

Journal Of Computer And Information Sciences



SAKARYA UNIVERSITY

e-ISSN 2636-8129

VOLUME 4

ISSUE 3

DECEMBER 2021

Effects of Binary Vectors Similarities on the Accuracy of MultiCriteria Collaborative Filtering

Twitter Sentiment Analysis Based on Daily Covid-19 Table in Turkey

Automated learning rate search using batch-level cross-validation

GraParT: A MATLAB Toolbox for Partitioning Directed Graphs



www.saucis.sakarya.edu.tr

A Smart Socket Equipped With IoT Technologies for Energy Management of Electrical Appliances

*A Digital Forensics Approach for Lost Secondary Partition Analysis
using Master Boot Record Structured Hard Disk Drives*

*Rain Rate and Rain Attenuation Prediction for Satellite
Communication in Ku Band Beacon over TURKSAT Golbasi*



SAUCIS

Sakarya University Journal of Computer and Information Sciences
Volume: 4 – Issue No: 3 (December 2021)
<http://saucis.sakarya.edu.tr/issue/67693>

Editor in Chief

Nejat Yumuşak, Sakarya University, nyumusak@sakarya.edu.tr

Associate Editors

İhsan Hakan Selvi, Sakarya University, Turkey, ihselvi@sakarya.edu.tr

Muhammed Fatih Adak, Sakarya University, Turkey, fatihadak@sakarya.edu.tr

Mustafa Akpınar, Sakarya University, Turkey, akpınar@sakarya.edu.tr

Unal Cavusoglu, Sakarya University, Turkey, unalc@sakarya.edu.tr

Veysel Harun Sahin, Sakarya University, Turkey, vsahin@sakarya.edu.tr

Editorial Assistants - Secretary

Deniz Balta, Sakarya University, Turkey, ddural@sakarya.edu.tr

Fatma Akalin, Sakarya University, Turkey, fatmaakalin@sakarya.edu.tr

Gozde Yolcu Oztel, Sakarya University, Turkey, gyolcu@sakarya.edu.tr

Ibrahim Delibasoglu, Sakarya University, Turkey, ibrahimdelibasoglu@sakarya.edu.tr

Muhammed Kotan, Sakarya University, Turkey, mkotan@sakarya.edu.tr

Sumeyye Kaynak, Sakarya University, Turkey, sumeyye@sakarya.edu.tr

Ahmet Erhan Tanyeri, Sakarya University, Turkey, tanyeri@sakarya.edu.tr

Editorial Board

Ahmet Ozmen, Sakarya University, Turkey, ozmen@sakarya.edu.tr

Aref Yelghi, Istanbul Ayyansaray University, ar.yelqi@gmail.com

Ayhan Istanbulu, Balikesir University, Turkey, iayhan@balikesir.edu.tr

Aysegul Alaybeyoglu, Izmir Katip Celebi University, Turkey, alaybeyoglu@gmail.com

Bahadir Karasulu, Canakkale Onsekiz Mart University, bahadirkarasulu@comu.edu.tr

Celal Ceken, Sakarya University, Turkey, celalceken@sakarya.edu.tr

Cihan Karakuzu, Bilecik Seyh Edebali University, cihan.karakuzu@bilecik.edu.tr

Fahri Vatansever, Bursa Uludag University, fahriv@uludag.edu.tr

Ibrahim Turkoglu, Fırat University, Turkey, iturkoglu@firat.edu.tr

Levent Alhan, Sakarya University, Turkey, leventalhan@sakarya.edu.tr

Kamal Z Zamli, Malaysia Pahang University, Malaysia, kamalz@ump.edu.my

Muhammed Fatih Adak, Sakarya University, Turkey, fatihadak@sakarya.edu.tr

Mustafa Akpınar, Sakarya University, Turkey, akpınar@sakarya.edu.tr



SAUUCIS

Editorial Board (Cont.)

Nuri Yilmazer, Texas A&M University, US, nuri.yilmazer@tamuk.edu

Nejat Yumuşak, Sakarya University, nyumusak@sakarya.edu.tr

Orhan Er, Bozok University, Turkey, orhan.er@bozok.edu.tr

Priyadip Ray, Lawrence Livermore National Laboratory, priyadipr@gmail.com

Resul Das, Firat University, Turkey, rdas@firat.edu.tr

Veysel Harun Sahin, Sakarya University, Turkey, vsahin@sakarya.edu.tr



SAUCIS

Sakarya University Journal of Computer and Information Sciences
Volume: 4 – Issue No: 3 (December 2021)
<http://saucis.sakarya.edu.tr/issue/67693>

Contents

Author(s), Paper Title	Pages
<i>Onur Cihan</i> GraParT: A MATLAB Toolbox for Partitioning Directed Graphs	277-286
<i>Burcu Demirelli Okkaloğlu</i> Effects of Binary Vectors Similarities on the Accuracy of Multi-Criteria Collaborative Filtering	287-301
<i>Buket Kaya, Abdullah Günay</i> Twitter Sentiment Analysis Based on Daily Covid-19 Table in Turkey	302-311
<i>Duygu Kabakcı, Emre Akbaş</i> Automated learning rate search using batch-level cross-validation	312-325
<i>Erhan Akbal, Omer Faruk Yakut, Sengul Dogan, Turker Tuncer, Fatih Ertam</i> A Digital Forensics Approach for Lost Secondary Partition Analysis using Master Boot Record Structured Hard Disk Drives	326-346
<i>Khaled Elorbany, Cüneyt Bayılmış, Seda Balta</i> A Smart Socket Equipped With IoT Technologies for Energy Management of Electrical Appliances	347-353
<i>Yasin Burak Kaya, Umit Cezmi Yılmaz</i> Rain Rate and Rain Attenuation Prediction for Satellite Communication in Ku Band Beacon over TURKSAT Golbasi	354-359

GraParT: A MATLAB Toolbox for Partitioning Directed Graphs

 Onur CİHAN

Marmara University, Faculty of Engineering, Department of Electrical and Electronics Engineering;
onur.cihan@marmara.edu.tr

Received 23 March 2021; Accepted 7 September 2021; Published online 31 December 2021

Abstract

Consensus algorithms are increasingly used in multi-agent systems due to their advantages in various applications. Recent results on consensus algorithms show that the number of groups formed in a network of agents utilizing consensus-based algorithms can be computed once its primary and secondary layer subgraphs are determined. In this study, we present GraParT -Graph Partitioning Toolbox- that can be used to partition directed graphs by determining its primary and secondary layer subgraphs and the vertices therein. The toolbox helps the user to build, modify, analyze and illustrate directed graphs in terms of the grouping behavior of the consensus algorithms with its user-friendly interface. GraParT is an open-source software that is available free of charge for academic and non-commercial use.

Keywords: directed graphs, consensus algorithms, group consensus, MATLAB toolbox

1. Introduction

The last two decades have seen the rapid development of consensus algorithms due to their use in practical problems consisting of multiple agents [1-6]. These algorithms find applications in the fields of robotics [1, 2], computer networks [3], distributed optimization [4], and social networks [5]. Most of the work consider the consensus problem as agreement on a single state which requires the underlying graph of the network to have a spanning tree [7, 8]. If there is no spanning tree in the graph, consensus on a single state is not possible and multiple groups will be formed in the network [8].

Recently, the number of groups that will be formed in a multi agent network utilizing a consensus based algorithm was investigated for networks with first [9], second [10] and third order agent dynamics [11]. While the stability conditions are different for these networks, the grouping behaviors are the same. The concepts of primary layer subgraphs (PLS) and secondary layer subgraphs (SLS), which were first introduced in [9], have been instrumental in our understanding of the grouping behavior of a multi-agent system. Once these subgraphs are determined, the topology designer can merge or divide the groups by adding new edges to the graph [12].

Hespanha proposed a MATLAB algorithm to partition undirected graphs which can be used to solve the l -bounded Graph Partitioning (l -GP) problem [13]. In this problem, the vertices of the given undirected graph are partitioned into k disjoint subsets where the sum of the edges connecting the vertices in different subsets is minimized. Çatalyürek and Aykanat built a hypergraph partitioning tool called PaToH that can be utilized to solve various combinatorial scientific computing problems including VLSI layout design and dynamic load balancing for parallel processing [14]. The objective of such partitioning is to divide the given graph into two (or more) subgraphs with equal (or nearly equal) number of vertices such that the cost function defined on the hyperedges connecting vertices in different subgraphs is minimized. Furthermore, depending on the problem, another objective may be to keep the sum of hyperedges connecting the vertices in the same subgraphs as close as possible (load balancing problem). For a detailed survey on graph partitioning, we refer the reader to [15] and the references therein. While some research has been carried out on developing graph partitioning tools for solving different graph-related problems, to the best of our knowledge, this is the first study that provides a tool for partitioning directed graphs to determine the groups and their members in a network of agents utilizing a consensus-based algorithm. Due to the applicability of the consensus algorithms in important real-world problems

(such as formation control of multi-robot systems, analysis of social network analysis, etc.), a toolbox for partitioning directed graphs may help the engineers in designing network topologies.

In this paper, our objective is to provide a tool for determining these groups and the vertices in each group by exploiting the graph structure. This tool enables the network topology designer to understand the grouping behavior of the multi-agent system without running a consensus based algorithm. More specifically, the topology designer can add new links or remove existing ones to form a new network or modify an existing one to obtain desired grouping when a consensus based algorithm is used. For instance, in the formation control problem of a multi-robot system, this tool can be used to design a network topology where the robots in the same group achieve a desired formation. To this end, we utilize the concepts of PLS and SLS; and implement the detection algorithms whose pseudocode were given in [8]. Moreover, we propose the notion of *reduced graphs* where each group is represented by a vertex and the interaction between the agents in two different groups are represented by an edge in the reduced graph. Finally, we propose an algorithm for obtaining the reduced graph of an arbitrary directed graph and discuss its time and space complexity. With this novel definition, one can analyze complex directed graphs in terms of the grouping behaviors of the agents that are utilizing a consensus based algorithm. To the best of authors' knowledge, this is the first study to provide such a graph theoretic concept and an algorithm its detection.

We organize the remaining parts of this paper as follows. We review graph theory concepts that are used throughout the paper and give the mathematical formulation of the conventional continuous-time and discrete-time consensus algorithms with linear agent dynamics in Section 2. In Section 3, we present the workflow of the toolbox; and demonstrate a short tutorial on how to use the toolbox, interpret the outputs and visualize the partitioned graph. Finally, Section 4 concludes the paper.

2. Graph Theory Preliminaries and Dynamical Model of Consensus

2.1 Graph Theory Preliminaries

Communication network of a multi-agent system is modeled by a directed graph $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ where $\mathbf{V} = \{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ is the finite set of vertices and $\mathbf{E} \subseteq \mathbf{V} \times \mathbf{V}$ is the set of edges representing the information flow between agents. We say that there exists a directed edge from \mathbf{v}_i to \mathbf{v}_j if $(\mathbf{v}_i, \mathbf{v}_j) \in \mathbf{E}$ and \mathbf{v}_i is called a neighbor of \mathbf{v}_j . The graph \mathbf{G} is said to consist of a spanning tree if there exists a vertex \mathbf{v}_r such that all vertices in \mathbf{G} receive information from \mathbf{v}_r directly or indirectly (in more than 1 step). Here, \mathbf{v}_r is called a root of the graph \mathbf{G} . When there is no spanning tree in a directed graph, it can be partitioned into its PLS and SLS whose definitions are given as follows.

Definition 1. (Primary and secondary layer subgraphs)

Let the multi-agent system be modeled by a directed graph $G = (V, E)$. Then G can be partitioned into $l_p + l_s$ subgraphs such that

- i) the vertices in l_p subgraphs (each consisting of the largest possible spanning tree) are not connected to the rest of the graph, and
- ii) the vertices in l_s subgraphs (each consisting of a spanning tree) are connected to the rest of the network through a single vertex which is a root of the subgraph and the only vertex to receive information from other subgraphs. Furthermore, this root vertex has at least two neighbors in two other subgraphs.

The subgraphs defined in items (i) and (ii) are called the PLSs and SLSs subgraphs of G and denoted by $G_{p,i}$ ($i = 1, \dots, l_p$) and $G_{s,j}$ ($j = 1, \dots, l_s$) respectively. Note that this partitioning is unique for a given directed graph.

2.2 Dynamical Model of Consensus

Consider a network of n agents evolving over a directed graph $G = (V, E)$. The conventional distributed consensus algorithm with continuous time first-order agent dynamics can be expressed as

$$\dot{x}_i(t) = \sum_{j \in N_i} a_{ij} (x_j(t) - x_i(t)), \quad i = 1, \dots, n \quad (1)$$

where $x_i(t) \in R^m$ is the state of agent i at time t , N_i is the set consisting of the neighboring vertices of agent i and a_{ij} is the (i, j) -th element of the adjacency matrix A defined as $a_{ij} > 0$ if $(v_j, v_i) \in E$ and $a_{ij} = 0$ otherwise. In matrix form, the system represented by Equation 1 can be expressed as

$$\dot{x}(t) = -(L \otimes I_m)x(t) \quad (2)$$

where I_m is the $m \times m$ identity matrix, \otimes is the Kronecker product operator and $L = [l_{ij}]$ is the Laplacian matrix defined by

$$l_{ij} = \begin{cases} \sum_{\substack{k=1 \\ k \neq i}}^n a_{ik}, & \text{if } i = j \\ -a_{ij}, & \text{otherwise} \end{cases} \quad (3)$$

In discrete-time, the distributed algorithm with first-order agent dynamics is given as follows

$$x_i(k + 1) = w_{ii}x_i(k) + \sum_{j \in N_i} w_{ij}x_j(k), \quad i = 1, \dots, n \quad (4)$$

where $x_i(k) \in R^m$ is the state of agent i at time step k , and $w_{ij} \geq 0$ is the weighting coefficient corresponding to the information exchange between agents j and i . The following assumption is necessary for the stability of the system represented by Equation 4.

Assumption 1. The following conditions hold for the weighting coefficients w_{ij}

$$i) \quad w_{ij} \begin{cases} > 0, & \text{if } (v_j, v_i) \in E \text{ or } i = j \\ = 0, & \text{otherwise.} \end{cases} \quad (5)$$

$$ii) \quad \sum_{j=1}^n w_{ij} = 1, \quad \text{for all } i \in \{1, \dots, n\} \quad (6)$$

Assumption 1(i) ensures that the weighting coefficients are always non-negative, and equal to 0 if there is no information flow between agent j to agent i . Assumption 1(ii) guarantees that the weighting coefficients sum up to 1 for all i .

The system represented by Equation 4 can be expressed in matrix form as

$$x(k + 1) = (W \otimes I_m)x(k) \quad (7)$$

where $x(k) = [x_1(k)^T, \dots, x_n(k)^T]^T$ and $W = [w_{ij}]$ is the weighting matrix of the system.

Given a multi-agent system with a directed graph that is not consisting of a spanning tree, the agents can not achieve consensus on a single equilibrium state. In such a case, the agents converge to different vectors and multiple groups are formed. The definition of group consensus is introduced as follows.

Definition 2. (Group consensus)

We say that the multi-agent system with the underlying graph G achieves group consensus if there are c nonempty sets S_l ($l = 1, \dots, c$) such that

$$\bigcup_{l=1}^c S_l = V, \quad S_l \cap S_q = \emptyset, \quad \text{for } l \neq q \text{ and } l, q = 1, \dots, c \quad (8)$$

and for the set S_l we have

$$\lim_{k \rightarrow \infty} \|x_i(k) - x_j(k)\| = 0, \quad \forall v_i, v_j \in S_l \quad (9)$$

for arbitrary initial conditions $x_i(0) \in R^m$ and arbitrary choice of averaging coefficients w_{ij} satisfying Assumption 1 (or equivalently, arbitrary choice of a_{ij}).

Note that for a multi-agent network with dynamics given in Equation 1 (or equivalently Equation 4 for discrete-time networks), the number of groups is related to the structure of the network [9-12, 16]. The following lemma states relationship between the number of groups and the graph partitioning based on Definition 1.

Lemma 1. (Number of groups)

Consider a multi-agent system with the underlying graph G where agents utilize the consensus algorithm given by Equation 1 (or equivalently Equation 4 for discrete-time networks). The multi-agent system forms $c = l_p + l_s$ groups where l_p and l_s are the number of PLSs and SLSs of G , respectively [9].

From Lemma 1, one can conclude that the number of groups can be determined from the numbers of PLS and SLS of the underlying network. It is shown in [9] that the agents' states in a particular subgraph converge to the same state. Furthermore, the states of the agents in the SLSs asymptotically converge to a convex combination of those of the agents in the PLSs. The main motivation of this study is to provide a useful tool for determining these groups and their members for a given directed graph. To better illustrate the groups in the network the following definition is introduced.

Definition 3. (Reduced graph)

Let $G = (V, E)$ be a directed graph consisting of l_p PLSs and l_s SLSs. Let $G_i = (V_i, E_i)$ denote the PLS for $i = 1, \dots, l_p$ and the SLS for $i = l_p + 1, \dots, l_p + l_s$. Then the graph $\bar{G} = (\bar{V}, \bar{E})$ is called the reduced graph of G where $\bar{V} = \{\bar{v}_1, \dots, \bar{v}_{l_p+l_s}\}$ is the vertex set and \bar{E} is the edge set of the reduced graph whose elements are defined as

$$(\bar{v}_i, \bar{v}_j) \begin{cases} \in \bar{E} & \text{if } \exists v_a \in V_i, v_b \in V_j \text{ such that } (v_a, v_b) \in E \\ \notin \bar{E} & \text{otherwise} \end{cases} \quad (10)$$

The adjacency matrix of the reduced graph is a block matrix of the form

$$\bar{A} = \begin{bmatrix} 0_{l_p \times l_p} & 0_{l_p \times l_s} \\ \bar{A}_{sp} & \bar{A}_s \end{bmatrix} \quad (11)$$

where \bar{A}_s is related with the communication between secondary layer subgraphs and \bar{A}_{sp} refers to the one-way communication between the PLS and SLS. Here $0_{m \times n}$ denotes the zero matrix of dimension $m \times n$.

In [9], two algorithms were introduced to determine the PLS and SLS of a given directed graph. Once these subgraphs are determined, the following algorithm can be used to determine the reduced graph \bar{G} .

Algorithm 1 Reduced graph extraction algorithm

1	procedure $\bar{G} = \text{ReducedGraph}(G, G_1, \dots, G_{l_p+l_s}, l_p, l_s)$
2	$\bar{V} \leftarrow \{\bar{v}_1, \dots, \bar{v}_{l_p+l_s}\}$
3	$\bar{E} \leftarrow \emptyset$
4	for $i \leftarrow 1, l_p$ do
5	for $j \leftarrow 1, l_s$ do
6	for all $v_a \in V_i$ do
7	for $v_b \in V_{l_p+j}$ do
8	if $(v_a, v_b) \in E$ then
9	


```

10          $\bar{E} \leftarrow \bar{E} \cup (\bar{v}_i, \bar{v}_{l_p+j})$ 
11     end if
12 end for
13 end for
14 end for
15 end for
16 end for
17 for  $i \leftarrow 1, l_s$  do
18     for  $j \leftarrow 1, l_s$  do
19         for all  $v_a \in V_{l_p+i}$  do
20             for  $v_b \in V_{l_p+j}$  do
21                 if  $(v_a, v_b) \in E$  then
22                      $\bar{E} \leftarrow \bar{E} \cup (\bar{v}_{l_p+i}, \bar{v}_{l_p+j})$ 
23                 end if
24             end for
25         end for
26     end for
27 end for
28 end for
29  $\bar{G} \leftarrow (\bar{V}, \bar{E})$ 
30 return  $\bar{G}$ 
31 end procedure

```

The algorithm initially generates a set consisting of $l_p + l_s$ vertices with an empty set of edges (time complexity $\mathcal{O}(n)$, space complexity $\mathcal{O}(n)$). For each agent in the primary layer subgraphs, the algorithm checks whether there is a directed path to an agent in the secondary layer subgraphs (time complexity $\mathcal{O}(n^2)$). If there exists such a path, a link is added to the edge set of the reduced graph (space complexity $\mathcal{O}(n^2)$). The same procedure is repeated for the links between the agents in the secondary layer subgraphs (time complexity $\mathcal{O}(n^2)$ and space complexity $\mathcal{O}(n^2)$). Consequently, the overall time complexity and space complexity of the reduced graph extraction algorithm is $\mathcal{O}(n^2)$. On the other hand, this algorithm requires the PLS and SLS of the graph to be determined prior to its execution. Since the time complexity of PLS and SLS detection algorithms are $\mathcal{O}(n^4)$ and $\mathcal{O}(n^6)$ (see [8]), we conclude that the time complexity of detecting the reduced graph of an arbitrary directed graph is $\mathcal{O}(n^6)$. Note from Definition 1 that the vertices in a PLS are not connected to the vertices in other PLS and SLS. Therefore l_p vertices in the reduced graph $\bar{G} = (\bar{V}, \bar{E})$ do not receive information from other vertices. We would like to note also that the concept of a *reduced graph* is novel and therefore there does not exist any other algorithm in the literature which can be used to determine the reduced graph of an arbitrary directed graph.

3. GraParT: Graph Partitioning Toolbox

In this section, we present GraParT: a MATLAB toolbox for partitioning directed graphs.

3.1 Installation

GraParT can be downloaded from <http://www.onurcihan.com/GraParT.html> and requires R2018b or a newer release of MATLAB to work properly. It is compatible with Windows, macOS and Linux operating systems.

3.2 The Workflow

GraParT allows the users to input the directed edges of a graph and computes the adjacency matrix A , the Laplacian matrix L , the weighting matrix W (assuming equal weighting is used for the information coming from different agents), the partitions of the graph (namely, the PLS and SLS and the vertices

therein) and the reduced graph. Furthermore, evolution of the states of the agents utilizing Equation 1 is given as a visual output. The workflow of GraParT is shown in Figure 1.

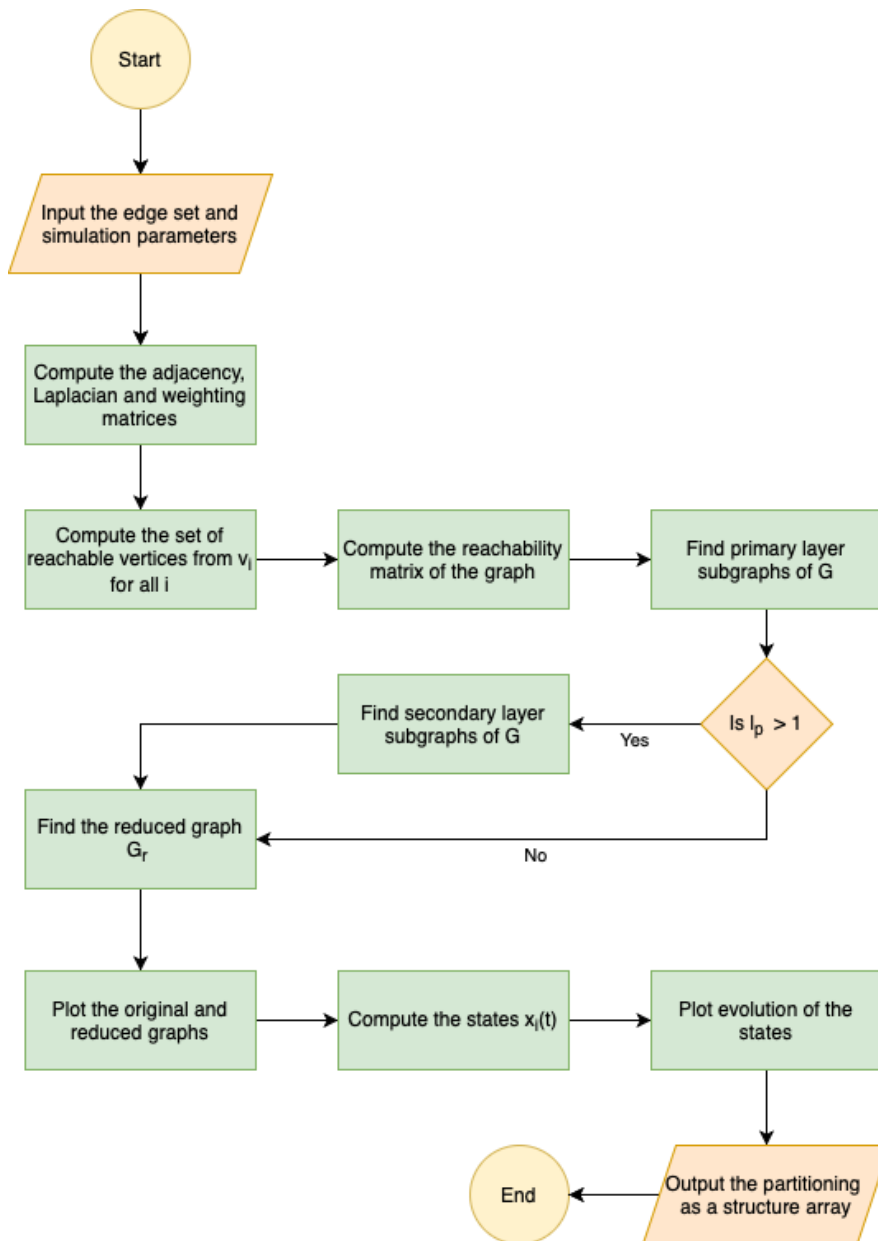


Figure 1 Overall workflow of GraParT

Note from Definition 1 that if a graph contains a single PLS, then all vertices of the graph is a member of the PLS and consequently the graph does not contain any secondary layer subgraphs. The PLS and SLS are determined by using the algorithms proposed in [9] and the reduced graph is determined by Algorithm 1.

3.3 The Graphical User Interface

GraParT has a user-friendly graphical user interface as illustrated in Figure 2.

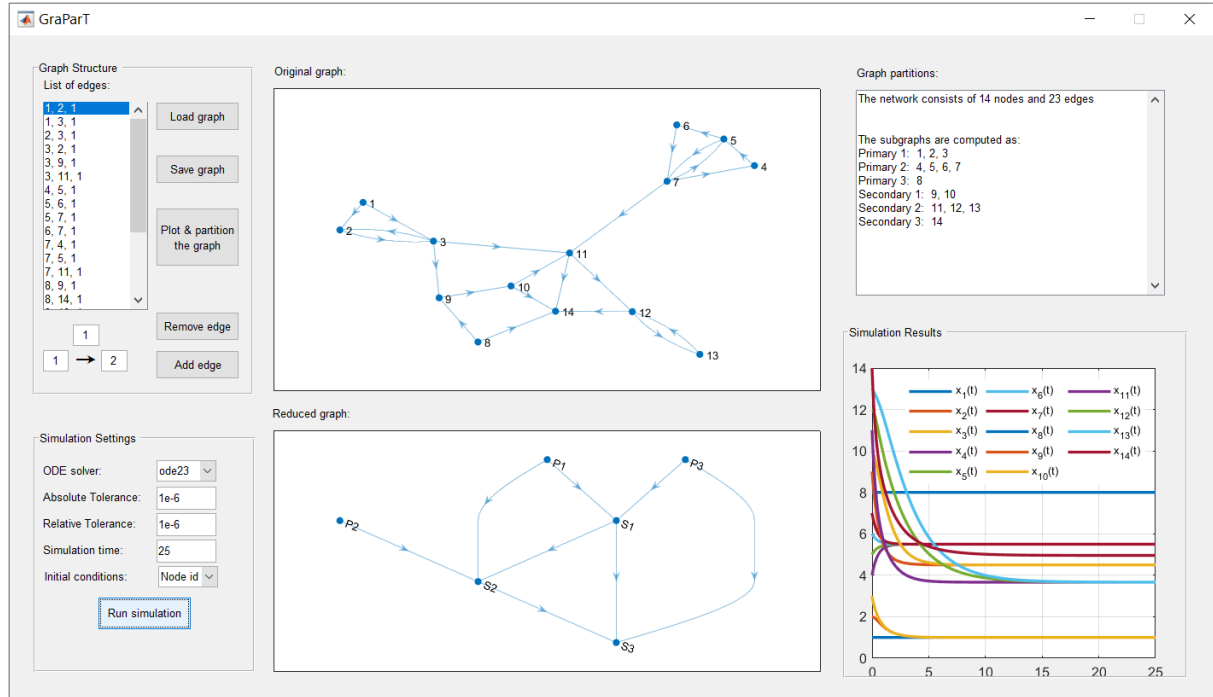


Figure 2 The Graphical User Interface of GraParT

The user enters the directed edges of the graph from the left-hand side of the user interface and the graphs can be saved as text files and loaded for future use. Once the user clicks the “Plot & partition the graph” button, GraParT computes the adjacency matrix, the Laplacian matrix and the weighting matrix; and partitions the graph into its PLS and SLS. The subgraphs and vertices are given as analysis results at the right-hand side of the interface. Furthermore, the reduced graph is computed and depicted together with the original graph.

The ordinary differential equation (ODE) solver that will be used to solve Equation 1, absolute and relative tolerances in the calculations, the maximum time for simulation (in seconds) and the initial states of the agents are simulation parameters to be entered by the user. The evolutions of the states are depicted as functions of time on the bottom-right side of the graphical interface and can be used to verify that the number of groups is equal to the number of subgraphs determined by GraParT.

In most multi-agent systems, the agents are mobile and as a result of this, the network connectivity may be time-varying. A new communication link may be built when two agents come closer, and an existing link may be broken when they move away from each other. In order to understand the grouping behavior of the network in such cases, we provide the following examples.

Example 1.

Consider a network of 14 agents whose edge set is given by

$$E_1 = \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_3, v_2), (v_3, v_9), (v_3, v_{11}), (v_4, v_5), (v_5, v_6), (v_5, v_7), (v_6, v_7), (v_7, v_4), (v_7, v_5), (v_7, v_{11}), (v_8, v_9), (v_8, v_{14}), (v_9, v_{10}), (v_{10}, v_{11}), (v_{10}, v_{14}), (v_{11}, v_{12}), (v_{11}, v_{14}), (v_{12}, v_{13}), (v_{12}, v_{14}), (v_{13}, v_{12})\}. \quad (12)$$

Since there is no spanning tree in the graph $G_1 = (V_1, E_1)$, multiple groups will be formed in the network. Once the directed edges are entered as inputs to GraParT, Figure 2 illustrates the network graph, the reduced graph; the PLS and SLS of G_1 , and evolutions of the states of the agents as functions of time. As can be seen from Figure 1, there are 3 PLS and 3 SLS in the network and the simulation

results verify that there are **6** consensus equilibria (groups of agents with same final values) in the network.

Example 2.

In order to understand the effect of adding edges to a graph on the grouping behavior of a multi-agent network, suppose that a new link between agents 8 and 13 is created. The edge set of the new graph can be written as

$$E_2 = \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_3, v_2), (v_3, v_9), (v_3, v_{11}), (v_4, v_5), (v_5, v_6), (v_5, v_7), (v_6, v_7), (v_7, v_4), (v_7, v_5), (v_7, v_{11}), (v_8, v_9), (v_8, v_{14}), (v_9, v_{10}), (v_{10}, v_{11}), (v_{10}, v_{14}), (v_{11}, v_{12}), (v_{11}, v_{14}), (v_{12}, v_{13}), (v_{12}, v_{14}), (v_{13}, v_{12}), (v_8, v_{13})\}. \tag{13}$$

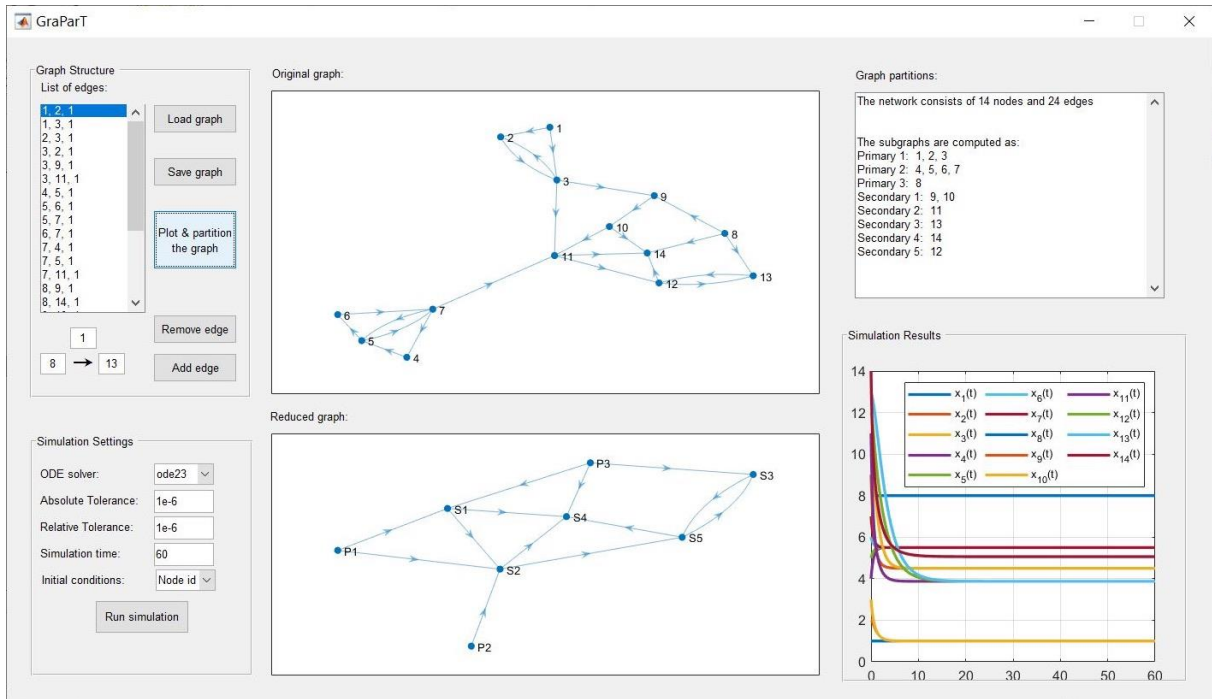


Figure 3 Partitioning of the network and the simulation results for Example 2

Figure 3 illustrates the network graph, the reduced graph; the PLS and SLS of G_2 , and evolutions of the states of the agents as functions of time. As can be seen from Figure 3, there are 2 PLS and 3 SLS in the network. In particular, the creation of the link between agents 8 and 13 resulted in the dissolution of the a secondary layer subgraph and formation of new secondary layer subgraphs in the modified graph. The total number of groups in the modified network is 8.

Example 3.

