IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 7, 2020, accepted April 11, 2020, date of publication April 20, 2020, date of current version May 1, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2988728

Evaluation of Li-Based Battery Current, Voltage, and Temperature Profiles for In-Service Mobile Phones

HAYDER ALI[1], (Student Member, IEEE), HASSAN ABBAS KHAN[1], (Member, IEEE), AND MICHAEL G. PECHT[2], (Life Fellow, IEEE)

[1]Department of Electrical Engineering, School of Science and Engineering, Lahore University of Management Sciences, Lahore 54792, Pakistan
[2]Center for Advanced Life Cycle Engineering (CALCE), University of Maryland, College Park, MD 20742, USA

Corresponding author: Hassan Abbas Khan (hassan.khan@lums.edu.pk)

This work was supported by the Center for Advanced Life Cycle Engineering (CALCE) at the University of Maryland and Its over 150 Funding Companies.

ABSTRACT A battery's state of charge or runtime, and state of health or life, will depend on the product's discharge current over time. For a mobile phone, the discharge current depends on the specific apps that are operated. This paper presents an experimental study to measure and evaluate the operational charge/discharge profile, temperature and terminal voltage of six Android apps; WhatsApp, Facebook, Facebook Messenger, Instagram, Snapchat, and TikTok on smartphones. The results show how the discharge current required by an app's operation, will affect the battery runtime and life, due to the combined effect of discharge current and temperature.

INDEX TERMS Android apps, C-rates, in-service performance, Li batteries, smartphones.

I. INTRODUCTION

The number of mobile (smartphone) users has surpassed 3.5 billion in 2020, which is 42% of the total global population [1], [2]. These users increasingly demand smartphones with improved functionality and performance to support gaming and social media applications (apps) [3]. This in turn stresses the battery system, which is widely considered as one of the weakest link in a modern mobile phone [4], [5]. In fact, some energy-intensive apps may reduce the battery operation time to as short as several hours for users [6], and were noted to be the likely cause of the Apple IOS slowdown problem in 2017 [7]. Krause *et al.* [8] showed the negative and highly non-linear impact of C-rate on the life of Li batteries. Dong *et al.* [9] also noted that accelerated battery aging occurred due to high discharge C-rates.

Reliability issues have also been highlighted with high C-rates as it causes high temperature which accelerates battery aging phenomena and at times causes thermal runaway [9], [10]. A significant drop in terminal voltage is also reported with high C-rates which impacts the smartphone operation [7]. Under high discharge C-rates, for a degraded battery with large internal resistance, heat dissipation becomes a major issue and unexpected smartphone shutdowns have been triggered even when there was substantial charge remaining in the battery [11]. This also results in lower effective battery capacity than the manufacturer's ratings.

Many efforts are made to address this problem of high C-rate in modern smartphones. For instance, some manufacturers suggest improving the algorithms for battery management system (BMS), but it can result in slowing down the operating system [7]. Another suggested method is to improve the system efficiency through low-power processors [5], [8]. Some researchers address the problem of high C-rate by improving battery capacity through improving the chemical operation of the battery by replacing the conventional electrodes of Li battery with the superior capacity (molybdenum disulfide) electrodes with enhanced functional capacity [12], [13].

With regards to in-use operation of smartphones, various studies suggests that traditional testing and qualification of Li-ion batteries may not be able to simulate the actual use of a smartphone in real environment [14], and that manufacturers must address the actual power dissipation of the apps as well as the batteries and smartphones circuitry [15], [16]. While some models have been developed

The associate editor coordinating the review of this manuscript and approving it for publication was Derek Abbott.

to profile power usage of smartphones, they rely on indirect methods to evaluate discharge patterns using aggregated current regimes [14], [17]–[20]. Direct measurement of discharge spike currents have also be conducted, but these rely on external sensors that can add significant measurement errors [17]–[19].

This paper presents the use of built-in mobile sensors to directly measure current discharge profile, terminal voltage, and battery temperature, when the phone is operated using six different apps: i.e., WhatsApp, Facebook, Facebook Messenger, Instagram, Snapchat, and TikTok. Our approach solves the problem of indirect measurements, and parameter averaging [21]. The proposed method uses the manufacturers' provided built-in mobile sensors data through an application program interface (API). We developed a relevant Android package kit (APK) to decipher battery parameters for the apps under investigation. We also provide the algorithm and API that we used (see Appendix) so other researchers can assess other mobile phones.

The rest of this paper is composed as follows. Section 2 summarizes Android-based phones and APIs that collect operation data. Section 3 presents discharge profile analyses of two case-study phones and discusses the impact of the six apps on battery drainage and temperature. Section 4 presents the conclusions.

II. ASSESSMENT OF SMARTPHONES FOR IN-SERVICE OPERATION

The availability of in-service battery data, including current, voltage, and temperature, depends on manufacturer-provided software interfaces as well as the capability of the built-in sensor hardware in a smartphone. Higher resolution of data is key for accurate parameter assessment, for instance, a resolution of microamperes for peak current drawn is more useful as compared to milliampere or ampere level resolution. Similarly, the update time of parameters (minimum interval after which each value is updated in the software APIs) is also important to accurately gauge the parameter variations.

Seven Android-based smartphone manufacturers were evaluated[1] using the developed APK to check the granularity (resolution and update time) of data for discharge current, terminal voltage, and battery temperature through the manufacturer-provided API (software). All the seven smartphone manufacturer support our developed APK, but most of the manufacturers do not provide high-granularity data to evaluate detailed current drawn from the batteries [22]. For instance, the current discharge profile for Samsung J7 is shown in Fig. 1(a), and no change in current was observed with varying app operation, although some variation in temperature was observed.

Similarly, a change in the current for Huawei Nexus P6 was averaged out for about 20 s, not allowing detailed information

