In [27], $Success@k$ measures whether the positive matching user identity will occur in top-k ($k <= K$) list or not. Note that for $AUC, Hit-Precision, MAP, MRR$ and $Success@k$, a higher value indicates better performance.

5. RELATED AREAS

In this section, we discuss areas related to the problem of user identity linkage across online social networks. We introduce these areas by briefly explaining the task goals, highlighting some popular methods and pointing out the differences from the user identity linkage problem.

5.1 Record linkage

Record linkage (or entity resolution) refers to the process of finding related entries in one or more related relations in a database and creating links among them [10]. This problem has been extensively studied in the database area and applied to data warehousing and business intelligence. Based on this survey [31], existing methods exploit features in three ways, namely numerical, rule-based and workflow-based. Numerical approaches combine the similarity score of each feature into a weighted sum to decide linkage [21]; Rule-based approaches derive match decision through a logical combination of testing separate rules of each feature with a threshold; Workflow-based methods apply a sequence of feature comparison in an iterative way. Both supervised such as TAILOR [19] and MARLIN [9], and unsupervised approaches such as MOMA [59] and SERF [7] are studied in the literature. Note that user identity linkage differs from the record linkage problem due to the specialty of online social network scenarios.

5.2 Network alignment

The network alignment task is to find a common subgraph across multiple input networks and can be categorized into local network alignment and global network alignment problems [55]. Local network alignment tries to multiple unrelated regions of isomorphism, while global network alignment maintains a consistent overall alignment for all nodes among networks. This problem has been widely applied in many application areas such as database matching [45], bioinformatics [29], computer vision [17], etc. Representative algorithms include IsoRank [55], NetAlign [6], etc. Recent approaches study network alignment problems under online social network scenarios with [75] or without attribute information [74]. The problem settings of network alignment and user identity linkage are very similar when network features are considered. However, they are different because: i) User identity linkage has its specialty which can be performed without network features ii) Network alignment aims to find a partial or overall alignment of subgraphs while user identity linkage focuses on node alignment.

5.3 De-anonymizing social networks

Social network anonymization refers to the process to replace each user identity's unique identifier (e.g. username) with a random string, but the network structure remains revealed [3]. From the attacker's perspective, both active and passive attacks can be performed on a single online social network with limited information to de-anonymize user identities [3]. By pointing out several drawbacks of active attacks, [48] proposes a large-scale passive social network de-anonymization method. Specifically, it utilizes known anchor links from the source social network as auxiliary information to de-anonymize user identities in the target social network. In this sense, social network de-anonymization problem are actually user identity linkage problem when only network structure information is leveraged.

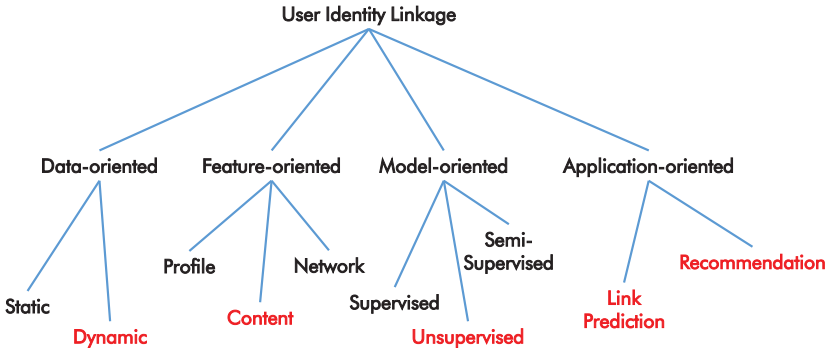


Figure 2: An overview of research aspects for User Identity Linkage. Those areas highlighted in red have not been extensively studied.

5.4 Link prediction

In the context of online social networks, traditional link prediction aims to predict missing links or future links between two user identities in a single social network [2]. Both supervised [1] and unsupervised [37] approaches are proposed to solve the link prediction task. User identity linkage can also be treated as link prediction problem. The difference is that for user identity linkage, we predict "link" between user identities of the same natural person on multiple social network media sites, while for link prediction, we usually predict links between two different users/objects on single homogeneous or heterogeneous network.

6. OPEN ISSUES AND FUTURE RESEARCH DIRECTIONS

In this section, we propose some remaining open issues for the user identity linkage problem and future research directions along this line. We will introduce the dataset challenge and evaluation challenge for the user identity linkage problem. Then, we will discuss some future research directions which have potential to attract more and more attention. As shown in Figure 2, we categorize the research aspects into four parts: data, feature, model, and application. We have already given detailed descriptions on feature and model in the previous sections, thus we focus on data-oriented and application-oriented research directions.

Data challenge: The user identity linkage problem is becoming popular in recent years and more and more methods are developed. However, as mentioned in Section 4.1, there is no agreed benchmark dataset to evaluate and compare existing methods. Existing publicly available datasets may contain *partial* features (such as network structures and user names), but dataset equipped with *comprehensive* profile, content and network features is limited. To obtain a comprehensive dataset for research purpose, we face following challenges: i) *User privacy*, how to access and use user identity features without invasion of user privacy? ii) *Ground truth*, how to obtain matching user identity pairs across online social networks when some social network sites may intentionally prevent user sharing contents; iii) *Limited access*, some online social network sites provide API to access their data for proper use, but they often set rate limits and restricted permission which make it hard to acquire data in large scale.

Evaluation challenge: In practical situation, we cannot get the entire social network datasets to perform the user identity task. Note that the matching and non-matching user identity pairs are very *imbalanced*, which may affect performance evaluation significantly [24]. Goga *et al.* also recommend precision and recall to be more reliable evaluation metrics [24]. In addition, different problem settings (such as exact matching and top-k matching) may cause the demand for choosing different evaluation metrics.

Dynamic user identity linkage: Social networks are dynamically changing over time. Profile, content and network features for user identities keep changing or being accumulated as time goes by. Thus, a more practical solution is to build online user identity linkage methods by extracting features dynamically. In [47], an online learning algorithm is developed to take advantage of incremental data to efficiently improve linkage performance. Nie *et al.* [50] shows that by modeling the "core interest" of user identities with accumulated content features, user identity linkage performance can be significantly improved. Along this line, we can consider to use other types of feature (e.g. network structure) properties dynamically. Moreover, existing approaches for dynamic (attribute) network analysis can also be extended to the multiple network scenario to perform dynamic user identity linkage task.

Jointly user identity linkage and recommendation: Cross-domain recommendations have attracted much attention from researchers recently. It aims to jointly leverage knowledge from source and target domains to build recommendation systems. Existing approaches focus on exploiting cross domain knowledge in following ways: linking, aggregating, sharing and transferring [12]. *Domain* can range in different levels such as attribute, type, item and system level. Note that social networks can be treated as a system level domain. For aggregating and sharing knowledge approaches, user and/or item *overlap* is needed among different social networks. Cross-domain recommendations can benefit from linking user identities from following aspects. First, when user overlap information is obtained via the user identity linkage task, user profiles can be enriched and social relations can be transferred to boost recommendation performance such as video [18], friend [66; 62] and product recommendations [41]. Second, user identity matching and recommendation task can be modeled jointly into a matrix factorization model [36]. For example, Li *et al.* provided a new viewpoint via collaborative filtering for user identity

linkage as well as item recommendations.

Jointly user identity linkage and link prediction: Link prediction problem has been proven to be an important research topic for decades. The major task of link prediction is to predict missing or future formed links in different social networks (homogeneous or heterogeneous social networks). In recent years, link prediction on *aligned networks*, where social networks share common users, are becoming popular [30]. User identity links (also refer to anchor links) can play very important roles in link prediction across aligned networks. Zhang *et al.* formalized the problem of *collective link prediction*, which jointly predicts anchor and social links together across heterogeneous online social networks [73; 64]. They demonstrated that link prediction can be performed simultaneously in multiple networks through meta-path based methods. Meanwhile, in [72] multiple anchor link (such as user and location anchor links) prediction has been explored with an unsupervised model recently.

7. CONCLUSION

Nowadays, people tend to join multiple online social networks for different purposes. Linking user identities across online social networks is of great value in many application areas such as recommendations, link prediction, etc. In this paper, we introduce a unified framework for the user identity linkage problem, which consists of two phases: i) Feature extraction and ii) Model construction. In detail, features can be obtained from profile, content and network information; while models can be conducted in supervised, semi-supervised and unsupervised ways. We further highlight some state-of-the-art approaches and introduce representative datasets and metrics . Moreover, we discuss data and evaluation challenges for this task. For future directions, many tasks in single social network can be properly adjusted and applied in cross network scenarios, and more practical problem settings for user identity linkage can be further explored.

8. REFERENCES

[1] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *SDM06: workshop on link analysis, counter-terrorism and security*, 2006.

[2] Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*. 2011.

[3] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW*, 2007.

[4] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 1999.

[5] Sergey Bartunov, Anton Korshunov, Seung-Taek Park, Wonho Ryu, and Hyungdong Lee. Joint link-attribute user identity resolution in online social networks. In *ACM (SNA-KDD)*, 2012.

[6] Mohsen Bayati, Margot Gerritsen, David F Gleich, Amin Saberi, and Ying Wang. Algorithms for large, sparse network alignment problems. In *ICDM*, 2009.