Reconsider the network under investigation in Example 2. Suppose that the link between agents 11 and 12 is removed. The edge set of the new graph can be written as

$$E_3 = \{(v_1, v_2), (v_1, v_3), (v_2, v_3), (v_3, v_2), (v_3, v_9), (v_3, v_{11}), (v_4, v_5), (v_5, v_6), (v_5, v_7), (v_6, v_7), (v_7, v_4), (v_7, v_5), (v_7, v_{11}), (v_8, v_9), (v_8, v_{14}), (v_9, v_{10}), (v_{10}, v_{11}), (v_{10}, v_{14}), (v_{11}, v_{14}), (v_{12}, v_{13}), (v_{12}, v_{14}), (v_{13}, v_{12}), (v_8, v_{13})\}. \tag{14}$$

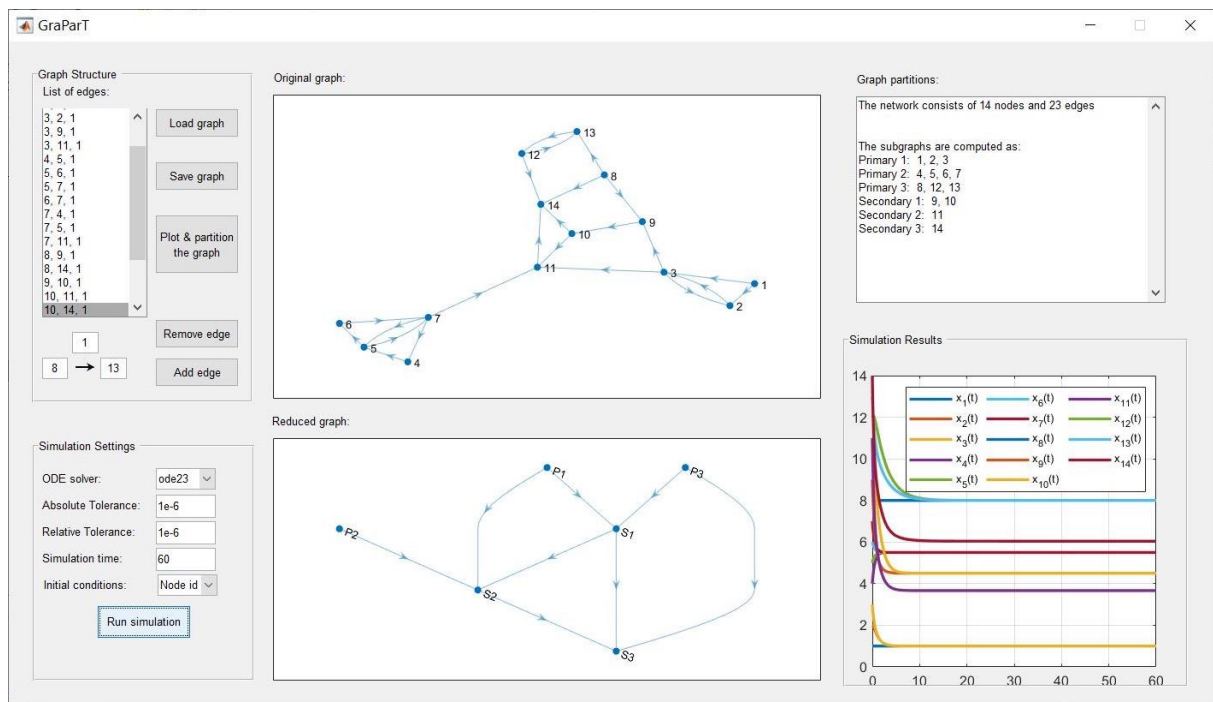


Figure 4 Partitioning of the network and the simulation results for Example 3

Figure 4 illustrates the network graph, the reduced graph; the PLS and SLS of G_3 , and evolutions of the states of the agents as functions of time. As can be seen from the figure, there are 3 PLS and 3 SLS in the network. In particular, the removal of the link between agents 11 and 12 resulted in unification of two SLS and a primary layer subgraph into a large primary layer subgraph (with the member agents 8, 12 and 13). The total number of groups in the modified network is 6.

Examples 1-3 show that modification of a directed graph by adding a new link or removing an existing link may result in an increase or a decrease of the number of groups formed in the network. By using the proposed partitioning tool, the network topology designer can create a directed multi-agent network with the desired grouping behavior.

4. Conclusion

In this paper, we provide a MATLAB toolbox for partitioning directed graphs into its PLS and SLS. The states of agents in these subgraphs asymptotically converge to the same values when they utilize a conventional continuous-time or discrete-time consensus algorithm. The toolbox enables the users to modify the graph and analyze the effect of these changes in the graph structure to the number of groups formed in the multi-agent network. We hope this toolbox will enrich the understanding of the grouping behavior of the multi-agent systems and will be a reference tool for network topology designers.

References

- [1] R. Aragues, J. Cortes, and C. Sagues, "Distributed consensus on robot networks for dynamically merging feature-based maps," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 840–854, 2012.
- [2] M. Mirzaei, H. Atrianfar, N. Mehdipour, and F. Abdollahi, "Asynchronous consensus of continuous-time lagrangian systems with switching topology and non-uniform time delay," *Rob. Auton. Syst.*, vol. 83, pp. 106–114, 2016.
- [3] N. Amelina, A. Fradkov, Y. Jiang, and D. J. Vergados, "Approximate consensus in stochastic

- networks with application to load balancing,” *IEEE Trans. Inform. Theory*, vol. 61, no. 4, pp. 1739–1752, 2015.
- [4] O. Cihan, “Distributed Solution of Road Lighting Problem Over Multi-Agent Networks,” *Sakarya University Journal of Computer and Information Sciences*, vol. 3, no. 2, pp. 89–98, 2020.
- [5] R. Hegselmann and U. Krause, “Opinion dynamics and bounded confidence: Models, analysis and simulation,” *J. Artif. Soc. Soc. Simul.*, vol. 5, no. 3, pp. 1–33, 2002.
- [6] O. Cihan, “Rapid solution of linear equations with distributed algorithms over networks,” *IFAC-PapersOnLine*, vol. 52, no. 25, pp. 467–471, 2019.
- [7] R. Olfati-Saber and R. M. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Trans. Automat. Control*, vol. 49, no. 9, pp. 1520–1533, 2004.
- [8] W. Ren and R. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Trans. Automat. Control*, vol. 50, no. 5, pp. 655–661, 2005.
- [9] Ö. F. Erkan, O. Cihan, and M. Akar, “Analysis of distributed consensus protocols with multi-equilibria under time-delays,” *J. Franklin Inst.*, vol. 355, no. 1, pp. 332–360, 2018.
- [10] O. Cihan and M. Akar, “Multi-consensus of second-order agents in discrete-time directed networks,” *Int. J. Syst. Sci.*, vol. 51, no. 10, pp. 1847–1861, 2020.
- [11] O. Cihan and M. Akar, “Necessary and Sufficient Conditions for Group Consensus of Agents With Third-Order Dynamics in Directed Networks,” *J. Dyn. Syst. Meas. Control*, vol. 142, no. 4, pp. 041003, 2020.
- [12] O. Cihan, “Topology design for group consensus in directed multi-agent systems,” *Kybernetika*, vol. 56, no. 3, pp. 578–597, 2020.
- [13] J. Hespanha, “An efficient MATLAB Algorithm for Graph Partitioning,” Technical Report, University of California, Oct. 2004.
- [14] Ü. Çatalyürek and C. Aykanat C, “PaToH (Partitioning Tool for Hypergraphs),” In: Padua D. (eds) *Encyclopedia of Parallel Computing*. Springer, Boston, MA, 2011.
- [15] A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz, “Recent Advances in Graph Partitioning,” In *Algorithm Engineering*, pp. 117–158, Springer International Publishing, 2016.
- [16] Ü. Develer and M. Akar, “Cluster consensus in first and second-order continuous-time networks with input and communication delays,” *International Journal of Control*, vol. 9 no. 4, pp. 961–976, 2021.

Effects of Binary Vectors Similarities on the Accuracy of Multi-Criteria Collaborative Filtering

 Burcu Demirelli Okkaloğlu¹

¹Yalova University, Faculty of Engineering, Computer Engineering Department;
bdokkalioglu@yalova.edu.tr

Received 16 June 2021; Revised 13 September 2021; Accepted 19 September 2021; Published online: 31 December 2021

Abstract

Recommender systems offer tailored recommendations by employing various algorithms, and collaborative filtering is one of the well-known and commonly used of those. A traditional collaborative filtering system allows users to rate on a single criterion. However, a single criterion may be insufficient to indicate preferences in domains such as restaurants, movies, or tourism. Multi-criteria collaborative filtering provides a multi-dimensional rating option. In similarity-based multi-criteria collaborative filtering schemes, existing similarity methods utilize co-users or co-items regardless of how many there are. However, a high correlation with a few co-ratings does not always provide a reliable neighborhood. Therefore, it is very common, in both single- and multi-criteria collaborative filtering, to weight similarities with functions utilizing the number of co-ratings. Since multi-criteria collaborative filtering is yet growing, it lacks a comprehensive view of the effects of similarity weighting. This work studies multi-criteria collaborative filtering and the literature of binary vector similarities, which are frequently used for weighting, by giving a related taxonomy and conducts extensive experiments to analyze the effects of weighting similarities on item- and user-based multi-criteria collaborative filtering. Experimental findings suggest that prediction accuracy of item-based multi-criteria collaborative filtering can be boosted by especially binary vector similarity measures which do not consider mutual absences.

Keywords: multi-criteria, collaborative filtering, similarity-weighting, binary vector similarity

1. Introduction

The number of people who have access to the Internet has been rapidly increasing. In parallel, e-commerce companies are increasing both in number and size. At this point, online companies like Amazon and Netflix utilize personalized recommendations to offer the right products to their customers. Recommender systems help companies understand their customers' tastes. By doing so, a customer might be interested in items otherwise he is not aware of. The first example of such a system was Tapestry [1], which was an experimental email filtering system at Xerox. Usually, recommender systems can be broadly categorized into collaborative filtering (CF), content-based filtering (CB), and hybrid approaches.

As the name suggests, CF systems require collaboration from their users [1], [2] to perform their tasks. Clearly, CF systems require user feedback. The feedback can be explicit or implicit. Explicit feedback is the ones that the users supply by casting their ratings. On the other hand, the CF system might utilize some implicit feedback such as browsing, click, or purchase history [3]. CF systems contain two approaches which can be either model or memory-based. Model-based approaches build a model and produce recommendations or predictions based on the model. On the other hand, memory-based approaches re-run the whole procedure to produce an output for the recommendation. The general idea in CF is that users who have similar preferences in the past will probably have similar preferences in the future as well. Therefore, memory-based CF approaches determine a set of neighbors for an active user. An active user is the one who is looking for a prediction or recommendation. The k nearest neighbor algorithm is the most dominant and widely used algorithm to select neighbors in user-based and item-based CF [4]. Although CF systems are widespread, they might suffer from data sparsity and cold start problems [5]. A cold start problem occurs when a new item or user is introduced into the system. Since there are not enough ratings for them, the CF system cannot produce a recommendation. CB utilizes contextual information of the items. Unlike CF, CB does not rely on different user feedbacks. Instead,

it essentially analyzes the content of the items rated by the active user and recommends similar items based on item contents. Such content can be described as a synopsis or actors for a movie-related domain. CB systems can be useful to avoid cold start problems for new items; however, the disadvantage is that it often produces obvious recommendations [6]. Hybrid approaches combine CF and CB to take advantage of them.

CF systems usually operate on $n \times m$ matrix, where n is the number of users while m is the number of items. Users rate items on a binary, interval, ordinal, or continuous scale [6]. In traditional CF systems, users express their preferences based on a single criterion. For example, a movie recommendation system collects the general preferences of the users about the movie. However, a movie evaluation might include several dimensions such as cast, story, direction, or action. Therefore, integrating multiple criteria might be useful to characterize items better. Adomavicius and Kwon [7] propose recommendation techniques for multi-criteria CF (MCCF) systems. Their seminal paper was the first to introduce the concept of MCCF and to propose two methods called similarity-based and aggregation function-based. In the similarity-based approach, the similarity values between users or items are calculated for each criterion using any existing similarity measures and then the overall similarity is calculated by averaging or selecting the smallest one. In the second method, the aggregation function-based approach, a relationship between the overall rating and sub-criteria ratings is extracted by utilizing an aggregation function. Since it is important to find an appropriate aggregation function, domain expertise, statistical techniques, and machine learning methods can be used.

Herlocker et al. [8] study the effectiveness of significance or similarity weighting in single criterion CF. The idea is that an active user might be highly correlated with some users on a small number of co-rated items. For example, two users might be perfectly correlated based on a single item and such correlation is stronger than a correlation of 0.9 over hundred co-rated items. Therefore, the authors [8] argue that the correlation between users can be adjusted by introducing significant weighting. In their original method, they use a threshold-based significance weighting factor. Although single criteria CF schemes have long been utilizing similarity weighting [9]-[12], two recent studies [13], [14] employ similarity weighting in MCCF. Scholars [13] improve the prediction performance of item-based MMCF. Yalcin and Bilge [14] propose a binary MCCF method and they apply binary similarity weighting to calculate the correlation between users. It is clear that MCCF literature lacks a systematic view of the effects of similarity weighting on the prediction performance.

In this study, we present a comprehensive study of the effects of binary vector similarity weighting on both item- and user-based MCCF. Binary vector similarity is used as a weighting factor in addition to the existing similarity measures to improve the prediction performance of MCCF. Considering the structure of the existing similarity measures in similarity-based approaches, similarities are only calculated based on co-rated users or items. Then, nearest neighbors are selected based on these calculated similarity values. On the other hand, higher similarity values may not always indicate a better correlation between users or items because these metrics do not consider how many co-rated items exist between two users or vice versa. In item- or user-based schemes, it is sufficient to have a single common item/user to calculate a similarity score between two users or items. In a user-based perspective, a user vector consists of item ratings while, in an item-based perspective, an item vector consists of users' ratings on the related item. At this point, binary vector representation of the rating vectors can reveal which items/users are rated by encoding rated items by 1 and unrated items by 0. Then, binary vector representations of two users' or items' vectors might be exploited in the similarity calculations since they inform the mutual and non-mutual presences as well as mutual and non-mutual absences of the ratings. Utilizing binary vectors similarities as weighting factors in addition to the existing methods is a solution to overcome the shortcoming of the limited number of co-ratings. This study presents extensive literature on binary vector similarity measures and their effect on similarity weighting in MCCF schemes. Besides, this work can serve as a reference point for future researchers to understand the contribution of binary vector similarity weighting on MCCF prediction accuracy.

The rest of the paper is organized as follows. In Section 2, the related work is given briefly. Section 3 represents detailed information about CF and MCCF. In Section 4, we give a review of binary vector similarity literature and related taxonomy. Then, we discuss how to apply binary vector similarities to

weight MCCF similarities. We experimentally show in Section 5 how binary similarity weighting affects the general prediction performance of user-based and item-based MCCF. Section 6 gives the discussion. The final section represents our conclusions.

2. Related Work

The implementation of multi-criteria recommendation systems dates to the early 2000s. First, Plantié, Montmain, and Dray [15] suggest using a decision support system that combines text mining techniques with multi-criteria analysis techniques to develop movie-based recommendation systems. Adomavicius and Kwon [7] state that using multi-criteria recommendation systems instead of traditional recommendation systems increase the likelihood that the correlations calculated between users. Researchers argue that rating items with more criteria, such as visual effects or scenario in a movie dataset, rather than giving a single rating about the items may also increase the accuracy of the recommendation. They show how traditional memory-based CF algorithms can be adapted to MCCF systems. First, researchers produce recommendations using traditional similarity-based approaches. While calculating the similarities between users, the system calculates the similarity value separately for each criterion as in the traditional similarity-based approach. Then, the average or smallest value of all criteria can be selected as a final similarity. They also show that similarity calculations can be made using multidimensional distance metrics. With this approach, it has been shown that only the traditional similarity calculation process has changed, the rest of the process will remain the same as in traditional CF algorithms to provide predictions. Researchers also mention that all other methods used in single-criteria or traditional recommendation systems can also be adapted to multi-criteria systems beside memory-based collaborative filtering algorithms. Bilge and Kaleli [16] propose a framework that adapts the work proposed by Adomavicius and Kwon [7] to multi-criteria item-based CF systems. The study shows that the performance of item-based multi-criteria systems is better than traditional item-based CF systems.

In memory-based CF algorithms, the number of items commonly rated among users is an important factor. Considering the nature of recommender systems, most items are not rated, and this situation causes similarity calculations to be made on a very small number of co-rated items when calculating a similarity value between two users. The similarity value calculated over a few common items may not reflect the actual correlation. Therefore, Herlocker et al. [8] propose to use a weighting method in traditional recommender systems while calculating the similarity between users. In the proposed method called significance-weighting, a threshold value is defined, and if the number of items voted commonly by two users is less than this value, the similarity is multiplied by the calculated weighting, thus the similarity between the two users is reduced to a certain extent. The goal is to obtain a more reliable similarity between two users. The more commonly rated items between two users, the more trust these two users have. Ma, King, and Lyu [9] modify the work proposed by Herlocker et al. [8] and apply the modified algorithm when calculating similarity both between users and items. Polatidis and Georgiadis [10] propose a new similarity calculation process inspired by the work [8] and divide the similarity process into multiple levels. If the conditions specified at each level are fulfilled, a value determined by the researchers is added to the calculated similarity. The aim of the researchers is to increase the similarity values of these users as more items are rated in common. In another study of the researchers, a dynamic multi-level CF system is also proposed [11]. If the criteria specified in the study are met, the correlation is affected positively by adding different numbers to the similarity calculation for different levels. On the other hand, if no criteria are met, the similarity is multiplied by the specified value and the similarity is devalued. Candillier, Meyer, and Fessant [12] suggest using the Jaccard similarity as a weighting method by multiplying it with the existing similarity methods for CF systems. The study shows that the proposed method gives better results than traditional methods. Shambour and Lu [17] propose to use the Jaccard similarity as a weighting method in item-based MCCF. First, adjusted cosine similarity (ACS) is used to calculate the similarity between items for each criterion. Then, the final similarity is calculated using the weighted average method as in the traditional multi-criteria algorithms. Researchers propose a new similarity calculation that multiplies the final calculated similarity value with Jaccard. Shambour, Hourani, and Fraihat [18] propose to apply the Dice metric as a weighting

method for multi-criteria item-based CF. Dice metric calculates a similarity between two items based on common users, such as Jaccard similarity. Euclidean distance is used while calculating the similarities between items for each criterion. Since the distance metric is selected to calculate similarity values, the similarity with the lowest value is chosen as the final similarity. Researchers propose a new similarity calculation process that multiplies the Dice metric with the final similarity in item-based MCCF systems. In a similar study to the last two, it is proposed for user-based MCCF systems [19]. The researchers integrate the Dice metric into ACS as a weighting method as in previous studies. Sadikoglu and Demirelli Okkalioglu [13] propose to apply Jaccard similarity and significance-weighting as weighting methods to improve the existing similarity calculation process in similarity-based approaches in item-based MCCF. The proposed two weighting methods are multiplied by Pearson Correlation Coefficient (PCC) and ACS for each criterion. The final similarity value is obtained by either the weighted average method or the minimum method. Researchers show that the proposed methods improve the accuracy and coverage of predictions compared to the traditional methods. Yalcin and Bilge [14] propose a binary MCCF method and they apply binary similarity weighting to calculate the correlation between users. Table 1 displays list of MCF work that utilize similarity weighting for comparison purposes.

Table 1 Comparison of the existing work

References	Data Type	Scheme	Similarity Weighting
Shambour and Lu [17]	Numeric	Item-based	Jaccard
Shambour, Hourani, and Fraihat [18]	Numeric	Item-based	Dice
Shambour [19]	Numeric	User-based	Dice
Sadikoglu and Demirelli Okkalioglu [13]	Numeric	Item-based	Jaccard and Significance-weighting
Yalcin and Bilge [14]	Binary	User-based, Item-based	Jaccard, Czekanowski, Simpson, Kulczynski, and Johnson

These studies state that high correlation values may not always ensure the best neighbors in similarity-based approaches. As the number of co-users or co-items increases, the correlation calculated between them is more reliable. Therefore, different weighting methods are proposed to change the neighbor selection process and to improve the prediction performance. Unlike the works presented here, we investigate the binary vector similarity literature and utilize a variety of binary similarity measures as weighting factors in the MCCF similarity calculation process. Thus, our work presents an enhanced view of the effects of binary vector similarities on the prediction accuracy of MCF schemes.

3. Multi-Criteria Collaborative Filtering

Before introducing MCF, brief information about a traditional CF is given. The best-known CF algorithm is the memory-based or neighborhood-based algorithm. The memory-based algorithms are also divided into two classes: user-based and item-based. The underlying approach of user-based is to find like-minded users when an active user (a) asks for a prediction for a target item (q). Equation 1 shows a formula how to calculate a prediction after users are selected as nearest neighbors for a .

$$pred(a, q) = \bar{r}_a + \frac{\sum_{u=1}^k (r_{u,q} - \bar{r}_u) sim(a, u)}{\sum_{u=1}^k sim(a, u)} \quad (1)$$

where k is k -nearest-neighbors of a . \bar{r}_a and \bar{r}_u are mean ratings for user a and user u , respectively. $r_{u,q}$ is the rating of user u on item q and $sim(a, u)$ illustrates the similarity value between user a and user u .

The similarity between users is crucial in CF. There are different methods used in the literature. One of the most popular similarity methods is Pearson Correlation Coefficient (PCC) [20]. PCC calculates similarities in the range [-1, 1], where 1 depicts the highest correlation whereas -1 illustrates the worst correlation between two users. Equation 2 shows how to apply PCC between two users, a and u :

$$sim(a, u) = \frac{\sum_{p \in P} (r_{a,p} - \bar{r}_a)(r_{u,p} - \bar{r}_u)}{\sqrt{\sum_{p \in P} (r_{a,p} - \bar{r}_a)^2} \sqrt{\sum_{p \in P} (r_{u,p} - \bar{r}_u)^2}} \quad (2)$$

where P demonstrates the set of items rated both an active user, a , and a user, u , $r_{a,p}$ and $r_{u,p}$ represent ratings for item p rated by a and u , respectively. \bar{r}_a and \bar{r}_u are the mean of P items' ratings of a and u .

Whereas user-based algorithms produce a prediction by utilizing user similarities, item-based algorithms generate a prediction based on similarities between items. In this case, an active user (a) asks for a prediction for a target item (q) as in user-based algorithms. The algorithm finds similarities between the target item and other items and k items are selected as the best neighbors. The prediction formula is given in Equation 3.

$$pred(a, q) = \frac{\sum_{i \in ratedItem(a)} sim(i, q) \times r_{a,i}}{\sum_{i \in ratedItem(a)} sim(i, q)} \quad (3)$$

Where $sim(i, q)$ shows the similarity value between item i and item q , $r_{a,i}$ represents the rating value given by user a to item i . Recall that among the selected neighboring items, the items rated by the user a are included in this calculation.

ACS is the common similarity method when calculating similarities between items [21]. The similarity value between two items in ACS is also in the range of $[-1, 1]$ as in PCC. ACS performs the similarity calculation between item i and q using Equation 4.

$$sim(i, q) = \frac{\sum_{u \in U} (R_{u,i} - \bar{R}_u)(R_{u,q} - \bar{R}_u)}{\sqrt{\sum_{u \in U} (R_{u,i} - \bar{R}_u)^2} \sqrt{\sum_{u \in U} (R_{u,q} - \bar{R}_u)^2}} \quad (4)$$

In Equation 4, U indicates common users who rated item i and item q . $R_{u,i}$ and $R_{u,q}$ show the rating given by user u to items i and item q , respectively. \bar{R}_u represents the average of ratings of user u .

In addition, PCC and ACS are the best-known similarity methods in CF. Therefore, these methods are widely used in both user-based and item-based algorithms. Equation 2 represents user-based similarity calculation; however, the formula can easily be converted into an item-based one. Likewise, Equation 4 shows item-based ACS, but it can be modified to user-based ACS. Since corresponding item- or user-based formulas are trivial, these formulas are briefly presented here.

A traditional CF system performs similarity calculations over a single criterion. Users send overall ratings about items and a prediction is produced using a single rating. On the other hand, researchers put forward an idea that multi-criteria systems could better reflect the characteristics of the users instead of a single rating-based system. Multi-criteria systems allow users to rate more than one criterion separately. Users reflect their personal preferences better due to the multi-criteria system. A rating function R in a multi-criteria CF is represented as follows:

$$R: users \times items \rightarrow R_0 \times R_1 \times \dots \times R_k \quad (5)$$

where R_0 shows overall ratings that users rate items and R_j represents that users rate items for j th criteria where $j = 1, 2, \dots, k$. k is the number of criteria.

In multi-criteria CF, Adomavicius and Known [7] propose similarity-based and aggregation function-based approaches to provide predictions. In this study, similarity-based approaches are applied. In this approach, similarity values between users/items are calculated using any existing similarity measures for each criterion separately as in traditional CF. Then, it is required to aggregate individual similarities to obtain an overall similarity. Adomavicius and Known [7] introduce two methods that aggregate individual similarities: average and worst-case similarity. In the average similarity approach as seen in Equation 6, the similarities of all individual criteria are averaged to get an overall similarity. In Equation 6, k defines the number of criteria, $sim_c(i, j)$ shows the similarity between item i and item j for criterion c .

$$sim_{avg}(i, j) = \frac{\sum_{c=0}^k sim_c(i, j)}{k + 1} \tag{6}$$

The second aggregation approach is called the worst-case or minimum similarity. The main purpose is to select minimum similarity as a final similarity among all individual criteria. The final similarity is estimated as seen in Equation 7.

$$sim_{min}(i, j) = \min_{c=0,1,\dots,k} sim_c(i, j) \tag{7}$$

Although Equation 6 and Equation 7 show how to calculate the overall similarity in item-based MCCF algorithms, user-based MCCF can be applied similarly [7].

4. Weighting Similarities in Multi-Criteria Collaborative Filtering

4.1 Binary Vector Representations and Similarities

In this section, our motivation is to present binary similarity measures to use them as weighting factors in the neighborhood selection. Since rating vectors are generally very sparse [22] in CF, similarity calculations can be boosted if weighting factors are utilized [23], [24]. PCC, ACS, or other similarity measures are applied over co-rated numeric items. Herlocker et al. [8] argue that the number of co-rated items can contribute to the similarity score as a weighting factor because the similarity between two vectors might have been calculated over few items. For example, the same similarity score, say 1.0 (perfect), may not indicate the same degree of similarity for vectors, which have few versus many co-rated elements. Therefore, weighting similarity scores is used in CF literature [8], [13], [14], [17]-[19], [23], [24] in the neighborhood selection.

A weighting factor can be calculated over binary representations of vectors. Assume that the rating range is a set of discrete numeric values, say $R = \{0, 1, 2, 3, 4, 5\}$, where 0 means unrated. For an item-based MCCF, an item vector for a criterion can be defined as $item_j = \{r_1, r_2, \dots, r_{n-1}, r_n\}$, where j is the item id and $r_i \in R, \forall i \in \{1, 2, \dots, n\}$. Such a vector can be also expressed in a binary form where each r_i is replaced either 1 if $r_i \neq 0$, or 0 otherwise. In binary vector format, 1s represent the presence while 0s represent the absence. For example, a rating vector of $item_1 = \{1, 2, 3, 0, 0, 4, 5, 0, 0, 2, 0\}$ can be transformed into a binary vector, $item_1 = \{1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0\}$. These binary vectors are also called binary basket data in the data mining literature [25].

While calculating the similarities of binary vectors, corresponding absence and presence become crucial. Table 2 displays how absence and presence value will be represented for two vectors, say v_1 and v_2 . a is $|v_1 \cdot v_2|_1$, which is the number of 1s after the dot product. In other words, a represents the number of common elements that both v_1 and v_2 have 1, which is the mutual presence. b is $|v_1 \cdot \overline{v_2}|_1$, which is the number of elements that is present in v_1 but absent in v_2 . c is $|\overline{v_1} \cdot v_2|_1$, which is the number of elements that is absent in v_1 but present in v_2 . $b + c$ can be called the non-mutual presence. d is $|\overline{v_1} \cdot \overline{v_2}|_1$ is the number of elements that are absent in both v_1 and v_2 , which is the mutual absence. Obviously, $|v_1 \cdot v_1|_1 = a + b$, $|v_2 \cdot v_2|_1 = a + c$, and $a + b + c + d$ is the number of features. In our case with item-item similarities, the number of features will be n , which is the number of users.

Table 2 Notation for binary vector similarities

v_1	v_2	
	Presence (1)	Absence (0)
Presence (1)	a	b
Absence (0)	c	d
	$n = a + b + c + d$	

Based on the notation in Table 2, many variations of binary vector similarities can be calculated. Table 3 lists 21 different measures including similarities and distances. The use of d is controversial [26], [27] while some measures do not include it some others include. Because the number of attributes that are absent in both vectors is very dominant, d might be omitted. This is true in CF, as well. These similarity and distance measures can be broadly categorized into two, ones ignoring d and ones considering d

[27], [28]. In addition, binary similarity functions that consider d in the calculation can be further classified as the ones treating both a and d equally or unequally [27].

Table 3 Binary similarity and distance measures

	Measure	Equation	Range	Ref
Similarity measures ignoring d	Jaccard	$\frac{a}{a+b+c}$	[0, 1]	[26], [27]
	Dice	$\frac{2 \times a}{2 \times a + b + c}$	[0, 1]	[26], [27]
	3W-Jaccard	$\frac{3 \times a}{3 \times a + b + c}$	[0, 1]	[26], [27]
	Sokal-Sneath-I	$\frac{a}{a + 2 \times b + 2 \times c}$	[0, 1]	[26], [27]
	Simpson	$\frac{a}{\min(a+b, a+c)}$	[0, 1]	[26], [27]
	Braun	$\frac{a}{\max(a+b, a+c)}$	[0, 1]	[26], [27]
	Johnson	$\frac{a}{a+b} + \frac{a}{a+c}$	[0, 2]	[26], [27]
	Kulczynski-I	$\frac{a}{b+c}$	[0, ∞)	[26], [27]
	Sorgenfrei	$\frac{(a \times a)}{(a+b) \times (a+c)}$	[0, 1]	[26], [27]
	Ochiai	$\frac{a}{\sqrt{(a+b)(a+c)}}$	[0, 1]	[26], [27]
Similarity measures considering d	Russel-Rao	$\frac{a}{a+b+c+d}$	[0, 1]	[26], [27]
	Sokal-Michener	$\frac{a+d}{a+b+c+d}$	[0, 1]	[26], [27]
	Sokal-Sneath-2	$\frac{2 \times (a+d)}{2 \times a + b + c + 2 \times d}$	[0, 1]	[26], [27]
	Roger-Tanimoto	$\frac{a+d}{a+2 \times (b+c)+d}$	[0, 1]	[26], [27]
	Gower-Legendre	$\frac{a+d}{a+0.5 \times (b+c)+d}$	[0, 1]	[26], [27]
	Faith	$\frac{a+0.5 \times d}{a+b+c+d}$	[0, 1]	[26], [27]
Distances ignoring d	Squared-Euclid	$\sqrt{b+c}^2$	[0, ∞)	[26], [27]
	Manhattan	$b+c$	[0, ∞)	[26], [27]
Distances considering d	Mean- Manhattan	$\frac{b+c}{a+b+c+d}$	[0, 1]	[26], [27]
	Size-Difference	$\frac{(b+c)^2}{(a+b+c+d)^2}$	[0, 1]	[26], [27]
	Shape-Difference	$\frac{n \times (b+c) - (b-c)^2}{(a+b+c+d)^2}$	[0, 1]	[26], [27]

Similarity measures that ignore d are generally useful for asymmetric binary vectors, where the presence is more important than absence, such as market basket data in the data mining literature. There might be many items that can go into a market basket; however, any two customers can have a very limited number of common items in their baskets. Therefore, calculating a similarity using d either in the numerator or denominator part of the equation devalues the mutual presence, which is a . The examples of such similarity measures that do not utilize d in Table 3 start from Jaccard until Ochiai. Jaccard is the ratio of dot products of two vectors in consideration over the union of presences (mutual presence plus non-mutual presence). The Jaccard measure does not prioritize mutual presence over non-mutual presences. However, Dice and 3w-Jaccard weigh the mutual-presence (a) by two and three, respectively. On the contrary, Sokal-Sneath-I imposes a penalty for non-mutual presence by a factor of two. Simpson and Braun measures are the ratio of mutual presence to the minimum and maximum non-mutual presence, respectively. Johnson, on the other hand, utilizes both. Kulczynski-I measure is the ratio of

mutual presence over non-mutual presence. In terms of set theory, it is the cardinality of the intersection over the sum of the cardinalities of relative differences. However, such a measure as Kulczynski range between 0 and ∞ . To avoid it, we use the updated Kulczynski formula [28]. Sorgenfrei and Ochiai measures are similar, where Ochiai is the extension of Cosine similarity when vectors are binary. Sorgenfrei, on the other hand, is the square of Ochiai measure.

Another group of measures utilizes d in a very similar nature. Russel-Rao is similar to Jaccard; however, it measures mutual presences over all elements. Sokal-Michener does not weigh any of a, b, c, d while Sokal-Sneath-2, Roger-Tanimoto, Faith, and Gower-Legendre utilize weighting on some of them. Distance measures need to be transformed into similarities. Squared-Euclid and Manhattan value can range between 0 and ∞ ; therefore, we converted them to $1/(1 + distance)$, where $distance$ is either Squared-Euclid or Manhattan score. Likewise, the rest of the distance metrics are converted into $(1 - distance)$ because their range is between 0 and 1.

4.2 Weighting Similarities in Multi-Criteria Collaborative Filtering by Binary Vector Similarity Measures

As introduced, PCC and ACS are the most prevalent techniques to calculate similarities in item- and user-based MCCF. Considering the item-based MCCF for brevity, both methods compute the item-item similarities based on users who rate the related items. Assume that we need to calculate $item_1$'s similarity to $item_2$ and $item_3$ and the number of common users who rate $item_1$ and $item_2$ is 1 while the number of users who both rate $item_1$ and $item_3$ is 100. The similarity score between $item_1$ and $item_2$, which is based on a single user, might be greater than the similarity between $item_1$ and $item_3$. Since PCC and ACS do not take the number of users who rate both items into account, the similarity scores may become controversial. According to Herlocker et al. [8], similarities become more credible if items are co-rated by many users. Therefore, similarities can be weighted by the number of users co-rating both items. For this purpose, we will utilize binary vector similarities to weigh item-item similarities as given in Equation 8, which gives the average aggregation, and Equation 9, which gives the minimum aggregation. Here, the item-item similarity for each criterion, sim_c , is multiplied by the binary similarity measure, which is $weightingFactor_c$, which can be adopted from Table 3.

$$sim'_{avg}(i, j) = \frac{\sum_{c=0}^k sim_c(i, j) \times weightingFactor_c(i, j)}{k + 1} \quad (8)$$

$$sim'_{min}(i, j) = \min_{c=0,1,\dots,k} sim_c(i, j) \times weightingFactor_c(i, j) \quad (9)$$

We briefly discuss item-based MCCF to weight similarities, the very similar idea can be applied to the user-based MCCF. In user-based settings, user-user similarities must be weighted by the binary vector similarities that are constructed for the co-rated items between users. Each calculated similarity between users for each criterion is multiplied by the corresponding binary vector similarity measure computed by the related user vectors. Again, these measures can be adopted from Table 3.

5. Experimental Evaluation

5.1 Data set


The most widely used dataset in MCCF, Yahoo!Movies, is used to test our proposed approaches. The dataset includes four sub-criteria and an overall rating for the movie domain. Users are asked to rate criteria such as direction, acting, story, and visuals. In addition, the average of four sub-criteria is calculated as the overall rating. Due to the extreme sparsity of the dataset, two subsets of Yahoo!Movies are created [29]. First, a subset of users and items with at least 10 ratings is created and called YM10. Likewise, YM20 dataset is also obtained with a subset of users and items with at least 20 ratings. Table 4 displays YM10 and YM20 data sets and their corresponding number of users, items and the ratings they have.

Table 4 Datasets

	YM10	YM20
# users	1293	202
# items	1164	247
# ratings	34846	8157

Yahoo!Movies includes 13 different ratings, and the rating range is from A+ to F. A+ indicates the highest rating, while F shows the lowest rating. Since CF systems usually use 5 star-scale, the ratings in Yahoo!Movies are converted into 5 star-scale. (A+, A, A-) is converted to 5, (B+, B, B-) is exchanged to 4, others are transformed to 3, 2, and 1 is assigned to the letter F [29]. A small example of the multi-criteria user-item matrix is given in Figure 1.