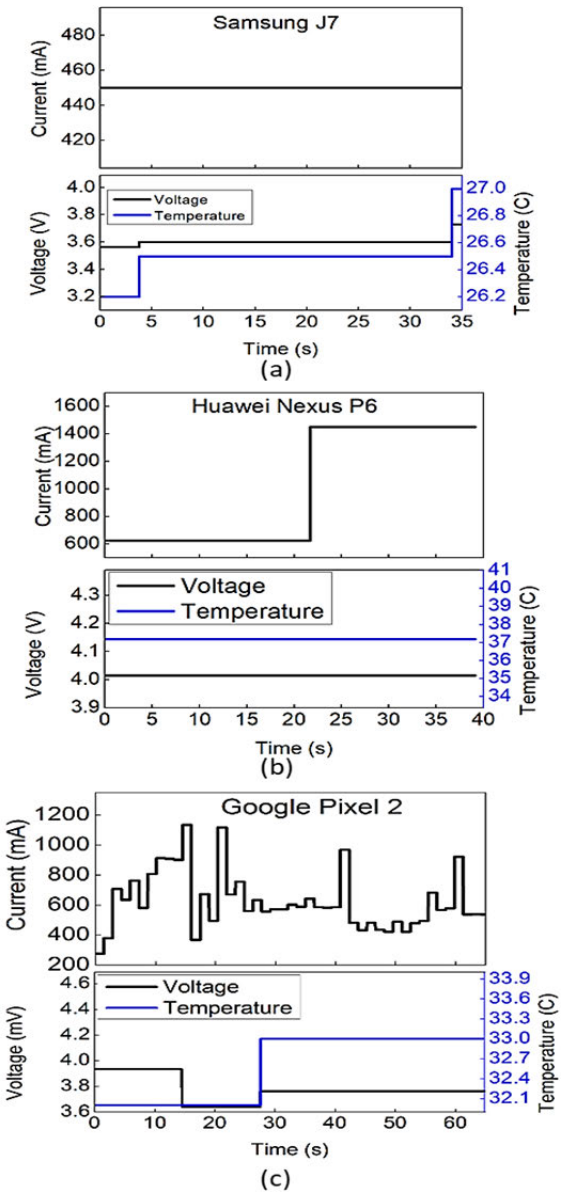


FIGURE 1. Current, voltage, and temperature measurements using the developed app: (a) Samsung (b) Huawei and (c) Google.

of current spikes with a particular app operation. Further, the update of voltage or temperature is large and as a result no change was observed. In the case of a Google phone (Pixel 2), the current discharge profile and temperature/voltage profile averaged out after every 2 s and 15 s, respectively, as shown in Fig. 1(c). Like Samsung and Huawei, it also limits the resolution on current drainage values.

The results for all seven manufacturers are summarized in Table 1. Because the highest resolution of electrical current and the fastest update time was measured with the Vivo (V9) and Motorola (Droid Turbo), those two smartphones were selected for further study. Both unused phones under test were fully charged to compare against nameplate battery capacities to check for any unusual degradation, and both phones passed

[1]Note that Apple® is not in the list. The reason is that although most companies allow APIs to share data at some levels, Apple does not. With the release of iOS 10 in September 2016, Apple decided to completely hide this information from developers and end-users [7], [20].

TABLE 1. Summary of the electrical current, terminal voltage, and temperature profile obtained by the developed APK.

Phone Manufacturer	Model	Application supported	Current		Voltage	Temperature
			Resolution	Value update time	Resolution	Resolution
Samsung	J7 [23]	Yes	Not available		mV	0.1 °C
Google	Pixel 2 [24]	Yes	µA	2 s	mV	0.1 °C
Huawei	Nexus P6 [25]	Yes	mA	20 s	mV	0.1 °C
Oppo	A57 [26]	Yes	mA	5 s	mV	0.1 °C
Vivo	V9 [27]	Yes	µA	0.01 s	mV	0.1 °C
Motorola	Droid Turbo [28]	Yes	µA	0.175 s	mV	0.1 °C
Xiaomi	Mi A1 [29]	Yes	µA	3 s	mV	0.1 °C

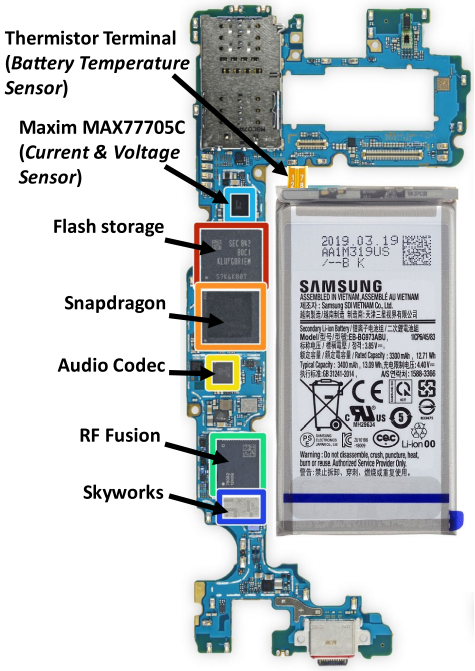


FIGURE 2. Internal circuit diagram of a high-end (Android-based) smartphone. This figure shows the typical placement of battery temperature sensor and current/voltage sensing chip alongside the battery.

the test. With regards to the market share, Vivo sold over 102 million phones worldwide in 2018 with a market share of about 7%, whereas Motorola sold over 35 million phones in 2018 [30].

Whereas a smartphone has numerous temperature sensors implemented across the device, battery temperature is typically sensed through a thermistor close to the terminals as shown for a typical assembly in Fig. 2 [31], [32]. In our developed app, the 'EXTRA_TEMPERATURE' API command is used, which specifically measures the battery temperature [33]. Battery current and voltage are generally monitored through a single chip such as Maxim MAX77705C, shown in Fig. 2 [34]. As the sensors are already embedded (built-in) in the mobile phone circuitry, our developed app uses a non-intrusive method to measure battery parameters through the manufacturer-provided APIs [22].

III. RESULTS AND DISCUSSION

To obtain the discharge profile, the Vivo (V9) and Motorola (Droid Turbo) phones were initially charged to full battery capacity (100%), and then each app was run for approximately 12-18 min, followed by the next charge cycle back to full charge. We found that 10+ min is sufficient to test all major functionalities in an app. Subsequently, the phone was discharged using the second app, and so on. During this time, most of the general capabilities of a specific app were tested. For example, in the case of WhatsApp, all features such as texting, photo send/download, voice note send/download, voice call, and video call were tested. The detailed features tested during the experiment for all apps are summarized in Table 2. Throughout the experiment, the experimental conditions were kept consistent to minimize variations from one app to the next. For instance, only the testing app (Snapchat or Facebook or others – only one at a time) along with our developed APK was run. All apps were tested on phones connected through WiFi and the sound volume and screen brightness were kept at 100% throughout the experiment ambient temperature at 26 ± 0.5°C.

For both phones under test, Fig. 3 shows the operation of various apps (i.e., WhatsApp, Facebook, Facebook Messenger, Instagram, Snapchat, and TikTok) on the current drain, the terminal voltage of the battery, and the battery temperature. The negative current represents charging, whereas the positive current shows discharging (Fig. 3(a)). Several positive spikes during charging appear which were triggered when the screen was turned on for viewing of the battery status. Terminal voltages (Fig. 3(b)) show an increasing trend while charging and a decreasing trend during the discharging, as expected. However, the rapid change observed in the current profile during charging is not seen in the voltage profile due to lower update time for the voltage (a software limitation