[7] Omar Benjelloun, Hector Garcia-Molina, David Menestrina, Qi Su, Steven Euijong Whang, and Jennifer Widom. Swoosh: a generic approach to entity resolution. *VLDB*, 2009.

[8] Nacéra Bennacer, Coriane Nana Jipmo, Antonio Penta, and Gianluca Quercini. Matching user profiles across social networks. In *International Conference on Advanced Information Systems Engineering*. Springer, 2014.

[9] Mikhail Bilenko and Raymond J Mooney. Adaptive duplicate detection using learnable string similarity measures. In *KDD*, 2003.

[10] David Guy Brizan and Abdullah Uz Tansel. A. survey of entity resolution and record linkage methodologies. *Communications of the IIMA*, 2015.

[11] Francesco Buccafurri, Gianluca Lax, Antonino Nocera, and Domenico Ursino. Discovering links among social networks. In *ECML/PKDD*, 2012.

[12] Iván Cantador, Ignacio Fernández-Tobías, Shlomo Berkovsky, and Paolo Cremonesi. Cross-domain recommender systems. In *Recommender Systems Handbook*. 2015.

[13] Francesca Carmagnola and Federica Cena. User identification for cross-system personalisation. *Information Sciences*, 2009.

[14] Deepayan Chakrabarti, Yiping Zhan, and Christos Faloutsos. R-mat: A recursive model for graph mining. In *SDM*, 2004.

[15] Peter Christen. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer Science & Business Media, 2012.

[16] William Cohen, Pradeep Ravikumar, and Stephen Fienberg. A comparison of string metrics for matching names and records. 2003.

[17] Donatello Conte, Pasquale Foggia, Carlo Sansone, and Mario Vento. Thirty years of graph matching in pattern recognition. *International journal of pattern recognition and artificial intelligence*, 2004.

[18] Zhengyu Deng, Jitao Sang, and Changsheng Xu. Personalized video recommendation based on cross-platform user modeling. In *ICME*, 2013.

[19] Mohamed G Elfeky, Vassilios S Verykios, and Ahmed K Elmagarmid. Tailor: A record linkage toolbox. In *ICDE*, 2002.

[20] P ERDdS and A R&WI. On random graphs i. *Publ. Math. Debrecen*, 6:290–297, 1959.

[21] Ivan P Fellegi and Alan B Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 1969.

[22] Lise Getoor and Ashwin Machanavajjhala. Entity resolution: theory, practice & open challenges. *VLDB*, 2012.

[23] Oana Goga, Howard Lei, Sree Hari Krishnan Parthasarathi, Gerald Friedland, Robin Sommer, and Renata Teixeira. Exploiting innocuous activity for correlating users across sites. In *WWW*, 2013.

[24] Oana Goga, Patrick Loiseau, Robin Sommer, Renata Teixeira, and Krishna P Gummadi. On the reliability of profile matching across large online social networks. In *KDD*, 2015.

[25] Oana Goga, Daniele Perito, Howard Lei, Renata Teixeira, and Robin Sommer. Large-scale correlation of accounts across social networks. 2013.

[26] David J Hand and Robert J Till. A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine learning*, 2001.

[27] Tereza Iofciu, Peter Fankhauser, Fabian Abel, and Kerstin Bischoff. Identifying users across social tagging systems. In *ICWSM*, 2011.

[28] Paridhi Jain and Ponnurangam Kumaraguru. Finding nemo: searching and resolving identities of users across online social networks. *arXiv preprint arXiv:1212.6147*, 2012.

[29] Gunnar W Klau. A new graph-based method for pairwise global network alignment. *BMC bioinformatics*, 2009.

[30] Xiangnan Kong, Jiawei Zhang, and Philip S Yu. Inferring anchor links across multiple heterogeneous social networks. In *CIKM*, 2013.

[31] Hanna Köpcke and Erhard Rahm. Frameworks for entity matching: A comparison. *Data & Knowledge Engineering*, 2010.

[32] Nitish Korula and Silvio Lattanzi. An efficient reconciliation algorithm for social networks. *VLDB*, 2014.

[33] Shamanth Kumar, Reza Zafarani, and Huan Liu. Understanding user migration patterns in social media. In *AAAI*, 2011.

[34] Sebastian Labitzke, Irina Taranu, and Hannes Hartenstein. What your friends tell others about you: Low cost linkability of social network profiles. 2011.

[35] Silvio Lattanzi and D Sivakumar. Affiliation networks. In *STOC*, 2009.

[36] Chung-Yi Li and Shou-De Lin. Matching users and items across domains to improve the recommendation quality. In *KDD*, 2014.

[37] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 2007.

[38] Jing Liu, Fan Zhang, Xinying Song, Young-In Song, Chin-Yew Lin, and Hsiao-Wuen Hon. What's in a name?: an unsupervised approach to link users across communities. In *WSDM*, 2013.

[39] Li Liu, Cheung K. William, Xin Li, and Lejian Liao. Aligning users across social networks using network embedding. In *IJCAI*, 2016.

[40] Siyuan Liu, Shuhui Wang, Feida Zhu, Jinbo Zhang, and Ramayya Krishnan. Hydra: Large-scale social identity linkage via heterogeneous behavior modeling. In *SIGMOD*, 2014.

[41] Chun-Ta Lu, Sihong Xie, Weixiang Shao, Lifang He, and Philip S Yu. Item recommendation for emerging online businesses. 2016.

[42] Anshu Malhotra, Luam Totti, Wagner Meira Jr, Ponnurangam Kumaraguru, and Virgilio Almeida. Studying user footprints in different online social networks. In *ASONAM*, 2012.

[43] Tong Man, Huawei Shen, Shenghua Liu, Xiaolong Jin, and Xueqi Cheng. Predict anchor links across social networks via an embedding approach. In *IJCAI*, 2016.

[44] Lydia Manikonda, Venkata Vamsikrishna Meduri, and Subbarao Kambhampati. Tweeting the mind and instagramming the heart: Exploring differentiated content sharing on social media. *arXiv preprint arXiv:1603.02718*, 2016.

[45] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *ICDE*, 2002.

[46] Marti Motoyama and George Varghese. I seek you: searching and matching individuals in social networks. In *Proceedings of the eleventh international workshop on Web information and data management*, 2009.

[47] Xin Mu, Feida Zhu, Zhi-Hua Zhou, Ee-Peng Lim, Jing Xiao, and Jianzong Wang. User identity linkage by latent user space modeling. In *KDD*, 2016.

[48] Arvind Narayanan and Vitaly Shmatikov. Deanonymizing social networks. In *ISSP*, 2009.

[49] Arvind Narayanan and Vitaly Shmatikov. Myths and fallacies of personally identifiable information. *Communications of the ACM*, 2010.

[50] Yuanping Nie, Yan Jia, Shudong Li, Xiang Zhu, Aiping Li, and Bin Zhou. Identifying users across social networks based on dynamic core interests. *Neurocomputing*, 2016.

[51] Olga Peled, Michael Fire, Lior Rokach, and Yuval Elovici. Entity matching in online social networks. In *SocialCom*. IEEE, 2013.

[52] Daniele Perito, Claude Castelluccia, Mohamed Ali Kaafar, and Pere Manils. How unique and traceable are usernames? In *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2011.

[53] Christopher Riederer, Yunsung Kim, Augustin Chaintreau, Nitish Korula, and Silvio Lattanzi. Linking users across domains with location data: Theory and validation. In *WWW*, 2016.

[54] Yilin Shen and Hongxia Jin. Controllable information sharing for user accounts linkage across multiple online social networks. In *CIKM*, 2014.

[55] Rohit Singh, Jinbo Xu, and Bonnie Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences*, 2008.

[56] Shulong Tan, Ziyu Guan, Deng Cai, Xuzhen Qin, Jiajun Bu, and Chun Chen. Mapping users across networks by manifold alignment on hypergraph. In *AAAI*, 2014.

[57] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *WWW*, 2015.

[58] Jiliang Tang, Yi Chang, and Huan Liu. Mining social

media with social theories: a survey. *ACM SIGKDD Explorations Newsletter*, 2014.

[59] Andreas Thor and Erhard Rahm. Moma-a mapping-based object matching system. In *CIDR*, 2007.

[60] Jan Vosecky, Dan Hong, and Vincent Y Shen. User identification across multiple social networks. In *2009 First International Conference on Networked Digital Technologies*. IEEE, 2009.

[61] Duncan J Watts and Steven H Strogatz. Collective dynamics of small-worldnetworks. *nature*, 1998.

[62] Ming Yan, Jitao Sang, Tao Mei, and Changsheng Xu. Friend transfer: cold-start friend recommendation with cross-platform transfer learning of social knowledge. In *ICME*, 2013.

[63] Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social media mining: an introduction*. Cambridge University Press, 2014.

[64] Reza Zafarani and Huan Liu. Connecting corresponding identities across communities. *ICWSM*, 2009.

[65] Reza Zafarani and Huan Liu. Connecting users across social media sites: a behavioral-modeling approach. In *KDD*, 2013.

[66] Reza Zafarani and Huan Liu. Finding friends on a new site using minimum information. In *SDM*, 2014.

[67] Reza Zafarani and Huan Liu. Users joining multiple sites: Distributions and patterns. In *ICWSM*. Citeseer, 2014.

[68] Reza Zafarani and Huan Liu. Users joining multiple sites: Friendship and popularity variations across sites. *Information Fusion*, 28:83–89, 2016.

