BILINGUAL
PUBLISHING
GROUP

Journal of Computer Science Research

Volume 5 | Issue 3 | July 2023 | ISSN 2630-5151（Online）

BILINGUAL
PUBLISHING
GROUP

Editor-in-Chief

Editorial Board Members

Volume 5 Issue 3 • July 2023 • ISSN 2630-5151 (Online)

Journal of Computer Science Research

Editor-in-Chief

Dr. Lixin Tao

Dr. Jerry Chun-Wei Lin

BILINGUAL
PUBLISHING
GROUP

BILINGUAL
PUBLISHING
GROUP

Volume 5 | Issue 3 | July 2023 | Page1-73

Journal of Computer Science Research

Contents

Articles

Review

BILINGUAL PUBLISHING GROUP

Journal of Computer Science Research
https://journals.bilpubgroup.com/index.php/jcsr

ARTICLE

Similarity Intelligence: Similarity Based Reasoning, Computing, and Analytics

Zhaohao Sun

Department of Business Studies, PNG University of Technology, Private Mail Bag, Lae 411, Morobe, Papua New Guinea

ABSTRACT

Similarity has been playing an important role in computer science, artificial intelligence (AI) and data science. However, similarity intelligence has been ignored in these disciplines. Similarity intelligence is a process of discovering intelligence through similarity. This article will explore similarity intelligence, similarity-based reasoning, similarity computing and analytics. More specifically, this article looks at the similarity as an intelligence and its impact on a few areas in the real world. It explores similarity intelligence accompanying experience-based intelligence, knowledge-based intelligence, and data-based intelligence to play an important role in computer science, AI, and data science. This article explores similarity-based reasoning (SBR) and proposes three similarity-based inference rules. It then examines similarity computing and analytics, and a multiagent SBR system. The main contributions of this article are: 1) Similarity intelligence is discovered from experience-based intelligence consisting of data-based intelligence and knowledge-based intelligence. 2) Similarity-based reasoning, computing and analytics can be used to create similarity intelligence. The proposed approach will facilitate research and development of similarity intelligence, similarity computing and analytics, machine learning and case-based reasoning.

Keywords: Similarity intelligence; Similarity computing; Similarity analytics; Similarity-based reasoning; Big data analytics; Artificial intelligence; Intelligent agents

*CORRESPONDING AUTHOR:
Zhaohao Sun, Department of Business Studies, PNG University of Technology, Private Mail Bag, Lae 411, Morobe, Papua New Guinea; Email: zhaohao.sun@gmail.com

ARTICLE INFO
Received: 19 March 2023 | Revised: 20 May 2023 | Accepted: 26 May 2023 | Published Online: 9 June 2023
DOI: https://doi.org/10.30564/jcsr.v5i3.5575

CITATION
Sun, Zh.H., 2023. Similarity Intelligence: Similarity Based Reasoning, Computing, and Analytics. Journal of Computer Science Research. 5(3): 1-14. DOI: https://doi.org/10.30564/jcsr.v5i3.5575

1. Introduction

Similarity, similarity relations, and similarity metrics have been playing an important role in computer science, artificial intelligence (AI), and data science [1-6]. Similarity has also played an important role in machine learning and case-based reasoning (CBR) [7,8]. Machine learning including deep learning has had important impacts on our life and work [7,9]. CBR as an AI technique has also played a significant role in experience-based reasoning and experience management [8-10]. From a meta viewpoint, what are the relationships between machine learning and CBR?

Intelligence has been also playing an important role in AI, business intelligence, machine learning, and CBR [3,8,11,12]. Intelligence can be defined as the collection, analysis, interpretation, visualization, and dissemination of strategic data, information, and knowledge for discovering and using the knowledge patterns and insights at the right time in the decision-making process [13]. Are machine learning and CBR related to similarity intelligence? This implies that similarity intelligence has been ignored in these disciplines. More specifically, research issues in this direction are:

1) Why is similarity intelligence important?

2) What are the relationships between similarity intelligence and experience intelligence, knowledge-based intelligence and data-based intelligence?

3) What are similarity computing and analytics and their impacts on similarity intelligence?

This article will explore similarity intelligence, similarity-based reasoning, similarity computing and analytics, and their relationships. To address the first question, this article looks at the similarity of intelligence and its impact on a few areas in the real world. To address the second question, it explores similarity intelligence that has been accompanying experience-based intelligence, knowledge-based intelligence and data-based intelligence to play an important role in computer science, AI, and data science. After reviewing the fundamentals of similarity, this article explores similarity-based reasoning (SBR) and proposes three similarity-based inference rules. This article then examines similarity computing and analytics, and a multiagent SBR system. The main contributions of this article are: 1) Similarity intelligence is discovered from experience-based intelligence consisting of data-based intelligence and knowledge-based intelligence. 2) Similarity-based reasoning, computing and analytics can be used to create similarity intelligence.

The rest of this article is organized as follows: Section 2 looks at why similarity intelligence is important. Section 3 examines machine learning and CBR as experience based intelligence. Section 4 examines the fundamentals of similarity. Section 5 explores similarity-based reasoning and proposes similarity-based inference rules for conducting SBR. Section 6 examines similarity computing and analytics. Section 7 proposes a multiagent architecture for an SBR system, and Section 8 ends this article with some concluding remarks.

2. Why is similarity intelligence important?

This section highlights why similarity intelligence is important.

Similarity has been playing an important role in mathematics, computer science, AI, and data science. Similarity has also played a significant role in fuzzy logic [2] and big data [5]. However, similarity intelligence has been ignored in these disciplines.

Similarity intelligence is a process for discovering intelligence from two or more objects or cases using similarity algorithms and techniques. The Turing test [14] has already mentioned that intelligence computing machinery is similar to that of human beings. This is a kind of similarity intelligence. Similarity intelligence includes similar relationships consisting of patterns and insights between machines, human beings, and software apps [14,5]. In other words, similarity intelligence is not only from human beings, but also from machines or software or apps.

Similarity also plays an important role in ChatGPT, because similarity is crucial in natural language understanding and processing. Based on the research analyzing 1000 texts produced by ChatGPT, it found that on average, the similarity varies between 70%

and 75% [15]. Therefore, one of the important tasks of ChatGPT (http://wwww.openAi.com) is to discover similarity intelligence from two or more objects, texts, and cases.

Similarity intelligence is important because it enables us to identify similarities and patterns in data sets, which can be used to make more informed decisions and predictions. By identifying similarities between different sets of data, objects, and cases [8], we can better understand relationships and draw insights that might not be immediately apparent, at least similarity intelligence can allow us to select one from a similarity class as a representative and then we can analyze it as a characteristic of the similarity class [16]. For example, in the field of customer relationship management [12], intelligence can be used to identify patterns and preferences in consumer behaviors through similarity metrics. The patterns and preferences can then be used to develop targeted advertising and product recommendations that are more likely to appeal to specific groups of consumers.