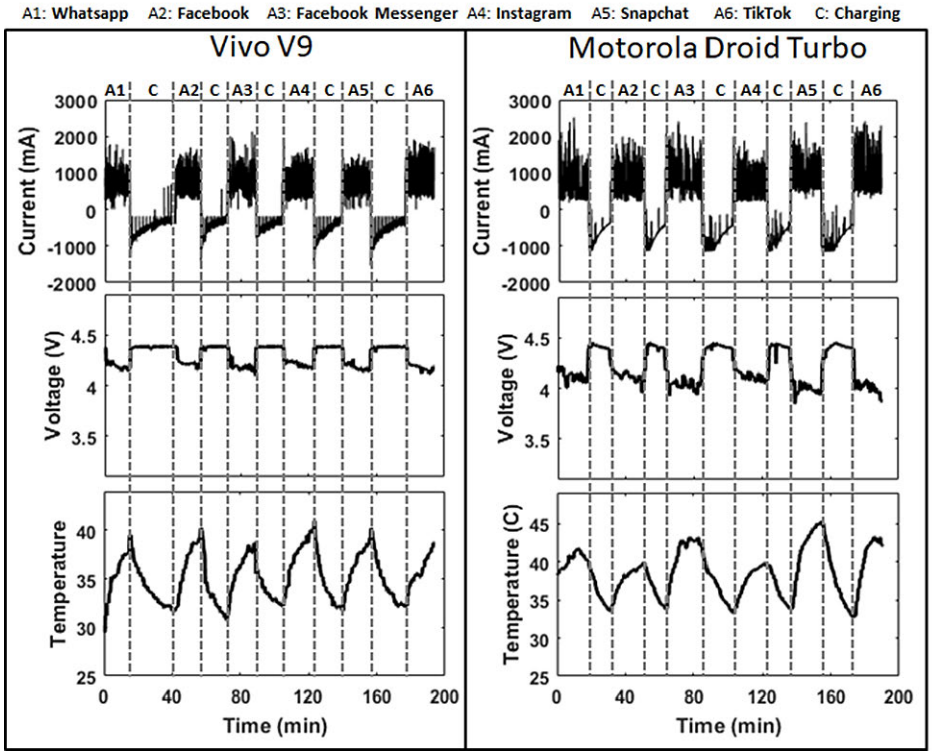


FIGURE 3. Charging/discharging profile of Vivo V9 and Motorola Droid for (a) current profile, (b) battery terminal voltage profile, and (c) battery temperature profile.

TABLE 2. Units features tested during experiment for all apps properties.

App	Features under observation
WhatsApp	Texting, send/receive photos and videos, voice note, 1 min video call twice, and 1 min audio call twice
Facebook	Scrolling, like and comment on post, and watch videos
Facebook Messenger	Texting, send/receive picture and videos, voice note, 1 min video call twice, and 1 min audio call twice
Instagram	Scrolling, like and comment on post, and watch videos
Snapchat	Texting, send/receive photos and videos by using different filter effects
TikTok	Scrolling, like, comment, and make TikTok videos

by the manufacturer). Each sensor has a separate resolution and update times as provided by the manufacturer and summarized in Table 1 where, for most of the manufacturers, the update time of the voltage sensor is high (several seconds) as compared to the current sensor (milliseconds).

The result in Fig. 3(a) show that the discharge rate depends on the apps and is also dictated by mobile phone technology (e.g., processor chipset, camera technology, screen type, and resolution). For instance, the processor used in Vivo V9 (Qualcomm MSM8953-Pro Snapdragon 626) is more energy-efficient than the one used in Motorola Droid Turbo (Qualcomm APQ8084 Snapdragon 805) due to improved process technology node from 28 nm (Motorola) to 14 nm (Vivo) [27], [28]. Further, the screen resolution for Motorola is 565 ppi in comparison with 400 ppi for Vivo. A higher resolution requires more pixel per inch and ultimately consumes more energy per inch. So, the behavior of apps vary significantly on various phones and ultimately the same app can perform in a varied manner on different mobile manufacturers [27], [28].

The temperature profile for both phones is shown in Fig. 3(c). The temperature consistently increases during app usage and decreases during the charging process (in comparison with the discharge operation under app usage) [35], [36]. For the Vivo V9 phone, the temperature is fairly consistent during the use of various apps and, during the charging interval, the temperature normalizes to below 32 °C with the highest battery temperature of about 40 °C. However, for the Motorola phone, usage of Snapchat, TikTok, and Facebook Messenger particularly increases the temperature beyond 43 °C and at least 3 °C higher than all other app usage or charging conditions. The amount of current spikes from the battery is also higher for Snapchat, TikTok, and Facebook Messenger for the Motorola phone, which could significantly affect aging and other related issues [37], [38]. To describe the current discharge profile, the mean current information is not sufficient to characterize these apps. In addition, current spike information is also essential as it can have many unintended consequences such as temperature rise and lifetime deterioration.

In order to evaluate this further, we calculate the probability density functions (PDFs) and normal distribution functions (NDFs) of the discharge current profiles for all the apps

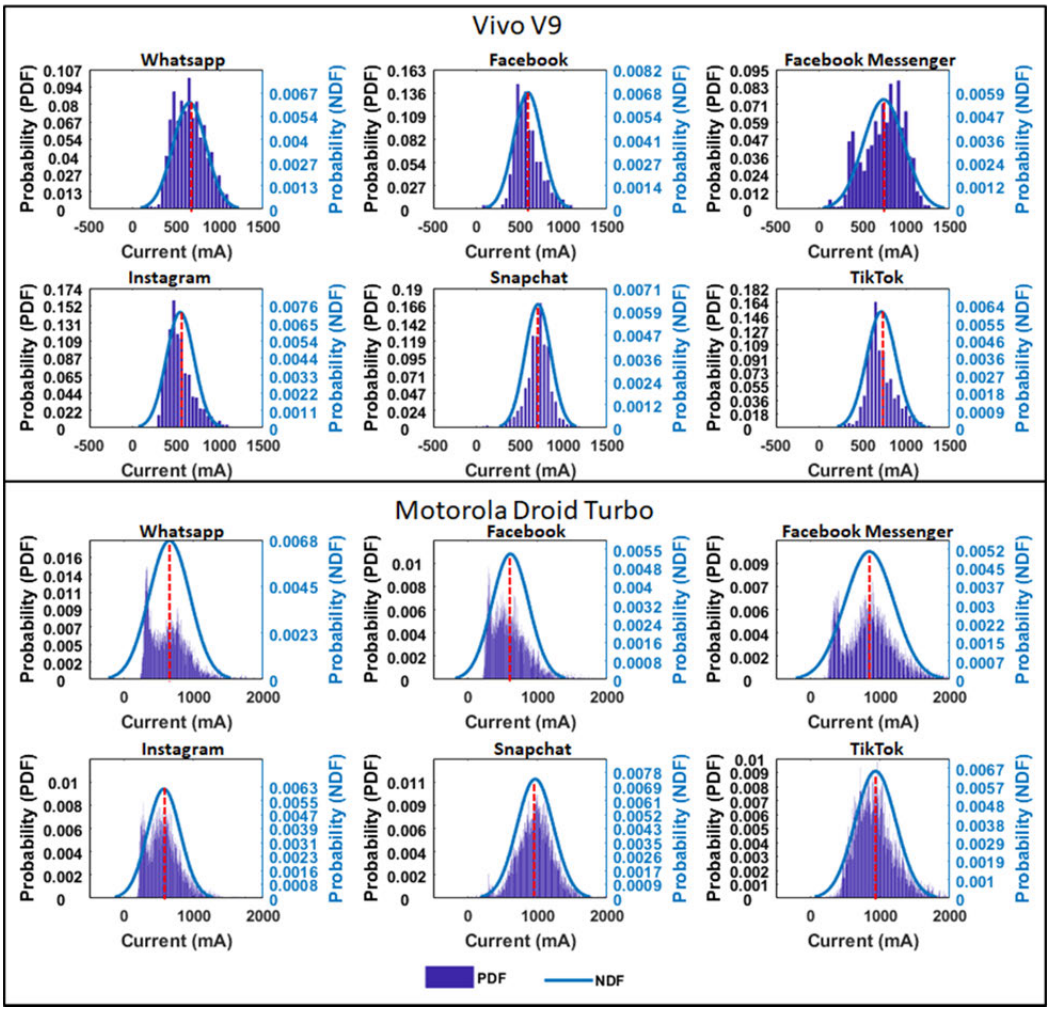


FIGURE 4. Probability density functions (PDFs) and normal distribution functions (NDFs) for all apps for the battery discharge cycle.