[69] Reza Zafarani, Lei Tang, and Huan Liu. User identification across social media. *TKDD*, 2015.

[70] Haochen Zhang, Min-Yen Kan, Yiqun Liu, and Shaoping Ma. Online social network profile linkage. In *Asia Information Retrieval Symposium*, pages 197–208. Springer, 2014.

[71] Haochen Zhang, Minyen Kan, Yiqun Liu, and Shaoping Ma. Online social network profile linkage based on cost-sensitive feature acquisition. In *Chinese National Conference on Social Media Processing*, 2014.

[72] Jiawei Zhang and Philip S Yu. Pct: partial co-alignment of social networks. In *WWW*, 2016.

[73] Jiawei Zhang, Philip S Yu, and Zhi-Hua Zhou. Meta-path based multi-network collective link prediction. In *KDD*, 2014.

[74] Jiawei Zhang and Philip Yu S. Multiple anonymized social networks alignment. In *ICDM*, 2015.

[75] Si Zhang and Hanghang Tong. Final: Fast attributed network alignment. In *KDD*. ACM, 2016.

[76] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S Yu. Cosnet: connecting heterogeneous social networks with local and global consistency. In *KDD*, 2015.

[77] Yuxiang Zhang, Lulu Wang, Xiaoli Li, and Chunjing Xiao. Social identity link across incomplete social information sources using anchor link expansion. In *PAKDD*, 2016.

[78] Xiaoping Zhou, Xun Liang, Haiyan Zhang, and Yuefeng Ma. Cross-platform identification of anonymous identical users in multiple social media networks. *TKDE*, 2016.

[79] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 2009.

Supervised Learning Techniques in Mobile Device Apps for Androids

Priyanka Basavaraju and Aparna S. Varde
Department of Computer Science, Montclair State University, NJ, USA
basavarajup1@montclair.edu, vardea@montclair.edu

ABSTRACT

Mobile devices have become an integral part of our daily lives. Most people carry smartphones today almost everywhere; and have other mobile devices such as tablets, often more convenient than full-fledged laptops for work transit, short trips etc. This had led to development of apps for mobile devices, easy to download and access anywhere anytime. An important field improving human experiences on mobile devices is machine learning. This constitutes technqiues involving acquisition of knowledge, skills and understanding by machines from examples, guidance, experience or reflection to learn analogous to humans. Among learning paradigms herein, supervised learning comprises situations where labeled training samples are provided to administer the process, making it more regulated, similar to human instructors providing such examples with notions of correctness to guide human learners. Supervised learning techniques are useful in designing mobile apps as they entail guided examples capturing specific human needs and their reasoning in activities, e.g., classification. This paper gives a comprehensive review of a few useful supervised learning approaches along with their implementation in mobile apps, focusing on Androids as they constitute over 50% of the global smartphone market. It includes description of the approaches and portrays interesting Android apps deploying them, addressing classification and regression problems. We discuss the contributions and critiques of the apps and also present open issues with the potential for further research in related areas. This paper is expected to be useful to students, researchers and developers in mobile computing, human computer interaction, data mining and machine learning.

1. INTRODUCTION

Many problems can easily be solved by computers, from simple computations that involve multiplying billions of numbers in few minutes to the complex computations of calculating the distance of asteroids in the galaxies. These computations seem quick and easy to computers when compared to the speed and processing power of humans. On the other hand, a simple task for a human is to recognize or differentiate between different classes of objects such as trees, cars, buildings and people; even though these may not always remain the same throughout. For example, a broken chair with three legs would easily be identified by a human as a chair, a STOP sign covered in snow is still clearly distinguished as the given sign, a dog wearing a floral headgear would be recognized as a dog etc. See Figure 1 for a few examples of objects in the real-world that are easily identifiable by human beings (Image Source: Google). This identification process is so intuitive that it seems almost effortless to humans. However, for computers, such tasks seem really complex to execute since things tend to change in the real world [20]. Consider the example of a broken chair. A computer might fail to recognize this as a chair unless it is programmed with the worst case scenarios (otherwise, if a piece of furniture has three legs, it is more likely to be identified as a stool than a chair). Hence, instead of writing specific programs to solve such invidiual problems, scientists try to make computers solve these catergories of problems using examples provided through certain techniques. Such an approach is a part of the field of machine learning, where machines can learn analogous to humans in guided and / or intuitive ways [1]. An effective comprehension of how to make machines, more specifically, computers learn has led to numerous new users of computers with new levels of competence and customization [10]. Furthermore, a detailed comprehension of data processing algorithms for machine learning has prompted a superior understanding of human learning capacities as well. Much research has been conducted to make computers learn nearly as well as humans. Algorithms have been created that are successful for certain types of learning tasks. For various problems ranging from text recognition, speech recognition and object recognition [20], it has been found that machine learning algorithms typically perform better than other prior computational approaches.

Figure 1. Example of objects as represented in real world

Accordingly, machine learning techniques have proved to be effective for mobile devices like smartphones, sensors, handheld and automotive computing systems. Smartphones are equipped with mobile sensors that are widely available, e.g., cameras, gyroscope, GPS and accelerometers that are becoming cheaper and widely available [11]. Thus, mobile application developers claim that a smartphone constitutes a *"sensor"* carried by humans that can be further exploited to provide even more services to users, e.g. safety features [12].

In order to validate the lateral growth of associated research publications, we searched publications containing the phrases "machine learning" and "mobile applications" in the digital libraries of ACM and IEEE, by filtering the results based on the year published, ranging from 2006 to 2016. The results including machine learning implementation in mobile devices and mobile applications can be seen in Figure 2. It can be seen that there is a peak in 2008, possibly due a surge in the use of mobile devices by then [9]. The publication trend continues to be high, depicting the continued interest in research and development in the area of machine learning for mobile apps.

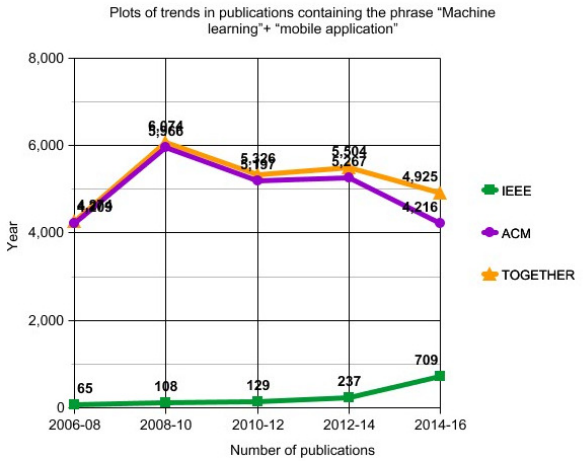


Figure 2. Publication trends in the areas of machine learning and mobile applications

It is thus useful to present a survey on machine learning techniques for mobile apps, which provides the motivation for our paper. In this survey paper, we fathom the techniques behind the tremendous growth in the machine learning field for decades, with particular emphasis on its use in recent technologies that seem almost indispensable, namely, mobile device apps. In order to make the survey more specific, we focus on supervised learning. This encompasses guided learning wherein labeled training data along with notions of correctness are provided in advance for the machine to learn. It is analogous to being taught by human teachers who are very specific in providing guided examples for learning with clear ideas of the right and the wrong that helps students learn in a regulated manner (as opposed to making them learn intuitively through trial and error, i.e., unsupervised learning, also a machine learning paradigm). It has been found that supervised learning techniques [15] are the more commonly used ones in mobile apps. This can be justified as follows. Mobile apps are designed for convenient, quick and easy access by humans. The guided learning processes of humans therefore seem to be more useful in incorporating their judgemeant and reasoning in the design of the apps. For making this survey paper even more focused, we consider the Android family of apps. This is because, based on the statistics, we observe that smartphones occupying around 52.8% of the worldwide market share today are Android devices [19]. Hence, to the best of our knowledge, a review of Android apps would be somewhat more useful. Thus, this survey paper concentrates on supervised learning techniques and their implementation in building Android mobile device apps. However, many of the details presented here with respect to techniques and applications can be helpful to other app developers and related professionals as well.

The rest of the paper is organized as follows. Section 2 provides an overview of supervised learning with selected approaches, namely, Support Vector Machines, AdaBoost, Artificial Neural Networks, Gaussian Process, Decision Trees and Naïve Bayes, all useful in mobile apps. Section 3 gives details of six Andorid apps in the six different areas of pedestrian help, fruit ripeness, application prediction, malware scanning, house pricing and healthcare monitoring that would be beneficial to Android users. It includes the data description, app learning details as well as contributions and critiques. Section 4 provides a discussion on the apps considering their utility value and open issues for further research. Section 5 states the conclusions.

2. SUPERVISED LEARNING METHODS

Machine learning paradigms fall in the categories of supervised learning, unsupervised learning and reinforcement learning as shown in Figure 3 herewith [10]. Among these, the supervised learning paradigm, our focus in this survey, consists of algorithms that learn a model from externally supplied instances of known data and known responses to result in a general hypothesis, such that the learned model can be used over new data to predict responses about future instances [15]. This is illustrated in Figure 4 (Image Source: MATLAB). In other words, such learning is performed with examples. The system is given existing data with examples that have been assigned one or more labels based on their input values and response values. These examples are used to train the system for the function it is expected to learn. The machine learns the given function and is then able to work on new data [15]. A simple analogy is a child identifying new fruits after being trained using existing fruits with their correct classification given in advance, e.g., fruit *x* is an apple, fruit *y* is an orange etc.