Therefore, similarity intelligence is important not only for computer science, AI, big data, and data science, but also for businesses and organizations in a wide range of industries, enabling decision makers to obtain more informed decisions in an intelligent experience-based, knowledge-driven, and data-driven world.

3. Experience-based intelligence

Experience-based intelligence is a process of discovering intelligence from experiences, based on experience-based reasoning [17]. Experience-based intelligence consists of data-based intelligence and knowledge-based intelligence. This section looks at similarity intelligence from experience-based intelligence using two examples, machine learning and CBR. Machine learning is data-based intelligence. CBR is knowledge-based intelligence.

3.1 Machine learning

Similarity has always been important in pattern recognition, graphical pattern recognition, machine learning [7,18], because as soon as we have created patterns, and we have to use similarity to match what was input to the systems and compare it with our patterns.

Machine learning is about how to build computers and apps that improve automatically through experience [19], that is, machine learning is a process of discovering intelligence from experience using computers and software. Therefore, machine learning is an experience-based Intelligence.

Machine learning is about how a computer can use a model and algorithm to observe some data about the world, and adapt to new circumstances and detect and extrapolate patterns [11]. Therefore, machine learning is a process of discovering intelligence from data, that is, machine learning is data-based intelligence, a process of discovering intelligence from data, because it is a process of using probabilistic models and algorithms on data to create intelligence through data [11].

One of the unsupervised machine learning is clustering [7]. How we calculate the similarity between two clusters or two objects is important for clustering [4,7,18]. There are a few methodologies that are utilized to calculate the similarity: For example, Min, Max, the distance between centroids and other similarity matrices mentioned in Section 4.4. Therefore, machine learning is similarity intelligence, a process for creating intelligence through similarity.

Overall, machine learning is an experience-based Intelligence, a process of discovering Intelligence through experience [4]. Machine learning is data-based intelligence, a process of discovering intelligence from data. Machine learning is also similarity intelligence, a process for creating intelligence through similarity.

3.2 Case-based reasoning

CBR is a process of discovering similarity intelligence from a case base, just as data mining is a process of discovering data intelligence from a large DB [12]. Similarity intelligence includes the exact case that has been used in the past for solving the problem encountered recently.

CBR is a reasoning paradigm based on previous experiences or cases [12,8]. CBR is based on two principles about the nature of the world [8]: The types of problems an agent encounters tend to recur. Hence, future problems are likely to be similar to current problems. The world is regular: similar problems have similar solutions or similar causes bring similar effects [8]. Consequently, solutions to similar prior problems are a useful starting point for new problem solving. The first principle implies that CBR is a kind of experience-based reasoning (EBR), while the second principle is the guiding principle underlying most approaches to similarity-based reasoning (SBR) [8]. "Two cars with similar quality features have similar prices" is one application of the above-mentioned second principle, and also a popular experience principle summarizing many individual experiences of buying cars. It is a kind of SBR. In other words, SBR is a concrete realization of CBR. The CBR system (CBRS) is an intelligent system based on CBR, which can be modelled as [8]:

$$\text{CBRS} = \text{Case Base} + \text{CBRE} \tag{1}$$

where the case base (CB) is a set of cases, each of which consists of the previously encountered problem and its solution. CBRE is a CBR engine, which is the inference mechanism for performing CBR, in particular for performing SBR. The SBR can be formalized as:

$$\frac{\mathrm{P}', \mathrm{P}' \sim \mathrm{P}\ \mathrm{P}, \rightarrow \mathrm{Q}}{\therefore Q^{'}} \tag{2}$$

where *P*, P', Q' and Q' represent compound propositions, P' ∼ P means that if P' and *P* are similar (in terms of similarity relations, metrics and measures, see Section 4) and then Q and Q' are also similar. (2) is called *generalized modus ponens*, that is, (2) is one of the inference rules for performing modus ponens based on SBR. Typical reasoning in CBR, known as the CBR cycle, consists of (case) Repartition, Retrieve, Reuse, Revise and Retain [8]. Each of these five stages is a complex process. SBR dominates all these five stages [16]. Therefore, CBR is a process for discovering intelligence through SBR, because Similar problems have similar solutions.

One significant contribution of CBR research and development is that it points out the importance of experience and similarity [9,16]. CBR is experience-based intelligence, a process for discovering intelligence based on experience. Because case base is a kind of knowledge base [10,8], so that, CBR is also a knowledge-based intelligence [11].

Overall, similarity intelligence accompanies experience-based intelligence [10,8], data-based intelligence [4] and knowledge-based intelligence [11] to provide constructive insights and decision supports for businesses and organizations.

4. Fundamentals of similarity

The similarity is a fundamental concept for many fields in mathematics, mathematical logic, computer science, AI, data science, and other sciences [16,9,20,21]. This section first briefly looks at similarity and then focuses on similarity relations, fuzzy similarity relations, and similarity metrics.

4.1 Introduction

The concept of similarity has been studied by numerous researchers from different disciplines such as in mathematics [20], big data [5], computer science [22,23], AI and fuzzy logic [1,2,21], to name a few. For example, Klawonn and Castro [24] examined similarity in fuzzy reasoning and showed that similarity is inherent to fuzzy sets. Fontana and Formato [25] extended the resolution rule as the core of a logic programming language based on similarity and discussed similarity in deductive databases. The concepts of similarity and similarity relations play a fundamental role in many fields of pure and applied science [26,20]. The notion of a metric or distance between objects has long been used in many contexts as a measure of similarity or dissimilarity between elements of a set [27,22,18]. Thus, there exist a wide variety of techniques for dealing with problems involving similarity, similarity relations, similarity measures, and similarity metrics [21,23]. For example, fuzzy logic [1,2], databases [5], data mining [18] and CBR [8] provides a number of concepts and techniques for dealing with similarity relations, similarity measures, and similarity metrics.

In what follows, we briefly introduce similarity relations, fuzzy similarity relations, and similarity metrics.

4.2 Similarity relations

The concept of a similarity relation is a natural generalization of similarity between two triangles and two matrices in mathematics [16,18]. More precisely:

Definition 1. A binary relation S on a non-empty set X is called a similarity relation if it satisfies:

(1) $\forall x$, xSx,

(2) If xSy, then ySx,

(3) If xSy, ySz then xSz

The conditions (1), (2), and (3) are the reflexive, symmetric, and transitive laws. If xSy we say that x and y are similar [20,16].