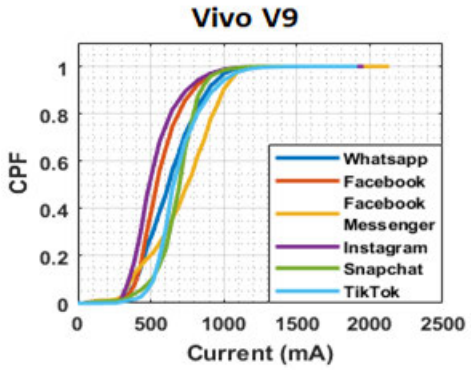
under investigation in Fig. 4. The PDFs/NDFs facilitates the reader to understand and reconstruct the results for further studies.

A PDF specifies the probability distribution for various levels of current drawn while using various apps. The highest point on the PDF relates to the mean value of current. In the case of the Motorola phone, several apps such as Snapchat have a higher mean value (967 mA) compared to Instagram (572 mA) with results summarized in Table 3. Higher currents drawn from the battery are undesirable from a heat dissipation perspective. In addition, high-frequency current spikes are not suitable for optimum battery operation as they rapidly degrade battery life [37], [39]. The values of standard deviation (SD) along with mean currents is also summarized in Table 3 for reconstruction of discharge profiles.

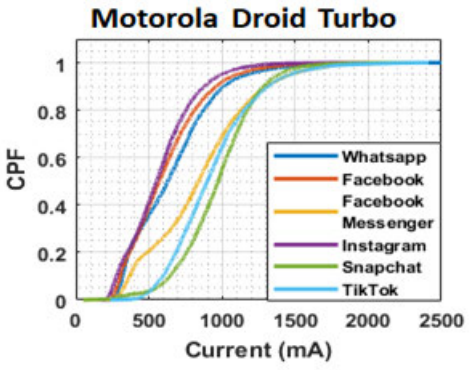
Experiments showed that the discharge profile for the six apps is significantly different on each phone. With regards to C-rate discharge on phones, higher rates are detrimental for both battery runtime and lifetime. Typically, up to 0.3C is considered a safe limit for accelerated lithium-ion battery degradation [40]–[42]. Therefore, the probability of current spikes above 0.3C is shown in Table 3 to assess the power usage of various apps under consideration. It can be noticed that the performance of energy-intensive apps such as Facebook Messenger and Snapchat vary from Vivo to Motorola. For instance, in the case of Vivo, the mean

TABLE 3. Summary of mean current drawn and probability of current spikes above 0.3C for Vivo and Motorola phones.

Apps	Vivo		Motorola	
	Mean current (mA) / SD (mA)	Probability of current spikes over 0.3C (%)	Mean current (mA) / SD (mA)	Probability of current spikes over 0.3C (%)
Whatsapp	653/190	5.8	653.8/293	5
Facebook	588/163	2.6	610.6/264	3.7
Facebook Messenger	740/235	16.7	845/353	17.2
Instagram	545/159	2.1	572/235	1.8
Snapchat	705/148	2.8	967/262	21.2
TikTok	713.4/170	8.8	933/292	20.1



(a)



(b)

FIGURE 5. Cumulative probability function for all apps under test. The y-axis shows the probability of not exceeding a current threshold on the x-axis.

current and the probability of current spikes above 0.3C for Facebook Messenger are 740 mA and 16.7%, respectively, which is higher than Snapchat with a mean current and probability of current spikes at 705 mA and 2.8%, respectively. Further, higher current spikes may not necessarily result in higher mean currents as seen with Snapchat, which has a mean current of 705 mA with a current spike probability of 2.8% compared with WhatsApp with a mean current of 653 mA and a current spike probability of 5.8%. Higher current spikes are associated with several well-documented issues such as degradation, capacity fade, and premature shutdowns [43]–[45].

With regards to Motorola, the performance of Facebook Messenger is relatively better than Snapchat as well as TikTok. The mean current for Facebook Messenger is 845 mA, compared to 967 mA for Snapchat and 933 mA for TikTok. Similarly, the current spike probability for Facebook Messenger is lower at 17.2% compared to 21.2% for Snapchat and 20.1% for TikTok. Fig. 5 plots the cumulative probability function (CPF) to link the current discharge profile with the probability of occurrence. The CPF also shows a variation in amplitude of the current spikes for all the apps. A spread across the x-axis (current) shows a higher probability of occurrence for a larger spike, which is undesirable. It is therefore evident from Fig. 5 that the performance of the Vivo V9 is better than the Motorola Droid Turbo for the use of apps under consideration.

With regards to the battery capacity and runtime of smartphones, power-hungry apps could reduce the operation time significantly. For instance, for the Motorola Droid Turbo model with a 3900 mAh battery [28], talk time of up to 48 h is suggested by the manufacturer, whereas our data shows that using Snapchat and TikTok for less than 5 h would fully drain the battery. On the other hand, using Instagram for at least 7 h is possible with the in-use discharge pattern. In addition to the average current drawn, C-rating is another critical parameter that plays an essential role in battery life span [41]. Typically, a C-rating of 0.1C to 0.4C for lithium-ion batteries is recommended by many manufacturers [42], [46], [47]. So, it is important to optimize app development to limit the battery current spikes. The C-rating of batteries for the phones investigated in this paper is not specified by the manufacturers, but our analysis shows that for the Motorola phone, current spikes of over 0.6C were observed, whereas the maximum current spikes for the Vivo phone were less than 0.5C. Therefore, manufacturers should provide this criterion while specifying the usage time of the phones under continuous operation.

IV. CONCLUSION

This study monitored and assessed the in-service charge/discharge profile (e.g. current discharge, terminal voltage, and battery temperature) of Android-based smartphones during active operation of the six most downloaded social media apps of 2019, i.e., WhatsApp, Facebook, Facebook Messenger, Instagram, Snapchat, and TikTok. The in-service charge/discharge profile was retrieved through our developed Android package kit (APK), which accesses the internal sensors through the manufacturer-provided application program interface (API). The interested readers can see our APP program in the Appendix.