	<i>item</i> ₁	<i>item</i> ₂	<i>item</i> ₃	<i>item</i> ₄
<i>user</i> ₁	-	$B_{C^+,A^+,C^+,A}^+$	-	$B_{B^+,C,B,A}$
<i>user</i> ₂	$D_{C,F,D,D}$	-	-	$F_{F,F,F,D}^-$
<i>user</i> ₃	$A_{A^+,A,A,A}$	-	$C_{B,C,C,D}$	-
<i>user</i> ₄	-	$A_{A^-,B^+,A,A}^-$	$B_{C,A,A,C}$	-



	<i>item</i> ₁	<i>item</i> ₂	<i>item</i> ₃	<i>item</i> ₄
<i>user</i> ₁	-	4 _{3,5,3,5}	-	4 _{4,3,4,5}
<i>user</i> ₂	2 _{3,1,2,2}	-	-	1 _{1,1,1,2}
<i>user</i> ₃	5 _{5,5,5,5}	-	3 _{4,3,3,2}	-
<i>user</i> ₄	-	5 _{5,5,4,5}	4 _{3,5,5,3}	-

Figure 1 An Example of the multi-criteria user-item matrix

5.2 Methodology

Experiments are conducted with YM20 and YM10 data sets. 5-fold cross-validation is utilized where each experiment is split into four training and one test set. Each time a prediction is produced for an active user (test user in the test set). For an active user, all rated items are listed and one of them is removed and the original value is stored for future tests, a prediction is produced for the removed target item, q . After the prediction is calculated, it is compared with the original value to calculate the accuracy metric, which is given in 6.3. Then, the removed original value is copied back for the next test. This strategy is repeated and known as leave-one-out. The number of neighbors is set to 40 [13], [16].

5.3 Evaluation Criteria

Mean absolute error, MAE, is used as the evaluation criteria. MAE measures the prediction error in terms of absolute difference over all queries. Its formulation is given in Equation 10, where p_i is the prediction, o_i is the original rating, and R is the number of predictions. It basically tells how much the predictions deviate from the original value.

$$MAE = \frac{1}{R} \sum_{i=1}^R |p_i - o_i| \quad (10)$$

5.4 Experiments

5.4.1 Effects of binary vector similarity on item-based MMCF

In the first experiment, we analyze how different binary similarity measures affect MAE scores when item-based MMCF is employed. Recall that PCC and ACS item-item similarities are aggregated with average and minimum. Table 5 displays the related results for both YM20 and YM10 data sets. Notice that bold table cells show that corresponding binary vector similarity weighting outperforms the original MCCF scheme.

The traditional MCCF method performs around 0.661, 0.681, 0.590, and 0.602 when PCC and ACS are aggregated by average and minimum functions, respectively, for YM20 data set. When the binary vector similarity measures are introduced in the item-item similarity calculation, most of them outperform the traditional MCCF method as seen in Table 5. The most remarkable measures are Sorgenfrei, Kulczynski-I, for YM20 data set. Sorgenfrei, Kulczynski-I are always among the top three measures in terms of MAE improvement compared to the traditional MCCF. Soregnfrei achieves 8%, 6.7%, 6.3%, and 5.8%, and Kulczynski-I achieves 7.5%, 6.5%, 6.6%, and 4.7% improvements for PCC-AVG, PCC-

MIN, ACS-AVG, and ACS-MIN, respectively. Sokal-Sneath-I accompanies these two measures in top-three in terms of MAE improvement when item-item similarities are calculated by PCC-MIN, ACS-AVG, and ACS-MIN. On the other hand, the worst performing measure is the Shape-Difference which always falls behind the traditional MCCF. However, it is prominent in this experiment to note that none of the binary vector similarities perform worse than the traditional MCCF except Shape-Difference for YM20 data set. Each of them, at least, achieves the same results with the traditional MCCF. Another interesting fact about binary similarity measures is that binary similarity measures ignoring d always perform better than both traditional MCCF and the rest of the binary similarity measures except Russel-Rao for YM20 data set.

Table 5 MAE Results When Binary Vector Similarities Are Applied in Item-based MCCF

	YM20				YM10			
	PCC AVG	PCC MIN	ACS AVG	ACS MIN	PCC AVG	PCC MIN	ACS AVG	ACS MIN
MCCF Traditional	0.661	0.681	0.590	0.602	0.760	0.772	0.777	0.777
Jaccard	0.617	0.652	0.555	0.577	0.677	0.711	0.605	0.623
Dice	0.622	0.654	0.558	0.579	0.679	0.714	0.609	0.624
3W-Jaccard	0.627	0.658	0.560	0.581	0.682	0.717	0.610	0.625
Sokal-Sneath-I	0.615	0.646	0.553	0.576	0.676	0.709	0.604	0.622
Simpson	0.624	0.660	0.562	0.582	0.692	0.727	0.632	0.639
Braun	0.623	0.655	0.559	0.580	0.685	0.715	0.613	0.628
Johnson	0.622	0.657	0.559	0.579	0.684	0.720	0.615	0.627
Kulczynski-I	0.611	0.637	0.551	0.574	0.674	0.707	0.603	0.620
Sorgenfrei	0.608	0.635	0.553	0.567	0.665	0.700	0.603	0.617
Ochiai	0.623	0.657	0.558	0.580	0.681	0.715	0.609	0.622
Russel-Rao	0.613	0.648	0.560	0.579	0.680	0.714	0.620	0.625
Sokal-Michener	0.658	0.681	0.584	0.600	0.757	0.769	0.755	0.752
Sokal-Sneath-2	0.661	0.681	0.586	0.601	0.760	0.770	0.758	0.750
Roger-Tanimoto	0.656	0.681	0.584	0.598	0.755	0.770	0.754	0.750
Gower-Legendre	0.661	0.681	0.586	0.601	0.760	0.770	0.758	0.750
Faith	0.652	0.678	0.581	0.598	0.755	0.768	0.747	0.748
Squared-Euclid	0.647	0.676	0.579	0.594	0.736	0.756	0.715	0.727
Manhattan	0.647	0.676	0.579	0.594	0.736	0.756	0.715	0.727
Mean- Manhattan	0.658	0.681	0.584	0.600	0.757	0.769	0.755	0.752
Size-Difference	0.662	0.681	0.588	0.602	0.760	0.770	0.759	0.750
Shape-Difference	0.864	0.942	1.030	1.060	0.888	0.905	1.032	1.065

Table 5 also displays the results for YM10 data set. The traditional method for MCCF records 0.760, 0.772, 0.777, and 0.777 for PCC and ACS when aggregated with average and minimum function, respectively. In all tests, the top three improvement scores are achieved by measures that ignore d , Sorgenfrei, Kulczynski-I, and Sokal-Sneath-I. Sorgenfrei performs 12.5%, 9.3%, 22.4%, and 20.6% improvement for PCC-AVG, PCC-MIN, ACS-AVG, and ACS-MIN, respectively. Moreover, Sorgenfrei is always the best-performing measure. Kulczynski-I improves the traditional MCCF by 11.3%, 8.4%, 22.4%, and 20.2%, while Sokal-Sneath-I achieves an improvement of 11.1%, 8.2%, 22.3%, and 19.9% for PCC-AVG, PCC-MIN, ACS-AVG, and ACS-MIN, respectively. Beside these top three measures, the other binary vector similarity measures except Shape-Difference achieve an improvement over traditional MCCF.

Since Table 5 includes experimental results of 22 different binary similarity measures for four different MCCF methods and two different data sets, we also illustrate the best performing binary similarity measures in Figure 2 for readers' convenience.

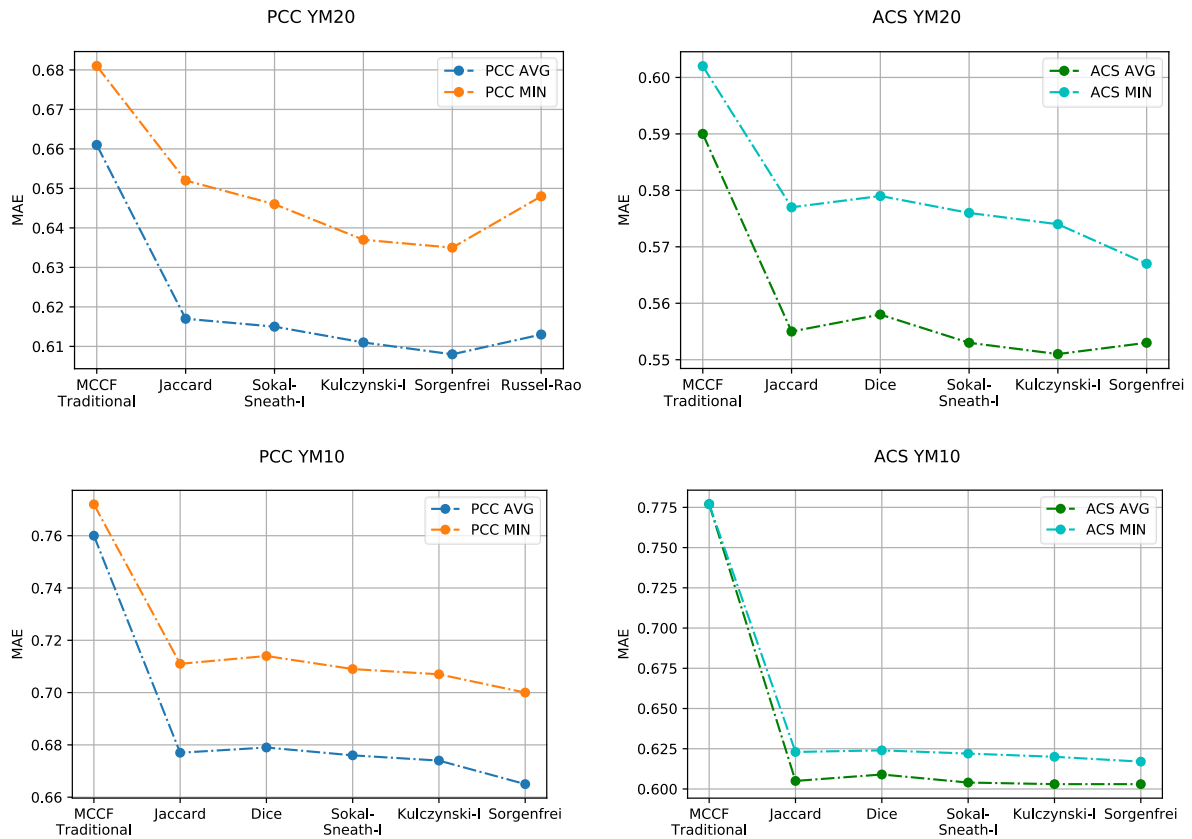


Figure 2 The best performing binary similarity measures against Traditional MCCF in item-based MCF

We analyzed how binary similarity measures affect the MAE results when item-based MCF is utilized. The experiment shows that weighting item-item similarities based on their binary vector representations can improve prediction accuracy in MCF. Based on the experimental findings, binary vector similarities that ignore d are relatively more successful than the measures that consider d . This phenomenon might occur due to the fact that mutual presences (occurrences of 1s) are more important than mutual absences (occurrences of 0s) in item-item similarity calculations. Because the data sets in CF are usually sparse; therefore, mutual absences do not reveal much. Besides, although dissimilarity measures usually contribute to the prediction accuracy in terms of MAE, Shape-Difference always falls behind the traditional MCF.

5.4.2 Effects of binary vector similarity on user-based MCF

This experiment analyzes the effect of binary vector similarities on the user-based MCF. Unlike the previous example, the binary similarity of two user vectors is multiplied by the user-user similarity. Table 6 displays the corresponding results. Recall that bold cells are better than the traditional MCF score.

When user-based MCF is applied on YM20 data set with binary vector similarity weighting, MAE scores are 0.552, 0.592, 0.497, and 0.623, for PCC-AVG, PCC-MIN, ACS-AVG, and ACS-MIN, respectively. Binary vector similarities ignoring d comparably achieve better results than the traditional MCF. In terms of measures considering d , they generally fall behind the traditional MCF with PCC-AVG and ACS-MIN; on the contrary, they are slightly better when item-item similarity calculation is based on PCC-MIN and ACS-AVG. The last group of binary vector similarities utilizes distance

functions, they perform similar to the group of measures that considers d . However, Shape-Difference, once more, is dramatically lower than the traditional MCCF.

When user-based MCCF is applied to YM10 data set which is sparser than YM20, weighting user-based similarities by binary vectors similarities, in general, does not introduce an improvement for PCC-AVG, PCC-MIN, and ACS-AVG. The only setting that an increase in MAE is obvious occurs when ACS-MIN is applied to weight user-user similarities. In this setting, the measures that consider d and dissimilarity functions (except Shape-Difference) record some improvement.

Table 6 MAE Results When Binary Vector Similarities Are Applied in User-based MCCF

	YM20				YM10			
	PCC AVG	PCC MIN	ACS AVG	ACS MIN	PCC AVG	PCC MIN	ACS AVG	ACS MIN
MCCF Traditional	0.552	0.592	0.497	0.623	0.584	0.633	0.407	0.513
Jaccard	0.548	0.585	0.495	0.596	0.606	0.645	0.528	0.538
Dice	0.548	0.588	0.496	0.602	0.606	0.646	0.526	0.536
3W-Jaccard	0.548	0.587	0.496	0.602	0.605	0.646	0.524	0.535
Sokal-Sneath-I	0.548	0.584	0.494	0.597	0.605	0.644	0.528	0.539
Simpson	0.551	0.586	0.501	0.593	0.609	0.640	0.522	0.532
Braun	0.549	0.584	0.495	0.605	0.612	0.649	0.532	0.537
Johnson	0.549	0.591	0.498	0.593	0.599	0.640	0.519	0.529
Kulczynski-I	0.546	0.579	0.493	0.598	0.605	0.643	0.529	0.538
Sorgenfrei	0.550	0.581	0.500	0.556	0.615	0.648	0.542	0.557
Ochiai	0.547	0.587	0.497	0.601	0.603	0.647	0.520	0.535
Russel-Rao	0.558	0.589	0.500	0.578	0.621	0.649	0.539	0.549
Sokal-Michener	0.552	0.593	0.495	0.634	0.584	0.636	0.427	0.488
Sokal-Sneath-2	0.552	0.590	0.496	0.629	0.583	0.634	0.425	0.486
Roger-Tanimoto	0.554	0.593	0.496	0.635	0.587	0.632	0.426	0.489
Gower-Legendre	0.552	0.590	0.496	0.629	0.583	0.634	0.425	0.486
Faith	0.551	0.590	0.493	0.626	0.584	0.634	0.418	0.485
Squared-Euclid	0.554	0.589	0.496	0.636	0.600	0.635	0.461	0.495
Manhattan	0.554	0.589	0.496	0.636	0.600	0.635	0.461	0.495
Mean- Manhattan	0.552	0.593	0.495	0.634	0.584	0.636	0.427	0.488
Size-Difference	0.551	0.589	0.496	0.629	0.583	0.633	0.431	0.485
Shape-Difference	0.881	0.694	0.768	0.891	0.969	0.938	0.861	0.917

Unlike the previous experiment where item-based MCCF is tested, the effect of introducing binary vector similarity to weight user-user similarities is relatively limited. These two experiments show that if the MCCF algorithm utilizes an item-based approach, binary vector similarities, especially the ones ignoring d can be integrated into the method for improved prediction accuracy. However, one should be informed about the fact that weighting might not help improve the prediction accuracy when user-based MCCF is applied if the data set is sparse like YM10.

Since Table 6 includes many experimental results and they are very close to each other five binary similarity measures are illustrated in Figure 3.

6. Discussion

MCCF approaches utilizing neighborhood methods need to calculate either user-user or item-item similarities. If the scheme is user-based, then the similarity calculation is performed on the co-rated

items of associated users. Otherwise, in item-based schemes, co-rating users of associated items are utilized in the similarity calculation. Similarities, on the other hand, are calculated based on co-ratings, regardless of how many there are. The number of co-ratings, whether a few or many, does not prevent the similarity calculation. Similarities calculated with very few co-ratings may be misleading as they are not built on an overwhelming mutual presence of ratings. Therefore, the lack of enough co-ratings may affect the neighbor selection and prediction performance negatively. Similarity weighting is utilized to alleviate such concerns. The general idea is to associate the similarity calculation with mutual presences and absences or non-mutual presences and absences of corresponding vectors. Such relations between different user or item vectors can be constructed by leveraging binary vector representations and similarities between binary vector representations can be employed as a weighting factor in neighbor selection while calculating similarities.

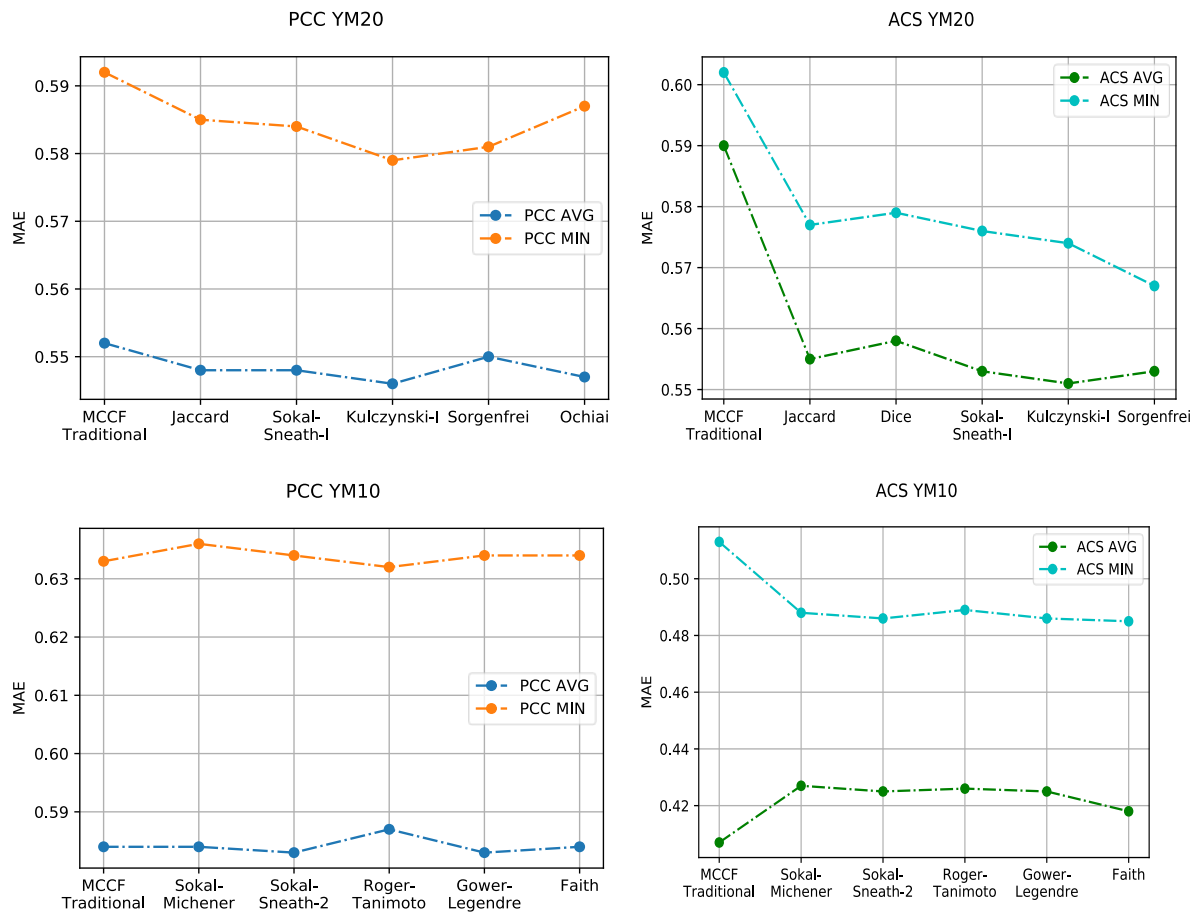


Figure 3 Successful binary similarity measures against Traditional MCCF in user-based MCCF

We, in this study, discuss item- and user-based MCCF schemes and the literature of binary vector similarity in detail. Then, we design and conduct extensive experiments to evaluate the effects of similarity weighting on the prediction accuracy of both item- and user-based multi-criteria collaborative filtering when item-item and user-user correlations are measured by PCC and ACS similarity, which are widely in use. The binary vector similarities measures are grouped under similarity- and distance-based methods. Those measures are further categorized as the ones considering mutual absences or not. Because collaborative filtering data is usually very sparse, the assumption is that mutual presence becomes more important than mutual absences. Experimental findings confirm the assumption: similarities weighted by binary vector similarity measures that ignore mutual absences achieve comparably better results (in terms of mean absolute error) than measures considering mutual absences as well as the traditional item-based MCCF where no weighting is applied. Among the binary vector similarity measures, Sorgenfrei, Kulczynski-I, and Sokal-Sneath-I are worth mentioning because they are generally the top three measures contributing the highest improvement percentages. The highest

improvement is achieved by Sorgenfrei and Kulczynski-I weightings by 22.4% for YM10 data set. Unlike item-based MCCF, in the user-based setting, the effect of binary vector similarity weighting on the prediction performance remains relatively limited. Therefore, one should consider the fact that similarity weighting may not improve the prediction accuracy in MCCF when user-based schemes are employed. Beyond the improvements achieved, one should be aware of the fact that employing Shape-Difference distance metric as a weighting factor never contributes to the prediction accuracy.

7. Conclusions

In this study, we perform a detailed analysis of the effects of binary vector similarities on the prediction performance of the multi-criteria collaborative filtering. Twenty-one different binary vector similarity measures have been discussed and they are used as weighting factor to scale item- and user-based correlations. Experimental findings suggest that measures not considering mutual absences contribute to the prediction accuracy. As a future work, we plan to scrutinize different aspects of binary multi-criteria collaborative filtering schemes in which the number of studies is limited.

Acknowledgments

The author has no conflict of interest to declare.

References

- [1] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," *Commun. ACM*, vol. 35, no. 12, pp. 61–70, Dec. 1992, doi: 10.1145/138859.138867.
- [2] Z. Zhang, T. Peng, and K. Shen, "Overview of collaborative filtering recommendation algorithms," *IOP Conf. Ser. Earth Environ. Sci.*, vol. 440, 2020.
- [3] S. Chen and Y. Peng, "Matrix factorization for recommendation with explicit and implicit feedback," *Knowledge-Based Systems*, vol. 158, pp. 109–117, Oct. 2018, doi: 10.1016/j.knosys.2018.05.040.
- [4] B. Demirelli Okkalioglu, "Improving Prediction Performance of Dynamic Neighbor Selection in User-Based Collaborative Filtering," *Sakarya University Journal of Computer and Information Sciences*, vol. 3, no. 2, pp. 73–87, Aug. 2020, doi: 10.35377/saucis.03.02.714969.
- [5] P. K. Singh, P. K. D. Pramanik, A. K. Dey, and P. Choudhury, "Recommender systems: an overview, research trends, and future directions," *Int. J. Bus. Syst. Res.*, vol. 15, no. 1, 2021.
- [6] C. C. Aggarwal, *Recommender Systems: The Textbook*, 1st ed. 2016. Cham: Springer International Publishing: Imprint: Springer, 2016.
- [7] G. Adomavicius and Y. Kwon, "New Recommendation Techniques for Multicriteria Rating Systems," *IEEE Intell. Syst.*, vol. 22, no. 3, pp. 48–55, May 2007, doi: 10.1109/MIS.2007.58.
- [8] J. Herlocker, J. A. Konstan, and J. Riedl, "An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms," *Information Retrieval*, vol. 5, no. 4, pp. 287–310, Oct. 2002, doi: 10.1023/A:1020443909834.
- [9] H. Ma, I. King, and M. R. Lyu, "Effective missing data prediction for collaborative filtering," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07*, Amsterdam, The Netherlands, 2007, doi: 10.1145/1277741.1277751.
- [10] N. Polatidis and C. K. Georgiadis, "A multi-level collaborative filtering method that improves recommendations," *Expert Systems with Applications*, vol. 48, pp. 100–110, Apr. 2016, doi: 10.1016/j.eswa.2015.11.023.
- [11] N. Polatidis and C. K. Georgiadis, "A dynamic multi-level collaborative filtering method for improved recommendations," *Computer Standards & Interfaces*, vol. 51, pp. 14–21, Mar. 2017, doi: 10.1016/j.csi.2016.10.014.
- [12] L. Candillier, F. Meyer, and F. Fessant, "Designing Specific Weighted Similarity Measures to

- Improve Collaborative Filtering Systems,” in *Advances in Data Mining. Medical Applications, E-Commerce, Marketing, and Theoretical Aspects*, 2008, pp. 242–255.
- [13] E. Sadikoğlu and B. D. Okkaloğlu, “Increasing Prediction Performance Using Weighting Methods in Multi-Criteria Item-Based Collaborative Filtering,” *European Journal of Science and Technology*, pp. 110–121, Aug. 2020, doi: 10.31590/ejosat.779171.
- [14] E. Yalçın and A. Bilge, “Binary multicriteria collaborative filtering,” *Turk J Elec Eng & Comp Sci*, vol. 28, no. 6, pp. 3419–3437, Nov. 2020.
- [15] M. Plantié, J. Montmain, and G. Dray, “Movies Recommenders Systems: Automation of the Information and Evaluation Phases in a Multi-criteria Decision-Making Process,” in *Database and Expert Systems Applications*, 2005, pp. 633–644.
- [16] A. Bilge and C. Kaleli, “A multi-criteria item-based collaborative filtering framework,” in *2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Chon Buri, May 2014, pp. 18–22. doi: 10.1109/JCSSE.2014.6841835.
- [17] Q. Shambour and J. Lu, “A Hybrid Multi-criteria Semantic-Enhanced Collaborative Filtering Approach for Personalized Recommendations,” *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, 2011, pp. 71–78, doi: 10.1109/WI-IAT.2011.109.
- [18] Q. Shambour, M. Hourani, and S. Fraihat, “An Item-based Multi-Criteria Collaborative Filtering Algorithm for Personalized Recommender Systems,” *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 8, 2016, doi: 10.14569/IJACSA.2016.070837.
- [19] Q. Shambour, “A user-based multi-criteria recommendation approach for personalized recommendations,” *International Journal of Computer Science and Information Security*, vol. 14, no. 12, 2016.
- [20] A. Gazdar and L. Hidri, “A new similarity measure for collaborative filtering based recommender systems,” *Knowl. Based Syst.*, vol. 188, no. 105058, 2020.
- [21] Y. Wang, P. Wang, Z. Liu, and L. Y. Zhang, “A new item similarity based on α -divergence for collaborative filtering in sparse data,” *Expert Syst. Appl.*, vol. 166, no. 114074, 2021.
- [22] J. Wei, J. He, K. Chen, Y. Zhou, and Z. Tang, “Collaborative filtering and deep learning based recommendation system for cold start items,” *Expert Systems with Applications*, vol. 69, pp. 29–39, Mar. 2017, doi: 10.1016/j.eswa.2016.09.040.
- [23] H.-J. Kwon, T.-H. Lee, J.-H. Kim, and K.-S. Hong, “Improving Prediction Accuracy Using Entropy Weighting in Collaborative Filtering,” in *2009 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing*, Jul. 2009, pp. 40–45. doi: 10.1109/UIC-ATC.2009.50.
- [24] M. Ghazanfar and A. Prugel-Bennett, “Novel Significance Weighting Schemes for Collaborative Filtering: Generating Improved Recommendations in Sparse Environments,” presented at the *DMIN’10, the 2010 International Conference on Data Mining!*, Jul. 12, 2010. Accessed: Jun. 02, 2021. [Online]. Available: <https://eprints.soton.ac.uk/270846/>
- [25] J. Han and M. Kamber, *Data mining*. Burlington, MA: Elsevier, 2012.
- [26] S. Choi and S. Cha, “A survey of Binary similarity and distance measures,” *Journal of Systemics, Cybernetics and Informatics*, pp. 43–48, 2010.
- [27] J. Podani, “Distance, similarity, correlation...”, in *Introduction to the exploration of multivariate biological data*. Leiden: Backhuys Publishers, 2000.
- [28] S.-H. Cha, C. Tappert, and S. Yoon, “Enhancing binary feature vector similarity measures,” *J. Pattern Recognit. Res.*, vol. 1, no. 1, pp. 63–77, 2006. doi: 10.13176/11.20.
- [29] D. Jannach, Z. Karakaya, and F. Gedikli, “Accuracy improvements for multi-criteria recommender systems,” in *Proceedings of the 13th ACM Conference on Electronic Commerce - EC ’12*, Valencia, Spain, 2012, doi: 10.1145/2229012.2229065.

Twitter Sentiment Analysis Based on Daily Covid-19 Table in Turkey

 Buket Kaya¹,  Abdullah Günay²

¹Corresponding Author; Firat University, Department of Electronics and Automation; bkaya@firat.edu.tr;
²Siirt University, Department of Call Center Services; abdullah.gunay@siirt.edu.tr;

Received 4 May 2021; Revised 26 October 2021; Accepted 3 November 2021; Published online 31 December 2021

Abstract

The coronavirus pandemic, which began to affect the whole world in early 2020, has become the most talked about agenda item by individuals. Individuals announce their feelings and thoughts through various communication channels and receive news from what is happening around them. One of the most important channels of communication is Twitter. Individuals express their feelings and thoughts by interacting with the tweets posted. This study aims to analyze the emotions of the comments made under the "daily coronavirus table" shared by the Republic of Turkey Ministry of Health and to measure their relationship with the daily number of cases and deaths. In the study, emotional classification of tweets was implemented using LSTM, GRU and BERT methods from deep learning algorithms. The results of all three algorithms were compared with the daily number of cases and deaths.

Keywords: Sentiment Analysis, Twitter, Deep Learning, NLP, LSTM, BERT, GRU, Covid19

1. Introduction

The rapid development of technology and related internet has created changes in people's social life. The fact that people want to share their activities, ideas and likes has made social media a popular platform and also an essential source of information. Today, while more than 4.5 billion people can surf the internet, almost 85% of them (3.8 billion) actively use social media. Twitter is still among the most effective social platforms, although its usage has dropped rapidly. As of 2020, the number of Twitter users is around 340 million. According to the Digital 2020 Global Outlook Report, Turkey ranks 6th globally with 11.8 million active Twitter users and 2nd in Europe [1].

With the Covid-19 virus outbreak that emerged in early 2020, people and governments found themselves in a crisis for which they were unprepared. With the pandemic, people began to physically stay away from social spaces. This situation has caused a change in individuals' behavior and daily habits. According to the study conducted by Ipsos [2], it has been revealed that individuals who follow the popular phrase 'Stay at Home' rule of the pandemic have acquired habits such as spending time in the kitchen, acquiring a habit of doing sports at home, taking advantage of online courses and trainings, and conducting video interviews. Later, these activities were shared on social media platforms. Examining the comments, sharing and discussions on social media platforms in order to observe the effects of the Covid-19 pandemic on the emotional world of people constitute an important resource for text mining. In its simplest definition, text mining is a type of data mining that sees what is written as a data source and analyzes this data. According to current data, more than 8,000 Tweets are shared on Twitter every second [3]. While writing the tweets, the words that were written inaccurately and without obeying the spelling rules, not being used in daily life and concepts specific to the social media environment have developed a unique language on Twitter. It is difficult to understand and analyze these tweets with human perception; Therefore, these data must be filtered, parsed and processed with natural language processing methods. There are many studies on the processing of data obtained from Twitter in the data mining sector by subjecting it to natural language processing One of them aims to predict pandemics with tweets posted between May and December 2009 [4]. Another study analyzed the side effects of drugs from tweets posted between the same year and months [5]. With 85% prediction success, social media dynamics changed human perception over time [6]. There is a study that analyzes perception over millions of tweets that tourists have sent for a touristic destination [7]. In another article, positive / negative classification was made with machine learning techniques over the comments shared on the

IMDB movie scaling platform [8]. Finally, there is a study on the analysis of emotions and thoughts in online environments [9].

Text mining and sentiment analysis aims to reveal the opinions, feelings and thoughts hidden in the comments and opinions shared by people on social media. Sentiment analysis has recently emerged as an active research analyzing people's opinions, feelings, and evaluations [10]. Working areas for sentiment analysis are Twitter and other social networks, blogs, forums and comments. Sentiment analysis studies conducted so far can be generally examined in two groups as dictionary and statistical (machine learning). The dictionary-based approach makes use of pre-prepared dictionaries of emotion concepts during classification. In the statistical / machine learning-based approach, machine learning algorithms and linguistic features are used during classification [11].

The purpose of this study is to automatically determine the emotionally positive / negative / neutral status of tweets posted by users on Twitter during the Covid-19 pandemic process. This study will be useful and important for decision makers by analyzing the mood of the society in the situation of social disasters and pandemics.

2. Related Work

Sabuncu and Atmış [12] recorded English tweets about Turkish Airlines with the R programming language and classified and interpreted the tweets using emotion analysis algorithm. Ataman and Ozguner [13] analyzed the tweets related to the Black Friday week in 2018 and analyzed the emotions contained in the tweets using the Python language and the Cognitus API. Kilimci [14] aimed to predict the direction of the Bist100 index with financial sentiment analysis. For this purpose, Kilimci has developed deep community models. In addition, in the system he developed, he achieved a high classification success of around 78% in Turkish and English data sets. In his study, [15] analyzed the positive, neutral and negative state of the shares after the social media platform Twitter subjected the shares of people to various pre-processes [11], on the other hand, compared the classification success using the multi-layer sensor (MLP), Naive Bayes (NB), Support Vector Machines (DVM) and Logistic Regression (LR) algorithms on user comments on an online book sales site. Chandra et al. [16] tried to find out whether tweets on Twitter are Islamophobic using sentiment analysis. The paper presented the CoronaBias dataset focusing on anti-Muslim hate, with more than 410,990 tweets from 244,229 users. This data set was used to perform longitudinal analysis. The study measured qualitative changes in the context associated with the Muslim community and performed macro and micro subject analysis to find common themes. Jianqiang et al. [17] presented a method of word embedding on large tweets based on unsupervised learning. The method uses semantic relationships between words in tweets and statistical features that occur together. Concerning the sentiment analysis, recently, Mungen et al. conducted a study on finding relationships between news and people's emotions in the Covid-19 pandemic [18].

Samuel et al. [19], using Tweets and R statistics software specific to coronavirus, determined the public sensitivity associated with the pandemic by emotion analysis method. The study achieved a 91% success rate for short tweets using the naive bayes-based classification method. In shorter tweets, it was observed that the logistic regression-based method remained at a success rate of 74%. On the other hand, it was determined that both methods performed relatively poorly in longer tweets. Chen et al. [20] used BERT to integrate data from emotion dictionaries. They showed that their approach consistently outperformed the state-of-the-art methods across all datasets. Lu et al. [21] first process the data they obtain through the BERT model. They then simulated linguistic functions in the sentence by combining Chinese grammar rules in the form of constraints with the Bi-gated recurrent neural network (GRU) and standardizing the output of adjacent positions. Wang et al. [22] analyzed 999,978 Weibo posts related to randomly selected COVID-19 from January 1, 2020 to February 18, 2020 in Sina Weibo, a popular Chinese social media. While the BERT model is used to classify sentiment categories, the TF-IDF model is more preferred to summarize the topics of the posts. Li et al. [23] presented a new method called GBCN, which uses a gate mechanism with context-sensitive direction placements to enhance and control BERT representation for perspective-based sentiment analysis. In the model created, SentiHood and SemEval-2014 data sets reached 88.0 and 92.9 F1 test results, respectively.

3. Data Description

3.1. Data Collection

Within the scope of the study, the emotions of individuals related to the COVID-19 pandemic, which is effective all over the world, were tried to be determined. It is planned to receive data from Twitter, a social media platform with a high rate of use in the world and in Turkey. However, due to the fact that there are a lot of tweets about COVID-19 and it is difficult to determine whether the tweets are related to the cases and deaths in Turkey or the world in general, a certain restriction was needed. At the same time, the fact that there are certain restrictions in terms of extracting data from Twitter required the narrowing of the area where the data will be collected. In this context, the comments made under the daily coronavirus tables announced by the Minister of Health of the Republic of Turkey on their official Twitter account were analyzed. Tweets about Turkey's daily coronavirus table announced by the Minister of Health are quoted, answered and retweeted by twitter users. While Figure 1 shows Turkey Daily Coronavirus Table, interaction of tweets explaining the Daily Coronavirus Table are given in Figure 2.