Example 1. Matrices B and C in $M_{n,n}$ are similar if $C = PBP^{-1}$ for an invertible P, in which case we write $B{\sim}C$. It is easy to prove that is $\sim$ is a similarity relation in $M_{n,n}$ [20].

This example implies that the concept of a similarity relation here is a generalization of the similarity between matrices in $M_{n,n}$.

Similarity relations can be used for classification through partition [16] and clustering [18].

4.3 Fuzzy similarity relations

As an extension of similarity relations, fuzzy similarity relations were introduced by Zadeh in 1971 [2] and have attracted much attention since then [1,27,21]. For example, fuzzy similarity relations have been used in CBR [16]. For the sake of brevity, we use standard fuzzy set theory notation for operations min and max, although there are many alternative choices for these operations available in fuzzy set theory (Zimmermann, 1996). S is still used to denote a fuzzy similarity relation if there is not any confusion arising.

Definition 2. A fuzzy binary relation S on a non-empty set is a fuzzy similarity relation in X if it is reflexive, symmetric, and transitive [28,2], that is:

$$S(x, x) = 1 \tag{3}$$

$$S(x, y) = S(y, x) \tag{4}$$

$$S \geq S \cdot S \tag{5}$$

where · is the composition operation of fuzzy binary relations based on min and max operations. A more explicit form of Equation (5) is

$$S(x,z) \geq \vee_y (S(x,y) \wedge S(y,z)) \tag{6}$$

Equation (6) is called max-min transitivity [2]. The revised form of this definition was given by Ovchinnikov in 1991 [28]. The main difference between the definition of Zadeh and that of Ovchinnikov lies in that instead of Equation (6), Ovchinnikov viewed the following model as max-min transitivity.

$$S(x,z) \geq S(x,y) \wedge S(y,z) \tag{7}$$

4.4 Similarity metrics

Generally speaking, similarity in mathematics is considered as a relation, while similarity in CBR is considered both a relation and a measure, a function, and a metric [16,8].

Definition 3. A relation, denoted by S_m, on non-empty X, is a similarity metric if it satisfies [16]:

1) S_m is a similarity relation on X;

2) 1- S_m is a metric on X; that is, it is a function from $X \times X$ to [0,1], provided that:

- For any $x \in X$, $S_m(x,x) = 1$
- For all $x, y \in X$, $S_m(x,y) = S_m(y,x)$ (8)
- For all $x, y, z \in X$, $S_m(x,z) \geq S_m(x,y) \wedge S_m(y,z)$

where $\wedge$ is min operator. Equation (8) in this definition is called the similarity inequality. It should be noted that the similarity metric here, S_m, can not directly satisfy the triangle inequality [16]. Equation (8) is based on Ovchinnikov's concept of fuzzy similarity relations [28].

In comparison with the definition of fuzzy similarity relations, we emphasize that the similarity metric here is first a traditional similarity relation, and also just a metric, maybe to some extent, because the similarity between two objects is the necessary condition to further discuss how similar they are in the context [16].

5. Similarity-based reasoning and inference rules

This section highlights similarity-based reasoning and its three inference rules.

5.1 Similarity-based reasoning

Similarity-based reasoning (SBR) has been studied by many researchers from different fields. For example, Sun [9] examined integration of rule-based and SBR from an AI viewpoint. He considered SBR as a reasoning-based similarity matching. Bogacz and Giraud-Carrier considered SBR as "reasons from similarity" from a neural network viewpoint [29]. The relationship between CBR and SBR has drawn some attention [8,16]. However, what is similarity-based reasoning? There is still no definition of it, to our knowledge. In fact, many methods of SBR seem to lack a sound theoretical or logical basis [30]. We need a relatively precise definition of SBR, in order to investigate similarity-based approaches to SBR.

Definition 4. Let *P*, *P*', *Q*, and *Q*' represent compound propositions, $P \to Q$ is a production rule, denoting if P then Q. A proposition can be inferred from propositions *P* and $P \to Q$, provided that *P*, and *P*' are similar ($P' \sim P$), and then *Q* and *Q*' are also similar; that is:

$$\frac{P', P' \sim P, P \to Q}{\therefore Q'} \tag{9}$$

Then, this reasoning paradigm is called similarity-based reasoning (SBR).

More generally, a proposition *Q*' can be similarity-based inferred from propositions P_1' $P_2', \ldots, P_n'$ provided P_1 P_2, ..., $P_n \to Q$, and $P_i' \sim P_i$ ($i = \{1, 2, \ldots, n\}$), *Q* and *Q*' are also similar; that is:

$$\begin{array}{l} P_1', P_2', \cdots, p_n' \\ \underline{P_1, P_2, \cdots, P_n \to Q} \\ \quad \therefore Q' \end{array} \tag{10}$$

It should be noted that the definition is based on modus ponens [31]. Therefore, the reasoning defined above can be considered as a kind of SBR with respect to modus ponens (which will be examined further in the next subsection). It can be also considered as a composite reasoning paradigm. Furthermore, the above definition is general, and its generality lies in that we have not assigned any special meaning or semantics to the similarity used in the definition.

Example 4. Google Chrome as a search engine is based on similarity-based reasoning, that is, "similarity-based reasoning" which is searched by https://www.google.com.au/ and found 50,000 results (on 02 March 2023). However, not every one of the found results is related to "similarity-based reasoning" but to "similarity" or "reasoning". This is the reason why we call Google Chrome as similarity-based reasoning. It does not really result in complying with the inference rule of modus ponens (see latter). This is also the reason why some hope to get exact results rather than 50,000 or millions of found results searched by Chrome and many other search engines.

In order to perform SBR, it is necessary to note the following three points:

1) What we examined in the previous subsections: Similarity relations, fuzzy similarity relations, and similarity metrics are concrete forms of similarity used in the above definition. In other words, each of them can lead to a class of SBR. We can, therefore, examine SBR from a viewpoint of either similarity relations or fuzzy similarity relations or similarity metrics. If so, we will make our investigation very complex, although the corresponding research results are of significance in applications.

2) SBR is treated in a more general way in this article; that is, two different forms, ~ and ≈ , of similarity (e.g., similarity relations, fuzzy similarity relations, and similarity metrics) are used in the context. The first ~ is associated with the similarity between *P* and *P′*, while the second ≈ is associated with the similarity between Q and *Q′*. In the context of CBR, these two similarity relations (or fuzzy relations or metrics) are in different worlds [8]; that is, the first ~ is associated with the possible world of problems, while the second ≈ is associated with the possible world of solutions.