As was demonstrated in this study, the discharge rate depended on the apps and their operation, as well as the phone hardware. For instance, the processor used in Vivo V9 (Qualcomm MSM8953-Pro Snapdragon 626) appears to be more energy efficient than the one used in Motorola Droid Turbo (Qualcomm APQ8084 Snapdragon 805). As a result the same app draws less current on Vivo phone as compare to Motorola phone.

This study also found that the operation of an app, for any given phone, will affect its long-term performance and reliability. For instance, Snapchat operation on Motorola was found to be the most energy and temperature intensive with a mean discharge current of 967 mA (for usage interval of approx. 20 min) with temperature rising to 45 °C. The apps Instagram, Facebook, WhatsApp, Facebook Messenger and TikTok showed lower mean discharge currents of 572, 610, 653, 854 and 933 mA, respectively during operation.

In addition to mean discharge current, electrical current spikes in current drawn is useful in modeling battery degradation through accelerated aging processes. For the Motorola phone, results showed that the probability

of current spikes beyond 0.3C in Snapchat and TikTok is more than 20%, which is significantly higher than all other apps. For the Vivo phone, the operation of Facebook Messenger was significantly more energy-intensive compared to Snapchat or TikTok. Higher spikes in current results in non-linear temperature rise due to I^2R dissipation resulting in accelerated aging process [37], [38].

While manufacturers suggest a runtime for smartphone batteries (the time the battery lasts in one full charge), our findings from in-service phone operation confirm that the actual (experimental) runtime is significantly lower than manufacturer-suggested talk time or operation time. In particular, the discharge currents are underestimated for in-service operation resulting in shortened operational duration. Taking the operation of Snapchat on a Motorola phone as an example, the mean discharge current is 967 mA (approx. C/4 rate – i.e., runtime to about 4 h), which is significantly smaller than the manufacturer-suggested phone runtime of up to 48 h. While the phrase ''up to 48 hours'' can be the highest limit, it is misleading for many consumers, since the actual runtime will be an order of magnitude lower for Snapchat operation. Similarly, the mean discharge current for Snapchat for Vivo V9 was 705 mA (approx. C/5 rate) with runtime of about 5 hr.

Finally, the information presented in this paper evaluates the discharge profiles for smartphones under different app usage for only two manufacturers. In the future, other phones could be considered. However, not all smartphone manufacturers allow/support measurement of discharge profile in high resolution (milli-second). For instance, Apple does not provide the relevant API, whereas Samsung and Google provides only an average current drainage profile [7], [48]. The average value provided by these manufacturers does not contain information on spike currents or effect on battery drainage by various app features on short time intervals, which are essential to characterize the operation of apps or model degradation based on varying consumer behavior.

APPENDIX

A. DATA ITEMS

```
\\ Declaring the type of variables
public class DataItem {
    long current;
    long time;
    int voltage;
    int temprature;
    long batteryCapacity;
}
```

B. MAIN ACTIVITY

```
\\ Import libraries
import android.content.Intent;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class MainActivity extends AppCompatActivity {

\\ Initialization
Long avgCurrent = null, currentNow = null;
private Intent myService;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
myService = new Intent(MainActivity.this,
CurrentService.class);
findViewById(R.id.start).setOnClickListener(new
View.OnClickListener() {
@Override

\\ Build program and start/stop button
public void onClick(View view) {
view.setSelected(!view.isSelected());
((Button)view).setText(view.isSelected()?"Stop":
"Start");
if (view.isSelected())
startService(myService);
else
stopService(myService);}});
// final TextView currentTv =
findViewById(R.id.currentTv);
// final TextView avgCurrentTv =
findViewById(R.id.avgCurrentTv);
// final TextView currPerMilliTv =
findViewById(R.id.currPerMilli);
// final Handler handler = new Handler();
// handler.post(new Runnable() {
// @Override
// public void run() {
\\ Main Commands for finding Current
// BatteryManager mBatteryManager = (BatteryManager)
getSystemService(Context.BATTERY_SERVICE);
//avgCurrent =
mBatteryManager.getLongProperty(BatteryManager.
BATTERY_PROPERTY_CURRENT_AVERAGE);
// currentNow =
mBatteryManager.getLongProperty(BatteryManager.
BATTERY_PROPERTY_CURRENT_NOW);
// handler.postDelayed(this, 1);// }// });
//      final Handler printHandler = new Handler();
//
//      printHandler.post(new Runnable() {
//          @Override
//          public void run() {
//              currentTv.setText("Current:" + currentNow +
"microamperes");
//              avgCurrentTv.setText("Avg. Current:" +
avgCurrent + "microamperes");
//
// printHandler.postDelayed(this, 100)
```