Figure 1 Turkey Daily Coronavirus Table

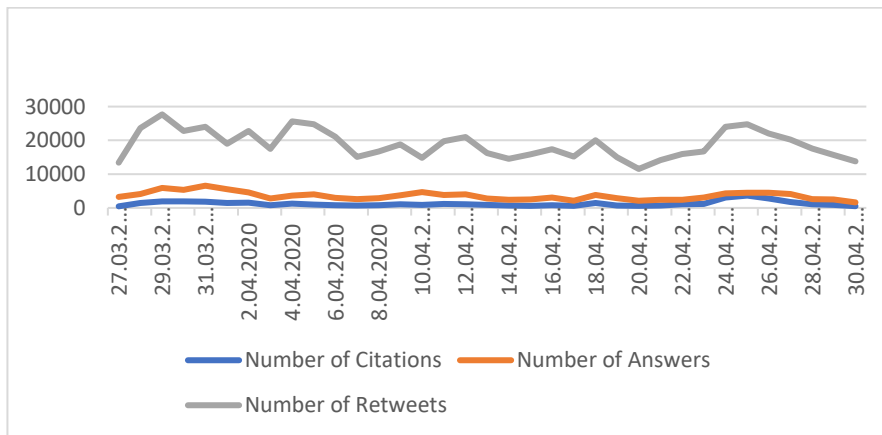


Figure 2 Interaction of Tweets Explaining the Daily Coronavirus Table

A total of 16240 tweets that were commented on the daily tables from March 27, 2020, when the coronavirus table was first announced in Turkey to April 30, 2020 using Twitter Premium API, were obtained through the Python program.

3.2. Data Pre-Processing

The data obtained from Twitter were subjected to various pre-processing using the Knime platform. Various filters are needed to process the words in the texts. In this context, repetitive expressions that are expressed as special characters, numbers and stop words in the texts and do not have any meaning on their own were found and filtered and all words were converted to lowercase letters. Then, using the Zemberek Library, which was prepared to process Turkish texts, word roots were found and separated into their elements. In addition, the usage frequencies of the words grouped in the "Bag of Words Creator" stage are calculated. The steps for preprocessing the data are given in Figure 3.

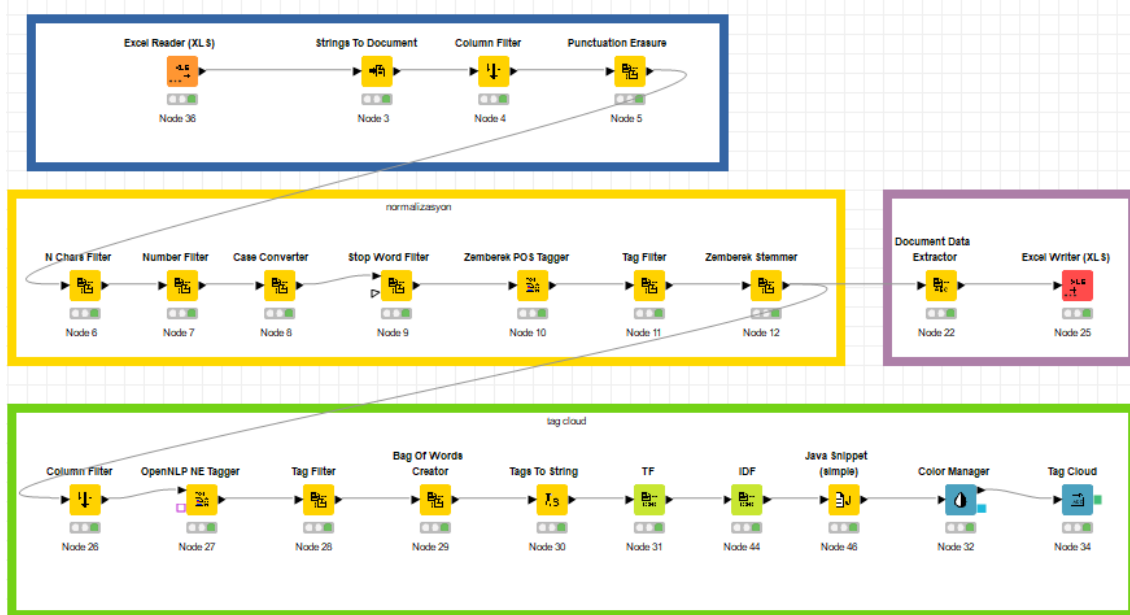


Figure 3 Pre-Processing of Data

The TF-IDF method is an approach that finds the relative frequency of words in a given document with the inverse ratio of words on the entire document corpus. The method uses two elements to determine the value: the i term frequency in the TF - j document and the reverse document frequency of the IDF - i term. TF-IDF can be calculated as follows [24]:

$$a_{ij} = tf_{ij}idf_i = tf_{ij} * \log_2\left(\frac{N}{df_i}\right) \quad (1)$$

In the formula, a_{ij} represents the weight of term i in document j . While tf_{ij} shows the frequency of the term i in document j , df_i gives to the document frequency of the term i in the collection. N is the number of documents in the collection.

As seen in Figure 4, using Zemberek-NLP, a natural language processing tool developed for Turkish, the word types in the data set were determined and the usage frequencies of the words were calculated. Figure 5 denotes word cloud derived from this data.

language. Many NLP implementations are based on language models that define probability distribution over words, characters or byte sequences in a natural language [25].

LSTM is a special type of RNN developed to address the vanishing / exploding problem encountered in RNNs (Figure 6). Similar to other RNNs, the output of the model is formed according to the input from the current time step and the output from the previous time step, and the current output is sent to the next time step. A memory cell (c_t) that stores the LSTM units state in random time intervals consists of three non-linear gates. These are an input gate (i_t), a forget gate (f_t), and output gate (o_t). The purpose of these gates is to regulate the flow of information entering and leaving the memory cell [26].

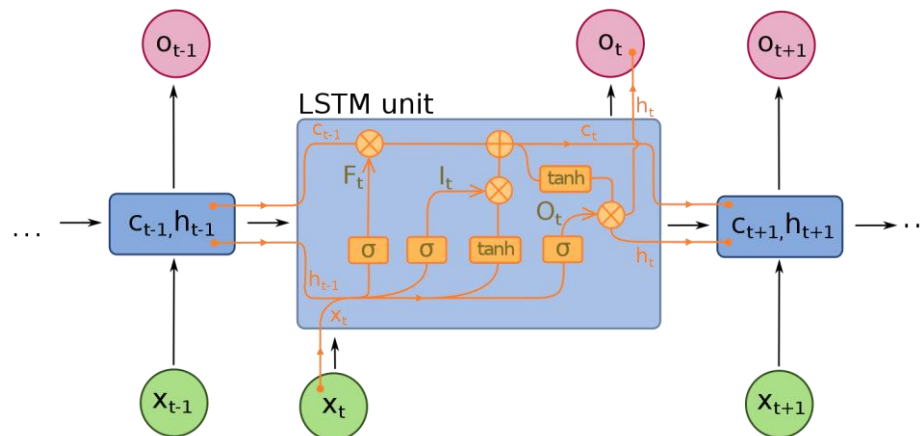


Figure 6 LSTM Model

GRU is an LSTM featured model that optimizes the LSTM network structure. When compared to the LSTM network structure, it is seen that the GRU network has two gateways, namely the update gate and the reset gate. This is because GRU is able to handle the long range long delay series prediction problem. While the reset gate determines the degree to which the information of the previous moment is ignored, the purpose of the update gate is to find out how close the information of the previous moment is to the current moment. A GRU model is shown in Figure 3. GRU has been noted to outperform LSTM units in terms of both CPU time convergence and parameter updates and generalization [23].

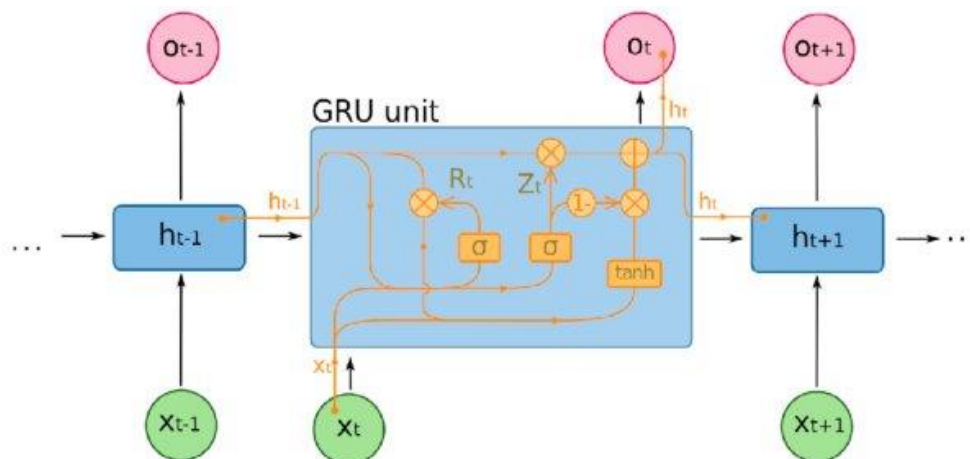


Figure 7 GRU Model.

There are many NLP techniques that automatically analyze human language and make it possible to represent it. The BERT algorithm, which consists of the initials of "Bidirectional Encoder Representations from Transformers", uses machine learning and artificial intelligence together.

The BERT given its architecture in Figure 8 is the most preferred natural language processing technique recently.

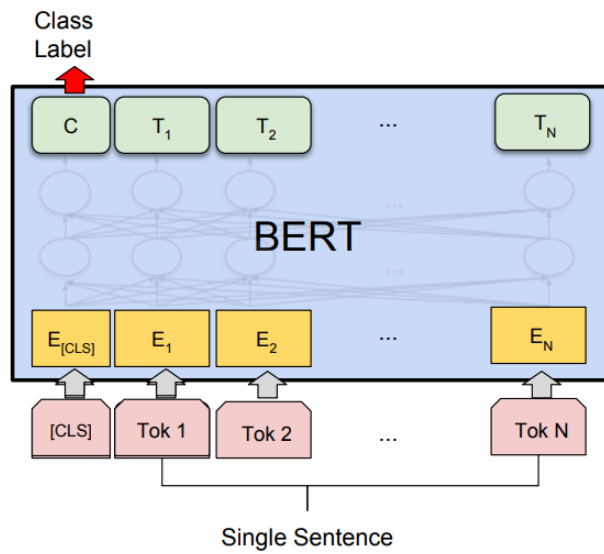


Figure 8 BERT Encoder

In order to analyze the data using the BERT algorithm, Python codes using Anaconda's Jupyter platform and open source deep learning libraries such as Keras neural network library and Tensorflow were used. Before analyzing the data, training and test data were created for the training of the machine, and this training data was tested. The results of the tests are shown in Figure 9.

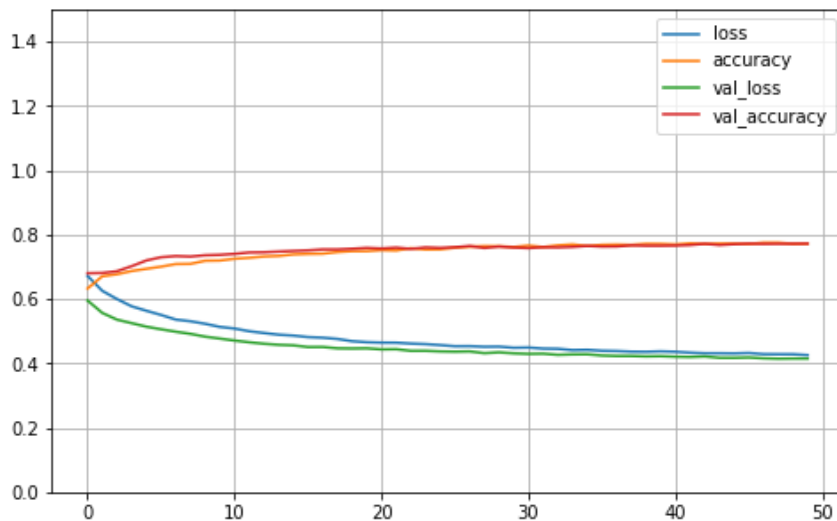


Figure 9. Test Results

Table 1 shows the results of three different architectures according to the performance metrics whose equations are given below.

$$Precision = \frac{TP}{TP+FP} \tag{1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (3)$$

$$F_1 \text{ Score} = 2 * \frac{Precision*Recall}{Precision+Recall} \quad (4)$$

Table 1 Performance Values of Three Different Architectures

Performance Metrics	GRU	LSTM	BERT
Accuracy	0.78	0.77	0.78
Precision	0.79	0.78	0.77
Recall	0.93	0.92	0.76
F1 Score	0.85	0.84	0.77
AUC	0.88	0.88	0.88

While creating the model, the embedding layer was added before the GRU layers and word vectors were used as input. The resources in the LSTM method were used to train the model and generate the word vectors. In this study, sentiment score is the ratio of positive comments to total comments. Since short comments are mostly classified as neutral, the number of neutral comments is high. Short reviews are labeled neutral, as in the review phase, as they are not evaluated at all or are labeled neutral. Also, in LSTM/GRU experiments, the result of the prediction is produced between 0 and 1, thanks to a sigmoid. It is understood that the result is positive as these values approach 1, and negative as they approach 0. As can be seen in Figures 10 and 11, the rate of negative comments exceeds positive on some days.

The daily average of sentiments in Figure 10 and Figure 11 and the number of daily cases and deaths were compared. In comparison, it is observed that the sentiment value increases, that is, more positive comments are made on the days when the number of cases decreases. However, when compared with the number of deaths, the relationship between the emotional value and the daily number of deaths was not determined. This is really an expected result because the death is a later consequence of positive cases.

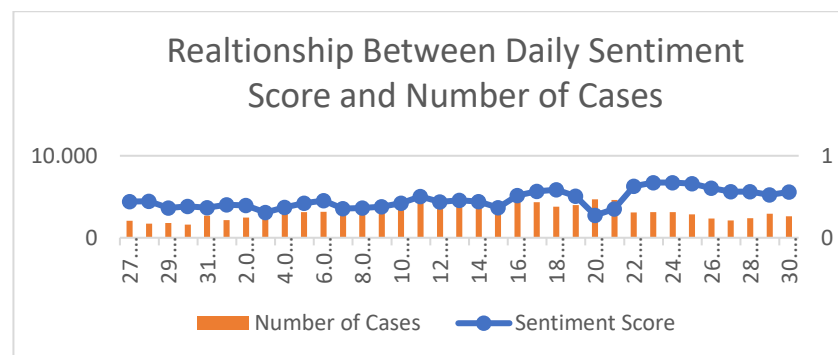


Figure 10 Relationship Between Daily Sentiment Score and Number of Cases

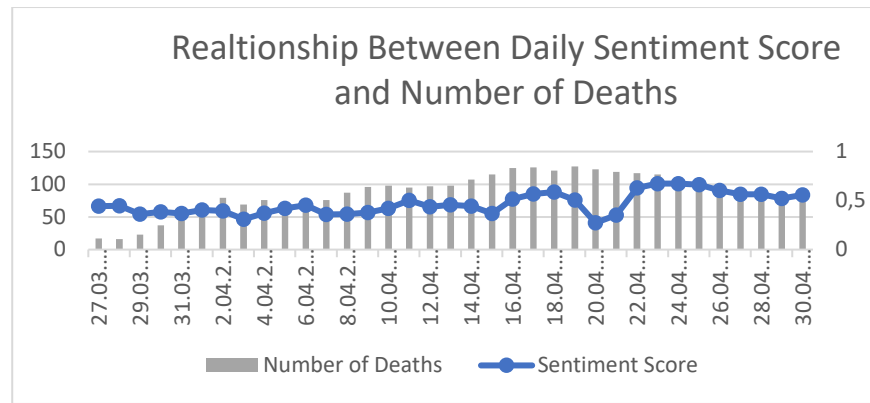


Figure 11 Relationship Between Daily Sentiment Score and Number of Deaths

5. Conclusions



Individuals who share their feelings and thoughts on social media channels have always been a great source of data for researchers observing society and individuals. In this paper, it is aimed to determine the feelings of individuals on Twitter about the number of positive cases and deaths related to the coronavirus pandemic, which started to affect the whole world at the beginning of 2020. As a dataset, only the comments made under the daily coronavirus tables announced by the Ministry of Health of the Republic of Turkey on its official Twitter account were used. The expressions in the obtained data set were subjected to various text preprocessing techniques and the emotional value of each tweet was calculated. Then, it has been tried to determine whether the values calculated using LSTM, GRU and BERT algorithms are related to the daily number of cases and deaths. The experimental results showed that there is a relationship between the number of positive cases per day and the sentiment score. On the other hand, since death is a later result of the positive case, it does not have an instant relationship with the sentiment score.

References

- [1] S. Kemp, "Digital 2020: 3.8 Billion People Use Social Media.", <https://wearesocial.com/blog/2020/01/digital-2020-3-8-billion-people-use-social-media>, 2020.
- [2] Ipsos, "Covid-19 Dönemi ve Evdeki Keşifler", https://www.ipsos.com/sites/default/files/ipsossia_trends_6nisan2020.pdf, Accessed August 20, 2020.
- [3] D. Murthy, *Twitter: Social Communication in the Twitter Age*. Cambridge, UK: Polity Press, 2018.
- [4] M. Szomszor, P. Kostkova and E. De Quincey, "# Swineflu: Twitter predicts swine flu outbreak in 2009", In *International Conference on Electronic Healthcare* (pp. 18-26). Springer, Berlin, Heidelberg, 2010.
- [5] J. Bian, U. Topaloglu and F. Yu, "Towards large-scale twitter mining for drug-related adverse events", In *Proceedings of the 2012 international workshop on Smart health and wellbeing* (pp. 25-32), 2012.
- [6] L. T. Nguyen, P. Wu, W. Chan, W. Peng and Y. Zhang, "Predicting collective sentiment dynamics from time-series social media", In *Proceedings of the first international workshop on issues of sentiment discovery and opinion mining* (pp. 1-8), 2012.
- [7] W.B. Claster, H. Dinh and M. Cooper, "Naïve Bayes and unsupervised artificial neural nets for Cancun tourism social media data analysis", In *2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC)* (pp. 158-163). IEEE, 2010.
- [8] B. Pang, L. Lee and S. Vaithyanathan, "Thumbs up? Sentiment classification using machine

- learning techniques", *arXiv preprint cs/0205070*, 2002.
- [9] R.M. Tong, "An operational system for detecting and tracking opinions in on-line discussion", In *Working Notes of the ACM SIGIR 2001 Workshop on Operational Text Classification* (Vol. 1, No. 6), 2001.
- [10] K. Ravi, V. Ravi. "A survey on opinion mining and sentiment analysis: tasks, approaches and applications", *Knowledge-based systems*, 89, 14-46, 2015.
- [11] M.L.B. Estrada, R.Z. Cabada, R.O. Bustillos, M. Graff. "Opinion mining and emotion recognition applied to learning environments", *Expert Systems with Applications*, 150, 113265, 2020.
- [12] İ. Sabuncu and M. Atmis, "Social Media Analytics for Brand Image Tracking: A Case Study Application for Turkish Airlines", *Yönetim Bilişim Sistemleri Dergisi*, 6(1), 26-41, 2020.
- [13] F. Ataman, Ö. Özgüner. "Determination of Social Media Users' Perceptions About "Black Friday" Activity via Sentiment Analysis", *Educational Research (IJM CER)*, 3(3), 361-371, 2021.
- [14] Z.H. Kilimci, "Financial sentiment analysis with Deep Ensemble Models (DEMs) for stock market prediction", *Journal of the Faculty of Engineering and Architecture of Gazi University*, 35(2), 635-650, 2020.
- [15] H.K. Küçükkartal, "Twitter'daki Verilere Metin Madenciliği Yöntemlerinin Uygulanması", *Eskişehir Türk Dünyası Uygulama ve Araştırma Merkezi Bilişim Dergisi*, 1(2), 10-13, 2020.
- [16] M. Chandra, M. Reddy, S. Sehgal, S. Gupta, A.B. Buduru, P. Kumaraguru. "A Virus Has No Religion: Analyzing Islamophobia on Twitter During the COVID-19 Outbreak", In *Proceedings of the 32nd ACM Conference on Hypertext and Social Media* (pp. 67-77), 2021.
- [17] Z. Jianqiang, G. Xiaolin, Z. Xuejun. "Deep convolution neural networks for twitter sentiment analysis", *IEEE Access*, 6, 23253-23260, 2018.
- [18] A.A. Müngen, İ. Aygün and M. Kaya, "Finding the Relationship Between News and Social Media Users' Emotions in the COVID-19 Process", *Sakarya University Journal of Computer and Information Sciences*, 3(3), 250-263, 2020.
- [19] J. Samuel, G.G. Ali, M. Rahman, E. Esawi and Y. Samuel, "Covid-19 public sentiment insights and machine learning for tweets classification", *Information*, 11(6), 314, 2020.
- [20] F. Chen, Z. Yuan and Y. Huang, "Multi-source data fusion for aspect-level sentiment classification", *Knowledge-Based Systems*, 187, 104831, 2020.
- [21] Q. Lu, Z. Zhu, F. Xu, D. Zhang, W. Wu and Q. Guo, "Bi-GRU Sentiment Classification for Chinese Based on Grammar Rules and BERT", *International Journal of Computational Intelligence Systems*, 13(1), 538-548, 2020.
- [22] T. Wang, K. Lu, K.P. Chow and Q. Zhu, "COVID-19 Sensing: Negative sentiment analysis on social media in China via Bert Model", *IEEE Access*, 8, 138162-138169, 2020.
- [23] W. Li, H. Wu, N. Zhu, Y. Jiang, J. Tan and Y. Guo, "Prediction of dissolved oxygen in a fishery pond based on gated recurrent unit (GRU)", *Information Processing in Agriculture*, 8(1), 185-193, 2021.
- [24] B. Trstenjak, S. Mikac and D. Donko, "KNN with TF-IDF based framework for text categorization", *Procedia Engineering*, 69, 1356-1364, 2014.
- [25] Y. Bengio, I. Goodfellow and A. Courville, *Deep learning*, Massachusetts, USA: MIT Press, 2017.
- [26] M.E. Basiri, S. Nemati, M. Abdar, E. Cambria and U.R. Acharya, "ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis", *Future Generation Computer Systems*, 115, 279-294, 2021.

Automated learning rate search using batch-level cross-validation

 Duygu Kabakcı¹,  Emre Akbaş²

¹Middle East Technical University; duygukabakci@gmail.com

²Corresponding Author; Middle East Technical University; emre@ceng.metu.edu.tr

Received 10 May 2021; Revised 02 November 2021; Accepted 04 November 2021; Published online 31 December 2021

Abstract

Deep learning researchers and practitioners have accumulated a significant amount of experience on training a wide variety of architectures on various datasets. However, given a network architecture and a dataset, obtaining the best model (i.e. the model giving the smallest test set error) while keeping the training time complexity low is still a challenging task. Hyper-parameters of deep neural networks, especially the learning rate and its (decay) schedule, highly affect the network's final performance. The general approach is to search the best learning rate and learning rate decay parameters within a cross-validation framework, a process that usually requires a significant amount of experimentation with extensive time cost. In classical cross-validation (CV), a random part of the dataset is reserved for the evaluation of model performance on unseen data. This technique is usually run multiple times to decide learning rate settings with random validation sets. In this paper, we explore batch-level cross-validation as an alternative to the classical dataset-level, hence macro, CV. The advantage of batch-level or micro CV methods is that the gradient computed during training is re-used to evaluate several different learning rates. We propose an algorithm based on micro CV and stochastic gradient descent with momentum, which produces a learning rate schedule during training by selecting a learning rate per epoch, automatically. In our algorithm, a random half of the current batch (of examples) is used for training and the other half is used for validating several different step sizes or learning rates. We conducted comprehensive experiments on three datasets (CIFAR10, SVHN and Adience) using three different network architectures (a custom CNN, ResNet and VGG) to compare the performances of our micro-CV algorithm and the widely used stochastic gradient descent with momentum in an early-stopping macro-CV setup. The results show that, our micro-CV algorithm achieves comparable test accuracy to macro-CV with a much lower computational cost.

Keywords: deep learning, neural networks, learning rate, hyper-parameter search, adaptive learning rate, cross-validation

1. Introduction

Training deep neural network models is not a trivial task. It is a delicate and complex process which aims to efficiently obtain a model that generalizes well to unseen test data. An excessive number of hyper-parameters including learning rate, learning rate schedule, mini-batch size, regularization parameter, weight decay constant and the network architecture increase the complexity of the training process as all of these parameters need to be tuned. The problem is exacerbated for new problem domains or new datasets. Guided sequential experiments with a different selection of hyper-parameters usually require prior knowledge on neural network's convergence, loss function topology and dataset to achieve a model that has smallest generalization error. After each experiment, the subsequent selection of hyper-parameters can be determined based on these factors using prior knowledge. In fact, human skill and expertise is a significant factor in the final performance of a deep learning architecture [1]. Even carefully designed sequential experiments conducted with knowledge-based assessments come with a very high amount of computational cost. Given the surprisingly large carbon-footprint of deep learning computations [2], electricity and hardware costs, efforts to shorten the training process become crucial.

Learning rate, i.e. step size, seems to be one of the most important hyper-parameters for deep neural networks that highly affects the model performance. Specifically, the learning rate adjusts the magnitude of the network's weight updates for minimizing the loss function. If the learning rate is too high, the model struggles to converge to a local minima; while a too small learning rate slows down convergence, hence increases the total training time.

Adaptive optimizers (e.g. Rmsprop[3], Adam[4]) have been proposed to address the problem of setting optimal learning rates. It is widely reported that “Adam” with its default parameters achieves high accuracy for many architectures. However, there is also evidence that “researchers kept evolving new architectures on which Adam works” [5]. Recent research ([6, 7]) also points out the convergence problem of Adam optimizer. According to these recent findings, hand-tuned stochastic gradient descent (SGD) with momentum optimizer can achieve better results than adaptive optimizers. Adaptive optimizers’ convergence problem can be solved with additional learning rate tuning. In short, selecting learning rate and its schedule is still an unsolved challenge for which an automated procedure would have significant impact. Our proposed methods (using micro cross-validation) explore some ideas towards automating the search for the optimal learning rate.

The standard way to find an optimal learning rate and/or decay parameter (more generally, a learning rate schedule) is to apply cross-validation (CV) to estimate model performance on unseen data for different learning rates and decay values. In classical cross-validation (CV), a random part of the dataset is reserved for the evaluation of model to estimate its future performance on unseen data. This process is repeated multiple times with random validation sets to determine the best learning rate settings. We name this standard CV method as “macro CV” since it is a dataset-level method. As an alternative, we propose “micro CV” method which works at the mini-batch level. We argue that “micro CV” may be more efficient because once the gradient vector is computed at the batch level, it could be re-used to test out several different learning rates. We propose an automated learning rate selection algorithm that aims to find optimal learning rate and learning rate schedules during training. In each iteration of our algorithm, we sample a mini-batch from the training set just like in all stochastic gradient algorithms. Then, we split this batch into two equal-size sets, one of which is used as the training mini-batch and the other as the validation mini-batch. We compute the gradient of the loss function with respect to network weights using only the “training” mini-batch. Once we have the gradient vector, we apply different step-sizes or learning rates along this vector to obtain different future models. Next, we evaluate these future models on the “validation” mini-batch, and accumulate the corresponding losses. When this process is repeated for the whole training set (i.e. all batches), we identify the step-size which has accumulated the least amount of (validation) loss, and set it as the learning rate for the next epoch. We implemented this method and a variant of it, where we not only accumulate the validation loss but also the training loss, and used them for the image classification task on three different image classification datasets (CIFAR10, SVHN, Adience) with three different CNN architectures. Our experiments show that, in terms of test set classification accuracy, macro-CV slightly outperforms micro-CV, however, in terms of computational cost, micro-CV algorithm is better by a large margin. Overall, our micro-CV algorithm yields promising results.

This paper has been compiled from the first author’s MSc thesis [8]. The rest of the paper is organized as follows. Section 2 discusses the SGD algorithm and provides a comparison with adaptive gradient descent algorithms as background. Section 3 describes our proposed micro CV algorithm. In Section 4, we describe the experiments we conducted for the purpose of comparing micro-CV and macro-CV methods. Finally, Section 5 concludes the paper by providing a brief summary and discussion.

2. Background and Related Work

Many methods have been proposed for automated learning rate selection [3, 4, 7, 9, 10, 11, 12, 13, 14, 15]. The ultimate goal is to obtain the lowest generalization error in a minimum time and memory budget. This section covers traditional hyper-parameter tuning approaches, adaptive learning rate methods, cyclical learning rate schedules, gradient based learning rate tuning methodologies and mini-batch validation based methods.

2.1 Hyper-parameter Tuning and Cross Validation

Automated hyper-parameter tuning can be considered as the ancestor of automated learning rate tuning. The classical way of tuning is to define a set of parameter values and estimate model performance for them. This basic method is known as “grid search” and it can be carried out in a sequential or parallel

manner, for more than one variable or hyper-parameter. The set of parameter values can be set beforehand or sampled randomly as well (i.e., random search [10]). The model performance is evaluated on a held-out set known as the “validation set”. Typically, 10 – 30% of the training set is reserved for this purpose. To reduce variability of results, the training set can be split into many folds and each fold can be used as the validation set while others are used for training. These methods are generally known as “cross-validation” methods. In this paper, we further call these methods as “macro cross-validation” since the train-validation split is done at the dataset level. As an alternative, this split can be carried out at the batch-level as we propose in this paper.

2.2 Adaptive Learning Rate Methods

The earliest heuristic to achieve adaptive learning is the idea of using separate learning rates per parameter based on the sensitivity of the cost function per parameter. AdaGrad [9], an earlier example of adaptive learning rate optimizers, scales learning rate per parameter with the square root of the sum of all historical squared values of the gradient. In this approach, parameters with larger partial derivatives have decreased learning rates compared to parameters with small partial derivatives. Even though this approach helps some models, keeping historical partial derivatives for the whole training session may lead to excessive decrease in the effective learning rate for some parameters. Rmsprop [3] modifies the AdaGrad algorithm to address its excessive decrease on learning rate by using a moving average of gradients to replace the whole-history based average. This introduces a new hyperparameter (which is usually not tuned). Adam optimizer [4] combines Rmsprop’s and momentum optimizer’s benefits. It computes two historical moving average estimates which keep averages of gradients and squared gradients respectively. There is also correction of initial bias of moving average of gradients and square of gradients in Adam that is also an addition to Rmsprop algorithm. Although Adam is capable to work with different models with default values, some cases still requires tuning the global learning rate and other hyper-parameters.

2.2.1 Revisiting SGD with Momentum

Even though adaptive methods are quite useful, recent research shows that SGD with momentum can obtain better test results over adaptive methods. Wilson et al. [6] conduct a comprehensive study which shows that, contrary to common belief, SGD and SGD with Momentum outperform adaptive optimizers on unseen dataset in designed tasks for over-parameterized models. Adaptive methods have faster progress during the earlier epochs of training while their final performance on test set is not promising. They also found out that tuning Adam classifier brings considerable improvement compared to using the default settings. Another research inspired by Wilson et al.’s [6] findings suggests a simple idea: using Adam on earlier epochs of deep neural network training, then switching to SGD with Momentum to address the saturation problem of Adam on later epochs [16]. They showed that using Adam and SGD with momentum instead of only using Adam results in better generalization. Several modifications have also been proposed for the Adam optimizer. The “YOGI” algorithm [17] proposes an additive adaptive rule which controls the increase in learning rate, as opposed to the rapid increase yield by Adam. Zhang et al. [7] conduct several experiments that compares hand tuned “SGD with momentum” and adaptive optimization methods. They report that models trained with hand-tuned SGD with momentum achieves faster convergence than Adam for many models. Indeed, many recent state-of-the-art image classification models on popular datasets, such as SVHN, CIFAR10 and ImageNet, use the “SGD with momentum” method [18] [19] [20] [21].

2.2.2 Cyclical Learning Rates

Cyclical learning rates (CLR) method [11] addresses the learning rate and learning rate schedule tuning problem. The method trains the neural network by cyclically varying learning rates within predefined boundaries instead of always decreasing the learning rate. Boundaries of cyclical learning rate can be found by linearly increasing the learning rate of the network for a few epochs which is called “LR range test”. The optimal learning rate is inside these boundaries. Then, the learning rates where model

accuracy starts to increase and decrease are taken as minimum and maximum boundaries of the cycle. Learning rate varies between these minimum to maximum and maximum to minimum boundary in a defined step size during training. These method also requires the selection of step size, i.e the number of iterations to take a half learning rate cycle from minimum to maximum. Cycle topology can be triangular or exponential. CLR experiments of different neural network architectures show that CLR achieves better accuracy than fixed learning rate methods. In a follow-up research, Smith et al. [12] propose super convergence phenomenon that can be achieved by using very large learning rates in the cyclical learning rate method. They present that using large learning rates helps to regularize the network that result in better model performance. In learning rate range test, increasing learning rate causes an increase on training loss while test loss is surprisingly decreased at the same time for Resnet-56 architecture. However, proposed rapid convergence was only demonstrated in a single task which limits the generalizability of this method.

Another adaptive method similar to cyclical learning rates is “SGD with warm restarts” (SGDR) [13]. In this work, the learning rate is initialized to some value that is scheduled to decay with an aggressive cosine annealing schedule. Warm restarts refers to restarting only learning rate while other model parameters remain the same with the latest step. SGDR improves many state-of-the-art models error rates on popular datasets.

These methods are important since they show that increasing the learning rate from time to time to reasonably higher values can be beneficial for final network performance.

2.2.3 Gradient Based Tuning Methods

Another adaptive approach is using gradients to find out the optimal learning rate. Schaul et al. [14] define a formula to obtain the optimal learning rates for SGD based on the variance of the gradients. This method can find either single global learning rate, or learning rates for each parameter or parameter group, based on the moving averages of gradients and the diagonal Hessian. This algorithm automatically decreases learning rate to zero when loss function is approaching to its optimal value without any manual learning rate search. Zhang. et al. [7] conduct experiments to show that carefully hand-tuned learning rate can be competitive with adaptive learning rate optimizers. They not only analyzed this phenomena but also, came up with an automated learning rate and momentum tuning approach called Yellowfin. They consider the learning rate tuning problem together with momentum tuning. Similar to Schaul et al.’s work [14], they use noisy quadratic model. In their tuner, hyper-parameters are tuned in every training step using curvature range and gradient variance estimates.

Another work proposes hypergradient descent method to tune the learning rate [15]. They define hypergradient descent as applying gradient descent to learning rate in each training step. This means calculating the partial derivative of the objective function at the previous time step with respect to the learning rate. Applying hypergradient descent on SGD, SGD with Nesterov momentum and Adam has showed that the need for manual tuning is reduced.

2.2.4 Mini-batch validation

We are not the first to explore mini-batch-level validation. Recently, Jenni and Favaro have extended the idea of cross-validation to validate a group of mini-batches with another mini-batch during training [22]. In their method, called “Deep Bilevel Learning” (DBL), at each iteration, they sample a number of training mini-batches and a validation mini-batch. Then, they calculate the gradient of the loss function on all of these mini-batches. Next, the inner product between the gradients of a training mini-batch and the validation mini-batch is computed. Each training gradient is weighted by its corresponding inner product value with the validation mini-batch. The final training gradient to be used for this iteration is the linear combination of all training mini-batch gradients weighted by their inner products with the validation mini-batch. If the gradient of a training mini-batch agrees with the gradient of the validation mini-batch, then their inner product, hence the training gradient’s weight, is a large positive value. They show that this procedure results in better generalization. They explain this achievement as the validation

of gradients helps to avoid memorization by encouraging model parameter updates that only reduce errors on shared sample patterns. Our proposed methods are similar to DBL in the sense that both use training and validation mini-batches during training. However, while DBL is based on gradient similarity (based on inner product), our methods are based on loss values obtained on training and validation mini-batches.