3) The readers can consider ~ and ≈, from now on in this article, as either similarity relations or fuzzy

similarity relations or similarity metrics in a consistent way in a real-world application. In other words, for a real-world problem, if one considers them similarity metrics, then she/he should use them consistently.

In order to perform SBR using Equation (9), we define a degree of similarity between propositions P and P' [9]) as

$$S(P, P') = 2\ \frac{|F_P \cap F_{P'}|}{|F_P| + \ |F_{P'}|} \tag{11}$$

where F_x is the set of the features of proposition x, and $|F_x|$ is the size of the set of features of F_x. The degree of similarity $S(P, P')$ has the following properties [18]:

1) $0 \le S(P, P') \le 1$.

Note that a similarity function is a special similarity relation.

2) If $P = P'$, that is, P and P' are the same propositions, then $S(P, P') = 1$.

5.2 Similarity-based inference rules

This section will look at similarity-based reasoning (SBR) and its three inference rules.

Similarity-based modus ponens

In the previous section, we defined SBR concerning modus ponens. In order to emphasize the importance of similarity between P and P', Q and Q' and show the difference of the inference rule of SBR from the generalized modus ponens of fuzzy reasoning [1], we call Equation (9) similarity-based modus ponens (SMP), and its form will be replaced with the following:

$$\frac{P', P' \sim P, P \to Q, Q \approx Q'}{\therefore Q'} \tag{12}$$

The reasoning paradigm of the similarity-based deductive system and similarity-based agent [32] is based on Equation (12), because similarity-based resolution [32] is an alternative form of Equation (12). Equation (12) is also a logical foundation for CBR [8]. SMP is one of the most important reasoning paradigms in many other disciplines, because it is the basic form of any similarity-based deductive reasoning paradigm [29,30,9].

In a CBR customer support system [8], P' is the problem description of the customer, $P' \sim P$ means that P' and P are similar, $P \to Q$ is the case retrieved from the case base C based on a similarity-based retrieval algorithm. $Q \approx Q'$ means that Q and Q' are similar, and Q' is the satisfactory solution to the requirement of the customer.

In the context of fuzzy similarity relations and similarity metrics [22], we assume that P' corresponds to $\widetilde{P_0}$, $P' \sim P$ corresponds to $\widetilde{F_{01}}$, $P \to Q$ corresponds to $\widetilde{F_{11}}$, $Q \approx Q'$ corresponds to $\widetilde{F_{10}}$, and Q' corresponds to $\widetilde{Q_1}$. Then, using the compositional rule of inference [1,33], we obtain:

$$\widetilde{Q_1} = \widetilde{P_0} \circ \widetilde{F_{01} \circ} \widetilde{F_{11}} \circ \widetilde{F_{10}} \tag{13}$$

where $\widetilde{P_0}$ is a fuzzy set in W_P. $\widetilde{F_{01}}$, $\widetilde{F_{11}}$, and $\widetilde{F_{10}}$ are a similarity metric, a fuzzy rule and a fuzzy similarity metric in $W_P \times W_Q$ respectively, and $\widetilde{Q_1}$ is a fuzzy set on W_Q. This is a computational foundation for similarity-based modus ponens [8]. In the case of $Q \approx Q'$, $\widetilde{F_{10}}$ is a unit metric, and Equation (13) is then simplified into:

$$\widetilde{Q_1} = \widetilde{P_0} \circ \widetilde{F_{01} \circ} \widetilde{F_{11}} \tag{14}$$

When $\widetilde{P_0}$, $\widetilde{F_{01}}$, $\widetilde{F_{11}}$, and $\widetilde{Q_1}$ are only a numerical similarity measure respectively, (14) essentially degenerates into the computational form.

In fact, many other reasoning paradigms also follow, to some sense, Equation (14), for example, analogical reasoning [1], although they have different semantics and operational algorithms for performing their own reasoning based on different real-world scenarios.

While fuzzy reasoning is essentially computational reasoning, SBR can be considered as both symbolic reasoning and computational reasoning [8]. If we regard SBR as computational reasoning, then we can consider it as a special kind of fuzzy reasoning, to some extent, because the similarity between P and P', $P \sim P'$, and the similarity between Q and Q', $Q \approx Q'$, are replaced by the fuzziness between them in the context of fuzzy logic. This is the reason why we can use fuzzy reasoning to examine the similarity-based modus ponens in CBR [8].

Similarity-based modus tollens

Similarity-based modus tollens (SMT) is another inference rule for SBR. From a traditional viewpoint, we can consider SMT as an integration of SBR and modus tollens. The general form of SMT is as follows:

$$\frac{\neg Q', Q' \approx Q, Q \to P, P \to P'}{\therefore \neg P'} \tag{15}$$

Although fuzzy modus tollens have not been investigated in fuzzy logic [1], this is the first time that similarity-based modus tollens is discussed. With the increasing importance of similarity, SMT and its corresponding SBR will find their applications in business and mathematics.

Example 5. Similarity-based modus tollens. Let:

- *RAT*: The applicant has a good credit rating,
- *REP*: The applicant has a good financial reputation,
- The loan officer has an experienced rule, *RAT→REP*: If the applicant has a good credit rating, then the applicant has a good financial reputation.

In this case, the loan officer knows the information from applicant *A*, *¬REP*: The applicant has an unsatisfactory financial reputation.

Because "a satisfactory financial reputation" is similar to "a good financial reputation", that is, *RAT→REP,* therefore, the loan officer uses the above SMT to make the decision and obtain *¬REP*: The applicant has an unsatisfactory credit rating, since "a good credit rating" is similar to "a satisfactory credit rating".

In the context of fuzzy similarity relations or similarity metrics [22], using the compositional rule of inference [1] to the above Equation (15) we obtain:

$$\widetilde{P_0} = 1 - (1 - \widetilde{Q_1}) \circ \widetilde{F_{10} \circ} \widetilde{F_{11}} \circ \widetilde{F_{01}} \tag{16}$$

This is a computational foundation for similarity-based modus tollens. In the case $Q \approx Q'$, $\widetilde{F_{10}}$ is a unit metric, and Equation (16) is then simplified into

$$\widetilde{P_0} = 1 - (1 - \widetilde{Q_1}) \circ \widetilde{F_{11}} \circ \widetilde{F_{01}} \tag{17}$$

Similarity-based abduction

Abduction has been used in system diagnosis or medical diagnosis [8] and scientific discovery [34].