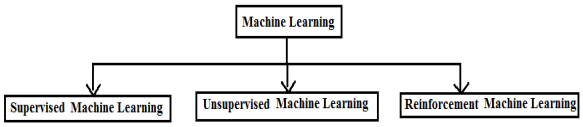


Figure 3. Machine learning paradigms

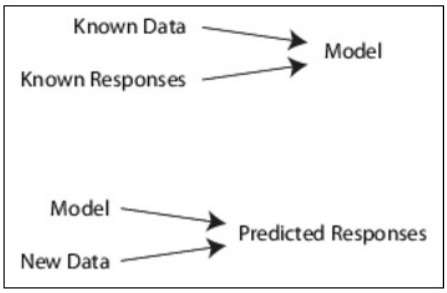


Figure 4. Graphical representation of supervised learning

Supervised learning includes two categories of algorithms:

1. Classification: This is used when the response values have differentiate between various discrete classes.
2. Regression: This is used when the response values are continuous, e.g., numerical data.

Some of the techniques in the supervised learning paradigm include: Support Vector Machines (SVM), Naïve Bayes Classifier, k - Nearest Neighbors (kNN), Decision Trees, Linear Regression and Nonlinear Regression, to name a few [15]. We focus on describing selected learning techniques that are useful in the mobile apps we discuss in this paper.

2.1 Support Vector Machines (SVM)

The technique of Support Vector Machines or SVM is a supervised learning method for classification proposed by Vladimir Vapnik and colleagues [6]. The goal of the SVM is to build decision planes or hyper planes that classify all training vectors into two different classes. The data for the training consists of a set of points that form the vectors. The learning process includes first training a support vector machine and then cross validating the classifier, explained as follows. Suppose we have two features x1 and x2 and have to classify the given input into either of the classes. The SVM draws hyperplanes that can separate the two different classes. The best classification is made by choosing the hyperplanes that leave maximum margins from all the classes.

Figure 5 provides a depiction of this. The SVMs make use of a nonlinear mapping function (Φ) that transforms the input data space to data in a feature space in such a manner as to make a problem linearly separable. SVM learning is used for data analysis and pattern identification, useful for classification and regression analysis [6]. Some practical applications of SVM are gene expression, text classification, image identification etc. SVMs are considered to provide good generalization accuracy and are fast in learning.

However, training a support vector machine causes a quadratic optimization problem with bound constraints and a linear equality constraint. Given this, there are many other issues to consider in designing the SVM learner. In particular, when training the machine with large sets of examples, the optimization techniques for general quadratic programs quickly become intractable in memory and time requirements.

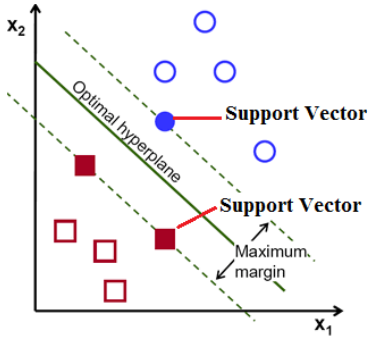


Figure 5. SVM showing the hyperplane

2.2 AdaBoost

The AdaBoost process is short for Adaptive Boosting and is a classification technique proposed by Yoav Freund and Robert Schapire [8]. Boosting is an ensemble method of creating a sequence of complex predictors made from blocks that are simple predictors. AdaBoost makes the model learn as follows. It sequentially trains a new model based on the errors of the previous model. In general terms, it builds a predictor model using any prediction algorithm and find the errors of the model results; it then focuses on these errors while building the next prediction model. This procedure helps in identifying the data points that are hard to predict and helps the latter classifiers in getting these examples correct. In the end, all the classifier models are combined together using a weighted combination. Models with errors are called weak classifiers. This learning process is represented in Figure 6.

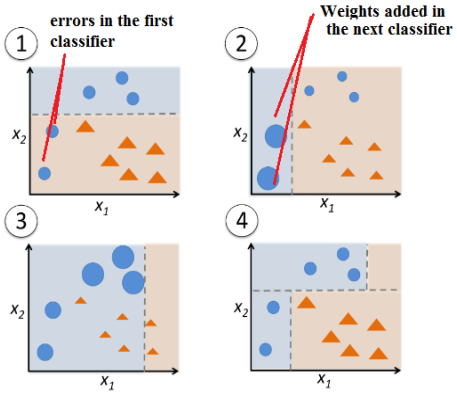


Figure 6. Graphical representation of AdaBoost

AdaBoost is often used for the following reasons [8]

1. Speed: It is fast, simple and easy to program.
2. Flexibility: It can be combined with any learning algorithm to form weak classifiers.
3. Effective: It produces the desired impacts, provided can consistently find rough rules of thumb.
4. Versatile: It can be used with data that is textual, numeric, discrete etc.

The classifier model with AdaBoost is easy to build; thus it is are highly used in applications with very large data sets. It requires no complicated iterative parameter estimation.

2.3 Artificial Neural Network (ANN)

Artificial neural networks serve as a model of the real neural network that exists in the human brain. They provide a general, practical method for learning real-valued, discrete-valued, and vector-valued functions from given training examples. The model is influenced by the observation of a human brain system that consists of very complex webs of interconnected neurons. Artificial Neural Network models comprise nonlinear computational components or processing units that work in parallel and are arranged in patterns resembling biological neural networks [4]. Nodes that form the computational components are connected by links that have weights. Once the model has been trained and tested, the predicted output should

ideally correspond to the actual output from the approximate mapping. An example of ANN appears in Figure 7.

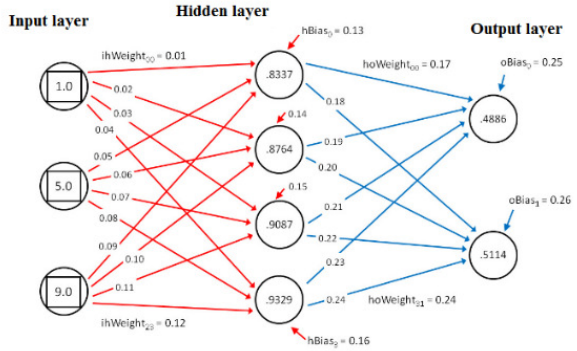


Figure 7. Example of an Artficial Neural Network

The ANN approach is known for its robustness to errors in the training data. A few of the applications that implement ANN recognize handwritten characters, human speech and robot control strategies. A very well-known ANN algorithm is the classical backpropagation method which uses gradient descent to tune network parameters in order to best fit a training set of input-output pairs [10, 4].

2.4 Gaussian Process

The Gaussian Process forms a simple regression model. A Gaussian Process (GP) can be defined as a collection of random variables, such that any finite number of them have a joint Gaussian distribution specified by its mean and covariance functions, i.e., vector and matrix, respectively [1]. Note that a Gaussian distribution also known as a normal distribution is a common continuous probability distribution often used to depict real-valued random variables whose distributions are unknown. It is a continuous function that approximates the exact binomial distribution of events. Training the GP model involves a model selection or a discrete choice between different functional forms for mean and covariance functions, followed by the adaptation of the hyper parameters of these functions. Hence, problems such as predicting prices, weather forecasting, disease tracking can deploy regression, since they typically involve estimating values that fall within a continuous range of outputs [1]. Figure 8 shows an example of a Gaussian model. In this example, the model is used for word prediction among a continuous range of possible words.

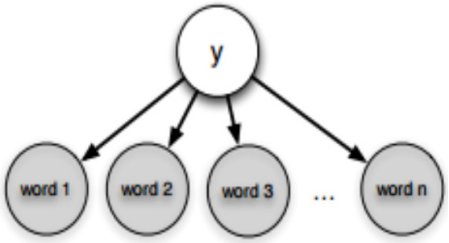


Figure 8. Example of a Gaussian model

2.5 Decision Tree

A Decision Tree learner is one of the most widely used practical methods for inductive inference [17]. This efficient nonparametric method, can be used for both classification and regression. It has a hierarchical data structure implementing a divide-and-conquer strategy for supervised learning where the local region is identified by a sequence of recursive splits in a smaller number of steps. A decision tree is composed of internal decision nodes, i.e., it classifies instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. Figure 9 portrays a typical decision tree example used in software tools and text books [17]. Here various aspects of the weather are used to predict whether a game of tennis should be played.

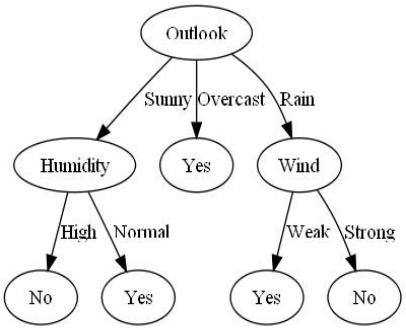
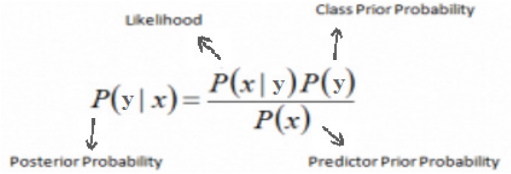


Figure 9. Decision Tree example for tennis playing

There are many problems that can be addressed using the decision trees. Here are some characteristics of problems that typically incorporate decision tree classifiers. 1. Instances are represented by attribute-value pairs; 2. The target function has discrete output values; 3. Disjunctive descriptions may be Required; 4. The training data may contain errors; and 5. The training data may contain missing attribute values [10].