3. Automated Learning Rate Search Methods using Micro Cross Validation

In this section, we describe our proposed micro-CV based automated learning rate search algorithm. The training scenario we consider is a typical one: we are given a supervised training set $T = \{(x_i, y_i)\}_{i=1}^n$ consisting of n examples, a neural network architecture represented by $f(x; \theta)$, where $f(\cdot)$ is the overall function computed by the network, θ is the set of learnable weights of the network and x is an input to the network, and a loss function \mathcal{L} . Our goal is to minimize \mathcal{L} on T by adjusting the network weights θ :

$$\theta^* = \arg \min_{\theta} \frac{1}{n} \sum_{(x_i, y_i) \in T} \mathcal{L}(f(x_i; \theta), y_i). \quad (1)$$

We consider the stochastic gradient descent (SGD) method where a mini-batch B consisting of m examples is randomly sampled ($m \ll n$), on which the gradient vector is computed:

$$g = \frac{1}{m} \sum_{(x_i, y_i) \in B} \nabla_{\theta} \mathcal{L}(f(x_i; \theta), y_i). \quad (2)$$

Then, the learning takes place by updating the network weights θ along the gradient vector g with a step-size η , called the learning rate:

$$\theta^{(t+1)} = \theta^{(t)} - \eta^{(t)} g^{(t)}, \quad (3)$$

where superscript (t) indicates the iteration number. For simplicity, we drop this notation for iteration in the following.

Given the widespread use and its more stable convergence, we consider the ‘‘SGD with momentum method’’ [23, 24], which accelerates learning if recent gradient vectors are consistently aligned. It has an additional parameter α , called the momentum parameter, to decide how much of past gradients are taken into account:

$$\vartheta = \alpha \vartheta - \eta g, \quad (4)$$

$$\theta = \theta + \vartheta. \quad (5)$$

Here the learning rate η is usually a small constant throughout the whole training or a variable which changes its value according to some schedule as a function of time (iteration number). Although decay schedules, where η monotonically decreases through time are more common; cyclical schedules, where η increases and decreases alternately [11], are also possible. In classical, macro (i.e. dataset-level) cross-validation (CV), these learning rate schedules are determined prior to training, mostly based on previous experience.

In this paper, we are interested in setting the learning rate η automatically using micro (i.e. batch-level) crossvalidation (CV). Before we describe how we do this, let us first look at the classical, macro CV.

In macro CV, the training set T is split into two mutually exclusive, training and validation sets (e.g. 80% to %20 is common). Together with this, a set of learning rate schedules are determined prior to training. Note that a ‘‘schedule’’ deterministically defines the values of η throughout a whole training episode. Then, a complete training episode is carried out for each different learning rate schedule. During training, ‘‘early-stopping’’ technique is widely used as a stopping criterion. Loss on the validation set is monitored and if it does not improve for a certain number of epochs, the training is terminated. At

the end of the training, performance is measured on the validation set and recorded. After all the learning rate schedules are used this way, the one yielding the highest performance (or the lower loss) on the validation set is identified as the “optimal learning rate (schedule)”. Many variations to this basic recipe is possible such as using more than one validation set (i.e. k-fold CV).

In contrast to macro CV, in micro CV, there is no need to split T into two prior to training. Instead, each minibatch is split randomly into two at each iteration. On one half-batch, the gradient is computed and on the other half-batch, validation losses for different learning rates are measured. These losses are accumulated for an entire epoch, at the end of which, the learning rate yielding the smallest validation loss is chosen as the learning rate to be used for the next epoch. This way, an “optimal” learning rate is identified for each epoch, which effectively produces a dynamic and adaptive learning rate schedule. Our micro-CV based automated learning rate search method is given in Algorithm 1, which is also illustrated in Figure 1.

Considering the inputs of the classical macro-CV stochastic gradient training algorithm, our micro-CV algorithm has an additional input parameter, λ , which weighs the importance of training and validation losses incurred during training. When λ is 1, only the validation loss is taken into account, and in that case, there is no need to execute line 14 in Algorithm 1.

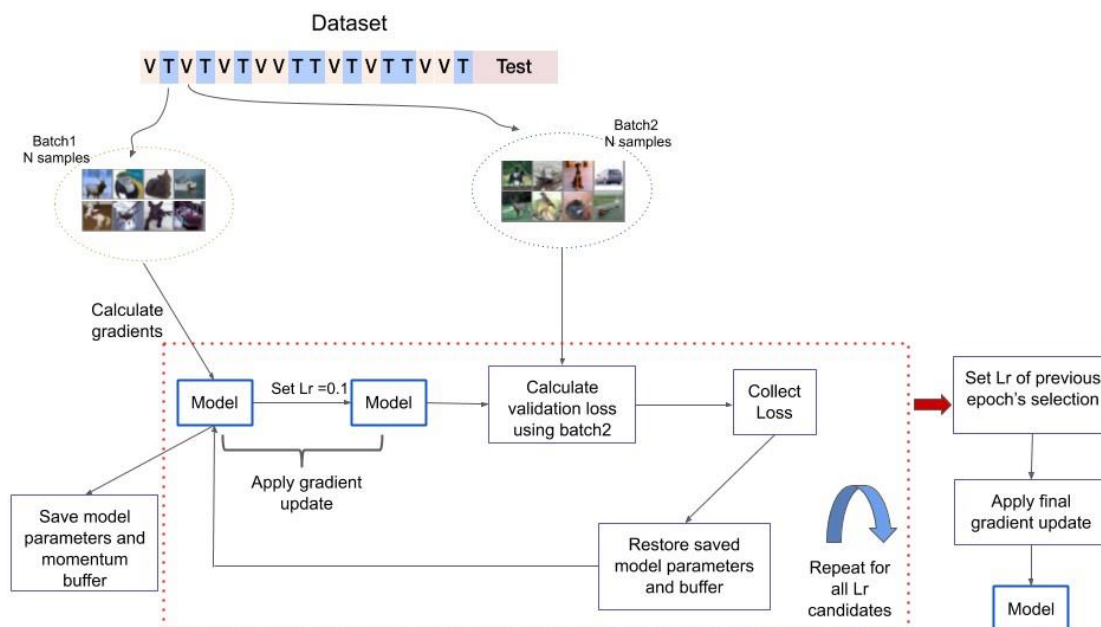


Figure 1 An overview of our micro-CV algorithm. In each training step, a training mini-batch (Batch1) and validation mini-batch (Batch2) are sampled from the training dataset. The gradient of the objective function with respect to the model parameters are computed on Batch1. This gradient is used for “validating” different learning rates on Batch2.

4. Experiments

In this section, we present our experimental results for the micro (i.e. batch-level) cross-validation (MCV) methods described in the previous section, and compare them to the performance of our baseline method. The baseline we consider is the classical macro-CV using stochastic gradient descent with momentum. We test whether our MCV methods that automatically select learning rates improve test set accuracy over the baseline. We also compare our method and the baseline in terms of time complexity.

Algorithm 1 Our “micro-CV” based training algorithm. This algorithm trains a given model by using micro cross-validation to automate the selection of learning rate schedule.

Input: Training set T , initial network weights θ , momentum parameter α , initial velocity ϑ , mini-batch size m , training-vs-validation trade-off parameter λ .

Output: Weights of trained model, θ .

```

1: Initialize learning rate  $\eta \leftarrow 0.00001$ 
2: epoch  $\leftarrow 1$ 
3: while true do
4:   learning_rates  $\leftarrow [5\eta, 2\eta, \frac{4}{3}\eta, \frac{3}{4}\eta, \frac{1}{2}\eta, \frac{1}{5}\eta]$  // List of learning rates to search for
5:   losses  $\leftarrow [0,0,0,0,0,0]$  // Accumulated loss for each learning rate
6:   for  $i = 1 \dots |T|/m$  do // Loop over mini-batches
7:     Sample  $B$ , a mini-batch of  $m$  examples from  $T$ 
8:     Randomly split  $B$  into two half-batches,  $B_1$  and  $B_2$  //  $B_1$  is for training,  $B_2$  for validation
9:     Compute gradient  $g$  on  $B_1$ :  $g \leftarrow \frac{1}{m} \nabla_{\theta} \sum_{(x_i, y_i) \in B_1} \mathcal{L}(f(x_i; \theta), y_i)$ 
10:    for  $j = 1 \dots |\text{learning\_rates}|$  do
11:       $\gamma \leftarrow \text{learning\_rates}_j$  //  $j^{\text{th}}$  element in learning_rates
12:      Create a temporary model using learning rate  $\gamma$ :  $\theta^{\text{tmp}} \leftarrow \theta + (\alpha\vartheta - \gamma g)$ 
13:      Compute the losses on  $B_1$  and  $B_2$  using this temporary model:
14:       $\ell_{B_1} \leftarrow \frac{1}{m} \sum_{(x_i, y_i) \in B_1} \mathcal{L}(f(x_i; \theta^{\text{tmp}}), y_i)$  // training loss
15:       $\ell_{B_2} \leftarrow \frac{1}{m} \sum_{(x_i, y_i) \in B_2} \mathcal{L}(f(x_i; \theta^{\text{tmp}}), y_i)$  // validation loss
16:      losses $_j \leftarrow \text{losses}_j + (1 - \lambda)\ell_{B_1} + \lambda\ell_{B_2}$ 
17:    end for
18:     $\vartheta = \alpha\vartheta - \eta g$ 
19:     $\theta \leftarrow \theta + \vartheta$  // The actual training update
20:  end for
21:   $k \leftarrow \text{argmin}(\text{losses})$ 
22:   $\eta \leftarrow \text{learning\_rates}_k$  // Pick the best learning rate with minimal accumulated loss
23:  overall_loss(epoch)  $\leftarrow \text{losses}_k$ 
24:  if no improvement in overall_loss then
25:    stop training
26:  end if
27:  epoch  $\leftarrow \text{epoch} + 1$ 
28: end while

```

Using just one dataset and a neural network architecture to explore whether MCV is effective would give us a limited picture. In order to increase the generality of our results, in our experiments, we used three image classification datasets, two of them being widely used small-scale benchmark datasets (CIFAR-10 [25] and SVHN [26]) and one of them being a larger scale dataset for age prediction from face images (Adience [27]). On these datasets, we evaluated three different convolutional neural network (CNN) architectures: a small, custom CNN, a ResNet [18] and a VGG [28] network.

Below we first describe the datasets (Section 4.1), the network architectures (Section 4.2) and our performance measures (Section 4.3). Then, we present and discuss the results of our experiments.

4.1 Datasets: CIFAR-10, SVHN, Adience

The CIFAR-10 [25] dataset contains 60,000 32x32 color images from 10 classes which are airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. The training set contains 50,000 images and the remaining 10,000 images are used as testing images.

The Street View House Numbers (SVHN) dataset [26] contains colored digits from Google Street View images, which are obtained from real world street house numbers. We used the cropped and centered version of SVHN that contains 73,257 training and 26,032 testing examples. Each image is 32x32 pixels.

The Adience [27] dataset includes face images for the tasks of age and gender prediction. The dataset consists of 26,580 images from 2,284 subjects which are labeled with 8 age interval classes $\{(0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53, 60-)\}$. We use the cropped and aligned images version of the dataset.

Provided train, test, validation sets include 11823, 4316 and 1284 images respectively. We performed only the age prediction task, skipping gender prediction.

4.2 Network Architectures

On CIFAR-10 and SVHN datasets, we used two well-known and widely used architectures: ResNet-18 [18] and VGG-11 [28], as well as a basic and small CNN, which has only convolutional and fully connected layers without any regularization such as batch normalization and dropout. The small CNN for CIFAR-10 includes six 3x3 convolutional layers which have 48, 48, 96, 96, 192, 192 output channels, respectively. Convolutions are followed by three fully connected layers which have 512, 265, 10 output channels, respectively. Rectified linear unit (ReLU) [18] is used as nonlinear activation function after each convolutional and fully-connected layer. For the SVHN dataset, since digit classification is an easier task, our small CNN architecture has four 3x3 convolutional layers which have 32, 32, 64, 64 output channels, respectively. Convolutions are followed by 2 fully connected layers with 512 and 10 output channels. The Adience dataset, unlike CIFAR-10 and SVHN, contains high-resolution images and it is a relatively more challenging dataset. A large capacity neural network can be beneficial for this dataset's age prediction task. For this reason, we choose to use ResNet-50 [18] architecture for this dataset.

4.3 Performance Measures

We compare the performances of baseline methods and our automated learning rate search methods in terms of accuracy and time. To compare accuracy, we simply use the ratio (percent) of correctly classified examples in the testing set. To compare time expenditure, we use two different performance measures: wall-clock time and theoretical computational complexity.

The “wall-clock time” is simply the time spent running all the experiments required for training a given model on a given dataset from start to end. Here, by training, we mean the whole cross-validation process.

The “theoretical computational complexity” refers to the number of complex operations (with large time expenditure), namely, the forward propagation, backward propagation and weight update operations. Forward operation refers to calculation of output layer's values through passing all neurons of DNN with input. A single forward is required to calculate the loss incurred by the prediction of network and the ground-truth class. Backward operation is performing backpropagation from networks last layer to the first layer by applying the chain rule to calculate gradients. After the backward operation, weights (i.e. parameters) of network are updated with the update operation. This operation calculates the final parameter update inside the optimizer with gradients (e.g. momentum buffer calculation for SGD with momentum). In the following, we present an experiment's theoretical complexity with two numbers: (i) number of total forward and backward operations, (ii) number of total update operations.

We conducted our experiments on computers that have two Xeon Scalable 6148 2.40 GHz CPU processor with 16GB RAM and 4x NVIDIA Tesla V100 16GB. The models are implemented in PyTorch but we also used Keras with Tensorflow backend in our earlier experiments.

4.4 CIFAR-10 Experiments

Macro-CV. To establish the baseline results, we used the classical macro cross-validation with early stopping. For each architecture, we performed two sets of experiments which are summarized below.

1. Use SGD with momentum (abbreviated as “Momentum SGD”) as the optimizer. No learning rate decay. Search for the best learning rate (LR) among $\{0.1, 0.01, 0.001, 0.0001\}$.
2. Use “Momentum SGD” as the optimizer. Use the best learning rate from the previous set. Search for the best learning rate decay parameter among $\{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$. We follow the standard, inverse-time decay rule:

$$\text{LR} \times \frac{1}{1 + \text{decay} \times \text{step}}, \quad (6)$$

where “decay” is the decay parameter and “step” refers to the number of learning updates on the model, or, simply the iteration number.

Both sets of experiments above have a common macro-CV setup. For a given learning rate schedule (i.e. learning rate and learning rate decay parameter), we randomly split the training set into %80 for training and %20 for validation, with stratified sampling. We monitor the validation loss and use early

Table 1 CIFAR-10 baseline macro-CV results for three different CNN architectures. Each row identifies a different training setup, which consists of a learning rate and a learning rate decay parameter. For each training setup, the training set is randomly split into 80%-20% training and validation. Early stopping with a patience of 20 epochs is used. Reported epoch numbers are for the epochs that yield smallest validation error. Optimizer is Momentum SGD. See Section 4.4 for other details.

	LR	LR Decay	Epochs	Test Loss	Test Acc.
Small CNN	10^{-1}	–	3	1.77	33.50
	10^{-2}	–	6	0.72	76.37
	10^{-3}	–	16.6	0.86	72.23
	10^{-4}	–	80	0.97	67.21
	10^{-2}	10^{-3}	7.4	0.77	74.63
	10^{-2}	5×10^{-4}	6.8	0.75	75.46
	10^{-2}	10^{-4}	6.4	0.72	76.60
ResNet-18	10^{-1}	–	6.2	0.63	80.70
	10^{-2}	–	4.4	0.63	79.30
	10^{-3}	–	6.8	0.75	76.12
	10^{-4}	–	13.6	0.92	68.07
	10^{-1}	10^{-3}	5.4	0.58	81.00
	10^{-1}	5×10^{-4}	6.4	0.60	81.50
	10^{-1}	10^{-4}	6.0	0.60	81.44
VGG-11	10^{-1}	–	None	None	10.00
	10^{-2}	–	9.8	0.80	75.40
	10^{-3}	–	42.4	0.96	73.26
	10^{-4}	–	80	1.77	31.30
	10^{-2}	10^{-3}	13.2	0.82	73.30
	10^{-2}	5×10^{-4}	9.8	0.78	75.72
	10^{-2}	10^{-4}	11	0.85	73.61

stopping with a patience of 20 epochs. Since the %80-%20 split introduces randomness on results, we repeat the mentioned process for 5 times to obtain average results. These 5 runs provide us with the required epoch count and the best learning rate and decay schedule. Then, we train the model one last time using these settings on the whole original training set (no splits). Finally, performance is reported on the testing set.

The results of these experiments are given in Table 1 for the three architectures mentioned in Section 4.2, namely, small CNN, ResNet-18 and VGG-11. For all three architectures, Momentum SGD with decay yields the best results: 76.60% for small CNN, 81.50% for ResNet-18 and 75.72% for VGG-11. Optimal learning rate (LR) and decay parameters are all different for different architectures: LR= 10^{-2} , decay= 10^{-4} for small CNN; LR= 10^{-1} , decay= 5×10^{-4} for small ResNet-18; LR= 10^{-2} , decay= 5×10^{-4} for VGG-11.

Micro CV. We obtained our micro-cross validation results for four different λ values ($\lambda \in \{1, 0.9, 0.7, 0.5\}$). Due to the randomness introduced by splitting each mini-batch into two (train and

validation half-batches), we repeated each experiment 5 times and reported the average epoch count, test loss and accuracy, theoretical time cost and wall-clock times. Results are presented in Table 2.

For all three architectures, macro-CV outperforms our micro-CV algorithm, in terms of test accuracy. However, we also observe that micro-CV performs reasonably well in a much shorter time window. For the small CNN, it achieves 73.59 test accuracy, which is 96% of macro-CV's test accuracy (76.60) in only 23% of the time spent by macro-CV (8min 37s vs. 36min 46s). The situation is similar for the other two architectures: for ResNet-18, 99% of macro-CV's test accuracy is achieved in 70% of time spent by macro-CV; for VGG-11, these numbers are 98% and 15%.

Table 2 Comparison of macro-CV and micro-CV results on CIFAR-10. Small CNN, Resnet-18 and VGG-11 test accuracy and loss values are reported with theoretical and time costs.

	Method	Epoch	Test Loss	Test Acc.	Theoretic Cost	Time Cost
Small CNN	Macro-CV baseline	6	0.72	76.60	(656K, 378K)	36min 46s
	Micro-CV ($\lambda = 1$)	53.2	0.91	68.48	(285K, 228K)	22min 56s
	Micro-CV ($\lambda = 0.9$)	35.6	1.03	73.59	(368K, 173K)	19min 38s
	Micro-CV ($\lambda = 0.7$)	5.2	1.68	38.37	(167K, 78K)	8min 54s
	Micro-CV ($\lambda = 0.5$)	4.4	1.49	45.68	(162K, 72K)	8min 37s
ResNet-18	Macro-CV baseline	6.2	0.61	81.50	(480K, 240K)	2h 2min
	Micro-CV ($\lambda = 1$)	100	0.96	65.30	(390K, 312K)	1h 40min
	Micro-CV ($\lambda = 0.9$)	100	0.95	68.71	(664K, 312K)	2h 12min
	Micro-CV ($\lambda = 0.7$)	55.9	1.01	75.29	(503K, 237K)	1h 41min
	Micro-CV ($\lambda = 0.5$)	44.4	0.76	81.03	(427K, 201K)	1h 25min
VGG-11	Macro-CV baseline	9.8	0.80	75.72	(416K, 208K)	1h 34min
	Micro-CV ($\lambda = 1$)	70.9	1.04	63.31	(178K, 142K)	37min 7s
	Micro-CV ($\lambda = 0.9$)	100	1.87	68.05	(332K, 156K)	52min 30s
	Micro-CV ($\lambda = 0.7$)	26	0.89	73.91	(153K, 72K)	24min 9s
	Micro-CV ($\lambda = 0.5$)	14.4	0.92	70.74	(114K, 54K)	18min 4s

Table 3 SVHN baseline (macro-CV) results for three different CNN architectures. Each row identifies a different training setup, which consists of a learning rate and learning rate decay parameter. For each training setup, the training set is randomly split into 80%-20% training and validation. Early stopping with a patience of 20 epochs is used. Reported epoch numbers are for the epochs that yield smallest validation error.

Optimizer is Momentum SGD. See Section 4.4 for other details.

	LR	LR Decay	Epochs	Test Loss	Test Acc.
Small CNN	10^{-1}	–	5	0.55	84.75
	10^{-2}	–	6	0.36	90.16
	10^{-3}	–	16.8	0.46	87.79
	10^{-4}	–	30	0.97	77.84
	10^{-2}	10^{-3}	5.6	0.39	89.16
	10^{-2}	5×10^{-4}	5	0.37	89.56
	10^{-2}	10^{-4}	4.5	0.36	89.66
ResNet-18	10^{-1}	–	4.2	0.21	94.17
	10^{-2}	–	3.4	0.22	93.69
	10^{-3}	–	6	0.26	92.50
	10^{-4}	–	23.6	0.30	91.02
	10^{-1}	10^{-3}	3.8	0.20	94.25

	10^{-1}	5×10^{-4}	3.2	0.20	94.30
	10^{-1}	10^{-4}	3.8	0.21	94.01
VGG-11	10^{-1}	–	None	None	None
	10^{-2}	–	8	0.27	92.30
	10^{-3}	–	37.8	0.26	90.34
	10^{-2}	10^{-3}	15.8	0.32	91.32
	10^{-2}	5×10^{-4}	8.6	0.30	91.94
	10^{-2}	10^{-4}	9.4	0.30	92.53

4.5 SVHN Experiments

We apply the experimental configuration of CIFAR-10 on the SVHN dataset. The only difference is in the architecture of the “small CNN”, which is described in Section 4.2.

Baseline macro-CV results are presented in Table 3. Micro-CV results are presented in Table 4. The summary of results is similar with that of CIFAR-10. In terms of test accuracy, macro-CV outperforms micro-CV, however, when time cost and test accuracy are considered together, micro-CV achieves a high relative test-accuracy with lower time cost. For the small CNN, micro-CV achieves 89.55% test accuracy, which is 99% of macro-CV’s test accuracy (90.16%) in only 21% of the time spent by macro-CV (87min 47s vs. 36min 30s). Similarly, for ResNet-18, micro-CV achieves 98% relative accuracy with 45% relative time cost; for VGG-11, these numbers are 93% and 61%.

Table 4 Comparison of macro-CV and micro-CV results on SVHN. Small CNN, Resnet-18 and VGG-11 test accuracy and loss values are reported with theoretical and time costs.

	Method	Epoch	Test Loss	Test Acc.	Theoretic Cost	Time Cost
Small CNN	Macro-CV baseline	4.8	0.36	90.16	(440K, 220K)	36min 30s
	Micro-CV $\lambda = 1$	27.8	0.37	89.55	(137K, 109K)	14min 25s
	Micro-CV $\lambda = 0.9$	33.4	0.62	89.15	(260K, 122K)	17min 37s
	Micro-CV $\lambda = 0.7$	6	0.55	83.30	(126K, 60K)	8min 35s
	Micro-CV $\lambda = 0.5$	3.6	1.20	58.85	(115K, 54K)	7min 47s
ResNet-18	Macro-CV baseline	3.2	0.20	94.30	(388K, 194K)	2h 47 min
	Micro-CV $\lambda = 1$	80.5	0.40	88.53	(286K, 229K)	2h 2 min
	Micro-CV $\lambda = 0.9$	92.6	0.31	91.45	(496K, 229K)	2h 53 min
	Micro-CV $\lambda = 0.7$	66.25	0.38	93.78	(420K, 197K)	2h 29min
	Micro-CV $\lambda = 0.5$	23.4	0.27	92.73	(211K, 99K)	1h 15min
VGG-11	Macro-CV baseline	9.8	0.30	92.53	(436K, 218K)	1h 37min
	Micro-CV $\lambda = 1$	100	0.45	86.25	(286K, 229K)	59min 55s
	Micro-CV $\lambda = 0.9$	13.7	1.46	48.70	(163K, 77K)	25min 54s
	Micro-CV $\lambda = 0.7$	3.4	2.05	26.30	(114K, 54K)	17min 59s
	Micro-CV $\lambda = 0.5$	2	2.23	19.60	(107K, 50K)	16min 54s

Table 5 Adience average baseline results on ResNet-50. Experiments with average epoch count, test accuracy and test loss are reported. They are conducted to obtain learning rate and learning rate decay setting that gives best accuracy result. Each setting of learning rate and learning rate decay is repeated 5 times.

LR	LR Decay	Epochs	Test Loss	Test Acc.
10^{-1}	–	26.6	1.47	50.12
10^{-2}	–	12.8	1.65	48.64
10^{-3}	–	16.4	1.68	44.81
10^{-4}	–	77	1.71	42.61

10^{-1}	10^{-2}	139.5	1.44	48.64
10^{-1}	10^{-3}	45.6	1.46	49.53
10^{-1}	5×10^{-4}	65.4	1.54	45.86
10^{-1}	10^{-4}	24.4	1.37	51.02

4.6 Adience Experiments

Compared to CIFAR-10 and SHVN, the Adience dataset contains higher resolution images. Therefore, on this dataset, we use a larger network, namely, ResNet-50.

Train and test experiment configuration is the same as that of CIFAR-10. One difference here is that experiments are repeated only 3 times, instead of 5. Another difference is in the selection of the validation set for the macro-CV experiments. Since train, test and validation sets are provided in the dataset, we used the provided validation set in all experiments instead of random validation set for each experiment. Baseline macro-CV results are presented in Table 5. And, the micro-CV results are presented in Table 6.

Table 6 Adience average adaptive learning rate search using MCV results on ResNet-50. Experiments with average epoch count, test accuracy and test loss, theoretical and wall clock time cost are reported.

Method	Epoch	Test Loss	Test Acc.	Theoretic Cost	Time Cost
Macro-CV baseline	24.4	1.37	51.02	(522K, 261K)	26h 16min
Micro-CV $\lambda = 1$	75.6	1.80	39.11	(176K, 141K)	3h 38min
Micro-CV $\lambda = 0.9$	96.7	1.63	46.75	(314K, 148K)	4h 13min
Micro-CV $\lambda = 0.7$	81.6	2.14	48.20	(314K, 148K)	4h 13min
Micro-CV $\lambda = 0.5$	72.6	2.28	52.50	(291K, 137K)	3h 54min

5. Conclusion

In this paper, we propose a minibatch-level (i.e. micro) cross-validation algorithm that can dynamically and adaptively choose the learning rate at each training epoch. As an alternative to the dataset-level, macro approach to cross-validation, in micro-CV, the randomly sampled minibatch is randomly split into training and validation halfbatches. The gradient vector computed on the training half-batch is re-used to evaluate several different learning rates (i.e. step sizes) on the validation half-batch. Using stochastic gradient descent with momentum as the optimizer, experiments on three different datasets with three different neural network architectures show the potential of our micro-CV training algorithm.

On CIFAR-10 and SVHN, micro-CV achieves a slightly lower test accuracy and macro-CV, however, it spends a much lower computational budget and time. For example, on CIFAR-10 using a ResNet-18 network, while macro-CV achieves 81.50% test accuracy in 2 hours 2 minutes, micro-CV achieves 81.03% in 1 hour 25 minutes. On SVHN, with the same architecture, macro-CV yields 94.30% accuracy in 2 hours 47 minutes, micro-CV yields 92.73% in 1 hour 15 minutes. Finally, on the relatively much larger and more realistic Adience dataset, our micro-CV algorithm outperforms macro-CV with a drastic reduction in total training time; macro-CV: 51.02% test accuracy in 26 hours 16 minutes, micro-CV : 52.50% test accuracy in 3 hours 54 minutes. These results show the promising potential of our micro-CV algorithm. Further research is required to comprehensively compare micro-CV to adaptive learning rate methods such as Adam, AdaGrad and Rmsprob.

Acknowledgments






The numerical calculations reported in this paper were fully performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources).

References

- [1] K. Anand, Z. Wang, M. Loog, and J. van Gemert, “Black magic in deep learning: How human skill impacts network training,” in *British Machine Vision Conference*, 2020.
- [2] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, “Green ai,” *Communications of the ACM*, vol. 63, p. 54–63, Nov. 2020.
- [3] G. E. Hinton, N. Srivastava, and K. Swersky, “Neural Networks for Machine Learning,” *COURSERA: Neural Networks for Machine Learning*, 2012.
- [4] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic gradient descent,” in *ICLR: International Conference on Learning Representations*, 2015.
- [5] “Neural networks (maybe) evolved to make adam the best optimizer – parameter-free learning and optimization algorithms.” <https://parameterfree.com/2020/12/06/neural-network-maybe-evolved-to-make-adam-the-best-optimizer/>. (Accessed on 03/01/2021).
- [6] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, “The Marginal Value of Adaptive Gradient Methods in Machine Learning,” in *Advances in Neural Information Processing Systems 30* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), pp. 4148–4158, Curran Associates, Inc., 2017.
- [7] J. Zhang, I. Mitliagkas, and C. Re, “YellowFin and the Art of Momentum Tuning,” *CoRR*, vol. abs/1706.0, 2017.
- [8] D. Kabakcı, “Automated learning rate search using batch-level cross-validation,” Master’s thesis, Middle East Technical University, Ankara, Turkey, July 2019. <https://open.metu.edu.tr/handle/11511/43629>.
- [9] J. Duchi, E. Hazan, and Y. Singer, “Adaptive subgradient methods for online learning and stochastic optimization,” *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.
- [10] J. Bergstra and Y. Bengio, “Random Search for Hyper-Parameter Optimization,” *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.
- [11] L. N. Smith, “Cyclical Learning Rates for Training Neural Networks,” *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 3 2017.
- [12] L. N. Smith and N. Topin, “Super-Convergence: Very Fast Training of Residual Networks Using Large Learning Rates,” *CoRR*, vol. abs/1708.0, 2017.
- [13] I. Loshchilov and F. Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts,” in *ICLR: International Conference on Learning Representations*, pp. 1–16, 2017.
- [14] T. Schaul, Z. Sixin, and Y. LeCun, “No More Pesky Learning Rates,” in *Proceedings of the 30th International Conference on Machine Learning*, 2013.
- [15] A. G. Baydin, R. Cornish, D. M. Rubio, M. Schmidt, and F. Wood, “Online Learning Rate Adaptation with Hypergradient Descent,” in *International Conference on Learning Representations*, 2018.
- [16] N. S. Keskar and R. Socher, “Improving generalization performance by switching from Adam to SGD,” *arXiv preprint arXiv:1712.07628*, 2017.
- [17] M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar, “Adaptive Methods for Nonconvex Optimization,” in *Advances in Neural Information Processing Systems 31* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), pp. 9793–9803, Curran Associates, Inc., 2018.
- [18] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [19] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017.
- [20] J. Hu, L. Shen, and G. Sun, “Squeeze-and-Excitation Networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6 2018.

- [21] Y. Huang, Y. Cheng, D. Chen, H. Lee, J. Ngiam, Q. Le, and Z. Chen, “Gpipe: Efficient training of giant neural networks using pipeline parallelism,” in *Advances in Neural Information Processing Systems (NIPS)*, 2019.
- [22] S. Jenni and P. Favaro, “Deep bilevel learning,” in *Proceedings of the European conference on computer vision (ECCV)*, pp. 618–633, 2018.
- [23] B. T. Polyak, “Some methods of speeding up the convergence of iteration methods,” *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.
- [24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
- [25] A. Krizhevsky, G. Hinton, *et al.*, “Learning Multiple Layers of Features from Tiny Images,” tech. rep., Department of Computer Science, University of Toronto, 2009.
- [26] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y Ng, “Reading Digits in Natural Images with Unsupervised Feature Learning,” in *Advances in Neural Information Processing Systems (NIPS)*, 2011.
- [27] E. Eidinger, R. Enbar, and T. Hassner, “Age and Gender Estimation of Unfiltered Faces,” *IEEE Transactions on Information Forensics and Security*, vol. 9, pp. 2170–2179, 12 2014.
- [28] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” in *ICLR: International Conference on Learning Representations*, pp. 1–14, 2015.

A Digital Forensics Approach for Lost Secondary Partition Analysis using Master Boot Record Structured Hard Disk Drives

 Erhan Akbal¹,  Omer Faruk Yakut²,  Sengul Dogan³,  Turker Tuncer⁴,  Fatih Ertam⁵

¹Corresponding Author; Firat University, Technology Faculty, Digital Forensics Engineering, Turkey; erhanakbal@firat.edu.tr; +90 533 493 46 03

² Firat University, Department of Digital Forensics Engineering, Turkey; omerfaruk.yakut@egm.gov.tr

³ Firat University, Department of Digital Forensics Engineering, Turkey; sdogan@firat.edu.tr

⁴ Firat University, Department of Digital Forensics Engineering, Turkey; turkertuncer@firat.edu.tr

⁵ Firat University, Department of Digital Forensics Engineering, Turkey; fatih.ertam@firat.edu.tr

Received 13 November 2021; Revised 24 November 2021; Accepted 07 December 2021; Published online 31 December 2021

Abstract

The development and widespread use of computer systems has increased the need for secure storage of data. At the same time, the analysis of digital data storage devices is very important for forensic IT professionals who aim to access information to clarify the crime. File systems of disk drives use partition structures to securely store data and prevent problems such as corruption. In this study, deletion or corruption of partitions on commonly used DOS / Master Boot Record (MBR) configured hard disk drives are investigated by using forensic tools. In order to analyze hard disk drives, Forensic Tool Kit (FTK), Magnet AXIOM, Encase, Autopsy and The Sleuth Kit (TSK), which are widely used as commercial and open source, are analyzed by using a presented scenario. In the scenario, the primary partition and the extended partition are created using the DOS / MBR partitioning structure on the test disk. Test files are added to the sections and the sections are deleted. The digital forensics tools were tested on the presented scenario. According to the obtained results, TSK and Encase are successful tools for DOS / MBR structured HDD analysis. However, FTK, Magnet AXIOM and Autopsy could not achieve information detection on DOS/MBR structured disks. These results clearly demonstrated that crime data can be hidden in MBR structured HDD. To carve these data, the correct methodology should be selected.

Keywords: digital forensics, dos/mbr partition, extended partition, lost partition, recovery partition, antiforensic

1. Introduction

The purpose of forensic analysis is to discover and present digital evidence in computer systems to reveal the reality of an event [1, 2]. Digital evidence is intended to be scientifically valid, reliable, and verifiable [3]. Knowledge extraction from big data in various formats is very important for digital forensics [4]. The digital evidence that is the main source of forensic information is stored on hard disk drives in computer systems. Digital forensics analysis is basically concerned with identifying, extracting, and analyzing file systems in these data storage systems [1]. With the rapid growth in computer systems, the use of large volume storage devices has increased. As a result of this increase in volume, the time allocated per case in forensic laboratories increases, and forensic processes are prolonged [5]. Digital forensics examiners use commercial or open-source forensics tools to minimize these delays and analyze errors in forensic processes. This age is called an information age [6]. Electronic evidence is crucial for judicial authorities. Therefore, the reliability of digital evidence is based on the correct use and reliability of forensic tools as well as the scientific implementation of the process [7]. Nowadays, digital devices for instance laptops, cameras, and mobile phones have evolved rapidly. Therefore, there are variable digital evidence in practice. In this rapidly changing dynamic environment, it became impossible to find a single forensic tool that could meet all needs [8].

The main task of digital forensic tools is to present obtained evidence in a convenient format for forensic examiners with superficial knowledge, and these tools have a crucial role in an investigation [9, 10]. Suspects may conceal data to prevent forensic analysts from accessing the data or disrupt the data

structure with Anti-Forensic methods. This may cause courts to decide on the basis of false or incomplete evidence [11, 12].

It has led to an increase in anti-forensic methods to prevent the judicial process and interfere with the evidence. The increase in the market share of anti-forensic tools clearly indicates this situation [13]. Moreover, users can easily access anti-forensic tools [14]. User-induced problems also cause data loss like anti-forensics tools. For example; The forensic tools that are dealing with the "42.zip" compression grenades cannot complete the evidence processing process because they cannot open the layered file [9]. Similarly, in the forensic analysis of the partition tables of Guid Partition Table disks, removing the Disk Protected Area and Device Configuration Overlay areas on the disk will prevent the analysis tools from accessing the spare part header and table at the end of the disk as it will change the position of the last sector [15]. In addition, disruption of extended partition structures on a disk configured with the widely used DOS / MBR partition table will prevent access to evidence and evidence metadata data within this corrupt partition.