Abduction is an important reasoning paradigm in SBR. Similarity-based abductive reasoning (SAR) is a natural development of abductive reasoning [35], or an application of SBR in abductive reasoning. Its general form is as follows:

$$\frac{Q', Q' \approx Q, Q \to P, P \to P'}{\therefore P'} \tag{18}$$

Example 6. Similarity-based abductive reasoning. As in Example 5, let:

- *RAT*: The applicant has a good credit rating,
- *REP*: The applicant has a good financial reputation,
- The loan officer has an experienced rule, *RAT→REP*: If the applicant has a good credit rating, then the applicant has a good financial reputation.

In this case, the loan officer knows the information from applicant *A*, *REP′* : The applicant has a satisfactory financial reputation. Because "a satisfactory financial reputation" is similar to "a good financial reputation"; that is, *REP ~ REP′*, the loan officer uses the above similarity-based abductive reasoning to make the decision and obtain *REP′*: The applicant has a satisfactory credit rating, because "a good credit rating" is similar to "a satisfactory credit rating". It is obvious that "The applicant has a satisfactory credit rating" is an explanation for "The applicant has a satisfactory financial reputation." Therefore, similarity-based abductive reasoning can be also used for generations of explanation, as abductive reasoning does scientific discovery [34,36].

In the context of fuzzy similarity relations and similarity metrics, using the compositional rule of inference [1] to the above Equation (18), we obtain:

$$\widetilde{P_0} = \widetilde{Q_1} \circ \widetilde{F_{10} \circ} \widetilde{F_{11}} \circ \widetilde{F_{01}} \tag{19}$$

This is a computational foundation for similarity-based abductive reasoning. In the case of $Q \approx Q'$, $\widetilde{F_{10}}$ is a unit metric, and Equation (19) is then simplified into:

$$\widetilde{P_0} = \widetilde{Q_1} \circ \widetilde{F_{11}} \circ \widetilde{F_{01}} \tag{20}$$

5.3 Summary

Table 1 summarizes the well-known inference rules: Modus ponens, modus tollens, abduction, and proposes three inference rules with respect to SBR, corresponding to the traditional forms: Modus ponens, modus tollens, and abduction [37,31]. So far, we have examined three different inference rules for SBR (see **Table 1**) in a unified viewpoint, each of them has been thoroughly used in computer science, mathematics, mathematical logic [38], and other sciences [30,39,34]. However, they are all the abstractions and summaries of SBR, natural reasoning, and ordinary reasoning in the real world. Furthermore, CBR has been only based on either modus ponens or modus tollens or abduction [33,8], whereas SBR is based on the mentioned three inference rules. It should be noted that reasoning paradigms can be classified into simple (atomic or first level) reasoning paradigms and composite (second level) reasoning paradigms [40], just as propositions can be divided into simple (atomic) propositions and compound propositions [39]. The simplest reasoning paradigm is an inference rule, which is the basis for any reasoning paradigm.

A composite reasoning paradigm consists of more than one inference rule. For example, fuzzy modus ponens [2] is a composite reasoning paradigm that integrates modus ponens and fuzzy rules. Any process model of a reasoning paradigm in AI is a method for obtaining composite reasoning paradigms. For example, the simplest rule-based expert system (RBES) can mainly consist of the knowledge base (KB) and an inference engine (IE), where IE is an inference mechanism for performing modus ponens or modus tollens or abduction. However, in order to manipulate the knowledge in the KB, the RBES must deal with knowledge representation, knowledge explanation, and knowledge utility which are the main components of the process model [11,8]. Therefore, the reasoning involved in RBES can be considered as a composite reasoning paradigm. In this way, we can differentiate reasoning paradigms in mathematical logic and AI. What we have examined in this article are simple or atomic inference rules for SBR. In future work, we will examine composite reasoning paradigms for SBR, which constitute a "reasoning chain" [3], "reasoning network" or "reasoning tree" with some depth, and correspond to natural reasoning in human professional activities.

It should be noted that the above-mentioned abductive reasoning and its SBR are unsound reasoning paradigms from a logical viewpoint [31]. However, like nonmonotonic reasoning, which is also unsound reasoning [8], this inference rule and its similarity-based abduction is the summarization of SBR used by people in the real-world situations.

6. Similarity computing and analytics

Similarity computing and analytics are science, technology, system and tools used in data, information, and knowledge analysis to measure and compare the similarity between different data, information, and knowledge sets. They are used in various fields such as AI including machine learning, data science, natural language understanding and processing, image recognition, and information retrieval. This section will examine similarity computing and

Table 1. Three inference rules for similarity-based reasoning.

	Modus ponens	**Modus tollens**	**Abduction**
Traditional form	$\dfrac{P, P \to Q}{\therefore Q}$	$\dfrac{\neg Q, P \to Q}{\therefore \neg P}$	$\dfrac{Q, P \to Q}{\therefore P}$
similarity-based form	$\dfrac{\begin{array}{c} P', P' \sim P, \\ P \to Q, Q \approx Q' \end{array}}{\therefore Q'}$	$\dfrac{\begin{array}{c} \neg Q', Q' \approx Q, \\ Q \to P, P \sim P' \end{array}}{\therefore \neg P'}$	$\dfrac{\begin{array}{c} Q', Q' \approx Q, \\ Q \to P, P \sim P' \end{array}}{\therefore P'}$

analytics in some detail.

Similarity computing is a science, technology, system, and tool for determining the degree of similarity or dissimilarity between two or more objects to create intelligence. That is, based on the research of Sun [41],

$$\begin{aligned}\text{Similarity computing} = {} & \text{Similarity science}\\ & + \text{Similarity engineering}\\ & + \text{Similarity technology}\\ & + \text{Similarity system}\\ & + \text{Similarity tools}\end{aligned} \tag{21}$$

Similarity relations, fuzzy similarity relations [2] and similarity metrics [1] such as Cosine similarity, Jaccard similarity, Euclidean distance, and Pearson correlation coefficient, among others are fundamentals for realizing similarity or dissimilarity between two or more objects for similarity computing [18].

Analytics is science, technology, system and tools for mining data, information, knowledge to discover meaningful intelligence, insights, patterns, and knowledge from big data in a database or data warehouse or knowledge in a knowledge base using similarity [42]. This can be achieved using database and data warehouse techniques, statistical techniques, knowledge base techniques, data visualization techniques, machine learning algorithms, and other data and knowledge processing tools [4,43]. Similarity analytics can be represented below [41],

$$\begin{aligned}\text{Similarity Analytics} = {} & \text{Similarity science}\\ & + \text{Similarity engineering}\\ & + \text{Similarity technology}\\ & + \text{Similarity system}\\ & + \text{Similarity tools}\end{aligned} \tag{22}$$

Basically, similarity analytics is a part of similarity computing, just as analytics is a part of computing [41]. Both aim to discover similarity intelligence in the domain. Even so, not only similarity computing but also similarity analytics can enable the analysis of large datasets, information sets and knowledge sets to identify and discover intelligence, patterns, knowledge and insights, and prediction of outcomes or cases. For example, in machine learning, similarity computing is used to find similarities between different data points, and analytics is used to train models that can make predictions based on those similarities [4,18].