2.6 Naïve Bayes

The Naïve Bayesian classifier is based on Bayes theorem proposed by Thomas Bayes (1702 - 1761); hence it can be stated that this is a conventional paradigm that has been around since the late 18th century. Bayesian classifiera are a part of the probabilistic family of classifiers [10]. They predict the class of a sample based on probability, i.e., they first predict the probability of the sample itself and then pick the class that has the highest probability given the observation. This is shown in the equation next.



The given equation provides a way of calculating posterior probability P(y|x) from P(y), P(x) and P(x|y). In practice, the denominator is often ignored since it does not affect the classification; the values of the features are given, so that the denominator is effectively constant.

Naive Bayes is a generative model that computes its predictions by modeling each class. Thus, given datasets with many attributes, it would be computationally expensive to compute P(x|y). In order to reduce the computation in evaluating P(x|y) P(y), the classifier assumes that the effect of an attribute value on a given class is independent of the values of the other attributes. This is called the class conditional independence assumption. It is made to simplify the computation involved and in this sense, it is considered *"naïve"*. The individual probabilities P(x1|y), P(x2|y), . . ., P(xn|y) can be estimated from the given training set.

Naïve Bayes classifiers are often used for the following reasons:

1. The model is simple, fast, space efficient and is convenient to build.
2. It handles real, discrete and streaming data well.
3. It is particularly used for dealing with large datasets as it does not need iterative parameter estimation.
4. The model is not sensitive to irrelevant features.
5. This classifier performs surprisingly well and outperforms other classification methods in many complex real-world situations.

The Naïve Bayes classifier serves as a popular method for text categorization, spam filtering and sentiment analysis [10]. Other applications include building recommendation systems to filter unseen data and real-time multiclass prediction systems such as face recognition.

Having thus described several pertinent classifiers, we now explain their usefulness in mobile app design with interesting real-world examples from the Android app family.

3. IMPLEMENTATION IN MOBILE APPS

This section describes a the implementation of some machine learning techniques for six mobile apps in the Android family of devices. In the first three apps shown here in the areas of pedestrian help, fruit classification and application prediction respectively, the model is trained by being embedded in the mobile devices. For the remaining three apps, in the areas of malware scanning, house pricing and healthcare monitoring respectively, the mobile devices are used for visualizing the data whereas the model training and data processing occur at the backend on local application servers.

3.1 Pedestrian Help App

According to recent data from the nonprofit National Safety Council, traffic fatalities raise closely up to 14% every year. Deaths and injuries in road traffic accidents pose a serious threat to global health and have a negative impact on social and economic progress [13]. Considering the death rate of traffic accidents involving pedestrians in 2009, a group of researchers have developed an Android mobile app for pedestrian help. This targets mobile users who get actively engaged with their mobile devices while walking on the streets. The developed app called *WalkSafe* is aimed to enhance pedestrian safety, by helping people that walk and talk, through alerts with a vibrate feature [21]. The app gives out a vibration signal once it detects an approaching vehicle, thus giving pedestrians the indication to be cautious. Since it has to predict accurate results in the real time, the app implements machine learning algorithms within itself to train the model. This application is a joint work of students, faculties and researchers from Computer Science Dartmouth College, USA and University of Bologna, Italy [21].

The contributions of their work are as follows. 1. The app includes intricate design with vehicle detection and pedestrian alert. 2. It incorporates machine learning algorithms on the phone to detect the front views and back views of moving vehicles. 3. It exploits the phone API (Application Programming Interface) to save energy by running the vehicle detection algorithm only during active calls and using mobile sensors such as a back camera to detect vehicles thet could be approaching the user. Figure 10 illustrates the WalkSafe app along with a suitable image of an approaching vehicle.

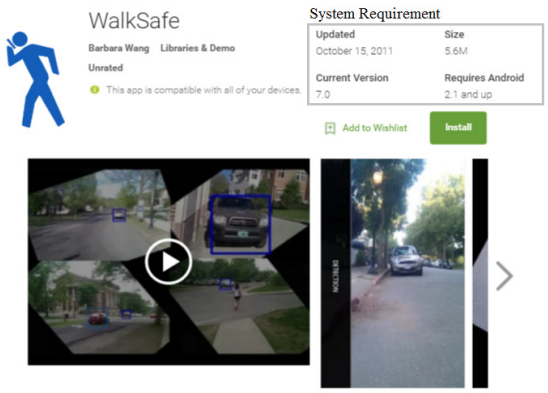


Figure 10. WalkSafe App with vehicle image

Their training dataset includes 7000 positive images (displaying the rear or front view of cars) and 3023 negative images (displaying pictures that show side views of cars or random urban environments).

As with many machine learning implementations, the design includes two stages, namely, the offline stage for training and the online stage for real time application, as described next.

Offline Stage: The core technology used in this project is the image recognition, that involves high computational intensive processing. Hence the application bases its vehicle recognition process on a model that is first trained offline; then uploaded and used for the online vehicle recognition. The process involves the following three steps. The first step consists of dataset building. In this step, images from different sources are collected to serve as the basis for further work by providing the training set. The second step involves training for image preprocessing. Most machine learning applications that involve images have to undergo preprocessing to extract the appropriate image with feature descriptors to train the classifier. Thus, 1032 positive samples are collected on which image processing techniques such as crop, resize, greyscale conversion and intensity variation are applied. This step creates 7000 different variations of the collected set, with the same sizes and alignments.

The third step entails feature extraction and classification. Here, a cascaded classifier that consists of several simpler classifiers is constructed based on Haar-like features. This is a well-known feature extraction technique often used because of its low computational property [21]. It works as follows. The process involves predefined sets of features, slid through the image from the top left to bottom right corner, reading pixel values. These features consider adjacent rectangular regions in a detection window; then sum up the pixel intensities in these regions and calculate the differences between them. The resulting value is called a Haar-like feature. The classifiers at every stage of the cascade pipeline "vote" on the classification of the same sample. These votes are then weighted according to the AdaBoost algorithm. The cascade classifier is trained on 20 stages. In addition to this, Decision Trees are used as weak classifiers to make the appropriate classification. The resulting Decision Tree is then uploaded and used for the online vehicle recognition.

Online Stage: In the online stage, similar steps are followed for the vehicle detection model that runs on the mobile device. In addition, we have the following functions for real-time processing in the app. The *image capture* function is for enhancing the phone performance and battery lifetime. The design team ensures triggering the image capture mode on the phone only during active phone calls. The application captures the information on its surroundings in the form of video frames. The *image preprocessing* function is available in real-time and its working is similar to that of the offline stage. The *feature extraction* function is for extracting important image features in real time. The relevant features are extracted and passed to the Decision Tree designed at the offline stage. If the prediction turns out to be positive, the application alerts the user by vibrating to inform the user about an approaching vehicle. Thus the online and offline stages together contribute to the effective performance of the WalkSafe app for Android mobile devices in the area of pedestrian help and safety.

The WalkSafe app is surely a useful innovation. However, we find that it could have certain drawbacks. For example, using the image processing technology could drain out the battery life of the mobile application completely if there are implementation problems. Also, observing the evaluation results provided in this work [21], the WalkSafe application detects cars upto 50 meters away from the pedestrians and provides just a few seconds for the pedestrian to react to the vibrate alert. The speed of the approaching car matters in such a situation, yet that aspect is not discussed in the paper [21]. Since the app involves camera, it will drain the phone camera early. This is reflected in the results shared by the team. The app has close to 10000 downloads and has received mixed reviews from the actual mobile phone users. Some of the users complain about no proper documentation on how the app works. All these issues provide the potential for further research, e.g., taking into account vehicle speed, providing user-friendly documentation and addressing the image processing issues such that they do not adversely affect battery life. However, on the whole, it has been found that WalkSafe is a good app for Android users.

3.2 Fruit Ripeness App

As mentioned in the introduction, smartphones come with sensors such as microphone, camera, GPS and accelerometer that enable them to interact with their environment in many ways. The application herewith exploits the nature of the microphone sensors. The role of the microphone is to collect audio signals and process them along with the necessary transformations when users are on the call. This nature of the microphone is used in an Android app called *Watermelon Classify* that is used to estimate the ripeness of watermelons [22]. During watermelon harvest, the actual ripeness of a given watermelon is identified by thumping on the watermelon to analyze the sound made by it. Note that once harvested, the watermelon will stop its ripening process and hence it becomes critically important to judge whether it is ripe or not prior to picking. A watermelon is around 90% water and the rest consists of sugar, an important indicator of the watermelon ripeness. The sugar content increases when watermelon becomes ripe and this leads to sound frequencies that differ in acoustic response from unripe watermelons. Knowing these factors, a group of researchers have developed an Android app that can differentiate between ripe and unripe watermelons. When a watermelon is patted, it produces an acoustic signal that can be captured by the mobile microphones. The signal undergoes various transformations to get its noise removed. Important features such as zero crossing rate, short-time energy, spectrogram, mel-frequency cepstral coefficients and brightness are extracted. These features are labeled and provided to a Support Vector Machine to train the model, for two different classes, ripe or unripe. The features extracted are known to accurately distinguish the watermelon classes based on earlier work of other researchers in the field. Once the model is trained, new input can be provided and the app can accurately classify the watermelon. Figure 11 shows the home screen of the Watermelon Classify app for Androids.