A disk configured with the DOS / MBR partition table allows up to four primary partitions to be created by nature. In BIOS-based systems, it assigns one of the four standard partitions as extended partitions. These extended partitions are special partitions that can be divided into logical partitions [15, 16].

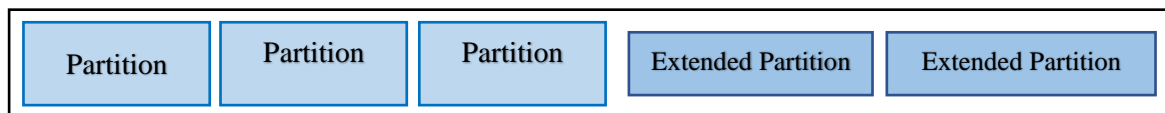


Figure1 Example section layout for bios based systems.

Extended partitions do not receive a partition ID, so logical partitions can be created as much as the capacity of the disk allows. Files can be stored on logical partitions and used as operating system partitions.

Knowing the capabilities of the forensic tool is very important for analysis. The copying and analysis processes of digital materials are completely dependent on forensic tools. For this reason, the findings of the forensic tools in the process of detecting guilt and innocence should be real, reliable, and reproducible. Errors occurring at any stage of the analysis can cause the potential data to be destroyed [17]. Forensic analysts generally rely on the used forensic tools, but different tools show different abilities in recovering and interpreting data on the same evidence. It should be known that all forensic tools have disadvantages as well as their advantages and that a single vehicle is not sufficient for all purposes [18, 19]. For this reason, if the partitions that are commonly encountered with DOS / MBR are corrupted or deleted, a schedule should be planned to detect these partitions.

1.1. Novelties and contributions

Novelties and contributions of our work are given below.

- A roadmap for the detection of lost extended partition structures on disks configured with the commonly used DOS / MBR partition table is proposed.
- The performances of popular forensic tools in comparison to the complexity of the extended section structures are compared.
- Data hiding and destruction methods on various disk areas are defined by using partition structures.
- The method for determining the location of the missing partitions on the disk by calculating all the parts on the disk is proposed.
- A comparison table regarding the detection of the deleted and destroyed Primary and Logical sections and the capabilities of the forensic tools in accessing the data within these sections were shown.
- Different forensics tools were examined.

1.2. Our scenario

In this scenario, a five volume HDD is presented. Three of them primary and two of them secondary. Then, we stored documents these volumes. Then, a primary and a secondary volume were deleted. To recover the deleted document, analyses have been performed by using six tools and a manual analysis. By using this scenario, a comprehensive benchmark is obtained and an optimum algorithm has been obtained for the MBR partitioned problem.

1.3. Organization

The rest of the article is organized as follows. The literature on anti-forensic methods has been examined. With the detection of deleted or destroyed partition structures on disks configured with DOS / MBR partition table, the methods of accessing the data in it are presented, and the competencies of commonly used forensic tools in this regard are compared according to the results of the sample scenario application and presented in the Methodology section. The results and the findings obtained in the discussion section are shown. The limits of our work are presented in the limitation section. Conclusion and future studies are presented in the last section.

2. Related Work

Digital forensics processes are directly proportional to the understanding of anti-digital forensics applications. For this reason, the results caused by the techniques that prevent digital forensics processes should be examined further [20]. Tools designed for anti-digital forensics purposes are typically divided into two categories. The first category is special anti-digital forensics tools. These tools can operate in the form of data hiding, data removal, data processing/editing/masking, Data confusion, and physical destruction. The second category is disruptive technologies. Disruptive technologies have a primary legitimate function and goals. Thus, any investigation may also have a detrimental effect on the relevant digital data in a device [21]. Anti-digital forensics tools and methods, which try to endanger the existence or reliability of the evidence throughout the judicial processes, are becoming more common and used every day [22].

There are different studies on anti-forensic information methods that address current problems and point out future methods. For example, one study interfered with six different anti-forensics tools. At the end of each intervention, the forensic copy of the disk was examined with the FTK forensics tool. In the study where the performance of these anti-forensics tools was observed; It has been observed that each anti-digital forensics tool performs incomplete deletion on unallocated areas. In this case, it was pointed out that it could allow the recovery of data from unallocated areas [13].

In another study, the possibility of using timestamps of the ext4 file system, which is an effective tool for data hiding in environments such as Linux operating systems and android devices, was analyzed. In this study, an ext4 digital forensics technique has been designed that shows that the nanosecond part of Ext4 timestamps can be used to create a system with steganographic power [23]. Apart from these, it has been shown that important signatures can be easily changed in order not to be detected by the malware by a one-byte cancellation factor attack method [24]. In addition, it has been revealed that with semantic value manipulation attack, data values with significant semantic meanings have been changed [25]. Moreover, studies have been carried out on anti-digital forensics methods such as attention deficit technique, which creates false objects and increases the analysis time in order to put researchers into wrong solution processes [26]. In another study where anti-digital forensics methods were examined in two different categories as provable and unproven, inconsistencies were examined by comparing the evidence that was attacked with reliable evidence for the detection of counterfeit information in the log and warning information examined as evidence [27]. Apart from these studies, in the literature, hard disk and file cleaning [28], changing file signature information [29], forgery of file timestamps, use of a restricted folder or file names, circular referencing and the use of ASCII character text, attacks, forensic information evasion techniques used to present the findings and digital forensics techniques applied to eliminate traces in Windows operating systems [30-32].

Different studies have been carried out on obtaining data from harddisks. [33] presented a methodology for obtaining OS-independent data from storage devices using UEFI firmware. [15] developed a tool for forensic examination of disks using the GUID partition table. [34] measured the performance of hard disk drives debug interfaces and service access methods. As a result, they have shown that data can be obtained from SATA disks. [35] demonstrated the impact of file systems, memory management, and disk partitioning structures on evidence acquisition. They found that by applying different scenarios, it would be difficult to collect evidence. [36] proposed a validation method for scenario-based file scraping from discs.

3. Preliminaries

The main purpose of our study is to explain how to access the data on the disk in case the partition structures are deleted or corrupted on a disk with MBR structured partition structures. In addition, by suggesting a calculation method to determine whether there is a lost partition in the disk partitions, the effects on the MBR methodology and data hiding were examined.

3.1. MBR Methodology

MBR is a disk partitioning system that was originally designed and started to be used in IBM systems. The first sector of MBR hard drives is 512 bytes in size, and bytes are stored as Little Endian [37]. It consists of 3 parts, and these parts are Master Boot Code, Master Partition Table, and MBR Signature [38].

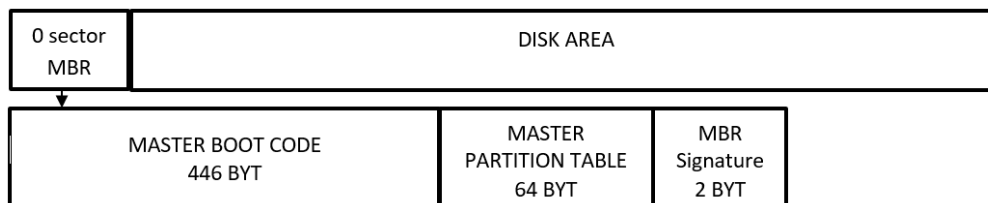


Figure 2 MBR disk location and basic structure.

The Master Boot Code in the MBR structure shown in Figure 2 contains the codes that the BIOS will read and run when the computer is first turned on. Master Partition Table is the part where primary partition information is kept. It can hold information with a range of to one from four sections. MBR signature defines MBR signature bytes. Partition Table size is kept in an area of 64 Bytes, and 16 Bytes are reserved for each section information on the MBR. This allows a maximum of 4 primary partitions to be created on a disk configured with MBR. The extended partition structure is used as a solution to this limitation.

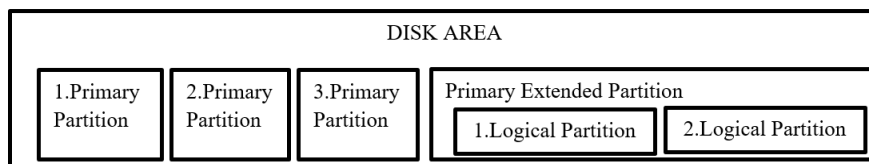


Figure 3 Schematical demonstration of the MBR of Partition Table.

The placement of the partition structures on the disk is shown in Figure 3. Three entries in the MBR define the primary partitions, and one extended partition is defined for the remaining disk space. Primary partition records and primary extended partition records are not kept on the MBR. Primary extended partitions are divided into Logical Partitions in themselves, and the addresses of these separated Logical partitions are kept in the MBR partition table in the first sector of the primary extended partition. Although this is a solution to the limitation of creating a limited number of sections in practice, the willful or unintentional destruction of the logical sections will cause losses related to the data on these sections [39].

In digital forensics analysis, as well as accessing the data, the location information of the data on the disk is also important in the evaluation phase. For example, while an inappropriate content that is inadvertently downloaded is in a location in the *WebBrowserName* \..... \..... within the user directory, the voluntarily downloaded content is usually located in the directories the user uses for storage. In order to reveal this difference, the correct location of the data in the examination process guides the forensic expert. For this reason, accessing the source information of the evidence recovered from deleted and destroyed departments is of great importance in terms of Forensic IT analysis [10].

3.2. Partitions can be hidden data

Digital forensic analysis uses data acquisition methods in different layers such as volume, file, and application layers. However, it is an accepted general approach to perform analysis operations on the entire copy of the hard disk. Regarding a forensic copy taken from the volume layer, for example, it will not be possible to access the sectors in the partition table that have been destroyed and the parts that have been deleted during the analysis, since the forensic tool will only copy the sectors in the addressed areas when making copies [40]. On the other hand, there is a reserved area between sectors 1-62 in disks using DOS partitioning structure, and evidence that can be hidden in these areas will not be available [39].

3.3. File system corruptions

Many data can be obtained on the partitions. However, basically the data is stored in file systems [41]. In file system analysis, it is possible to reach the location (path), content, and meta data information where the available data are stored. In case the partition structures are corrupted or deleted, the file system structure will also be corrupted as the file systems reside on the partition structures. In this case, although data recovery tools access data within the sectors, they will not be able to access the location (path), content, and meta data information of the data. Therefore, it is important to know the details of disk partitioning structures as well as knowing how a forensic tool works [15].

4. Scenario Analysis

A scenario has been created in the laboratory environment for forensic analysis on the disk partitions. On the created scenario, a review was performed with *six* forensic analysis tools. Many file systems have emerged in accordance with the needs arising with technological developments. The developed file systems bring many innovations with it. File systems generally show similarities to each other [42]. However, it differs according to the structure, storage method, and intended use. These differences also affect the analysis and data recovery methods in file systems in terms of Digital Forensics. One of the widely used file systems, New Technology File System (NTFS), was designed by Microsoft and used as the default file system for Microsoft Windows NT, Windows 2000, Windows XP, Windows 7,8,10 and Windows Server [43, 44]. The NTFS file system has been developed to replace the previously widely used File Allocation Table (FAT) file system. When the FAT file system was widely used, the NTFS file system was preferred only on the server-side. It is now widely used in personal computers. NTFS is a much more complex file system than FAT because it is capable and scalable. NTFS is designed for reliability and large storage devices, and thanks to its scalable structure, it allows changes to be made over time in line with new demands. Each byte of data in an NTFS file system is divided into one file [45]. The first entry in the NTFS file system is the "Boot Metadata" file that starts from Sector 0 and can take up to 16 Sectors in length. This file holds the base unit and location of \$MFT. One sector on each allocated NTFS volume belongs to a file. The MFT startup is in Volume Boot Record (VBR). VBR is in a \$Boot record in the MFT [45, 46].

Other legacy and widely used file system FAT is by far the simplest of the file systems supported by Windows NT. It is the map of the disk that indicates the areas in which the information of the files in a disk is recorded. In the FAT file system, the partition is divided into clusters, each containing a certain amount of sectors. Where and how files are written on these clusters is defined on the FAT system.

When the operating system wants to access any file, it takes advantage of this information overwritten by FAT [47].

Recovering data by using MFT / FAT records is a method frequently used by digital forensics experts. Most of the digital forensics tools use MFT records to successfully and rapidly perform data recovery. This situation allowed for the rapid progress of judicial processes [48]. Although manual data recovery methods can be applied using the hex editor, this can cause serious time and labor loss. Manual recovery methods should also be applied when necessary [49].

Digital forensics tools are achieved in data recovery methods with MFT / FAT records. However, in the analysis, there are cases where the location and meta-data information of the recovered data on lost and deleted partition structures on disk areas configured with MBR are missing. In some cases, it is not sufficient for digital forensics tools to search only on file system defined sections. Because in this case, it will not be possible to access the data in the deleted partition that cannot be determined to have a valid file system. The analysis process of disks configured with MBR and not using extended partition structure is easy, but extended partition complicates the analysis process [50].

There has been a significant increase in the size of storage media over the years. However, the size of the logical block format known as the sector, which forms an important part of hard drives, remained constant. In the 2010s, hard drive companies started moving from the old sector size of 512 bytes to a larger, more efficient sector size of 4096 bytes, often referred to as 4K sectors and now referred to as the Advanced Format by IDEMA (International Disk Equipment and Materials Association). However, long-term advantages and potential dangers have emerged during the transition from 512 bytes to 4K sectors. A 512-byte sector can usually correct defects up to 50 bytes in length. Hard drives today are pushing the limits of error correction. Consequently, it has become a basic need to improve the transition to larger sectors, error correction and format efficiencies in the hard drive industry.

It is not possible for the entire hard disk industry to move to the new 4K standard and to change all of these old assumptions suddenly. The methods and calculations to be applied in the study were carried out using traditional 512-byte sectors and disks addressed. It is not available for disk structures with a sector size of 4096 bytes or less 2048 byte, referred to as Advanced Format. For these reasons, in the scenario created for forensic analysis on disk partitions in the laboratory environment, the Test Disk was structured with MBR and created with 3 main partitions and 2 extended partitions. The partition structures created are formatted with NTFS and FAT file system. By labeling the section structures, documents with the same label as the section were copied into these sections, and the created test disk was analyzed with 6 forensic tools and the results were evaluated. The main purpose of the study is to determine the behavior of digital forensics tools related to the complete and lossless recovery of data within the partition structures structured with MBR and to propose a roadmap that will help identify lost partition structures.

4.1. Analysis method

The existing partition tables should be examined on a deleted disk with suspect partitions, and the consistency of the total size of the disk with the total size of the current partitions should be confirmed. Also, after the control of the end and start sectors of the sections, if there are lost partitions, these sections should be tried to be saved. To determine this process, performing the calculation suggested in Figure 4 on the disk will ensure the accuracy of the analysis process.

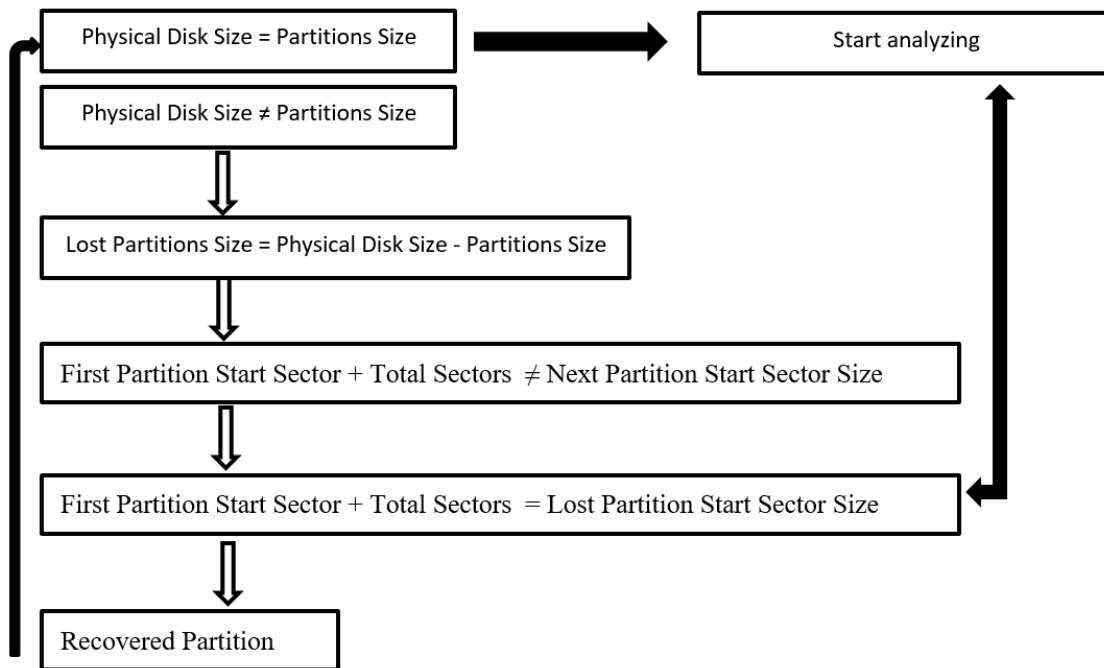


Figure 4 Flow chart of the proposed lost partition detection.

The steps of this process are given below.

Step 1: The Total size of the disk to be analyzed is compared with the total size of the available partitions.

Step 1.1. If the size is equal, after the confirmation of the operation with the controls in Step 2, the analysis process is started.

Step 1.2. If the size is not equal, it should be considered that a lost partition state occurred.

Step 2: The sum of the starting sector of the first partition on the disk and the total number of sectors is expected to give the starting sector of the next partition. If the result obtained does not match the results in the section table, it indicates that there may be a lost partition structure. The result obtained in this case gives the physical address of the first sector of the lost partition on the disk. The above procedure is performed on all partitions, respectively, to confirm whether the partition's start and end sectors are the same as in the partition table.

Step 3: File system tags (NTFS, FAT etc.) and signature value of 55AA are searched in the first sector of the lost part detected. With this label and signature value confirmed, lost partition recovery is performed.

Step 4: After the lost partition is recovered, the calculations in Step 1 are repeated and any lost partitions are detected. Operations continue until 1.1 step confirmation is achieved and then disk analysis is started.

4.2. Experimental setup

For this study, a Work Station Computer with HP Z840 intel® Xeon® CPU E5-2680 @ 2.40 GHz (2 Processors), 128 GB RAM and Samsung brand 160 GB hard disk (Test Disk) is used, and this disk is configured with MBR.

Since a maximum of 4 primary partitions can be created on the partitions configured with MBR, the test disk is divided into *five* sections in total, *three* primaries, and *two* extended logical partitions in order to use the extended partition structures. Test.docx, Test.xlsx, and Test.zip documents were created in these *five* sections. These files will be used in the analysis phase to evaluate the success of forensic tools. After deleting *one* primary partition and *one* extended partition from the partitions created in the second part of the study, the test disk will be analyzed using licensed and open-source forensic tools, and the results will be evaluated. The steps of this process are given below.

Step 1: The hard disk with 160 GB capacity of Samsung brand is configured with MBR and it is divided into *five* sections as *three* primary partitions and *two* extended partitions.

Step 2: Sections are named as VOLUME_A, VOLUME_B, VOLUME_C, VOLUME_D, and VOLUME_E.

Step 3: Zip, excel, and word documents are created in the sections and are named to be related to the section letters.

Step 4: Test documents were added to the Primary Section named VOLUME_B and to the Extended Section named VOLUME_E.

Step 5: VOLUME B and VOLUME_E sections were deleted.

The accuracy of the findings to be obtained in case of analyzing the mentioned processes after this scenario was determined.

Disk size and available partitions can be easily calculated with different software as well as using existing analysis software. The values obtained for the initial disk are shown in Table 1.

Table 1 Initial size and sector information of the test disk

Device Name	Disk Image
File Path	O:\SAMSUNG_160_GB_HDD_TEST_IMAGE_V\image.001
Total Size	160.041.888.256 Bytes (149.1 GB)
Total Sectors	312.581.813
Disk Signature	9201C3C1
Partitions	Valid

4.3. The used digital forensics tools for analysis of the defined scenario

After preparing the scenario environment, licensed AccessData Forensic Toolkit, Magnet Axiom, Encase open-source licensed Autopsy, and TSK tools were used in forensic investigations. Whether or not test data can be obtained from the scenario created with the analyzes made and accessibility to details such as location and date were investigated. The software and version information used are given in Table 2.

Table 2 The used digital forensics tools.

Software Tool	Version	License
Access Data Forensic Tool Kit	7.0.0.163	Licensed
Encase	8.7.00.93	Licensed
Magnet Axiom	3.0.0.13673	Licensed
Autopsy	4.11.0	Open Source
The Sleuth Kit	4.7.0	Open Source

4.3.1. Analysis of the scenario using AccessData Forensic Toolkit

After reading the size and sector information of the test disk, when the partitions determined by the software are checked, a total of 4 partition information was found. It was understood that one of these sections was labeled as a recovered section.

Name	Path	P-Size	L-Size	Category
[Recovered] Partition 1	image.001/[Recovered] Partition 1	29,30 GB	29,30 GB	Partition
Partition 1	image.001/Partition 1	22,09 GB	22,09 GB	Partition
Partition 2	image.001/Partition 2	24,41 GB	24,41 GB	Partition
Partition 5	image.001/Partition 5	19,53 GB	19,53 GB	Partition
Unpartitioned Space [basic disk]	image.001/Unpartitioned Space [basic disk]	n/a	n/a	Unpartitioned Space

Figure 5 Section information detected by FTK software

While the total disk size is supposed to be 149.1 GB, which is the initial calculated value, the total size of the partitions detected by the software; It appears to be 29.30 GB + 22.09 GB + 24.41 GB + 19.53 GB = 95.33 GB.

Consequently, since Physical Disk Size \neq Partitions Size, the existence of the lost partition structure should be considered. Figure 5 shows that the size of the recovered partition is 29.30 GB. When the files in the partition structure are given the same letter label as the partition label, when the documents in the recovered partition are checked; It is understood that there are documents named test_file_B.docx, test_file_B.rar and test_file_B.xlsx in VOLUME_B. (See Figure 6)

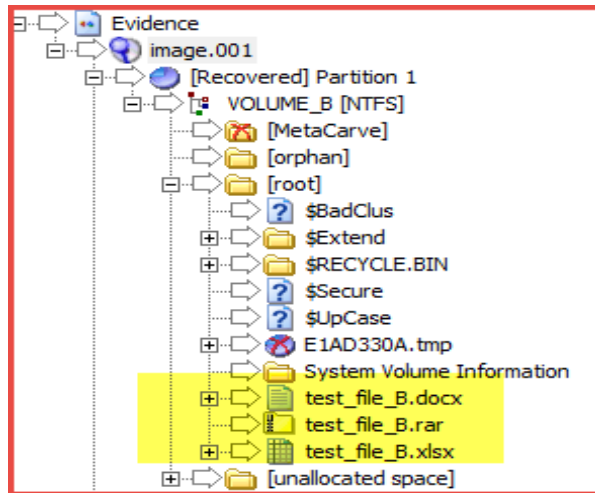


Figure 6 Documents labeled with the section name in the Recovered Section.

When analyzing the Extended Partition on the disk, it is seen that the deleted extended partition cannot be recovered. Since the forensic tools also perform sector-based data recovery operations, the data in these areas should be analyzed since the Unpartitioned Space may have recovered existing files on the area.

File List		
✓	Name	Path
<input type="checkbox"/>	Carved [33214464].zip	image.001/Unpartitioned Space [basic disk]/[unallocated space]/199940096*Carved [33214464].zip
<input type="checkbox"/>	Carved [33206272].zip	image.001/Unpartitioned Space [basic disk]/[unallocated space]/199940096*Carved [33206272].zip
<input type="checkbox"/>	Carved [33198080].zip	image.001/Unpartitioned Space [basic disk]/[unallocated space]/199940096*Carved [33198080].zip
<input type="checkbox"/>	Carved [33067008].zip	image.001/Unpartitioned Space [basic disk]/[unallocated space]/199940096*Carved [33067008].zip

Figure 7 Source information of recovered documents on Unpartitioned Space.

It can be seen in Fig. 8 that the names of the documents are Carved, and the extensions do not match their categories when the data on Unpartitioned Space is examined. If we need detailed information on the data obtained as carved, metadata records should be checked.

File List							
✓	Name	Category	Accessed	Modified	Created	P-Size	L-Size
<input type="checkbox"/>	Carved [33214464].zip	Excel 2016	n/a	n/a	n/a	n/a	8192 B
<input type="checkbox"/>	Carved [33206272].zip	Excel 2016	n/a	n/a	n/a	n/a	6604 B
<input type="checkbox"/>	Carved [33198080].zip	Excel 2016	n/a	n/a	n/a	n/a	6178 B
<input type="checkbox"/>	Carved [33067008].zip	Microsoft Word 2016 XML	n/a	n/a	n/a	n/a	11,41 KB

Figure 8 Metadata information of recovered documents on Unpartitioned Space.

In search of date information of the documents, it is shown in Figure 8 that these data cannot be recovered. When the processes in the scenario were examined, it was understood that the FTK forensic tool detected MBR partitioned primary partition (Volume_B) but did not show the partition information about the extended Partition, Volume_E. Also, information such as file type, name, and time information of the documents in Volume_E obtained by data scraping could not be reached.

4.3.2. Analysis using Encase software

Sections of the test disk obtained using Encase software are shown in Figure 9.

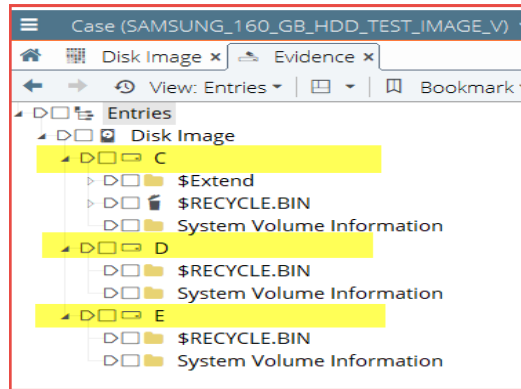


Figure 9 Section structures obtained with Encase

It is understood from Table 3 that there are *three* partition structures belonging to the test disk, and the total size of the partitions is 22.1 GB + 24.4 GB + 19.5 GB = 66 GB.

Table 3 Partition information obtained using Encase of the test disk

ID	Type	Start Sector	Total Sectors	Size
07	NTFS	2.048	46.336.000	22.1 GB
0c	FAT32X	107.778.048	51.200.000	24.4 GB
0c	FAT32X	158.980.033	40.960.000	19.5 GB

When the partition information of the test disk obtained with Encase is examined in Table 3; The total size of the *three* partitions detected was calculated as 66 GB. It is understood from Table 1 that the total disk size is 149.1 GB. When the total size of the available partitions and the total size of the test disk are compared, it is seen that Physical Disk Size is \neq Partitions Size. In this case, it is understood that the sections should be examined manually using the start and end sectors.

MBR records are kept in sector 0 in the partitions configured with MBR, and "sectors between 0-2047" are reserved as MBR area. For this reason, the 1st partition starts in the 2048 th sector, and the department signature information is at the end of this sector. The sum of the first partition start sector and the total number of sectors will give us the start part of the *two* partitions. As a result of the calculation, it is seen that it is $2048 + 46.336.000 = 46.338.048$, and it should be partition starting from 46.338.048 sector. However, when the partition table was examined, it was understood that there was no such partition. This situation shows us that there is a lost partition.

The Disk View feature of the encase forensic tool was used to see the 46338048 startup sector of the lost partition. It should be confirmed that the sector specified as a result of the addition is correct. 55AA signature value of the NTFS file system was seen in the last bytes of the sector in Figure 10.

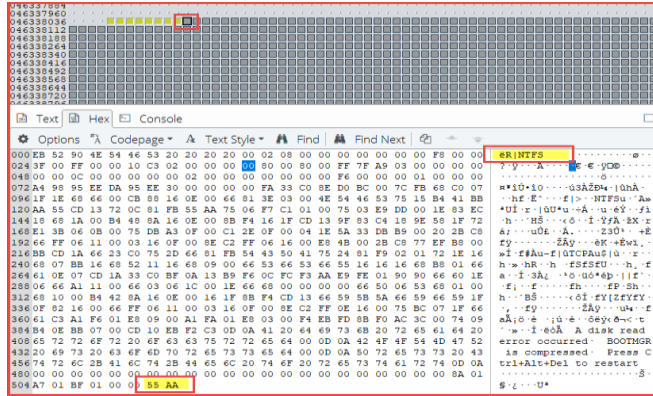


Figure 10 Signature of 55AA at the end of 46338048 sector and NTFS file system tag.

In the initial sector of 46338048 lost partitions, adding partition was performed. Thus, the contents of the lost partitions that were not visible at the beginning but as a result of calculations can be accessed.

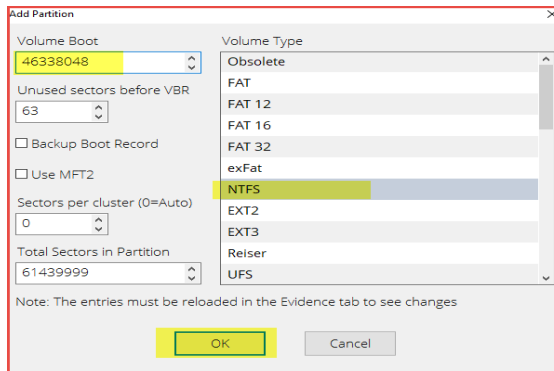


Figure 11 46338048 section information within the sector.

Manual addition is determined according to the starting sector determined in Figure 11. It is possible to access lost partition information after entering file type and start sectors.

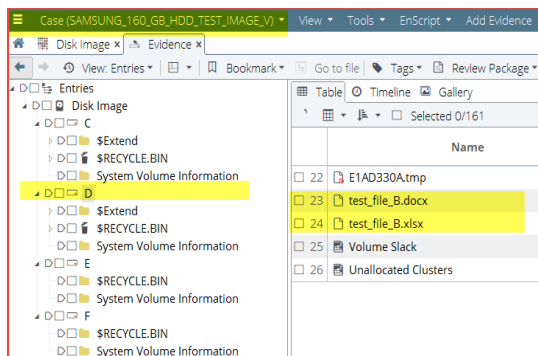


Figure 12 File structure of the disk partitions after adding the recovered primary partition of the test disk and the labeled documents in the partition

After performing the specified operations, the partition table view changed as in Table 4. However, the total number of sectors still does not match the number of sectors of the disk. Therefore, calculations are required.

Table 4 Partition table view after adding the recovered partition of the test disk

ID	Type	Start Sector	Total Sectors	Size
07	NTFS	2,048	46.336.000	22.1 GB

00	Recovered	46.337.985	61.440.062	29.3 GB
0c	FAT32X	107.778.048	51.200.000	24.4 GB
0c	FAT32X	158.980.033	40.960.000	19.5 GB

The partition table obtained after adding the lost partition is shown in Table 4. When the total size of the sections determined in this section table is calculated, 22.1 GB + 29.3 GB + 24.4 GB + 19.5 GB = 95.3 is obtained. When the total size of the available partitions and the total size of the test disk are compared, it is seen that Physical Disk Size is ≠ Partitions Size. As a result of this calculation, it is concluded that there are other lost parts. According to the calculation, a new lost partition asset has emerged. This situation shows the existence of at least 5 sections. In this case, it is concluded that there is an extended partition on the test disk.

In the future operations, the lost part should be made considering that it is an extended part. After the existence of the lost partition was understood, the determination process of the starting sector of the lost partition was started. According to the section table in Table 4, it is expected that the 4th partition will give the 5th partition total of the start sector and the total sector. When the calculation related to the 4th partition is realized, it is understood that 158.980.033 + 40.960.000 = 199.940.033. However, since the 4th and the lost partition are extended sections, the 63 sectors reserved for Volume Boot Record should be added to this total after 2048 sectors are added to this sector. As a result, the starting sector of the lost partition was calculated as 2048 + 63 + 199.940.033 = 199.942.144.

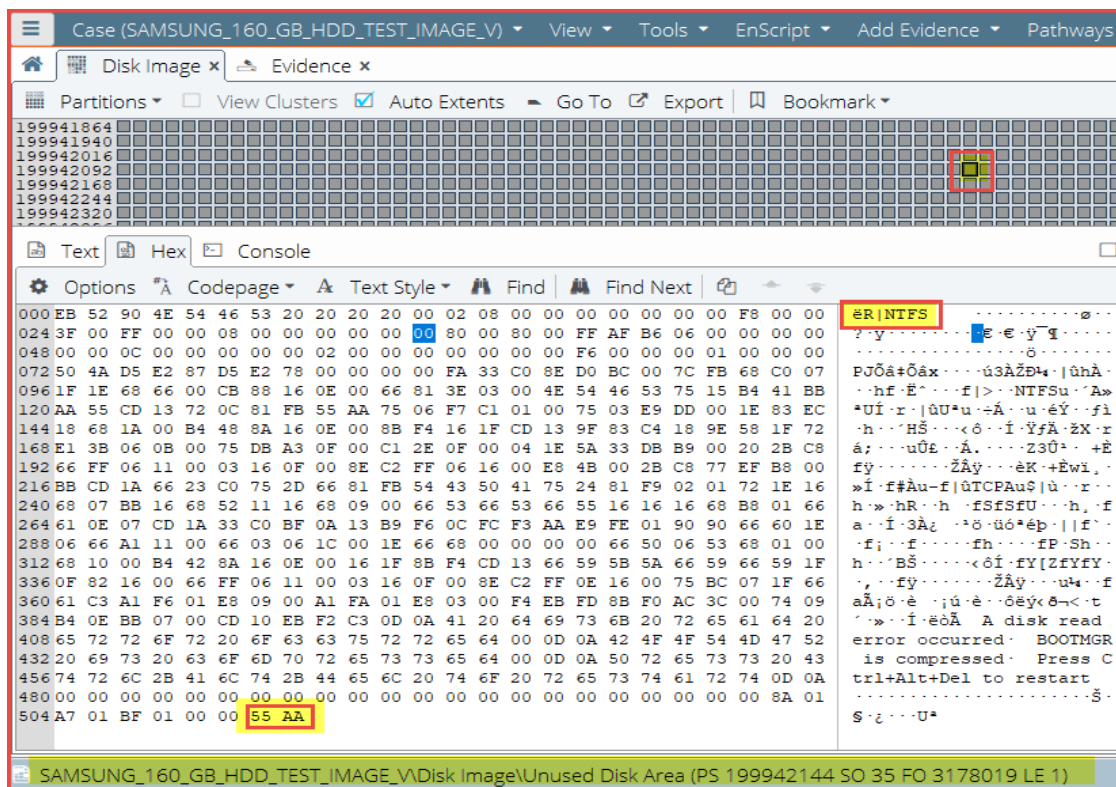


Figure 13 Section signature of 55AA at the end of 199942144 sector

Figure 13 shows the 55AA signature value of the NTFS file system in the last bytes of the sector. Thus, it is understood that there is one more section content. In the next stage, manual section addition was performed. The process for the addition is shown in Figure 14.

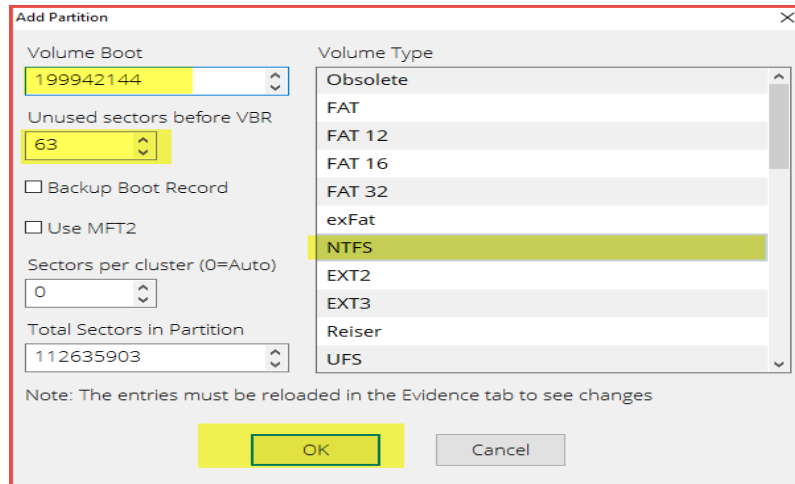


Figure 14 199942144 section information held within the sector.