Although similarity science has not been proposed in academia, similarity engineering, similarity technology, similarity systems (see the next section) and similarity tools based on similarity models, methods, and algorithms are well-known in the market [1,7,44].

Overall, similarity computing and analytics are science, technology, and system in modern data, information, and knowledge analysis to enable researchers and practitioners to gain similarity intelligence, knowledge and insights, and make predictions in various fields.

7. A multiagent SBR systems

In AI, a reasoning paradigm usually corresponds to an intelligent system. This section proposes a multiagent SBR system as an example, which constitutes an important basis for developing any multiagent SBR systems (MSBRS).

7.1 A general architecture of an SBR system

Similarity case base (SCB) is similar to a case base in a case base system [8] illustrated in **Figure 1**. The SCB is a text case base in natural language processing systems [4] and an insight base in data mining system and data analytics systems [41]. SCB consists of all the cases that the SBR System collects periodically. A user interface is used to interact with the SCB and MIE in the SBR System (see Section 7.3). The MIE is a multi-inference engine that consists of the mechanism for implementing three reasoning paradigms based on the above-mentioned three similarity-based inference rules and their algorithms for SBR with manipulating the SCB to infer similarity-based problem solving and decision making requested by the user. The remarkable difference between the mentioned SBRS and the traditional CBR system (CBRS) lies in that the latter's inference engine is based on a unique reasoning paradigm (or inference rule), while the MIE is based on many different reasoning paradigms. This implies that a CBRS is only a subsystem of the SBRS. Therefore, this SBR System is the extension of CBRS and similarity-based reasoning [8,31].

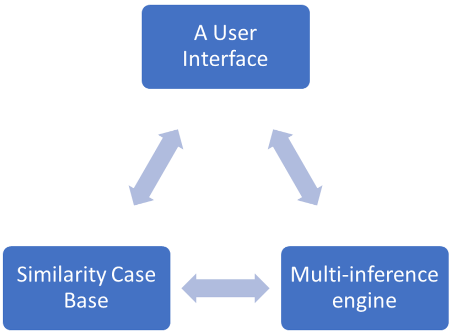


Figure 1. A general architecture for a SBR system.

7.2 MEBIE: A multiagent framework for similarity based inference engine

As mentioned in the previous subsection, the MIE is a multi-inference engine for SBRS [45]. MIE could automatically adapt itself to the changing situation and perform one of the mentioned similarity-based inference rules for SBR (**Figure 2**). However, any existing intelligent system has not reached such a high level [46]. The alternative strategy is to use multiagent technology to implement the MIE. Based on this idea, we propose a multi-agent framework for a similarity-based inference engine (for short MABIE), which is a core part of a multiagent SBR system (MSBRS), as shown in **Figure 1**. In this framework, three rational agents (from SMP agent to SAR agent) are semiautonomous [8]. These three agents are mainly responsible for performing SBR corresponding to three similarity-based inference rules in the SBRS respectively. In what follows, we discuss each of them in some detail.

1) The SMP agent in the MABIE is responsible for manipulating the SCB based on similarity-based modus ponens and its algorithm (also see Section 5.1) to infer the similarity-based problems and solutions requested by the user. This agent can be considered as an agentization of an inference engine in a traditional CBR system. The function of the SMP agent can be extended to infer the cases in the SCB based on fuzzy modus ponens [46,23].

2) The SMT agent manipulates the SCB to infer the case requested by the user based on similarity-based modus tollens and its algorithms (see Section 5.2).

3) The SAR agent is responsible for manipulating the SCB to infer the case requested by the user based on similarity-based abductive reasoning and its algorithm (see Section 5.3). This agent can generate the explanation for the experience-based reasoning inferred by the MEBIE. This agent can be considered as an agentization of an inference engine in an abductive CBR system [33].

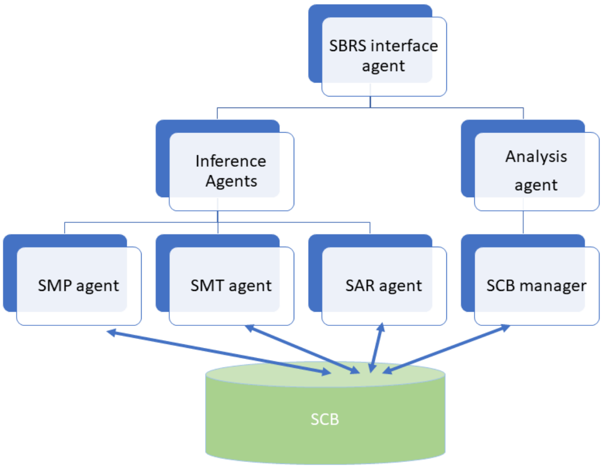


Figure 2. MIE and other agents in a MSBRS.

7.3 Some other agents in MSBRS

For the proposed MSBRS, there are some other intelligent agents, shown in **Figure 2**. These are an interface agent, an analysis assistant and a SCB manager. In what follows, we will look at them in some depth [45].

The SBRS interface agent is an advisor to help the MSBRS user to know which reasoning agent she/he should ask for help. Otherwise, the SBRS interface agent will forward the problem of the user to all agents in the MIE for further processing.

The output provided by the MIE can be considered as a sub output. The final output as the solutions to the similarity-based problem of the user will be processed with the help of the analysis agent. Since different agents in the MIE use different inference rules, and then produce different, conflicting results with knowledge inconsistency. How to resolve such knowledge inconsistency is a critical issue for the

MSBRS. This issue will be resolved by the Analysis assistant of the MSBRS. The analysis assistant will:

- Rank the degree of importance of the sub outputs from the MAMIE taking into account the knowledge inconsistency,
- Give an explanation for each of the outputs from the MIE and how the different results are conflicting,
- Combine or vote to establish the best solutions,
- Forward them to the SBRS interface agent who then forwards them to the user.

The SCB manager is responsible for administering the SCB. Its main tasks are SCB creation and maintenance, similarity case base evaluation, reuse, revision, and retention. Therefore, the roles of the SCB manager are an extended form of the functions of a CBR system [8], because case base creation, case retrieval, reuse, revision and retention are the main tasks of the CBR system [16].