C. CURRENT SERVICE

```
\\ Import libraries
import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.media.MediaScannerConnection;
import android.net.Uri;
import android.os.BatteryManager;
import android.os.Build;
import android.os.Environment;
import android.os.IBinder;
import androidx.core.app.NotificationCompat;
import android.util.Log
import java.io.File;
import java.io.FileWriter;
mport java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.concurrent.TimeUnit;

\\ Initialization
public class CurrentService extends Service {
private static final String CHANNEL_ID =
''battryCurrNotiChan'';
private static final long EACH_FILE_TIME = 5 * 60 *
1000;
Long avgCurrent = null;
StringBuilder string = new StringBuilder();

\\ Saved file directory
private FileWriter fileWriter;
   public static String dataDirectoryPath =
Environment.getExternalStorageDirectory() +
''/BatteryCurrent'';
   private Thread thread;

\\ Declare data type and decimal place
long startTime;
// ArrayList<Pair<Long,Long>> dataItems = new
ArrayList<>(10000);
ArrayList<DataItem> dataItems = new
ArrayList<>(10000);
Integer syncObj;
private long batteryCapacityInitial;
private int currTemp;
private int currVoltage;
private BroadcastReceiver batteryBcReceiver;
private int notifyId = 8192;
@Override
public void onCreate() {
   super.onCreate();
   if (android.os.Build.VERSION.SDK_INT <=
Build.VERSION_CODES.M)
      return;
   Intent notificationIntent = new Intent(this,
MainActivity.class);
   notificationIntent.setAction(Intent.ACTION_MAIN);

\\ Add Notification
notificationIntent.addCategory(Intent.CATEGORY
_LAUNCHER);
//
notificationIntent.addFlags(Intent.FLAG_ACTIVITY_NEW
_TASK);
      PendingIntent pendingIntent =
PendingIntent.getActivity(this, 0, notificationIntent, 0);
      if (android.os.Build.VERSION.SDK_INT >=
android.os.Build.VERSION_CODES.O) {
         NotificationManager mNotificationManager =
(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
         NotificationChannel channel =
mNotificationManager.getNotificationChannel(CHANNEL
_ID);
         if (channel == null) {
            channel = new
NotificationChannel(CHANNEL_ID, ''Beam Channel'',
NotificationManager.IMPORTANCE_DEFAULT);
mNotificationManager.createNotificationChannel(channel);
         }
         Notification notification = new
Notification.Builder(this, CHANNEL_ID)
               .setContentTitle(''Battery Current Running..'')
            .setSmallIcon(R.drawable.ic_launcher_foreground)
               .setContentIntent(pendingIntent)
               .setTicker(''bTicker'')
               .build();
         startForeground(notifyId, notification);
      } else {
         Notification mNotification = new
NotificationCompat.Builder(this)
               .setContentTitle(''Battery Current Running..'')

.setSmallIcon(R.drawable.ic_launcher_foreground)
               .setContentIntent(pendingIntent)

.setPriority(NotificationCompat.PRIORITY_DEFAULT)
               .setTicker(''bTicker'')
               .build();
         startForeground(notifyId, mNotification);
      }

@Override
public IBinder onBind(Intent intent) {
   return null;
```

```
    }
    @Override
    public void onDestroy() {
        synchronized (syncObj) {// TODO: 3/13/2019 this
call should be in onService stop?
            syncObj = Integer.valueOf(0);
    //        thread.interrupt();
            try {
                syncObj.notifyAll();
                syncObj.wait();

    \\ Make sure thread is interrupted
    Thread.sleep(100);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                } catch (Exception e) {
                    e.printStackTrace();
            writeToFile(makeString());
                unregisterReceiver(batteryBcReceiver);
            }
            super.onDestroy();
        }

    \\ Declare Time
     private String makeString() {
        StringBuilder string = new StringBuilder();
        long lastLineTime = 0;
        for (DataItem dataItem: dataItems) {
            if (lastLineTime ==
TimeUnit.MILLISECONDS.convert(dataItem.time,
TimeUnit.NANOSECONDS)) //skip same micro/milli
seconds
                continue;
            lastLineTime =
TimeUnit.MILLISECONDS.convert(dataItem.time,
TimeUnit.NANOSECONDS);

string.append(lastLineTime).append(",").append(dataItem.
current).append(",").append(dataItem.voltage).append(",").
append(dataItem.temprature).append(",").append(dataItem.
batteryCapacity).append("\n");
        }
        return string.toString();
    }
    @Override
    public int onStartCommand(Intent intent, int flags, int star-
tId) {
        thread = new Thread() {
            public void run() {
                syncObj = 1;
                getCurrentValues();
        }
    };
    thread.start();

    batteryBcReceiver = new BroadcastReceiver() {
    //        int scale = -1;
    //        int level = -1;
    //        int voltage = -1;
    //        int temp = -1;
            @Override

    \\ Command for Battery level Temperature and
voltages
    public void onReceive(Context context, Intent intent) {
    //            level =
intent.getIntExtra(BatteryManager.EXTRA_LEVEL, -1);
    //            scale =
intent.getIntExtra(BatteryManager.EXTRA_SCALE, -1);
                currTemp =
intent.getIntExtra(BatteryManager.EXTRA_TEMPERATU
RE, -1);
                currVoltage =
intent.getIntExtra(BatteryManager.EXTRA_VOLTAGE,
-1);
    //            Example:
    //            Log.e("BatteryManager", "level is
"+level+"/"+scale+", temp is "+temp+", voltage is
"+voltage);
    //            ERROR/BatteryManager(795): level is 40/100
temp is 320, voltage is 3848

    \\ So this means that the battery is 40% full, has a tem-
perature of 32.0 degree C, and has voltage of 3.848 Volts.
            }
        };
        IntentFilter filter = new
IntentFilter(Intent.ACTION_BATTERY_CHANGED);
        registerReceiver(batteryBcReceiver, filter);

        return START_STICKY;  }

    \\ Command Charge Counter
    private void getCurrentValues() {
    startTime = System.nanoTime();
    BatteryManager mBatteryManager =
(BatteryManager)
getSystemService(Context.BATTERY_SERVICE);
    batteryCapacityInitial =
mBatteryManager.getLongProperty(BatteryManager.BATTE
RY_PROPERTY_CHARGE_COUNTER);
        synchronized (syncObj) {
    //          while (!thread.isInterrupted()) {
              while (syncObj == 1) {
                try {
                    syncObj.wait(0,500);
    //                Thread.sleep(0, 500);
              } catch (InterruptedException e) {
                e.printStackTrace(); }
    \\ Command for instant and average Current
            DataItem dataItem = new DataItem();
    //          if (!thread.isInterrupted()) {
```

```
//      avgCurrent =
mBatteryManager.getLongProperty(BatteryManager.BATTE
RY_PROPERTY_CURRENT_AVERAGE);
            dataItem.time = System.nanoTime() - startTime;
            dataItem.current =
mBatteryManager.getLongProperty(BatteryManager.BATTE
RY_PROPERTY_CURRENT_NOW);
            dataItem.voltage = currVoltage;
            dataItem.temprature = currTemp;
            dataItem.batteryCapacity =
mBatteryManager.getLongProperty(BatteryManager.BATTE
RY_PROPERTY_CHARGE_COUNTER);
            dataItems.add(dataItem)
          if
(TimeUnit.MILLISECONDS.convert(dataItem.time,
TimeUnit.NANOSECONDS) > EACH_FILE_TIME)

            saveFileAndResetState(); }
          try {
            syncObj.notifyAll();
          } catch (Exception e) {
            e.printStackTrace();
          }
  //        }
      }
      }
      private void saveFileAndResetState() {
        writeToFile(makeString());
        dataItems.clear();
        startTime = System.nanoTime();
      }
  // @SuppressWarnings(''deprecation'')
  // @Override
  // public void onStart(Intent intent, int startId) {
  //     super.onStart(intent, startId);
  //
  // }
      private void writeToFile(String string) {
        try {
          File dir = new File(dataDirectoryPath);
          if (!dir.exists())
            dir.mkdirs();

          SimpleDateFormat dateFormat = new Simple-
DateFormat(
          ''yyyy_MM_dd_HH_mm_ss'');
      Date date = new Date();
      String filename = dateFormat.format(date) +
''.csv'';
      // Write the file into the folder
      File file = new File(dir, filename);
      fileWriter = new FileWriter(file);
      //Headings to add for the first time
      BatteryManager mBatteryManager =
(BatteryManager)
getSystemService(Context.BATTERY_SERVICE);
      long batteryCapacityEnd =
mBatteryManager.getLongProperty(BatteryManager.BATTE
RY_PROPERTY_CHARGE_COUNTER);
      fileWriter.append(''Time (milliSeconds),Current
(microAmps),Voltage(milliVolts),Temperature(celsius x
10),Capacity microAmp-hours (' +
batteryCapacityInitial+'':''+batteryCapacityEnd+'')\n'');
      fileWriter.append(string);
      fileWriter.flush();
      fileWriter.close();
      fileWriter = null;
    MediaScannerConnection.scanFile(getApplicationContext
(), new String[]{dataDirectoryPath}tim, null, new
MediaScannerConnection.MediaScannerConnectionClient()
{
          @Override
          public void onMediaScannerConnected() {
          }
          @Override
          public void onScanCompleted(String s, Uri uri)
{
          }
        });
      } catch (Exception e) {
       Log.e(''ExceptionHandler'', e.getMessage());   }
}}
```

ACKNOWLEDGMENT

The authors thank the Center for Advanced Life Cycle Engineering (CALCE) at University of Maryland and its over 150 funding companies for enabling research into advanced topics in reliability, safety, and sustainment. The authors thank Mr Usman Khan for developing the APK for deciphering battery profile for the smartphones under investigation. The authors also thank Cheryl Wurzbacher and Saurabh Saxena for editing and comments to improve the article's quality.