Figure 11. Layout of the Watermelon Classify app

The contributions of this work as seen in [22] are listed herewith. 1. The work involves understanding and extracting key acoustic features related to the ripeness of watermelons using relatively low-complexity measures including zero crossing rate, short-time energy and others. 2. The app implementing the machine learning technique of SVM for supervised learning is suitably harnessed here for effective results. 3. A crowdsourcing application has been designed by collecting user feedback to improve the classification methods. 4. The app achieves an accuracy of 89.9 % in correctly classifying ripe and unripe watermelons, which is very good.

The datasets in this application have been trained by thumping acoustic response signals. The relevant data has been collected

by patting on ten ripe and unripe watermelons, respectively. The features extracted for the training purposes are: zero crossing rate (ZCR), short-time energy (STE), sub-band short-time energy ratio, mel-frequency cepstral coefficients (MFCC), brightness and spectrogram [22].

While this is a very interesting app, it has some pitfalls, as we have noticed. It is a crowdsourcing application, in which the model has been trained with test data collected from 20 watermelons and has then been released to the users. The model which is embedded with the application files, receives the training data whenever a user runs the app, this process helps in refining the accuracy. With all the implementation, the app still provides 89% of accuracy, which is better than other other existing solutions. However, it may still not be enough to keep the harvest at stake for new farmers. Thus, further work with respect to crowdsourcing for enhancing the training data would help to make the app more accurate and generic.

3.3 Application Prediction App

This app targets users who have multiple applications installed on their smartphones and find it difficult to reach a particular application in the device, scrolling through the screens. The team in [3] has developed a *Home Screen App*. This home screen app acts as an intelligent layer between the underlying mobile operating system and the user interface that automatically organizes the installed apps in a more intelligent and personalized way; hence it also predicts the next app the user will deploy in near future.

The contributions of this work include the following. 1. For the implementation of the proposed idea, the team has conducted exhaustive studies on several domain-specific features for app usage prediction. 2. Considerably high accuracy is achieved using machine learning techniques. 3. The work proposes a new algorithm, i.e., a Parallel Tree Augmented Naïve Bayesian Network. This works around removing the strong assumptions of independence in Naïve Bayes for more accurate predictions.

In order to serve as the data for this app, two set of features are collected, i.e., human-engineered and automatic. For the human-engineered features, the data is collected by the Aviate sample log, where Aviate is a Yahoo based application. The log sample collected includes 60 million records with around 200,000 anonymized users, having a total of more than 70,000 unique apps. For the other type, namely, the automatic features, the sets of features collected include the real time spatiotemporal contexts as sensed by the home screen app.

Given this data, the next application is predicted incorporating two aspects. The first aspect is based on popularity, which assumes that the given user would be installing a concerned app for the first time and hence relies on its recognition among other users. The second aspect is based on the sessions, where various features are extracted based on the given user's spatiotemporal context collected using mobile sensors. These include Time, Latitude, Longitude, Speed, GPS Accuracy, Context Trigger Context Pulled, Charge Cable and Audio Cable. The app uses Parallel Tree Augmented Naive Bayesian Network (PTAN) proposed in this work as the prediction model. It builds upon Naïve Bayes and removes the strong assumptions of independence in Naïve Bayes, thus exploiting correlations among attributes. The training procedure involves two phases. Phase 1 involves structure training that learns the structure of the Bayesian network. Phase 2 comprises parallel parameters estimation that encompasses deducing a set of parameters for each individual user. Based on all these aspects, the next app that the user would install is predicted to assist the user in decision-making by taking into account overall popularity of apps as well as the individual user preferences.

This app thus constitutes research on how the home screen can be taken to the next level of predicting the app a given user might install in the near future. Note that the Aviate app is used here to collect log data and understand the attributes. Figure 12 shows an example of the home screen mobile app where 12(a) shows the changes through the day as automatically detected while 12(b) shows the auto-categorized apps that the user would be likely to install.

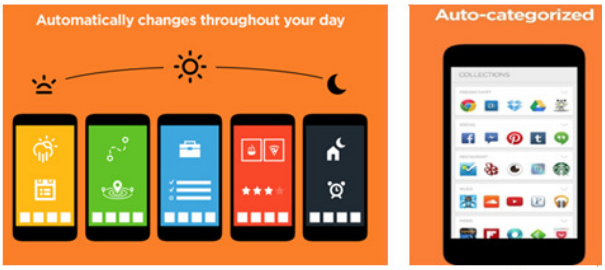


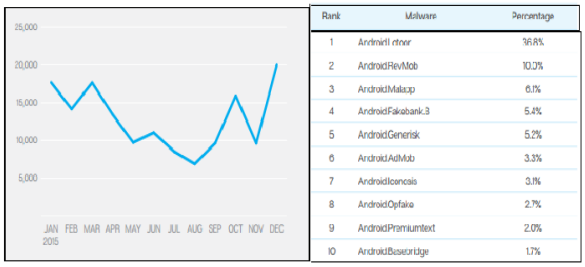
Figure 12. Example of home screen mobile app
(a) automatic changes in the day
(b) auto-categorization for app prediction

Like most other apps, this home screen app for application prediction also has its critiques. For this app, organizing the application based on user interests when the user installs the application for the first time is challenging. In this work, the team has already done enormous research in addressing the issues that occur due to the app or user cold start problem. The app cold start problem is that which occurs when a user installs a new app on their device. Likewise, the user cold start problem arises when a user installs and open the home screen app for the first time. Further research in this area could enhance the quality of the app even further.

3.4 Malware Scanning App

Malware applications tend to gain access to mobile devices with the intent of stealing data, damaging the device, annoying the user and other such issues. The attacker, i.e., the sender of malware manipulates the user into installing various harmful applications and / or increases unapproved remote access by exploiting the vulnerability of the device. Such applications give no lawful notification to the user. Their threat includes Trojans, worms, botnets, and viruses. Figure 13 depicts Android malware, such that 13(a) shows the projection of the overall malware volume while 13(b) shows a list of top ten malware examples. In order to address the malware problem, the design team in [5] claims that their proposed system can effectively perform real time malware detection on the Android platform. This application is a cloud based Android malware

analysis service called *ScanMe* mobile. The objective of this app is to bring about awareness of the Android application package (APK) files to the intended users before they make any attempt to install them on the given Android device. This is achieved by sharing a detailed report of the analyzed files to the user.



Rank	Malware	Percentage
1	Andmidl.ctnor	36.8%
2	AndroidRevMob	10.0%
3	AndroidMalapp	6.1%
4	AndroidFakebank.3	5.4%
5	AndroidGenerisk	5.2%
6	AndroidAdMob	3.3%
7	AndroidIconosis	3.1%
8	AndroidOpfake	2.7%
9	AndroidPremiumtext	2.0%
10	AndroidBaseBridge	1.7%

(a) (b)

Figure 13. Android malware
(a) Projection of volume
(b) Top ten examples

There are important contributions of this work as stated next. 1. Though there exist other applications for malware detection, this work implements machine learning techniques that use training data to predict the malicious contents in the APK file, thus making the app more specific and useful. 2. ScanMe provides a plug and play solution that allows users to install their own local sandbox systems. These are application environments to analyze data locally by the user to facilitate faster and easier analysis of the APKs. 3. Apart from compiling a comprehensive report of the analyzed APK, the app shares such a report by publishing it through a Web interface that helps in bringing awareness of such malware applications to other users. 4. This app provides a static as well as dynamic analysis of the APK file which gives helpful information to the user. 5. The app incorporates advanced technologies and tools such as Google App Engine (GAE) datastore and its Website, Google Cloud Endpoint technology and Google Cloud Messaging (GCM) services to provide the best facilities to the users.

In order to provide data to train the model, the team has collected 100 known malware samples and 100 benign applications from the Android Malware Genome Project [2]. The training focuses on the Manifest.xml file which is the application file present in the APK. The team has a list of 206 Android permissions that are placed in three categories: *normal, dangerous and signature/system*. As the name suggests, the latter two categories can cause the most harm and have the most risky privileges.

The layout design of the ScanMe mobile app is portrayed in Figure 14. When users install the app for the first time, they are asked to register the device GCM service to establish a communication channel between the Android client application and GAE instances. If users choose to register the application, then it utilizes Google Cloud Endpoint technology to integrate mobile clients with a Google App Engine backend. This feature is implemented to conduct analysis of the APK file in the background and notify the users with messages when the analysis is completed. If the users fail to register, they have to login to the Website to obtain the results. Once the application is registered, the users can choose to upload one or more downloaded APK files (read from a Secure Digital memory card) into a sandbox server. This sandbox server is an Ubuntu system that analyzes the APK file. The system first runs a file, which extracts the permissions from the manifest file of the APK and sends it to the detection module.