7.4 Workflows of agents in MSBRS

Now let us have a look at how the MSBRS works. The user, U, asks the SBRS interface agent to solve the problem, *p*. The SBRS interface agent asks U whether a special reasoning agent is needed [45]. U does not know. Thus, the SBRS interface agent forwards p (after formalizing it) to all agents in the MIE for further processing. The agent in the MIE manipulates the case in the SCB based on p, and the corresponding reasoning mechanism, and then obtains the solution, which is forwarded to the Analysis assistant. After the Analysis assistant receives all solutions to p, it will rank the degree of importance of the solutions, give an explanation for each of the solutions and how the results are conflicting or inconsistent, and then forward them (with p) to the SBRS interface agent who would then forward them to U. If U accepts one of the solutions to the problem, then the MSBRS completes this mission. In this case, the SCB manager will look at whether this case is a new one. If yes, then it will add it to the SCB. Otherwise, it will keep some routine records to update the SCB. If U does not accept the solution provided, the SBRS interface agent will ask U to adjust some aspects of the problem p, which is changed into *p*', then the SBRS interface agent will once again forward the revised problem *p*' to the MIE for further processing.

8. Conclusions

Artificial intelligence (AI) has addressed experience-based intelligence and knowledge-based intelligence at their early stage. Big data has been experiencing significant progress in the past 10 years, AI has been developing machine learning and deep learning to address data-based Intelligence. In fact, similarity intelligence has been accompanying experience-based intelligence, knowledge-based intelligence, and data-based Intelligence to play an important role in computer science, AI, and data science in general and similarity computing and analytics in particular. The main contributions of this article are:

1) It explored similarity intelligence, based on the similarity discovered from experience-based intelligence in machine learning and CBR. Similarity intelligence will be developed and created by many systems and algorithms in AI, computer science, and data science.

2) It explored similarity-based reasoning and proposed its three different rules, which constitute the fundamentals for all SBR paradigms.

3) It highlighted similarity-based reasoning, computing, and analytics to create similarity intelligence. As an example, the article also proposed a multiagent architecture for an SBR system (MSBRS).

Overall, similarity intelligence is discovered from big data, information, and knowledge using similarity relations, fuzzy similarity relations and metrics, SBR, similarity computing and semantics.

Furthermore, the similarity-based approach to similarity intelligence, SBR, similarity computing and analytics proposed in the article opens a new way to integrate machine learning (e.g. machine learning algorithms such as instance-based learning and k-Nearest Neighbor (kNN) classifier and experience-based reasoning based on SBR, which will be examined in future work. Knowledge management and experience management have drawn increasing attention in business, e-commerce, and computer science. Their correspondence to intelligent systems is similarity-based systems such as CBR systems and machine learning. How to apply similarity intelligence in Knowledge management, experience management, and similarity-based systems will be also examined in future work.

Measurement of intelligence is based on the ability to solve difficult problems. How to define the measurement of similarity intelligence is still a weakness of this article. In future work, we will explore the measurement of similarity intelligence.

Conflict of Interest

There is no conflict of interest.

References

[1] Zimmermann, H.J., 2011. Fuzzy set theory—and its applications. Springer Science & Business Media: Berlin.

[2] Zadeh, L.A., 1971. Similarity relations and fuzzy orderings. Information Sciences. 3(2), 177-200.

[3] Minsky, M., 1988. Society of mind. Simon and Schuster: New York.

[4] Aroraa, C., Chitra, L., Munish, J., 2022. Data analytics: Principles, tools, and practices. BPB Publications: New Dalhi.

[5] Sun, Z., 2022. A mathematical theory of big data. Journal of Computer Science Research. 4(2), 13-23.

[6] Zhang, D.G., Ni, C.H., Zhang, J., et al., 2022. A novel edge computing architecture based on adaptive stratified sampling. Computer Communications. 183, 121-135.

[7] Milošević, P., Petrović, B., Jeremić, V., 2017. IFS-IBA similarity measure in machine learning algorithms. Expert Systems with Applications. 89, 296-305.

[8] Finnie, G., Sun, Z., 2004. Intelligent techniques in E-commerce: A case based reasoning perspective. Springer-Verlag: Berlin.

[9] Sun, R., 1995. Robust reasoning: Integrating rule-based and similarity-based reasoning. Artificial Intelligence. 75(2), 241-295.

[10] Bergmann, R., 2002. Experience management: Foundations, development methodology, and internet-based applications. Springer: Berlin.

[11] Russell, S., Norvig, P., 2020. Artificial intelligence: A modern approach (4th Edition). Prentice Hall: Upper Saddle River.

[12] Laudon, K.G., Laudon, K.C., 2020. Management information systems: Managing the digital firm (16th Edition). Pearson: Harlow.

[13] López-Robles, J.R., Otegi-Olaso, J.R., Gómez, I.P., et al., 2019. 30 years of intelligence models in management and business: A bibliometric review. International Journal of Information Management. 48, 22-38.

[14] Turing, A., 1950. Computing machinery and intelligence. Mind. 49, 433-460.

[15] Schwab, P.N., 2023. ChatGPT: 1000 Texts Analyzed and up to 75,3% Similarity [Internet] [cited 2023 Mar 17]. Available from: https://www.intotheminds.com/blog/en/chatgpt-similarity-with-plan/

[16] Sun, Z., Finnie, G., Weber, K., 2004. Case base building with similarity relations. Information Sciences. 165(1-2), 21-43.

[17] Finnie, G., Sun, Z., 2003. R5 model for case-based reasoning. Knowledge-Based Systems. 16(1), 59-65.

[18] Kantardzic, M., 2011. Data mining: Concepts, models, methods, and algorithms. John Wiley & Sons: Hoboken.

[19] Jordan, M.I., Mitchell, T.M., 2015. Machine learning: Trends, perspectives, and prospects. Science. 349(6245), 255-260.

[20] Epp, S.S., 2010. Discrete mathematics with applications. Cengage Learning: Boston.

[21] Zhang, D.G., Ni, C.H., Zhang, J., et al., 2022. New method of vehicle cooperative communication based on fuzzy logic and signaling game strategy. Future Generation Computer Systems. 142, 131-149.

[22] Finnie, G., Sun, Z., 2002. Similarity and metrics

in case-based reasoning. International Journal of Intelligent Systems. 17(3), 273-287.

[23] Zhang, D., Wang, W., Zhang, J., et al., 2023. Novel edge caching approach based on multi-agent deep reinforcement learning for Internet of vehicles. IEEE Transactions on Intelligent Transportation Systems. 24(6), 1-16.

[24] Klawonn, F., Castro Peña, J.L., 1995. Similarity in fuzzy reasoning. Mathware & Soft Computing. 2(3), 197-228.