REFERENCES

[1] S. Latif, R. Rana, J. Qadir, A. Ali, M. A. Imran, and M. S. Younis, "Mobile health in the developing world: Review of literature and lessons from a case study," *IEEE Access*, vol. 5, pp. 11540–11556, 2017.

[2] Venturebeat. *Newzoo: Smartphone users will top 3 billion in 2018, hit 3.8 billion by 2021*. Accessed: Feb. 20, 2020. [Online]. Available: https://venturebeat.com/2019/06/07/facebook-will-no-longer-pre-install-its-apps-on-huawei-phones/

[3] P. K. D. Pramanik, N. Sinhababu, B. Mukherjee, S. Padmanaban, A. Maity, B. K. Upadhyaya, J. B. Holm-Nielsen, and P. Choudhury, "Power consumption analysis, measurement, management, and issues: A State-of-the-Art review of smartphone battery and energy usage," *IEEE Access*, vol. 7, pp. 182113–182172, 2019.

[4] K. Park, Y. Choi, W. J. Choi, H.-Y. Ryu, and H. Kim, "LSTM-based battery remaining useful life prediction with multi-channel charging profiles," *IEEE Access*, vol. 8, pp. 20786–20798, 2020.

[5] Y. Guo, C. Wang, and X. Chen, "Understanding application-battery interactions on smartphones: A large-scale empirical study," *IEEE Access*, vol. 5, pp. 13387–13400, 2017.

[6] D. Ferreira, A. K. Dey, and V. Kostakos, "Understanding human-smartphone concerns: A study of battery life," in *Proc. Int. Conf. Pervas. Comput.*, 2011, pp. 19–33.

[7] Y. Sun, L. Kong, H. Abbas Khan, and M. G. Pecht, "Li-ion battery reliability—A case study of the apple iPhone," *IEEE Access*, vol. 7, pp. 71131–71141, 2019.

[8] F. C. Krause, J. A. Loveland, M. C. Smart, E. J. Brandon, and R. V. Bugga, "Implementation of commercial li-ion cells on the MarCO deep space CubeSats," *J. Power Sources*, vol. 449, Feb. 2020, Art. no. 227544.

[9] T. Dong, P. Peng, and F. Jiang, "Numerical modeling and analysis of the thermal behavior of NCM lithium-ion batteries subjected to very high C-rate discharge/charge operations," *Int. J. Heat Mass Transf.*, vol. 117, pp. 261–272, Feb. 2018.

[10] K. Wang, F. Gao, Y. Zhu, H. Liu, C. Qi, K. Yang, and Q. Jiao, "Internal resistance and heat generation of soft package Li4Ti5O12 battery during charge and discharge," *Energy*, vol. 149, pp. 364–374, Apr. 2018.

[11] J. Bräckner, S. Thieme, H. T. Grossmann, S. Dörfler, H. Althues, and S. Kaskel, "Lithium–sulfur batteries: Influence of C-rate, amount of electrolyte and sulfur loading on cycle performance," *J. Power Sources*, vol. 268, pp. 82–87, Dec. 2014.

[12] L. Wang, Q. Zhang, J. Zhu, X. Duan, Z. Xu, Y. Liu, H. Yang, and B. Lu, "Nature of extra capacity in MoS_2 electrodes: Molybdenum atoms accommodate with lithium," *Energy Storage Mater.*, vol. 16, pp. 37–45, Jan. 2019.

[13] M. Huang, H. Chen, J. He, B. An, L. Sun, Y. Li, X. Ren, L. Deng, and P. Zhang, "Ultra small few layer MoS_2 embedded into three-dimensional macro-micro-mesoporous carbon as a high performance lithium ion batteries anode with superior lithium storage capacity," *Electrochimica Acta*, vol. 317, pp. 638–647, Sep. 2019.

[14] C. Wang, F. Yan, Y. Guo, and X. Chen, "Power estimation for mobile applications with profile-driven battery traces," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, Sep. 2013, pp. 120–125.

[15] L. Hou, L. Zhang, and J. Kim, "Energy modeling and power measurement for mobile robots," *Energies*, vol. 12, no. 1, p. 27, 2019.

[16] L. Cruz and R. Abreu, "EMaaS: Energy measurements as a service for mobile applications," in *Proc. IEEE/ACM 41st Int. Conf. Softw. Eng., New Ideas Emerg. Results (ICSE-NIER)*, May 2019, pp. 101–104.

[17] M. F. Tuysuz, M. Ucan, and R. Trestian, "A real-time power monitoring and energy-efficient network/interface selection tool for Android smartphones," *J. Netw. Comput. Appl.*, vol. 127, pp. 107–121, Feb. 2019.

[18] X. Chen, N. Ding, A. Jindal, Y. C. Hu, M. Gupta, and R. Vannithamby, "Smartphone energy drain in the wild: Analysis and implications," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 1, pp. 151–164, Jun. 2015.

[19] R. W. Ahmad, A. Gani, S. H. A. Hamid, F. Xia, and M. Shiraz, "A review on mobile application energy profiling: Taxonomy, state-of-the-art, and open research issues," *J. Netw. Comput. Appl.*, vol. 58, pp. 42–59, Dec. 2015.

[20] W. Jung, C. Kang, C. Yoon, D. Kim, and H. Cha, "DevScope: A nonintrusive and online power analysis tool for smartphone hardware components," in *Proc. 8th IEEE/ACM/IFIP Int. Conf. Hardw./Softw. codesign Syst. Synth. (CODES+ISSS)*, 2012, pp. 353–362.

[21] J. Chung, H. Jung, J. Koo, Y. Kim, and U.-M. Kim, "A development of power consumption measurement system for Android smartphones," in *Proc. 11th Int. Conf. Ubiquitous Inf. Manage. Commun. (IMCOM)*, 2017, p. 95.