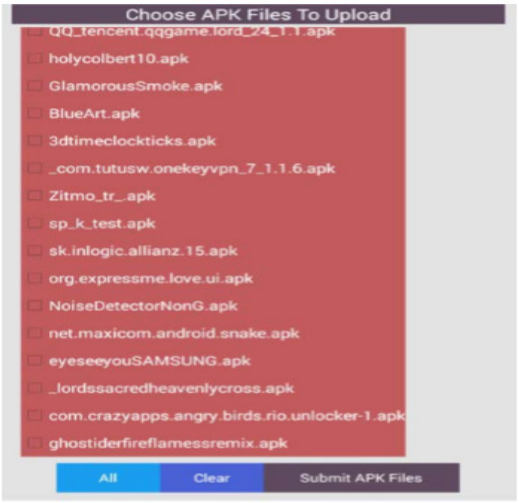


Figure 14. Layout design of ScanMe mobile app

The detection module is trained as follows. The sandbox identifies all the permissions and separates them into the three givevn categories of normal, dangerous and signature/system. It then lists all the custom permissions, features, services, receivers and activities, and writes all the data to the APK's report file. This data is fed to the Artificial Neural Network based malware detection classifier and is mapped to a format required by the ANN. The module learns patterns of permissions and gets trained to model a good classifier. The trained model is then used to carry out malware classification in the online detection process. Once the report is compiled at the sandbox, it is sent to the Google App Engine data store.

We have noticed that the paper in this work [5] has detailed information for implementing the approach, including clear architecture and methods. However, though the paper is well organized, it does not provide enough details on the features obtained to train the model, even though they seem critical. ScanMe Mobile users can locally scan APK files on the SD (Secure Digital) memory card of their phone. Yet, when the users download a mobile application, they are directed to the internal memory of the mobile device and there is a need to make necessary changes in the settings file forcing them to download APK files in the SD card. This app would have a lower utility value if users are unaware of this scenario. The app is also not found in the Google Play Database to verify the scan results. All these issues present the potential for further work, to enhance the appeal the malware scanning application.

3.5 House Pricing App

The housing market has always been a topic of interest. The trends in housing markets in any country are very interesting to house owners or potential buyers. Moreover, the trend also

affects the economic situations of the respective countries. There are many Web applications available that provide the trends of the housing markets and these have inspired the development of mobile apps for the same. We review herewith an app designed specifically for London users [14], given that London is a city with a very high cost of living and yet is one of the most sought after cities for residential purposes, having a diverse international population. The objectives of this app are to: 1. Predict future price changes of London properties that fall within a user-defined search radius. 2. Refer locations to the user that can be considered within a search radius based on the budget provided by the user.

This app has multiple contributions 1. The app handles a large set of data and produces accurate results in terms of future price changes of London properties. This is accomplished by the implementation of machine learning techniques. 2. The prediction is often trained with updated values of the housing market; hence it provides the latest information. 3. Most data processing occurs at the server side of the application, thus decreasing the computational task on the client side, thereby not adversely affecting the mobile device performance.

In order to generate the data in the app, the design team has collected the details of all property transactions that have occurred in Greater London from 1995 to 2013, bringing up the total to more than 2.4 million in the period. The data collected has been classified into different feature values, ID (Transaction ID), Date (Date processed, Month of transaction, Year of transaction), Transaction Price, Property classification (Type, Build, Tenure etc.), Address information (Postcode, Local authority, Full address, Borough, Ward etc.).

Figure 15 shows the layout design of the house pricing app [14]. The app implementation involves two frameworks: client side and server side. The client side is developed to collect the information from the user, the area code, other search refinement information and budgets. All this data is then passed to the server side.

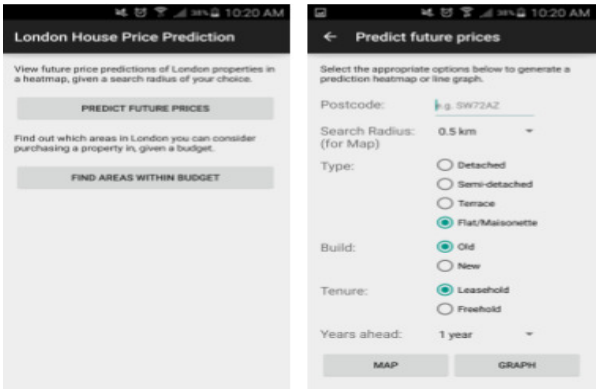


Figure 15. Layout design of house pricing app

The data gets processed at the server side and the results are displayed back to the client side in the form of map and graph information. The server side of this application contains two servers. The Web server acts as a middle layer and the application server is where the computations are handled.

The data training process involves the following [14]. In order to capture geographical variation, the map is represented in terms of latitudes and longitudes that are converted to appropriate postcode addresses. The values for the collected feature sets are noted. Since the data is too large, the designers use the technique of dividing the data into map grids and train the prediction model seperately. In the end, the predictions for each test input are made by combining the predictions of all the local models. This data is stored at the server end. The data is then sent to the application server where the model is trained and stored by Web server which acts as a middle layer to transfer the data.

It is noteworthy to observe the critiques of this app. The house pricing app deals with large sets of data, which is good for designing an effective classifier. However, the app may not be ideal for predicting the trends as the data is not updated on a regular basis. Note that the trends in the housing market can change within a day. According to their paper [14], it takes a day to train the model again with the new data since there are only a few servers. This flaw of a less number of servers has to addressed for catering better to the users. Also, the future price search options in the current app could be enhanced to provide more refinement choices for the users. These issues offer the scope for further work.

3.6 Healthcare Monitoring App

There are applications of machine learning techniques in the field of medicine. Sleeping disorder is an important problem; if not identified and treated right, it might lead to various health issues including chronic disorder, heart failures and in the worst case death [18]. According to the latest statistics from a leading hospital, around 70 million Americans suffer from disorders of sleep and wakefulness. Most of them happen to be undiagnosed. Researchers in [16] have contributed to this area by building a mobile app that can help people track sleeping patterns and take immediate action upon seeing disturbances. This app can also be used by doctors to remotely monitor their patients who may have sleep disorders.

This app entails significant contributions. 1. The work here involves developing a low-cost system for a robust, non-invasive multimodal sleep monitoring. 2. The app uses machine learning to automatically analyze the collected data, recognize sleep patterns and track the users' breathing behavior. 3. This particular app is also available for Apple iPhones in addition to Androids, which is an added advantage, increasing clientele.

Regarding data collection, this sleep pattern monitoring app is a continuation of prior work by the same researchers. They have designed a pressure mat capable of producing 1728 features based on the 1728 pressure sensors embedded inside the mattress. The values here can range from 0 to 100 for each feature. Thereafter, ten important data features can be selected such that they clearly identify different classes. The app also uses a Web camera for monitoring breathing behavior of the users. It observes chest movements by capturing frames of images for comparing the data with a given threshold value. Apart from these values, the team has 500 feature vectors collected by 5 different sleeping positions. This is done by

making various volunteers sleep on the mattress and thereby monitoring their sleeping positions, considering people of different sizes and heights. There are totally 2000 samples in the dataset. This app works in two stages, offstage and onstage as described next. Figure 16 portrays the layout design of this sleep pattern monitoring app [16].

Offstage: In this stage, the concerned data on sleep positions and related features is collected and the model is trained based on the input values and response values. Here, they use SVM, given that the app involves five different classes pertaining to sleep positions, namely, left, up, down, side and sitting. It is to be noted that SVM is usually used with two classes, however, when there are more than two classes, multiclass SVMs are used, as in this application. The breathing patterns are also monitored in the offstage.

Onstage: In this stage, the data is aggregated and then displayed to the people who are responsible for monitoring the sleep patterns. The app displays a graphical representation of the monitor information. This can be utilized by the concerned app users as well as by the doctors who would be interested in analyzing any disturbances.

Figure 16. Layout design of sleep monitoring app

Given that this app is used in healthcare, its critique becomes even more significant, since the data is highly sensitive. We have noticed the following negative aspects that deserve further attention. The app provides a graph that makes it quite hard for any new user to analyze the data. Moreover, the monitoring process involves setting up a huge lab which might make it very expensive to use. The application works well if the set-up is at the hospital and the users are given thorough knowledge on how to track their information to share it with the doctors. The issues listed herewith motivate further research in the area, in order to provide greater enhancement.

Having thus described various mobile apps with respect to their features, learning techniques and contributions, we now present a general discussion on these.

4. DISCUSSION

Mobile devices have become such an important part of our daily lives that they seem almost indispensable today. Accordingly, apps, providing quick and easy access on the fly to a variety of services, have become very popular on mobile devices such as smartphones. Among various mobile devices, the Android family is found to have the greatest market share in the world. Android OS is the most popular across the globe, considering the various mobile operating systems including iOS by Apple, Blackberry OS etc. Thus, mobile apps on Android devices deserve special attention. It is found that machine learning and data mining techniques, especially those in the category of supervised learners are very useful in designing mobile apps. Considering these factors, we have focused our survey on the use of supervised learning techniques in mobile device applications, with particular emphasis on Android apps.

Various apps are surveyed here and are found to be useful in several aspects of everyday life. We next discuss the utility value of each app from a user perspective.

The *pedestrian help app*, namely, *WalkSafe* is likely be very helpful to people in busy cities, e.g., New York City, London, Mumbai, Shanghai and so on. It is found that people are often on their smartphones, while waiting for buses, crossing streets, halting at signals etc. Thus, a safety signal given by WalkSafe would prove useful here. It could also be useful in smaller towns, however people are more likely to drive there than walk and also roads there are more clear.