After entering the file type and initial sectors, it is possible to access the lost partition information and its contents.

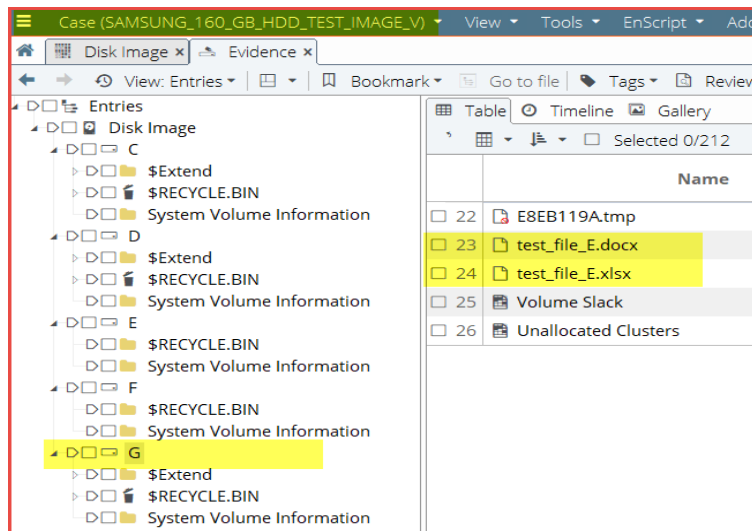


Figure 15 File structure of the disk partitions after the recovered extended partition of the test disk is added and the labeled documents in the partition.

After all manual additions were performed, all of the sections that were not visible at the beginning but deleted were accessed, and the lost partition information was reached. Table 5 shows the latest partition information.

Table 5. Partition table view after recovered partitions

ID	Type	Start Sector	Total Sectors	Size
07	NTFS	2.048	46.336.000	22.1 GB
00	Recovered	46.337.985	61.440.062	29.3 GB
0c	FAT32X	107.778.048	51.200.000	24.4 GB
0c	FAT32X	158.980.033	40.960.000	19.5 GB
00	Recovered	199.942.081	112.635.966	53.7 GB

As a result of the examination carried out with the Encase forensics tool, all the section structures were not shown directly to the user. However, encase allows the specialist to add a manual partition structure. Therefore, as a result of the size mismatch, the investigator should search the section signature

information. Then, it is necessary to add a manual section by looking at the section start sector and total size information. In this case, all section structures, location information, and metadata data could be accessed.

4.3.3. Analysis using Magnet AXIOM Examine software

Magnet AXIOM program is one of the programs widely used in forensics science. Figure 16 was obtained when the image file obtained from the test disk was opened with the magnet program.

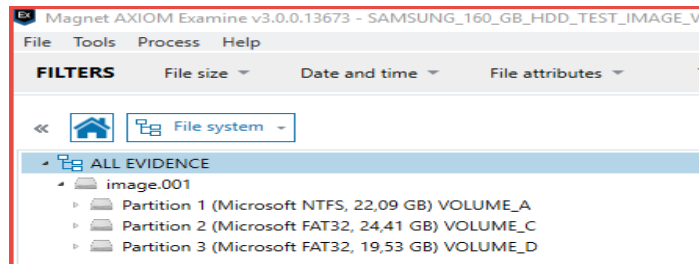


Figure 16 Section information detected by Magnet Axiom software

While the total disk size of the test disk should be 149.1 GB, the total size of the partitions detected by Magnet appears to be 22.09 GB + 24.41 GB + 19.53 GB = 66.03 GB.

As a result, since Physical Disk Size \neq Partitions Size, the existence of the lost partition structure is considered. Figure 13 shows that primary and secondary Partition structures are not recovered by the software. Since the forensics tools perform the sector-based rescue, document files belonging to the deleted sections are searched on the unpartitioned areas. As a result of the search, Word documents that were previously labeled with the VOLUME tag, whose name and path information could not be found, were identified and shown in Figure 17. The findings show that the data cannot be found in the correct location and content. For this reason, it is not possible to obtain content that may be evidence. This situation may affect the forensics process negatively.

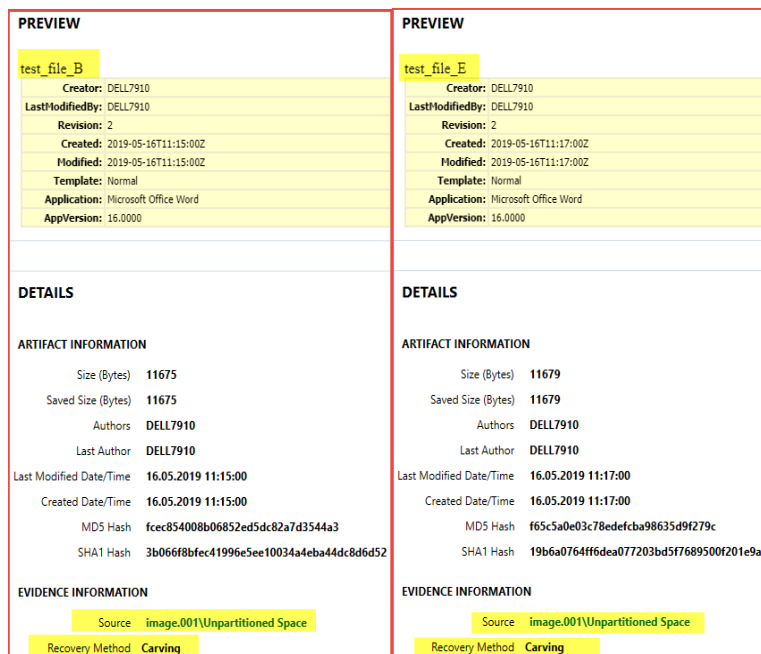


Figure 17 Metadata information of recovered documents on Unpartitioned Space.

Considering the findings obtained with Magnet Axiom, it was seen that no content related to primary and extended section information was available. The software does not offer the ability to add an

external partition. As a result of data carving, initial test data could be accessed with different names, but location information could not be accessed.

4.3.4. Analysis using Autopsy 4.11.0 software

Autopsy program is an open-source licensed and widely used forensics tool that can perform many operations manually. The partition structures of the test disk were obtained with Autopsy as shown in Figure 18. In the figure, Vol3 corresponds to the Volume_A that we initially configured, and Vol9 corresponds to the Volume_E partitions. According to the section descriptions, the field type is marked as Unallocated. Although Volume_B and Volume_E were initially deleted, only the primary partition has been recovered, but the data in the secondary extended partition have not been accessed.

Name	ID	Starting Sector	Length in Sectors	Description	Flags
vol1 (Unallocated: 0-2047)	1	0	2048	Unallocated	Unallocated
vol2 (NTFS / exFAT (0x07): 2048-46338047)	2	2048	46336000	NTFS / exFAT (0x07)	Allocated
vol3 (Unallocated: 46338048-107778047)	3	46338048	61440000	Unallocated	Unallocated
vol4 (Win95 FAT32 (0x0c): 107778048-158978047)	4	107778048	51200000	Win95 FAT32 (0x0c)	Allocated
vol7 (Unallocated: 158978048-158980095)	7	158978048	2048	Unallocated	Unallocated
vol8 (Win95 FAT32 (0x0c): 158980096-199940095)	8	158980096	40960000	Win95 FAT32 (0x0c)	Allocated
vol9 (Unallocated: 199940096-312581807)	9	199940096	112641712	Unallocated	Unallocated

Figure 18 Partition information detected by the Autopsy 4.11.0 software of the test disc

It is shown in Figure 19 that there is one recovered partition among the sectors Vol3 (Unallocated: 46338048-107778047), existing metadata information of the documents is recovered, but location information cannot be recovered.

Name	Modified Time	Change Time	Access Time	Created Time	Location
test_file_B.xlsx	2019-05-16 14:15:23 EET	2019-05-16 14:15:24 EET	2019-05-16 14:15:23 EET	2019-05-16 14:15:15 EET	
test_file_B.rar	2019-05-16 14:15:28 EET	2019-05-16 14:15:31 EET	2019-05-16 14:15:28 EET	2019-05-16 14:15:28 EET	
test_file_B.docx	2019-05-16 14:15:08 EET	2019-05-16 14:15:09 EET	2019-05-16 14:15:08 EET	2019-05-16 14:14:53 EET	

Figure 19 Tagged documents in Vol 3 Unallocated section

It can be seen in Figure 19 that only primary partition structures are recovered by the software, but secondary partition structures are not recovered. Forensic tools perform the sector-based rescue. When searching the document files of the deleted sections on the non-partitioned areas, documents whose name and path information are not known and which were previously tagged by us with the VOLUME label were identified. However, it was not possible to reach the information in which position the documents are or by whom.

4.3.5. Analysis using The Sleuth Kit (TSK) tools

TSK is open-source software that offers many libraries and command lines to the user to analyze disk images. TSK provides an analysis of the partition and file system data independent of the partition structure. When the test disk image was opened with the TSK tool mmls, the areas shown in Figure 20 were obtained.


```

root@kali:~/media/root/97EB-BDE2/SAMSUNG_160_GB_HDD_TEST_IMAGE_V# mmls image.001
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

    Slot  Start      End          Length      Description
000: Meta  0000000000  0000000000  0000000001  Primary Table (#0)
001: -----  0000000000  0000002047  0000002048  Unallocated
002: 000:000  0000002048  0046338047  0046336000  NTFS / exFAT (0x07)
003: -----  0046338048  0107778047  0061440000  Unallocated
004: 000:001  0107778048  0158978047  0051200000  Win95 FAT32 (0x0c)
005: Meta  0158978048  0312578047  0153600000  Win95 Extended (0x0f)
006: Meta  0158978048  0158978048  0000000001  Extended Table (#1)
007: -----  0158978048  0158980095  0000002048  Unallocated
008: 001:000  0158980096  0199940095  0040960000  Win95 FAT32 (0x0c)
009: -----  0199940096  0312581807  0112641712  Unallocated
    
```

Figure 20 Partition information obtained using TSK tools

It should be considered that there is a lost partition in the unallocated area between sectors 46338048-107778047 shown in Figure 20. In addition, it is seen that there is an extended partition area among sectors 158978048-312578047 (Win95 Extended). However, it is understood that the extended partition identified is between the sectors 158980096-199940095. Thus, it is observed that there is one lost partition between sectors 199940096- 312581807. The information obtained with the fsstat command is given in Figure 21.

```

root@kali:~/media/root/97EB-BDE2/SAMSUNG_160_GB_HDD_TEST_IMAGE_V# fsstat -o 0046338048 image.001
FILE SYSTEM INFORMATION
-----
File System Type: NTFS
Volume Serial Number: 30EE95DAEE9598A4
OEM Name: NTFS
Volume Name: VOLUME_B
Version: Windows XP
    
```

Figure 21 2. partition partition information deleted by using test disk with TSK tools

When fls command is used to detect the files in the section, *three* documents are accessed, which are labeled with the section tag shown in Figure 22.

```

root@kali:~/media/root/97EB-BDE2/SAMSUNG_160_GB_HDD_TEST_IMAGE_V# fls -o 0046338048 image.001
r/r 4-128-1: $AttrDef
r/r 8-128-2: $BadClus
r/r 8-128-1: $BadClus:$Bad
r/r 6-128-4: $Bitmap
r/r 7-128-1: $Boot
d/d 11-144-4: $Extend
r/r 2-128-1: $LogFile
r/r 0-128-0: $MFT
r/r 1-128-1: $MFTMirr
d/d 40-144-1: $RECYCLE.BIN
r/r 9-128-3: $Secure:$SD5
r/r 9-144-11: $Secure:$SDH
r/r 9-144-14: $Secure:$SII
r/r 10-128-1: $UpCase
r/r 10-128-4: $UpCase:$Info
r/r 3-128-3: $Volume
d/d 36-144-1: System Volume Information
r/r 44-128-3: test file B.docx
r/r 39-128-1: test file B.rar
r/r 45-128-3: test file B.xlsx
-r/r * 43-128-3: E1AD330A.tmp
V/V 256: $OrphanFiles
    
```

Figure 22 File information in deleted partition 2 obtained using the test disk with TSK tools

Since the extended partition table is in sector 0158978048, the content of the Extended partition table is shown in Figure 23 using the mmls command. Here, it is clearly seen that 2048 sectors are reserved areas.

```

root@kali:~/media/root/97EB-BDE2/SAMSUNG_160_GB_HDD_TEST_IMAGE_V# mmls -o 0158978048 image.001
DOS Partition Table
Offset Sector: 158978048
Units are in 512-byte sectors

    Slot  Start      End          Length      Description
000: Meta  0000000000  0000000000  0000000001  Primary Table (#0)
001: -----  0000000000  0000002047  0000002048  Unallocated
002: 000:000  0000002048  0040962047  0040960000  Win95 FAT32 (0x0c)
003: -----  0040962048  0312581807  0271619760  Unallocated
    
```

Figure 23 Extended Partition table information obtained by using Testdisk with TSK tools

It is understood that deleted extended partition started in 0199940096 sectors. Thus, $2048 + 199940096 = 199942144$ will give the sector where the sector information is available. The `fsstat` command was used to get detailed information about the deleted extended partition and the information seen in Figure 24 was obtained. It has been accessed that the file format of the Partition is NTFS, and the partition name is `VOLUME_E`.

```
root@kali: /media/root/97EB-BDE2/SAMSUNG_160_GB_HDD_TEST_IMAGE_V# fsstat -o 0199942144 image.001
FILE SYSTEM INFORMATION
-----
File System Type: NTFS
Volume Serial Number: 78E2D587E2D54A50
OEM Name: NTFS
Volume Name: VOLUME_E
Version: Windows XP
```

Figure 24 Partition information of deleted extended partition obtained by using the test disk with TSK tools

The `fls` command was used to detect the files in the section, and *three* documents that we labeled with the section tag in Figure 25 were accessed.

```
root@kali: /media/root/97EB-BDE2/SAMSUNG_160_GB_HDD_TEST_IMAGE_V# fls -o 199942144 image.001
r/r 4-128-1: $AttrDef
r/r 8-128-2: $BadClus
r/r 8-128-1: $BadClus:$Bad
r/r 6-128-4: $Bitmap
r/r 7-128-1: $Boot
d/d 11-144-4: $Extend
r/r 2-128-1: $LogFile
r/r 0-128-6: $MFT
r/r 1-128-1: $MFTMirr
d/d 39-144-1: $RECYCLE.BIN
r/r 9-128-8: $Secure:$SDS
r/r 9-144-11: $Secure:$SDH
r/r 9-144-14: $Secure:$SII
r/r 10-128-1: $UpCase
r/r 10-128-4: $UpCase:$Info
r/r 3-128-3: $Volume
d/d 36-144-1: System Volume Information
r/r 43-128-3: test_file_E.docx
r/r 38-128-1: test_file_E.rar
r/r 44-128-3: test_file_E.xlsx
-/r * 42-128-3: E8EB119A.tmp
V/V 256: $OrphanFiles
```

Figure 25 File information in deleted extended partition obtained using TSK tools.

The tools in the TSK allow many operations related to a disk image to be performed manually. Therefore, the process using this tool can also be evaluated as a manual analysis. As a result of the analysis, it has been seen that by performing the calculations correctly, all the partitions and files on the disk can be accessed together with the location information. However, it is important to know the disk structure and partitioning structures in order to ensure the full and correct use of the TSK.

4.3.6. Analysis using X-Ways Forensics tools

X-Ways Forensic analysis software is one of the most widely used forensics tools by digital forensics analysts. When the partition structures are checked by opening the image file obtained from the Test Disk with X-Ways Forensic Analysis software; There are five partition structures identified in Figure 26.

Name	Ext.	Size	Created	Modified	Record changed	Attr.	1st sector
Partition 1	NTFS	22,1 GB					2.048
Partition 2	NTFS	29,3 GB					46.338.048
Partition 3	FAT32	24,4 GB					107.778.048
Partition 4	FAT32	19,5 GB					158.980.096
Partition 5	NTFS	53,7 GB					199.942.144

Figure 26 Section information detected by X-Ways software

It has been determined that partition 2 and partition 5 on the test disk image are primary and extended partition structures created and deleted in the scenario part. When the detected deleted partition

structures are checked, it is shown that the data in the partition structures are completely recovered and presented in Figure 27.

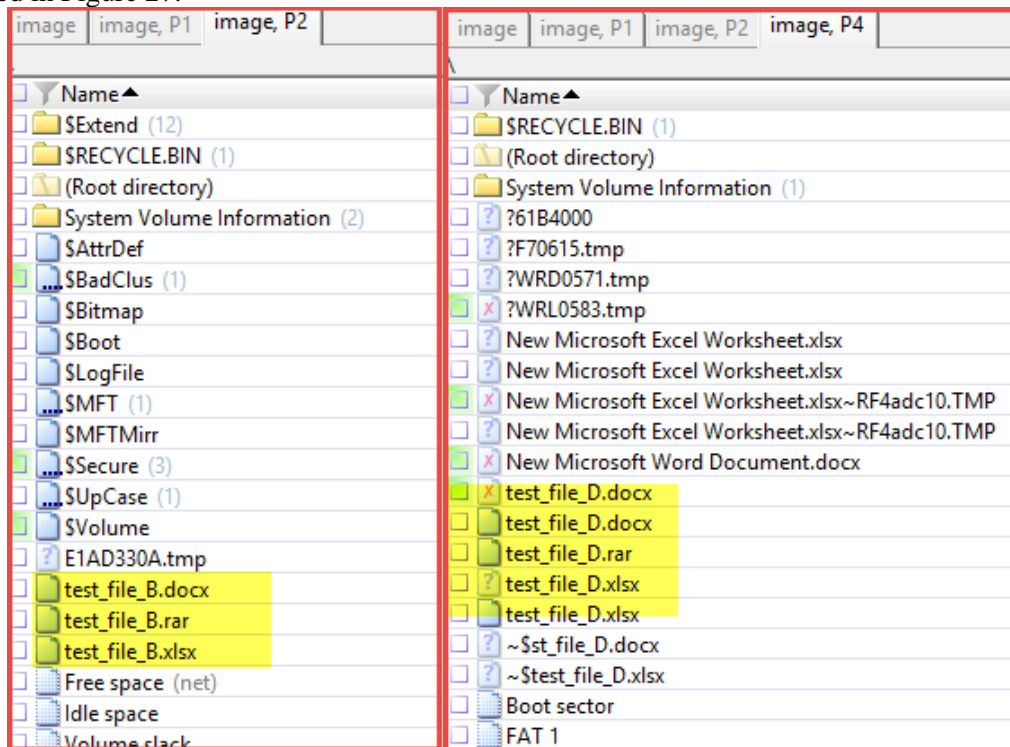


Figure 27 Data in deleted partitions.

When the test disk image was analyzed with X-Ways software, it was determined that deleted primary and extended partition structures were automatically detected and added by the software. In addition, it has been understood that the data in it was recovered without the need for any manual intervention.

4.3.7. Discussion

[15] demonstrated methods of obtaining data from disks partitioned with GPT. Similarly, [33] there are studies on obtaining data from UEFI disks. However, there are no studies showing the methods of obtaining data from MBR disks. However, it is still possible to encounter MBR disks.

In this work, a MBR disk analysis methodology has been presented by using a scenario and the widely used forensics tools. According to the obtained findings, these points are highlighted.

Benefits;

- A new scenario is presented to analysis MBR partitioned disks.
- Six analyses methods have been presented and results of them have been given.
- A comprehensive benchmark for MBR forensics analysis have been presented.

Limitation;

- The results are depended on the presented scenario. More scenarios can be analyzed for obtaining comprehensively results.

5. Conclusions

A sample scenario was prepared in the study. A hard disk image that is configured with MBR created in this scenario and deleted partition structures are used. Thus, the performance of 5 different forensics tools was tested on this disk image with this study. With this study, it was observed that some forensics tools could not recover the partition structures, while others could not recover the extended partition structures by recovering the primary partition structures. Some forensic tools were evaluated to allow manual examination of the section structures.

In this work, there is a primary deleted partition labeled VOLUME_B and deleted Extended partition labeled VOLUME_E. These sections only allow manual evaluation of TSK and Encase forensics tool section structures. Thus, the structure of the section and the documents in it have been recovered without any problems (Metadata, Location, etc.).

Magnet Axiom failed during partition recovery and was able to recover only document and metadata. The FTK can only recover deleted primary partitions. It was also able to recover documents without metadata data. In the analysis with Autopsy, similar results were obtained with the FTK forensics tool. X-Ways has the ability to automatically recover primary and extended partition structures. The findings obtained are given in Table 6.

Table 6. Comparison table of the findings obtained

	FTK	Encase	Magnet Axiom	Autopsy	TSK Tool	X-Ways
Partition 1 recovery	X	X	-	X	X	X
Partition 2 recovery	-	X	-	-	X	X
Extended partition recovery	-	X	-	-	X	X
Document recovery	X	X	X	X	X	X
Metadata information detection	X	X	X	X	X	X
Location information detection	-	X	-	-	X	X

The basic principles of digital forensic tools are the same. However, the analyst should not be content with the results obtained by the forensic tools and should manually intervene when necessary. For this reason, possible situations that may be caused by the complex structure of extended partitions were taken into account in the analysis of disks configured with MBR. In such a case, sample analysis was carried out by proposing an evaluation, and the results obtained from this analysis are presented in a comparison table. It should be confirmed that especially the initial size of the disk and the sum of sectors obtained from forensics software are the same. Otherwise, in the forensics investigation, the lost partitions will not be examined, and many contents that may be of evidence will not be available.

Our future directions are given as follows. This paper show advantages and disadvantages of the widely used digital forensics analyses tools. We are planning to propose a new successful MBR analyses tool for overcoming the disadvantages of the exist analyses analysis models.

Authors' Contributions: All authors contributed equally to the study.

Statement of Conflicts of Interest: There is no conflict of interest between the authors.

Statement of Research and Publication Ethics: The author declares that this study complies with Research and Publication Ethics.




References

- [1] C. Altheide and H. Carvey, *Digital forensics with open source tools*. Elsevier, 2011.
- [2] B. Carrier, "Open source digital forensics tools: The legal argument," *stake*, 2002.
- [3] R. Harris, "Arriving at an anti-forensics consensus: Examining how to define and control the anti-forensics problem," *digital investigation*, vol. 3, pp. 44-49, 2006.
- [4] G. Horsman, "Formalising investigative decision making in digital forensics: proposing the Digital Evidence Reporting and Decision Support (DERDS) framework," *Digital Investigation*, vol. 28, pp. 146-151, 2019.

- [5] T. Vidas, B. Kaplan, and M. Geiger, "OpenLV: Empowering investigators and first-responders in the digital forensics process," *Digital Investigation*, vol. 11, pp. S45-S53, 2014.
- [6] S. L. Garfinkel, "Digital forensics research: The next 10 years," *digital investigation*, vol. 7, pp. S64-S73, 2010.
- [7] Y. Guo, J. Slay, and J. Beckett, "Validation and verification of computer forensic software tools—Searching Function," *digital investigation*, vol. 6, pp. S12-S22, 2009.
- [8] A. C. Bogen and D. A. Dampier, "Unifying computer forensics modeling approaches: a software engineering perspective," in *First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05)*, 2005: IEEE, pp. 27-39.
- [9] M. Wundram, F. C. Freiling, and C. Moch, "Anti-forensics: the next step in digital forensics tool testing," in *2013 seventh international conference on it security incident management and it forensics*, 2013: IEEE, pp. 83-97.
- [10] G. C. Kessler, "Anti-forensics and the digital investigator," 2007.
- [11] A. Khan, U. K. Wiil, and N. Memon, "Digital forensics and crime investigation: Legal issues in prosecution at national level," in *2010 Fifth IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering*, 2010: IEEE, pp. 133-140.
- [12] R. W. Taylor, E. J. Fritsch, and J. Liederbach, *Digital crime and digital terrorism*. Prentice Hall Press, 2014.
- [13] M. Geiger, "Evaluating Commercial Counter-Forensic Tools," in *DFRWS*, 2005.
- [14] S. Garfinkel, "Anti-forensics: Techniques, detection and countermeasures," in *2nd International Conference on i-Warfare and Security*, 2007, vol. 20087, pp. 77-84.
- [15] B. J. Nikkel, "Forensic analysis of GPT disks and GUID partition tables," *Digital Investigation*, vol. 6, no. 1-2, pp. 39-47, 2009.
- [16] Y. Liu, J. Fang, and C. Han, "A new R-tree node splitting algorithm using MBR partition policy," in *2009 17th International Conference on Geoinformatics*, 2009: IEEE, pp. 1-6.
- [17] G. Horsman, "Tool testing and reliability issues in the field of digital forensics," *Digital Investigation*, vol. 28, pp. 163-175, 2019.
- [18] S. Bommisetty, R. Tamma, and H. Mahalik, *Practical mobile forensics*. Packt Publishing Ltd, 2014.
- [19] H. Mahalik, R. Tamma, and S. Bommisetty, *Practical Mobile Forensics*. Packt Publishing Ltd, 2016.
- [20] M. Al Fahdi, N. L. Clarke, and S. M. Furnell, "Challenges to digital forensics: A survey of researchers & practitioners attitudes and opinions," in *2013 Information Security for South Africa*, 2013: IEEE, pp. 1-8.
- [21] G. Horsman and D. Errickson, "When finding nothing may be evidence of something: Anti-forensics and digital tool marks," *Science & Justice*, vol. 59, no. 5, pp. 565-572, 2019.
- [22] K. Conlan, I. Baggili, and F. Breitingner, "Anti-forensics: Furthering digital forensic science through a new extended, granular taxonomy," *Digital investigation*, vol. 18, pp. S66-S75, 2016.
- [23] T. Göbel and H. Baier, "Anti-forensics in ext4: On secrecy and usability of timestamp-based data hiding," *Digital Investigation*, vol. 24, pp. S111-S120, 2018.
- [24] T. Haruyama and H. Suzuki, "One-byte modification for breaking memory forensic analysis," *Black Hat Europe*, 2012.
- [25] A. Prakash, E. Venkataramani, H. Yin, and Z. Lin, "Manipulating semantic values in kernel data structures: Attack assessments and implications," in *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2013: IEEE, pp. 1-12.
- [26] K. Lee, H. Hwang, K. Kim, and B. Noh, "Robust bootstrapping memory analysis against anti-forensics," *Digital Investigation*, vol. 18, pp. S23-S32, 2016.
- [27] S. Rekhis and N. Boudriga, "A system for formal digital forensic investigation aware of anti-forensic attacks," *IEEE transactions on information forensics and security*, vol. 7, no. 2, pp. 635-650, 2011.
- [28] A. Regenscheid, L. Feldman, and G. Witte, "NIST Special Publication 800-88 Revision 1, Guidelines for Media Sanitization," National Institute of Standards and Technology, 2015.
- [29] D. Hurlbut-AccessData, "Fuzzy Hashing for Digital Forensic Investigators," 2009.

- [30] K. Hausknecht, D. Foit, and J. Burić, "RAM data significance in digital forensics," in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2015: IEEE, pp. 1372-1375.
- [31] E. Casey and G. J. Stellatos, "The impact of full disk encryption on digital forensics," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 3, pp. 93-98, 2008.
- [32] B. Dolan-Gavitt, "Forensic analysis of the Windows registry in memory," *digital investigation*, vol. 5, pp. S26-S32, 2008.
- [33] D. Jeong and S. Lee, "Forensic signature for tracking storage devices: Analysis of UEFI firmware image, disk signature and windows artifacts," *Digital Investigation*, vol. 29, pp. 21-27, 2019.
- [34] M. Gruhn, "Forensic limbo: Towards subverting hard disk firmware bootkits," *Digital Investigation*, vol. 23, pp. 138-150, 2017.
- [35] F. Freiling, T. Glanzmann, and H. P. Reiser, "Characterizing loss of digital evidence due to abstraction layers," *Digital Investigation*, vol. 20, pp. S107-S115, 2017.
- [36] H. Kim, J. Kim, and T. Kwon, "A Study of Verification Methods for File Carving Tools by Scenario-Based Image Creation," *Journal of the Korea Institute of Information Security & Cryptology*, vol. 29, no. 4, pp. 835-845, 2019.
- [37] T. Sammes and B. Jenkinson, *Forensic computing*. Springer, 2007.
- [38] B. Carrier, *File system forensic analysis*. Addison-Wesley Professional, 2005.
- [39] K. Sindhu and B. Meshram, "Digital forensics and cyber crime datamining," 2012.
- [40] B. Carrier, "Defining digital forensic examination and analysis tools using abstraction layers," *International Journal of digital evidence*, vol. 1, no. 4, pp. 1-12, 2003.
- [41] K. Eckstein and M. Jahnke, "Data Hiding in Journaling File Systems," in *DFRWS*, 2005.
- [42] F. Ahsan, M. I. Lali, I. Ahmad, A. Ishaq, and S. Mohsin, "Exploring the effect of directory depth on file access for FAT and NTFS file systems," *ISTASC*, vol. 8, pp. 130-135, 2008.
- [43] J. Davis, J. MacLean, and D. Dampier, "Methods of information hiding and detection in file systems," in *2010 Fifth IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering*, 2010: IEEE, pp. 66-69.
- [44] W. Hong-biao, "The Features and Applications of FAT and NTFS File Systems [J]," *Computer Knowledge and Technology (Academic Exchange)*, vol. 6, 2007.
- [45] P. Nabity and B. J. Landry, "Recovering deleted and wiped files: A digital forensic comparison of FAT32 and NTFS file systems using evidence eliminator," ed: SWDSI, 2013.
- [46] M. Trawicki, "File Systems," 2002.
- [47] J. M. Rodriguez and J. Duffany, "Computer Forensics Tutorial Disk File Systems (FAT16, FAT32, NTFS)," POLYTECHNIC UNIV OF PUERTO RICO SAN JUAN, 2012.
- [48] N. Zhang, Y. Jiang, and J. Wang, "The Research of Data Recovery on Windows File Systems," in *2020 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, 2020: IEEE, pp. 644-647.
- [49] S. G. Taskin and E. U. Kucuksille, "Recovering Data Using MFT Records in NTFS File System," *Academic Perspective Procedia*, vol. 1, no. 1, pp. 448-457, 2018.
- [50] K. L. Rusbarsky and K. City, "A forensic comparison of NTFS and FAT32 file systems," http://www.marshall.edu/forensics/files/RusbarskyKelsey_Research-Paper-Summer-2012.pdf. *Fetched: July*, vol. 6, p. 2017, 2012.

A Smart Socket Equipped With IoT Technologies for Energy Management of Electrical Appliances

 Khaled ELORBANY¹,  Cüneyt BAYILMIŞ²,  Seda BALTA³

¹ Department of Computer Engineering, Sakarya University, Turkey; khaled.wagdy554@gmail.com

² Corresponding Author; Department of Computer Engineering, Sakarya University, Turkey; cbayilmis@sakarya.edu.tr

³ Department of Information System Engineering, Kocaeli University, Turkey; seda.balta@kocaeli.edu.tr

Received 18 January 2021; Revised 16 November 2021; Accepted 7 December 2021; Published online 31 December 2021

Abstract

Internet of Things is an ecosystem of devices that are capable of detecting, processing and communicating with the Internet. Recently, the Internet of Things applications that make our business and social life easier are spreading rapidly. In this study, a software controlled smart socket equipped with the Internet of Things technologies for energy management of electrical appliances and its development processes is presented. The implemented system consists of two main parts, hardware and software. In the hardware part, Wemos D1 with ESP8266 Wi-Fi module as a microcontroller for calculation and communication processes, an ACS712 current sensor to measure current consumption and a relay for on-off control is used. Schematic drawings of the hardware design are given to guide IoT-based application developers. The software part consists of Firebase cloud platform for data storage and a mobile application developed for smart socket control. The developed smart socket offers users many features such as on-off control over the internet, real-time and past monitoring of energy consumption, awareness of planned electrical outages, and overcurrent protection definition as software etc.

Keywords: Internet of Things (IoT), Smart socket, Remote monitoring and control, Power consumption.

1. Introduction

The communication of things over the Internet makes our lives easier and makes our business more efficient. For this reason, the Internet of Things has a wide range of applications from building control to health. The Internet of Things is a global network of devices capable of detecting, processing and communicating [1].

Smart sockets can be used to remotely control electrical devices. Control of smart sockets over the internet allows users to access their devices at any time from anywhere.

There are many studies on device control over the internet in the literature. A smart switch control system was developed that provides internet connection through the ESP8266 Wi-Fi module. The system developed has an on-off function only through a mobile application [2]. A smart socket was developed where electrical devices can be observed in on-off function and energy consumption over the Internet [3]. Another study also developed a simple algorithm application that shows the decrease in electricity consumption during peak hours using smart meter functions [4]. Real-time voltage, current and power consumption can be calculated, and a work was developed as a power meter to calculate electricity costs [5]. A new smart socket design was introduced to manage and reduce the demand for homes. With the smart socket tool offered, a study was developed that reduced the highest demand of the house by approximately 18% with voltage control for passive loads [6]. IoT technologies based smart energy consumption monitoring and energy management system of a house is presented. NodeMCU with ESP8266 Wi-Fi module, current sensor and blynk platform were used in the study. Energy consumption is given as quantity and cost [7]. A smart socket named IntelliPlugs having both WiFi and GSM communication unit has been developed. The IntelliPlugs can monitor in real-time and control selected electric devices [8]. The smart socket based on the cloud platform of the IoT is presented. With the cloud platform can remotely control and monitor the working status of smart socket [9]. To monitor and control energy consumption in a home IoT-enabled smart plug socket is proposed.

The proposed smart plug includes ESP32 Wi-Fi module. Energy consumption status can be shows and analyzes over internet [10].

This study presents design and implementation of the smart socket equipped with IoT technologies for energy management of electrical appliances.

The overall working architecture of the system performed is presented in Figure 1. As seen Figure 1, the system consists of smart socket, mobile application and Firebase cloud platform. The smart socket includes Wemos D1 microcontroller equipped with ESP8266 Wi-Fi module, ACS712 current sensor and role.

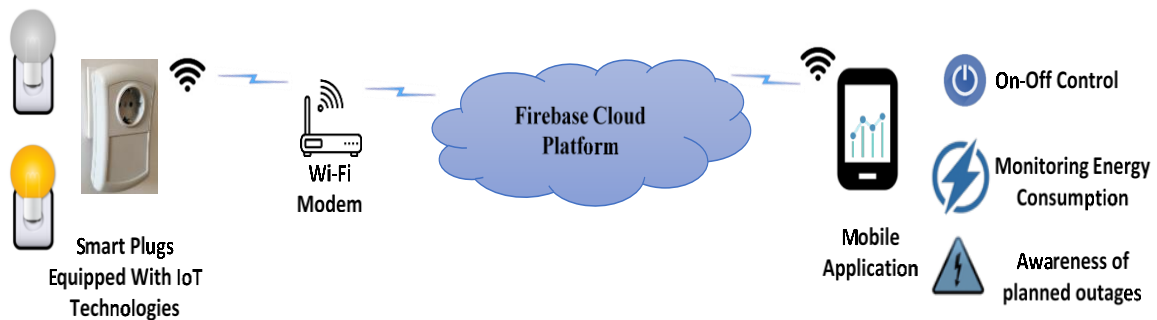


Figure 1 General System Architecture

Considering the short literature summary given above, the differences between this study and the studies that are mentioned in the literature are:

- Like studies in the literature, it allows monitoring of on-off and energy consumption through mobile application. In addition, the user can report historical energy consumption (weekly, monthly, etc...). With this feature, the user's future energy consumption profile can also be removed.
- The software of socket is designed to protect the device that is plugged in it (TV, refrigerator etc.) from overcurrent withdrawal.
- Smart socket tracks planned power outages on related websites and informs users via mobile application.
- The smart socket can automatically connect to the Wi-Fi network that the user chooses by scanning the Wi-Fi networks nearby without having any need to write any code. This feature revealed a product that the end user can use directly as plug&play.