[22] Source Android. *Measuring Device Power*. Accessed: Feb. 20, 2020. [Online]. Available: https://source.android.com/devices/tech/power/device

[23] Samsung. *Galaxy J7*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.samsung.com/pk/smartphones/galaxy-j7-j700h/

[24] Gsmarena. *Google Pixel 2*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.gsmarena.com/google_pixel_2-8733.php

[25] Gsmarena. *Huawei Nexus 6P*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.gsmarena.com/huawei_nexus_6p-7588.php

[26] Gsmarena. *Oppo A57*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.gsmarena.com/oppo_a57-8468.php

[27] Gsmarena. *Vivo V9*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.gsmarena.com/vivo_v9-9117.php

[28] Gsmarena. *Motorola DROID Turbo*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.gsmarena.com/motorola_droid_turbo-6727.php

[29] Gsmarena. *Xiaomi Mi A1 (Mi 5X)*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.gsmarena.com/xiaomi_mi_a1_(mi_5x)-8776.php

[30] Counterpoint. *Global Smartphone Market Share: By Quarter*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.counterpointresearch.com/global-smartphone-share/

[31] Sensor Scientific. *Thermistor and Rechargeable Batteries*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.sensorsci.com/thermistor-and-rechargeable-batteries

[32] Vishay Bccomponents. *Fast Charging Control with NTC Temperature Sensing*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.vishay.com/docs/29089/fastappl.pdf

[33] Developers. *Battery Manager*. Accessed: Feb. 20, 2020. [Online]. Available: https://developer.android.com/reference/android/os/BatteryManager

[34] IFIXIT. *Samsung Galaxy S10 and S10e Teardown*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.ifixit.com/Teardown/Samsung+Galaxy+S10+and+S10e+Teardown/120331

[35] D. H. Jeon and S. M. Baek, "Thermal modeling of cylindrical lithium ion battery during discharge cycle," *Energy Convers. Manage.*, vol. 52, nos. 8–9, pp. 2973–2981, Aug. 2011.

[36] G. Liu, M. Ouyang, L. Lu, J. Li, and X. Han, "Analysis of the heat generation of lithium-ion battery during charging and discharging considering different influencing factors," *J. Thermal Anal. Calorimetry*, vol. 116, no. 2, pp. 1001–1010, May 2014.

[37] K. Uddin, A. D. Moore, A. Barai, and J. Marco, "The effects of high frequency current ripple on electric vehicle battery performance," *Appl. Energy*, vol. 178, pp. 142–154, Sep. 2016.

[38] L. W. Juang, P. J. Kollmeyer, A. E. Anders, T. M. Jahns, R. D. Lorenz, and D. Gao, "Investigation of the influence of superimposed AC current on lithium-ion battery aging using statistical design of experiments," *J. Energy Storage*, vol. 11, pp. 93–103, Jun. 2017.

[39] S. Buso, L. Malesani, and P. Mattavelli, "Comparison of current control techniques for active filter applications," *IEEE Trans. Ind. Electron.*, vol. 45, pp. 722–729, May 1998.

[40] F. Leng, Z. Wei, C. M. Tan, and R. Yazami, "Hierarchical degradation processes in lithium-ion batteries during ageing," *Electrochimica Acta*, vol. 256, pp. 52–62, Dec. 2017.

[41] S. Saxena, Y. Xing, D. Kwon, and M. Pecht, "Accelerated degradation model for C-rate loading of lithium-ion batteries," *Int. J. Electr. Power Energy Syst.*, vol. 107, pp. 438–445, May 2019.

[42] Samsung. *Samsung 21700 INR21700-33J 3300mAh 3.2A Battery cell 3.7V*. Accessed: Feb. 20, 2020. [Online]. Available: http://queenbattery.com.cn/our-products/541-samsung-21700-inr21700-30t-3000mah-35a-battery-cell-37v.htm

[43] M. J. Brand, M. H. Hofmann, S. S. Schuster, P. Keil, and A. Jossen, "The influence of current ripples on the lifetime of lithium-ion batteries," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 10438–10445, Nov. 2018.

[44] M. Uno and K. Tanaka, "Influence of high-frequency charge–discharge cycling induced by cell voltage equalizers on the life performance of lithium-ion cells," *IEEE Trans. Veh. Technol.*, vol. 60, no. 4, pp. 1505–1515, May 2011.

[45] S. Saxena, Y. Xing, and M. Pecht, "A unique failure mechanism in the nexus 6P lithium-ion battery," *Energies*, vol. 11, no. 4, p. 841, 2018.

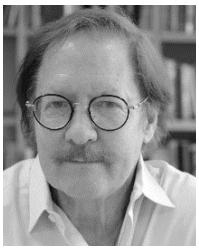
[46] Gauangzhou Markyn Battery. *Li-ion Rechargeable Battery (GMB103448S)*. Accessed: Feb. 20, 2020. [Online]. Available: https://www.powerstream.com/p/H103448S.pdf

[47] Panasonic. *Panasonic BM 18650BM 3200mah Battery Cell 3.6V*. Accessed: Feb. 20, 2020. [Online]. Available: http://queenbattery.com.cn/our-products/154-panasonic-bm-18650bm-3200mah-battery-cell-36v.html

[48] Apple. *Battery Level*. Accessed: Feb. 20, 2020. [Online]. Available: https://developer.apple.com/documentation/uikit/uidevice/1620042-batterylevel

HAYDER ALI (Student Member, IEEE) received the B.S. degree in electrical engineering from the University of Engineering and Technology (UET), Lahore, Pakistan, in 2015, and the M.S. degree in electrical engineering from the Lahore University of Management Science (LUMS), Lahore, in 2018, where he is currently pursuing the Ph.D. degree with the Department of Electrical Engineering. His current research interests include lithium-ion battery failure analysis, renewable energy, and efficient power processing of renewable resources.

HASSAN ABBAS KHAN (Member, IEEE) received the B.Eng. degree in electronic engineering from the Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Topi, Pakistan, in 2005, and the M.Sc. degree (Hons.) and Ph.D. degree in electrical and electronic engineering from the School of Electrical Engineering, The University of Manchester, Manchester, U.K. From 2005 to 2010, he was with the School of Electrical Engineering, The University of Manchester. He is currently an Assistant Professor with the Department of Electrical Engineering, Lahore University of Management and Sciences, Lahore, Pakistan. His current research interests include renewable energy and its uptake in developing countries and novel grid architectures for low-cost rural electrification through solar energy. He is also working with efficient and reliable solar PV deployments in urban settings to maximize their performance ratios.



MICHAEL G. PECHT (Life Fellow, IEEE) received the B.S. degree in physics, the M.S. degree in electrical engineering, and the M.S. and Ph.D. degrees in engineering mechanics from the University of Wisconsin–Madison, Madison, WI, USA, in 1976, 1978, 1979, and 1982, respectively.

He is currently a Professional Engineer and a Chair Professor. He is also the Director of the Center for Advanced Life Cycle Engineering, University of Maryland, which is funded by more than 150 of the world's leading electronics companies at more than U.S.$6 M/year. He has authored or coauthored more than 20 books on product reliability, development, and use and supply chain management. He has also authored or coauthored more than 700 technical articles, ten patents, and a series of books with the electronics industry, in China, South Korea, Japan, and India. He is an ASME Fellow, a SAE Fellow, and an IMAPS Fellow. He was the recipient of the IEEE Exceptional Technical Achievement Award, in 2008 and 2010, and the IEEE Reliability Society's Lifetime Achievement Award. He has served for the three U.S. National Academy of Science studies and two U.S. Congressional Investigations in Automotive Safety. He served as an Expert for U.S. FDA. He also served as an Editor-in-Chief for IEEE Access, for a period of six years, the IEEE Transactions on Reliability, for a period of nine years, and *Microelectronics Reliability*, for a period of 16 years.