The *fruit ripeness app* called *Watermelon Classify* seems a boon to farmers as it would assist them in detecting whether a given watermelon is ripe. Also, it could help shoppers who are not experts in estimating ripeness which could be useful as it would avoid buying low quality fruit or being cheated by shopkeepers. It would also serve as a simple learning tool for novices in the area of horticulture. The principles used here could potentially be extended to some other types of fruits as well.

The *application prediction app* could be helpful for almost all users, especially to those that tend to install several apps. This *Home Screen app* would help such users get organized by predicting the next app most likely to be useful to them based on overall popularity and individual user preferences. It would thus reduce excessive installation of apps, as many of them would seem low on the priority list. It would also save time by giving users suggestions rather than having them browse all available apps before making an installation decision.

The *malware scanning app* here aka the *ScanMe mobile app* appears to be almost a necessity on mobile devices. This could be considered analogous to having virus scanners and other malware detection tools on desktops and laptops that seem to be a must today. As mobile devices gain popularity, attackers find ways to cause trouble to good users. Thus, protective apps on handheld devices are very useful as they could prevent serious attacks and reduce the threat of harmful software.

The *house pricing app* appears to be a gift to those interested in seriously buying a house or window shopping for one. It could

also be useful to renters who could be potential home-owners. The *London housing app* described here would give users in that city an idea of what to expect in the housing market, preparing them for an informed discussion with a realtor, property owner or agent. Having such an app handy on their phone would help them get the information at-a-glance and help them negotiate better. On the other hand, the sellers could also benefit from this in addition to buyers by getting to know the market prices before offering a quote on their house. It would help them have more well-guided discussions with prospective clientele as well. While London has been selected here due to being an expensive yet popular metropolis for house-buying, the discussions applied here would be useful in potential housing apps for other towns and cities as well.

The *healthcare monitoring app* reviewed herewith is by far the most interesting and useful among the six apps discussed. Since it monitors *sleep patterns* of the users, it could help in early detection of sleep disorders, thereby preventing further health problems. This app is usable on Apple and Android devices, thus enhancing its utility value. Given that it is a medical app, the nature of the data is very sensitive, and this would be a risk-intensive app that needs to be used with caution, using doctor advice as needed, for greater effectiveness. While this app, like the others here, has it advantages, it also has certain concerns. These concerns present open issues for future work.

Accordingly, we now list some open issues pertaining to the various apps presented here that could potentially lead to further research. These are as follows.

- Incorporating vehicle speed while giving a vibrate signal to warn of a moving vehicle in the WalkSafe app
- Performing image processing in such a manner as to enhance battery life in this pedestrian help app
- Improving the crowdsourcing for the Watermelon Classify app to increase ripeness estimation accuracy
- Extending the app for ripeness of other fruits taking into account aspects of Watermelon classification
- Further addressing the app cold start problem in the Home Screen app (when the app is new to a given user), thus providing better service
- Fully solving the user cold start problem (when the user is new to a given app), in order to augment the usage of the application prediction app
- Conducting deeper studies on the verification of the malware scanning results in the ScanMe mobile app
- Making the download of this malware scanning app more user-friendly in order to enhance its utility
- Giving frequent data updates to the London housing app to incorporate recent trends, thus improving its own usefulness and also exemplifying the design for other house pricing apps
- Improving future price searches in the current app for offering more refinement choices to various users in this house pricing app and thus other similar apps
- Augmenting the interpretation of the results in the sleep pattern app, given that its present graph display is difficult to understand, especially for new users
- Heading towards making this healthcare monitoring app more cost-effective, since its current version seems expensive to implement, with lab costs etc.

Another point to note is that the healthcare app described here is generic to the Apple and Android families of mobile devices. It would be interesting to provide this facility in some other apps as well to make them more appealing to a wider range of users across multiple platforms. This calls for further research.

Hence, the various open issues listed herewith provide the potential for future work in the respective areas. They motivate further research and development that entails supervised learning techniques and mobile app design. This would involve studies by data mining and machine learning researchers in conjunction with those in fields such as human computer interaction (HCI) and mobile computing. Mobile device usage is ever-growing and thus inspires the development of more apps as well as the enhancement of existing apps, taking into account these and other related aspects.

5. CONCLUSIONS

In this survey paper, we present a brief overview of various supervised learning techniques useful in the implementation of mobile apps. We also provide examples of a few interesting mobile apps with specific reference to the Android family of handheld devices. We describe the design layout of the apps, and the respective learning technique(s) as deployed within the apps. We also outline a list of contributions of each app as entailed in the corresponding piece of research leading to its development. Furthermore, we offer a critique of each app from a user perspective. We also present a discussion on the apps overall, entailing their utility value and putting forth open issues that could potentially lead to further research. This survey paper would be useful to various professionals in areas such as human computer interaction (HCI) and mobile computing. It would also be useful to the machine learning and data mining communities, from the research as well as the development angle.

6. REFERENCES

[1] Alpaydin, Ethem, *Introduction to Machine Learning,* MIT Press, 2014.

[2] *Android Malware Genome Project*, URL reference: http://www.malgenomeproject. Org/

[3] Baeza-Yates, Ricardo., Jiang, Di., Silvestri, Fabrizio, and Harrison, Beverly, "Predicting The Next App That You Are Going to Use", *ACM International Conference on Web Search and Data Mining (WSDM),* 2015, New York, NY, USA, pp. 285-294.

[4] Bishop, Christopher. M., *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[5] Cole, Yevgeniy., Zhang, Hanlin., Ge, Linqiang., Wei, Sixiao., Yu, Wei., Lu, Chao., Chen, Genshe., Shen, Dan., Blasch, Erik, and Pham, Khanh D., "ScanMe mobile: a local and cloud hybrid service for analyzing APKs", *ACM Conference on Research in Adaptive and Convergent systems (RACS)*, 2015 New York, NY, USA, pp. 268-273.

[6] Cortes, Corinna and Vapnik, Vladimir. "Support Vector Networks", *Machine Learning*, 1995, 20: 273-297.
[7] Felt, Adrienne Porter., Finifter, Matthew., Chin, Erika., Hanna, Steve, and Wagner, David, "A survey of mobile malware in the wild", *ACM workshop on Security and Privacy in Smartphones and Mobile devices (SPSM)*, 2011 New York, NY, USA, pp. 3-14.
[8] Freund, Yoav and Schapire, Robert, "A short introduction to boosting", *Journal of the Japanese Society for Artificial Intelligence*, 1999, 14: 771-780.
[9] *History of Mobile Phones*, URL reference: http://www.knowyourmobile.com/nokia/nokia-3310/19848/history-mobile-phones-1973-2008-handsets-made-it-all-happen
[10] Michalski, Ryszard S., Carbonell, Jamie G. and Mitchell, Tom. M. *Machine learning: An Artificial Intelligence Approach*. Morgan Kaufmann, 1985.
[11] *Mobile App Survey*, URL reference: https://www.sans.org/reading-room/whitepapers/analyst/2013-mobile-application-survey-35080
[12] Nasar, J., Hecht, P., and Wener, R., "Mobile telephones, distracted attention, and pedestrian safety", *Accident Analysis and Prevention*, 2008, 40(1), 69-75.
[13] National Safety Council, URL reference: http://www.nsc.org/pages/home.aspx?var=mnd
[14] Ng, Aaron, and Deisenroth, Marc, *Machine Learning for a London Housing Price Prediction Mobile Application*, Technical Report, June 2015, Imperial College, London, UK.
[15] Omary, Zanifa., Mtenzi, Fredrick, "Machine learning approach to identifying the dataset threshold for the performance estimators in supervised learning", *International Journal for Infonomics*, 3:314–325, 2010.
[16] Papakostas, Michalis., Staud, James., Makedon, Fillia, and Metsis, Vangelis, "Monitoring breathing activity and sleep patterns using multimodal non-invasive technologies", *ACM International Conference on Pervasive Technologies Related to Assistive Environments (PETRA)*, 2015, New York, NY, USA, Article 78, 4 pages.
[17] Quinlan, J.R., "Induction of Decision Trees", *Machine Learning*, 1986, 1: 81-106.
[18] *SleepMed Publication*, URL reference: http://www.sleepmedsite.com/page/sb/sleep_disorders/sleep_statistics
[19] *Smartphone Market*, URL reference: https://www.comscore.com/Insights/Rankings/comScore-Reports-January-2016-US-Smartphone-Subscriber-Market-Share
[20] Viola, Paul, and Michael Jones, "Rapid object detection using a boosted cascade of simple features", *Computer Vision and Pattern Recognition*, 2001, 1: 511–518.
[21] Wang, Tianyu., Cardone, Giuseppe., Corradi, Antonio., Torresani, Lorenzo, and Campbell, Andrew T., "WalkSafe: a pedestrian safety app for mobile phone users who walk and talk while crossing roads", *ACM Workshop on Mobile Computing Systems & Applications (HotMobile)*, 2012, New York, NY, USA, Article 5, 6 pages.
[22] Zeng, Wei., Huang, Xianfeng., Arisona, Stefan Müller, and McLoughlin, Ian Vince, "Classifying watermelon ripeness by analysing acoustic signals using mobile devices". *Personal & Ubiquitous Computing*, 2014, 18(7): 1753-1762.