The work of the developed system is summarized in the flow chart given in Figure 2. The implemented smart socket first scans for any available Wi-Fi networks nearby and connects to any saved Wi-Fi network. Then checks to see whether the socket is open or not. As long as the socket is working, it reads the withdrawn current value and checks the limit that is defined over the mobile app. The smart socket on / off control can be done with mobile application. Also, mobile app shows real-time and past energy consumption of the smart socket.

2. Design and Implementation of The Proposed System

2.1 Hardware

2.1.1 Circuit Setup

In this study, Wemos D1 Mini based on the Low Energy Energy ESP-12E Wi-Fi module is used. The embedded software is developed in the Arduino IDE environment. The developed smart socket circuit

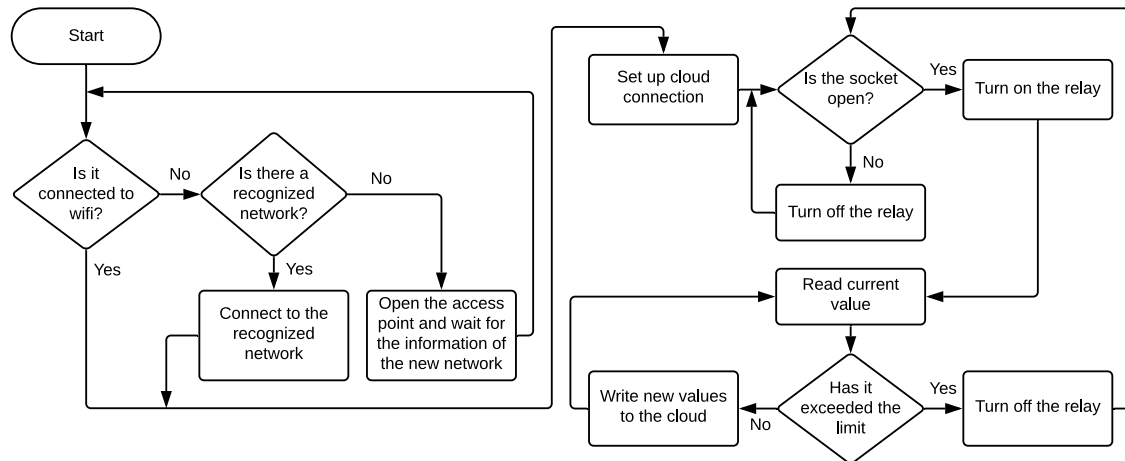


Figure 2 Embedded System Operating Principle

is installed directly in the existing socket box, the adaptor is providing supplement at range of 5v to the used components. The ACS712 current sensor is used to measure the AC current, calculates one ampere for each 185mV. After the raw value is taken from the sensor, the RMS formula is applied to obtain the AC current value. The ACS712 current sensor, which is produced by Allegro firm, produces an analog signal that is measured by the external ADC module (ADS1115). The ADS1115 module is 16-bit (1-bit sign bit) and its input voltage range is 2.0-5.5V. Switching circuit is applied using relay, transistor and resistance components to switch high-voltage AC. The circuit of the smart socket equipped with IoT technologies is presented in Figure 3 and the electrical circuit diagram is represented in Figure 4.

2.1.2 Details of The Used Modules

- Microprocessor (Wemos D1 Mini):

The Wemos D1 Mini is an open source IoT platform based on the esp8266-12F module. At very affordable prices, it is like "Small Arduino with Wi-Fi". It runs with range 3.3-5 volts. It has 16 general-purpose pines and its internal ADC works with 10-bit sensitivity.

- Current Sensor (ACS712):

ACS712-05B is a linear current sensor attached to magnetic effect field. This version allows two-way current input up to ± 5 amps. And gives 5V analog voltage (185 mV/A) as an output.

- Switching Circuit:

It is the circuit we have installed using electronic components as (relay, resistance and transistor) to control the high voltage AC current.

- External ADC (ADS1115):

The external ADC is used because the output of the ACS712 current sensor we used to measure AC current is 5V and the internal ADC in the ESP8266-12F based Wemos D1 Mini microprocessor only accepts up to 3.2V. This ADC works in accordance with the ACS712 current sensor, which we use because it is 16-bit (1-bit mark bit) and input voltage range $\pm 0.3V$.

2.1.3 Circuit Usage and Working Scenario

The normal AC 220 volts are reduced to 5 volts via the adapter, thus feeding the microprocessor. The same current passes through the ACS712 sensor and reaches to the connected device. The sensor measures the current and generates an analog signal and sends it to the external ADC (ADS1115). The ADC reads that signal and send the output to the microprocessor. The microprocessor takes that data

and sends it to the cloud or turns the switching circuit on and off according to the data it receives. When the socket is started, the "WifiManager" software library is used for automatic network detection.



Figure 3 Smart Socket

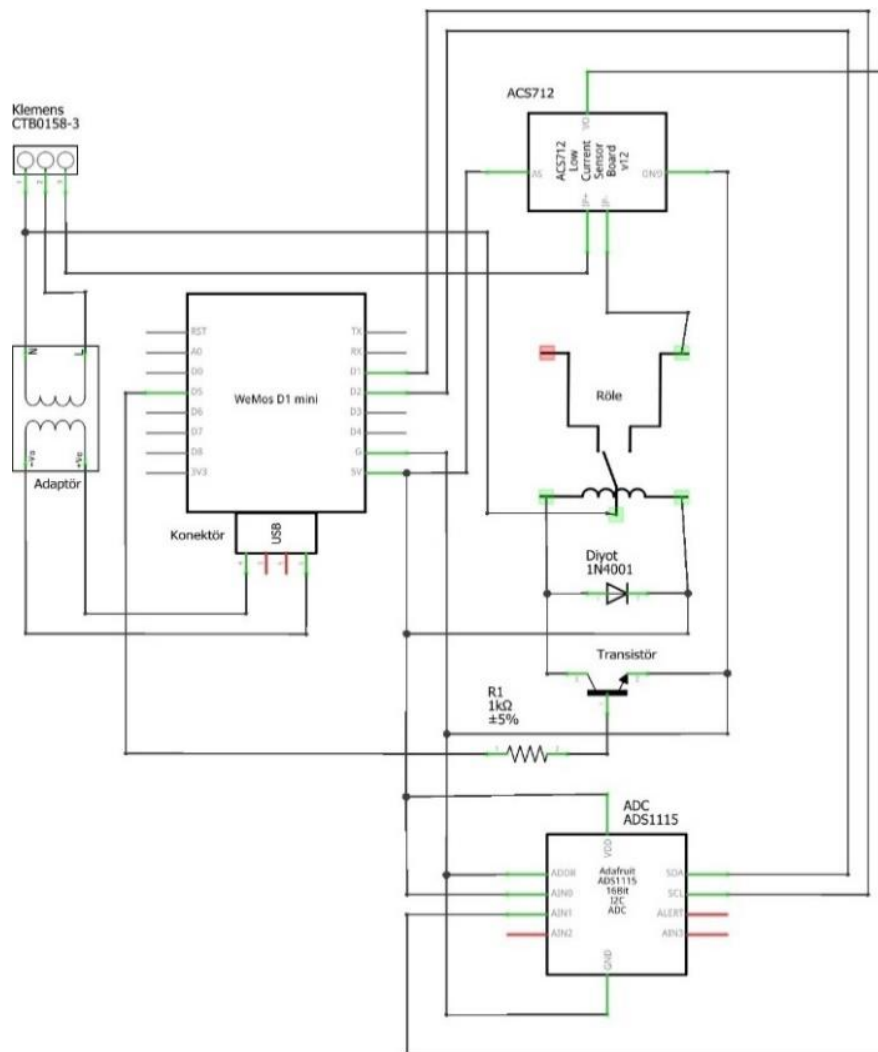


Figure 4 Schematic Drawing

2.2 Software

2.2.1 Software Design

The software includes embedded software for the Wemos D1 Mini development card and the android mobile app. Embedded software consists of 2 basic parts: the AC signal processing and the cloud functions. The RMS formula is used to process the AC signal wave. Firebase, a Cloud-based platform developed by Google, is used for cloud operations. Firebase provides data storage in real time.

2.2.2 Cloud Operations

In this study, Firebase, a cloud-based platform developed by Google is used. Firebase provides data storage in real time. Using a NoSQL type database, it stores and syncs data between users and devices in real time. Updated data is synchronized between connected devices in milliseconds, and if our app is offline, the data is stored and synchronized when there is a network connection.

2.2.3 Signal Processing

The "RMS" value is the square root of the average value of the squares of snapshots. Also, it can be defined as "AC power quantity that produces the same heating effect as an equivalent DC power".

$$I_{sum} = \sqrt{I_1^2 + I_2^2 + I_3^2 + \dots + I_{n-1}^2 + I_n^2} \quad (1)$$

For the calculation of the RMS current value, by using the scale, the sensitivity and the reference voltage value of the external ADC (ADS1115) the I_{RMS} value is obtained as a result after processing the collected raw value.

$$I_{RMS} = I_{sum} / ADC \text{ scale} * \text{reference voltage} / \text{current sensor sensitivity} \quad (2)$$

The block code for the current measurement is shown below.

Code 1 Block code for the current measurement

```

1 float getCurrentAC() {
2     float vRef = (ads.readADC_SingleEnded(0) * 0.1875) / 1000;
3     Serial.print(String("V = ") + vRef + " Vs,")
4     uint32_t period = 1000000 / frequency;
5     uint32_t t_start = micros();
6     uint32_t Isum = 0, measurements_count = 0;
7     int32_t Inow;
8
9     while (micros() - t_start < period) {
10         Inow = zeroRef - ads.readADC_SingleEnded(1);
11         Isum += Inow * Inow;
12         measurements_count++;
13     }
14     float Irms = sqrt(Isum / measurements_count) / adcScale
15                 * vRef/ sensitivity;
16     return Irms;
17 }
```

2.2.4 Wi-Fi Connection

In the first step, it searches for a known saved network, connects to it again if it is found. If it is not found, the microprocessor works as an access point that is easily configured by a name and maybe a password. When any device is connected to the access point, it is lead to a web server to search for the new network. When the object is defined, the constructor function in the library written in C++ is called

and tries to search and connect to the networks it knows, if it does not find it, it moves to the next line and calls the function “autoConnect ()” that opens the accesspoint, and the name between the parentheses is used as the parameter. Wi-Fi connection processes are presented in Figure 5.

2.2.5 Tracking of Planned Electrical Power Outages

Pre-learning power outages allows us to prevent malfunctions that may occur in devices. Planned power outages in our country are announced on official and unofficial websites. With the “<https://guncelkesintiler.com/>” web site, electricity outage information in many cities can be accessed. In this study, electricity outage information is checked at certain intervals by using task timer in mobile app.

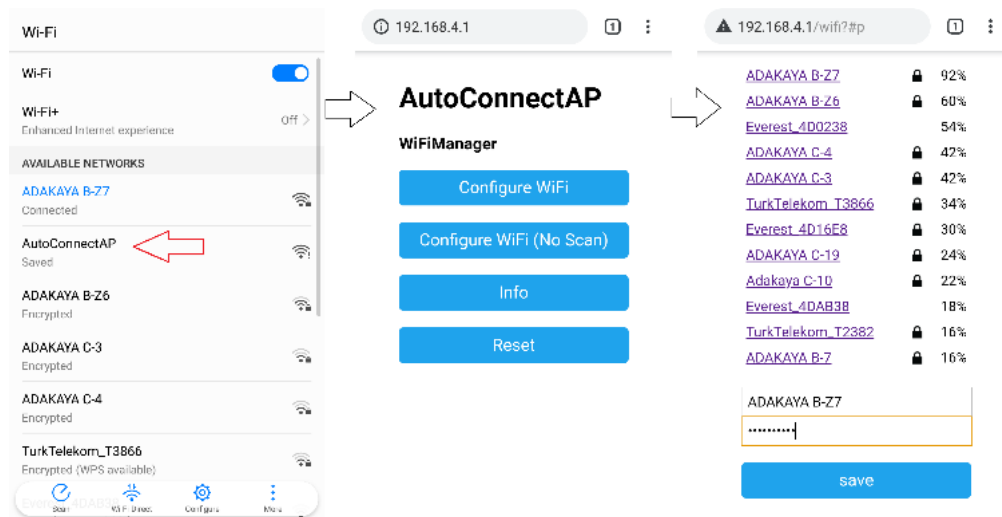


Figure 5 - Wi-Fi Connection Process

2.2.6 Mobile App

Android is a giant operating system developed by Google and used by millions of people. The mobile app was developed in the Android Studio IDE environment. The IDE allows us to test the application developed using an emulator. The application consists of two parts: design, and background running codes. For the design XML language, and for the background codes we used Java programming language. The design is used to place blocks on the interface. The background Java codes contain the code portion that specifies the operations of the blocks we design. Thanks to the mobile app, the socket is monitored live. The Android operating system version we use in the project is Nougat, which is version 7. The software development version is API Level 29. The interfaces of the application is presented in Figure 6. Figure 6-a shows main screen of mobile application. Choose location for planned electrical outages and definition overcurrent limit screen is given in Figure 6-b. Figure 6-c shows real-time and past monitoring of energy consumption of smart socket.

3. Conclusions

In order to control and monitor electrical appliances, designed and implemented a smart socket based on IoT technologies is presented in this study. The smart socket has simple design and low cost. The system consists of smart socket, mobile application and Firebase IoT cloud platform. The smart socket offers many features. The user can easily open or close the socket through the mobile application. The current limit determined by the user, automatically causes the socket to turn itself off when it is exceeded. Thus, the socket offers extreme current protection. The user can monitor energy consumption through the graphics in the mobile app. Also, users can learn about planned power outages using the mobile app and they can protect their electric devices against possible malfunctions.

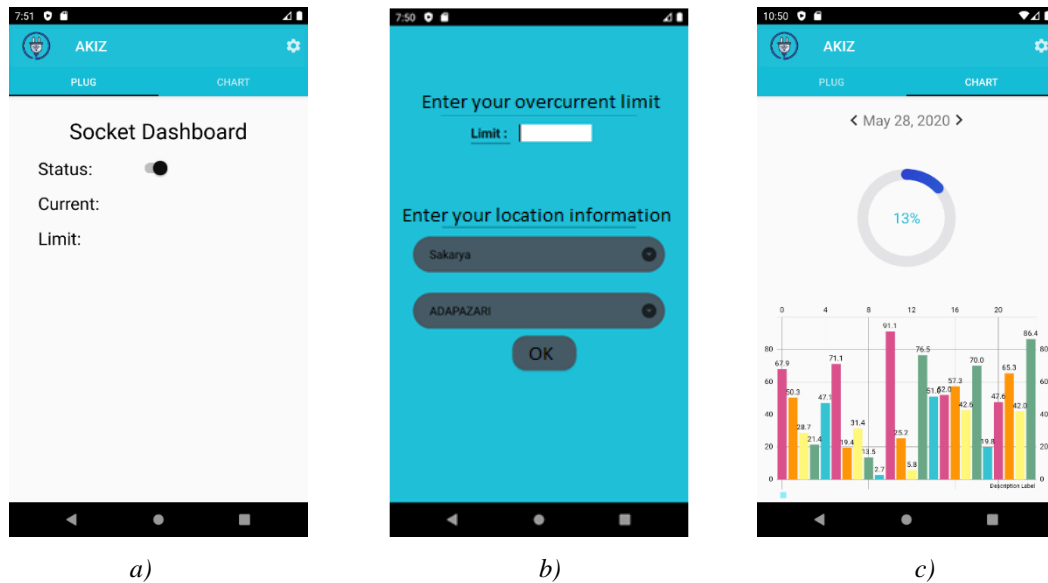


Figure 6 - Mobile App Interfaces (a-main screen, b-choose location for planned electrical outages and definition overcurrent limit screen, c-real-time and past monitoring of energy consumption)

Future works will be continued on this study. The use of artificial intelligence technologies in smart home and city applications is becoming widespread. We plan to add new features to smart plug presented in the study and integrate it with AI technologies. As a future study, the analysis of the energy consumption data obtained using smart socket can be performed by machine learning algorithms. Thus, the user's profile can be obtained and the best electric tariff can be offered to the user. In addition, users can contribute to savings.

References

- [1] C. Bayılmış and K. Küçük, *Nesnelerin İnternet'i: Teori ve Uygulamaları*, İstanbul, Papatya Bilim Yayınevi, 2019.
- [2] S. P. Makhanya, E. M. Dogo, N. I. Nwulu and U. Damisa, "A Smart Switch Control System Using ESP8266 Wi-Fi Module Integrated with an Android Application," *Proc. - IEEE 7th Int. Conf. on Smart Energy Grid Eng.*, pp. 125-128, 2019.
- [3] Y. Thongkhao and W. Pora, "A Low-cost Wi-Fi Smart Plug with On-off and Energy Metering Functions," *Proc. - 13th Int. Conf. on Electrical Eng./Electronics, Comp., Telecom. and Inf. Tech.*, pp. 1-5, 2016.
- [4] H. Morsali et al., "Smart Plugs for Building Energy Management Systems," *Proc. - 2nd Iranian Conf. on Smart Grids*, pp. 1-5, 2012.
- [5] A. A. Mohammed and E. Erçelebi, "Development of Embedded System for Making Plugs Smart," *Proc. - 4th Int. Conf. on Elect. and Electronic Eng.*, pp. 148-152, 2017.
- [6] O. Elma and U. S. Selamoğlu, "A Home Energy Management Algorithm With Smart Plug for Maximized Customer Comfort," *Proc. - Int. Conf. on Electric Power and Energy Conv. Syst.*, pp. 1-4, 2015.
- [7] M. Taştan, "Internet of Things based Smart Energy Management for Smart Home," *KSII Trans. on Int. and Inf. Sys.*, vol. 13, no. 6, pp. 2781-2798, Jun. 2019, doi:10.3837/tiis.2019.06.001.
- [8] R. C. Castro et al., "IntelliPlugs: IoT-based Smart Plug for Energy Monitoring and Control Through WiFi and GSM," *Proc. - 12th Int. Conf. on Hum., Nano., Inf. Tech., Comm. and Cont., Envir., and Manag.*, pp. 1-4, 2020.
- [9] J. Guan, K. Zhang, X. Zhong, "Design of Intelligent Socket Based on Cloud Platform of IoT," *Proc. - IEEE Int. Conf. on Art. Intell. and Comp. App.*, pp. 148-151, 2021.
- [10] Md. Rokonzaman, et al., "Design and Implementation of an IoT-Enabled Smart Plug Socket for Home Energy Management," *Proc. - 5th Int. Conf. on Smart Grid and Smart Cities*, pp. 50-54, 2021.

Rain Rate and Rain Attenuation Prediction for Satellite Communication in Ku Band Beacon over TURKSAT Golbasi

 Yasin Burak Kaya¹,  Umit Cezmi Yılmaz²

¹Corresponding Author; Türksat Satellite Communication and Cable TV Operation AS. Satellite Control Center, 06839, Golbasi, Ankara, Turkey; ybkaya@turksat.com.tr

²Türksat Satellite Communication and Cable TV Operation AS. Satellite Control Center; cyilmaz@turksat.com.tr

Received 4 August 2021; Revised 04 December 2021; Accepted 13 December 2021; Published online 31 December 2021

Abstract

The most effective technique used to measure rain attenuation is experimentally monitoring the received signal strength of satellite beacon. A satellite beacon is a signal which does not modulated and sends to the ground in constant frequency with a specifically designed power. Beacon signals are used by ground station antenna users to track the satellite easily. The satellite operators generally choose beacon signal at the Ku Band frequency band since this band is more resistant to rain attenuation. Ku band satellite system performance describe by the contribute of rainfall rate and rain attenuation. This paper includes the comparison of rain attenuation in Ku band beacon and takes as a reference ITU – 838 model caused by rain fall rate in TURKSAT premises. It is compared by calculating that how signals affected from rain between 2012 and 2019 observations for a TURKSAT satellite and in this measurement, both theoretical formula and data are used.

Keywords: TURKSAT, rain attenuation, ku band beacon signal, ITU-838

1. Introduction

The most dominant sources of attenuation in satellite communication systems come from atmospheric losses. These losses arise from rain, snow and dense clouds individually or as a combined effect. Rain attenuation becomes more dominant loss for carrier frequencies above 10 GHz. Precipitation rates (determining the amount of rain) and cumulative precipitations show differences from region to region on earth. Calculating and predicting path losses are therefore not accurate without comprehensive meteorological information obtained locally. In ITU-R proposal [1], earth is divided into regions according to precipitation rates. However, precipitation rates vary because of geographical locations of countries and climate conditions [2]. There might be some discrepancies on precipitations between ITU-R proposals and real case, considering that the ITU-R should have been used more conservative approach. That's why, the operators are motivated to revisit the rain attenuation models on their region. Rain precipitation model of ITU-R designated Turkey region 'K'. The main motivation for this work is to analyze whether there is differences between region K values versus long term rain statics maintained under all local weather conditions. The data used in this study belongs to the Golbasi /Ankara /Turkey for the dates starting from June 2012 up to September 2019 at yearly basis. Ku band beacon signals coming from an operational TURKSAT GEO satellite are used. They are examined according to weather conditions, space propagation (or path) losses, cable losses and atmospheric losses. This paper is organized as follows. In section 2, link budget of path losses of the TURKSAT Ku band beacon signal is provided. Section 3 presents rain losses comparing ITU-R-838 and TURKSAT data. In section 4, results are summarized, and a solution method is proposed to compensate for the differential errors that might endanger the reliability and even the continuity of communication.

2. Link Budget of TURKSAT Satellite Ku Band Beacon

Satellite operators broadcast at high frequencies such as C, Ku or Ka band. Ku band used most commonly to minimize such attenuations. Frequency of the C-band is small compared to the Ku and Ka bands, and the antenna diameter is quite large compared to the other frequencies used. In C-band, more power is required to transmit the signal to the satellite. This is not a preferred frequency. As the frequency increases, the antenna diameters will decrease proportionally. For satellite operations, larger antennas with tracking capabilities are used in Ku and Ka band and this kind of antennas cost more comparing the others [3], [4], [5], [6]. One of the biggest motivation for calculating satellite losses due rain as accurate as possible is financial gain especially on the uplink side. For this study, two different antennas were selected and their parameters are given in Table 1.

Table 1 Türksat Satellite Beacon link budget information for Two Antennas [6]

Parameter	Symbol	Antenna-A	Antenna-B
Frequency:	f	11.120 GHz	11.120 GHz
Diameter:	d	7.2 m	13.2 m
Efficiency	η	0.6	0.6
Satellite EIRP	EIRP	15 dBW	15 dBW
Free Space Loss Downlink	LS	204.88 dB	204.88 dB
Receiver Gain	Gr	56.26 dBW	61.53 dBW
Receiver System Temp	Ts	127K	125.8 K
Receiver G/T	G/T	35.26 (dB/K)	40.30 (dB/K)
Received power	Pr	-103.62 dBm	-98.152 dBm
Received Power LNA	Pr (LNA)	-43.62 dBm	-68.152 dBm
Gain of LNA	GLNA	60 dBm	30 dBm
Bandwidth	BW	2 kHz	2 kHz
Gas Absorption Losses	Lg	1 dB	1 dB
Pol. and Misalign losses	Lp	1 dB	1 dB
Rain Rate	R0.01	16.35 mm/h	15.36 mm/h
Carrier to Noise Ratio	C/N0	70.28 (dB/Hz)	75.70 (dB/Hz)

3. Rain Fade Calculations

3.1. Based on ITU-R

Rain fade which is the main source of error in path loss calculations, changes with frequency, location, polarization and rainfall rate. The amount of loss due to rain can be calculated using

$$L_{RAIN} = \gamma_R D_S \quad (1)$$

Where L_{RAIN} is the rain loss (or attenuation) in dB, γ_R is the specific attenuation in dB/km, and D_S is the path length through the troposphere in km which is measured from the freezing point (zero-degree isotherm, the level at which it is assumed that all rain originates) in the atmosphere to the receiving antenna. To calculate the rain loss, latitude (ϕ) and longitude of the earth station should be known. In addition the antenna altitude of the station; h_s km, the frequency of operation, and the polarization of the signal should also be known. Schematic presentation of an Earth-space path is shown in Fig.1

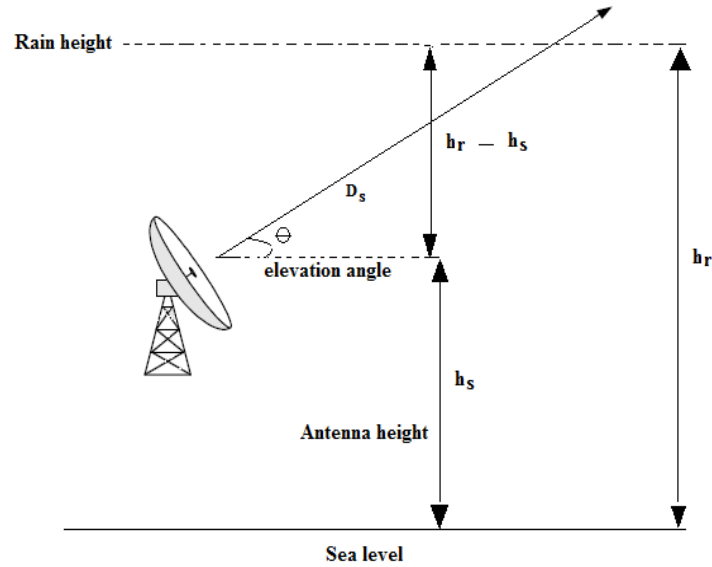


Figure 1. Schematic representation of an Earth-space path [7], [8], [9], [10]

$$D_s = \frac{(h_r - h_s)}{\sin\theta} \tag{2}$$

Where h_r is the rain height from the sea level, h_s is the distance of the satellite receiving antenna to the sea level, and θ is the elevation angle. ITU-R Recommendation P.839 relates rain height to location which is reproduced in Table 2.

Table 2 Rain height, h_r for different regions of the Earth [11]

Latitude ϕ	h_r (km)	Region
$\phi > 23$ North	$5 - 0.075 (\phi - 23)$	Northern hemisphere (except North America & Europe west of 35°W)
$0 < \phi < 23$ North	5	Northern hemisphere (except North America & Europe west of 35°W)
	$3.2 - 0.075 (\phi - 35)$	Northern hemisphere North America & Europe west of 35°W
$0 > \phi > 21$ South	5	Southern hemisphere
$21S > \phi > 71$	$5 + 0.1(\phi + 21)$	Southern hemisphere
71 South $> \phi$		Southern hemisphere

The elevation angle θ and the rain path D_s can be calculated as 43.145° and 3.8976 km respectively. Besides, Antenna-A was used between 2012-2015 whose h_s is 1086 meters, Antenna-B was used between 2016-2019 whose h_s is 1095 meters from the sea level. The elevation angle θ and the rain path D_s can be calculated as of 43.145° and 3.885 km respectively. Using equation the specific attenuation, γ_R can be found from for each year [6].

$$\gamma_R = kR^\alpha \tag{3}$$

where k (either k_H or k_V), and α (either α_H or α_V) are frequency-dependent parameters given in ITU-R Recommendation P-838 [8] as shown in the first three rows of Table 3. If the beacon signal is horizontally polarized then k_H , and α_H are used in equation (3).

Table 3 Frequency-dependent coefficients for estimating specific rain attenuation [7]

Frequency (GHz)	k_H	α_H	k_V	α_V
11.000	0.01772	1.2140	0.01731	1.1617
11.120	0.01840	1.2097	0.01790	1.1563

TURKSAT carrier frequency is 11.120 GHz [6] and corresponding; k and α factors that are calculated using the ITU recommendation P 838 [7]. The rainfall rate R is another parameter necessary to calculate the attenuation coefficient, γ_R . Table 4 links the rainfall rates to the percentage of the time it is exceeded in any year by rainfall regions of the world [5] as per ITU recommendation. It indicated before [5], Turkey falls in region K. For γ_R calculations, it is assumed that the rainfall rate that will only be exceeded 0.01% of the time for a satellite terminal in Turkey, Golbası (i.e. the rainfall rate that will give 99.99% availability). From that, the rain rate to be used as per ITU recommendation becomes 42 mm/h. TURKSAT is located in the central Anatolian region of Turkey, in the Gölbaşı of Ankara. It has 39°46' N and 32°49' E values as coordinates. Since the meteorology station in the center of Gölbaşı is 20 km between TURKSAT premises, measurements is taken with the meteorology equipment (Davis Vantage Pro2) in the center of the ground station. Since meteorology equipment and antennas are in the same place, it enabled us to get more accurate results. Annual rain rate and rain attenuation are calculated according to the data obtained from the measuring device located at the ground station [6].

Table 4 Rainfall rate R to the percentage of the time it is exceeded in any year by region [11]

%R time exceeded	A	B	C	D	E	F	G	H	J	K	L	M	N	P	Q
1.0	<0.1	0.5	0.7	2.1	0.6	1.7	3	2	8	1.5	2	4	5	12	24
0.3	0.8	2	2.8	4.5	2.4	4.5	7	4	13	4.2	7	11	15	34	49
0.1	2	3	5	8	6	8	12	10	20	12	15	22	35	65	72
0.03	5	6	9	13	12	15	20	18	28	23	33	40	65	105	96
0.01	8	12	15	19	22	28	30	32	35	42	60	63	95	145	115
0.003	14	21	26	29	41	54	45	55	45	70	105	95	140	200	142
0.001	22	32	42	42	70	78	65	83	55	100	150	120	180	250	170

3.2 Rain Fade Based on Long Term Meteorological Data at Golbası/Ankara

ITU-R P.618-12 recommends to use 42 mm/h rain rate for rainfall attenuation for region 'K'. Whereas, as can be seen from Table 5 and Table 6, smaller the rain rates are calculated by ITU-R-838 Model by using the measured values. The related tables show also the calculated values and actual values derived from antenna system. There are minor differences between the calculated and measured L_{RAIN} due to the differences in antenna system (i.e, diameter, rain sensor and rain blower).

Table 5 Rain rates for years Turksat Golbası region [6], [12], [13]

Year	Rain Rate at 0.01% (mm/hr)	E° Antenna-A	D _s (km)	Calculated L _{RAIN} (dB)	Measured L _{RAIN} (dB)	Difference (dB)
2012	9.54 mm/h	43.146	3.897	1.099	0.529	0.570
2013	10.21 mm/h	43.146	3.897	1.193	0.533	0.660
2014	16.35 mm/h	43.146	3.897	2.110	0.912	1.198
2015	14.78 mm/h	43.146	3.897	1.867	0.757	1.110

Table 6 Rain rates for years Turksat Golbası region [6], [12], [13]

Year	Rain Rate at 0.01% (mm/hr)	E° Antenna-B	D _s (km)	Calculated L _{RAIN} (dB)	Measured L _{RAIN} (dB)	Difference (dB)
2016	13.54	43.145	3.885	1.674	0.630	1.374
2017	12.06	43.145	3.885	1.455	0.589	0.866
2018	15.36	43.145	3.885	1.950	0.715	1.235
2019	14.23	43.145	3.885	1.778	0.678	1.100

4. Conclusion

While examining the losses in satellite communication link budgets, calculation of margins become very significant. Climatic conditions and rain precipitation value of Golbasi region are different from the region 'K', determined by ITU-R. Considering this region, same climatic area as South Asia and North Iraq does not always give a correct result [5]. Rain precipitation rate in tropical areas is different from continental climate. It has precipitation rate between 10-25 mm/h in accordance with data taken from meteorology and it is rarely above 25 mm/h. When precipitation rate gets higher, loss rate also gets higher according to ITU-R 838 model. According to this study, the value of 42 mm/h seem to be very conservative value comparing the long term meteorological rain data for Turkey [13], [14], [15]. Considering the region 'K' rain rate is 42 mm/h, this corresponds to about 6.6 dB rain attenuation for TURKSAT location. On the other hand, for Antenna-A case, the maximum observed value at 2014 was 16.35 mm/h which corresponds to 2.11 dB rain attenuation. In this case, the diameter of Antenna-A can be reduced from 7.2 meters to 4.5 meters as per calculations result. For Antenna-B, the maximum observed rain rate was 15.36 mm/h at 2018 which corresponds to 1.95 dB of rain loss. In that case, the diameter of Antenna-B can be reduced from 13.2 meters to 7.7 meters as per calculations. It is shown with this research that ITU-R-612 has conservative values comparing the actual results. The major gain from the study is to have opportunity of reduction of antenna diameter as well as minor profits due to the other units of the antenna systems (i.e HPA, LNA).

Acknowledgments

We would like to thank Turksat AS for its' invaluable support.

References

- [1] Dr. Y. L. Hui, W. K. Sain, Y. L. Sin *et al.*, "Overcoming Rain Fade Obstacle-Understanding Singapore's," *Rain Dynamics and Feasible Countermeasures Against Rain Fading-DSTA Horizons.*, pp. 6-15.
- [2] S. J. Mandeep and E. J. Allnut, "Rain attenuation Predictions at Ku-Band in South East Asia Countries," *Progress in Electromagnetics Research, PIER 76.*, pp. 65-74, 2007.
- [3] C. J. Kikkert and B. Bowthorpe, "A Satellite Beacon Receiver using Digital Signal Processing Techniques," *Progress in Electronic Research, PIER 76.*, pp. 65-74, 2007.
- [4] C. J. Kikkert, B. Bowthorpe, G. H. Allen, "Satellite Beacon Receiver Improvement using Digital Signal Processing," *ISSPA '96.*, pp. 517-520, Gold Coast, Australia, August 1996.
- [5] H. J. J. Ameen., "Rain effect on Ku Band satellite system," *Electrical and Electronics Engineering: An International Journal (ELELIJ).*, vol. 4, no. 2, May 2015.
- [6] Türksat Satellite Communication and Cable TV Operation A.Ş / Ground Stations /Document., "11.120 GHz Ku band Beacon Telemetry Information," , 2012-2019.
- [7] *Specific attenuation model for rain for use in prediction methods* (2005, September, 29), E27367 (03-05): RECOMMENDATION ITU-R P.838-3
- [8] O. Olabisi and E. A. Oladeji., "Effects of Rain on Vertical and Horizontal Polarized Ku Band Radio Propagation in Tropical Region," (*IJCST.*), vol. 6, no. 1, January-February 2018.
- [9] *Propagation data and prediction methods required for the design of Earth-space telecommunication systems* (2015, August, 11), E70000, (206/3): RECOMMENDATION ITU-R P.618-12
- [10] A.Samed, D. F. Diba, and D.Y. Choi., "A Survey of Rain Fade Models for Earth-Space Telecommunication Links-Taxonomy, Methods and Comparative Study," *Remote Sens.*, 2021, 13, 1965. [https://doi.org/10.3390/rs13101965.](https://doi.org/10.3390/rs13101965), 2021.
- [11] P. Charlesworth., "Notes on a Technique to estimate Rain Margins/ University of Wales," , 2010-2015.
- [12] B. I. Asianuba, and U. Agomuo., "Effect of Rain on Satellite Television Transmission," (*EJET.*), vol. 4, no. 4, 2019.

- [13] Turkish State Meteorological Service Department of Golbasi: Station no: 17134., “Amount of rain year, day and hours,” 2012-2014.
- [14] Y. A. Abdulrahman, A. T. Rahman, and A. K. S. Rahim., “ A new Rain Attenuation Conversion Technique for Tropical Regions, ” *Progress in Electromagnetics Research B.*, no. 26, pp. 53-67, 2010.
- [15] C. M. Kestwal, S. Joshi, and S. L. Garia., “ Prediction of Rain Attenuation and Impact of Rain in Wave Propagation at Microwave Frequency for Tropical Region (Uttarakhand, India),” *International Journal of Microwave Science and Technology.*, vol. 2014, ID 958498, pp. 6, 11-June-2014.