[25] Fontana, F.A., Formato, F., 2002. A similarity-based resolution rule. International Journal of Intelligent Systems. 17(9), 853-872.

[26] Biacino, L., Gerla, G., Ying, M., 2000. Approximate reasoning based on similarity. Mathematical Logic Quarterly. 46(1), 77-86.

[27] Kundu, S., 2000. Similarity relations, fuzzy linear orders, and fuzzy partial orders. Fuzzy Sets and Systems. 109(3), 419-428.

[28] Ovchinnikov, S., 1991. Similarity relations, fuzzy partitions, and fuzzy orderings. Fuzzy Sets and Systems. 40(1), 107-126.

[29] Bogacz, R., Giraud-Carrier, C., 2000. A novel modular neural architecture for rule-based and similarity-based reasoning. Hybrid neural systems. Springer: Berlin. pp. 63-77.

[30] Hüllermeier, E., 2001. Similarity-based inference as evidential reasoning. International Journal of Approximate Reasoning. 26(2), 67-100.

[31] Sun, Z., 2017. A logical approach to experience-based reasoning. New Mathematics and Natural Computation. 13(1), 21-40.

[32] Loia, V., Senatore, S., Sessa, M.I., 2004. Combining agent technology and similarity-based reasoning for targeted E-mail services. Fuzzy Sets and Systems. 145(1), 29-56.

[33] Sun, Z., Finnie, G., Weber, K., 2005. Abductive case-based reasoning. International Journal of Intelligent Systems. 20(9), 957-983.

[34] Magnani, L., 2011. Abduction, reason and science: Processes of discovery and explanation. Springer Science & Business Media: Berlin.

[35] Baral, C., 2000. Abductive reasoning through filtering. Artificial Intelligence. 120(1), 1-28.

[36] Console, L., Dupré, D.T., Torasso, P., 1991. On the relationship between abduction and deduction. Journal of Logic and Computation. 1(5), 661-690.

[37] Sun, Z., Finnie, G., Sun, J. (editors), 2005. Four new fuzzy inference rules for experience based reasoning. Fuzzy Logic, Soft Computing and Computational Intelligence (IFSA2005); 2005 May 30; Beijing. p. 188-193.

[38] Reeves, S., Clarke, M., 1990. Logic for computer science. Addison-Wesley: Wokingham.

[39] Hurley, P.J., 2000. A concise introduction to logic. Thomson Learning: Wadsworth.

[40] Dosen, K., 1993. A historical introduction to substructrual logics. Substructrual logics. Clarendon Press: Oxford. pp. 1-30.

[41] Sun, Z., 2022. Problem-based Computing and Analytics. International Journal of Future Computer and Communication.11(3), 52-60.

[42] Sun, Z., Stranieri, A., 2021. The nature of intelligent analytics. Intelligent analytics with advanced multi-industry applications. IGI-Global: Hershey. pp. 1-22.

[43] Sun, Z., Pambel, F., Wu, Z., 2022. The elements of intelligent business analytics: Principles, techniques, and tools. Handbook of research on foundations and applications of intelligent business analytics. IGI-Global: Hershey. pp. 1-20.

[44] Iantovics, L.B., Kountchev, R., Crișan, G.C., 2019. ExtrIntDetect—A new universal method for the identification of intelligent cooperative multiagent systems with extreme intelligence. Symmetry. 11(9), 1123.

[45] Sun, Z., Finnie, G., 2016. A Similarity Based Approach to Experience Based Reasoning (Prepprint). Available: https://ro.uow.edu.au/

[46] Sun, Z., Finnie, G., 2005. MEBRS: A multiagent architecture for an experience based reasoning system. Knowledge-based intelligent information and engineering systems. Springer: Berlin. pp. 972-978.

BILINGUAL PUBLISHING GROUP

Journal of Computer Science Research

https://journals.bilpubgroup.com/index.php/jcsr

ARTICLE

Development of New Machine Learning Based Algorithm for the Diagnosis of Obstructive Sleep Apnea from ECG Data

Erdem Tuncer

Faculty of Technology, Biomedical Eng. Department of Kocaeli University, Kocaeli, 41001, Turkey

ABSTRACT

In this study, a machine learning algorithm is proposed to be used in the detection of Obstructive Sleep Apnea (OSA) from the analysis of single-channel ECG recordings. Eighteen ECG recordings from the PhysioNet Apnea-ECG dataset were used in the study. In the feature extraction stage, dynamic time warping and median frequency features were obtained from the coefficients obtained from different frequency bands of the ECG data by using the wavelet transform-based algorithm. In the classification phase, OSA patients and normal ECG recordings were classified using Random Forest (RF) and Long Short-Term Memory (LSTM) classifier algorithms. The performance of the classifiers was evaluated as 90% training and 10% testing. According to this evaluation, the accuracy of the RF classifier was 82.43% and the accuracy of the LSTM classifier was 77.60%. Considering the results obtained, it is thought that it may be possible to use the proposed features and classifier algorithms in OSA classification and maybe a different alternative to existing machine learning methods. The proposed method and the feature set used are promising because they can be implemented effectively thanks to low computing overhead.

Keywords: ECG; Sleep apnea; Classification; Dynamic time warping; Median frequency

1. Introduction

Obstructive sleep apnea (OSA) is a sleep-related breathing disorder. It becomes evident with the obstructions in the upper respiratory tract during sleep and the waking periods following these obstructions. OSA can seriously reduce a person's quality of daily life and cause the development of many cardiovascular diseases. Therefore, early diagnosis and treatment of obstructive sleep apnea is important. Electrocardi-

*CORRESPONDING AUTHOR:
Erdem Tuncer, Faculty of Technology, Biomedical Eng. Department of Kocaeli University, Kocaeli, 41001, Turkey; Email: erdemtuncerr@gmail.com

ARTICLE INFO
Received: 6 June 2023 | Revised: 4 July 2023 | Accepted: 5 July 2023 | Published Online: 14 July 2023
DOI: https://doi.org/10.30564/jcsr.v5i3.5762

CITATION
Tuncer, E., 2023. Development of New Machine Learning Based Algorithm for the Diagnosis of Obstructive Sleep Apnea from ECG Data. Journal of Computer Science Research. 5(3): 15-21. DOI: https://doi.org/10.30564/jcsr.v5i3.5762

ogram (ECG) is the process of recording the electrical activity of the heart. In today's conditions, ECG signals are used in the diagnosis of OSA. Apnea diagnosis from the ECG signal is measured by heart rate variability. It will be economical and practical to determine whether a person has OSA syndrome with the proposed machine learning technique using single-channel ECG recordings. Because with such a system, there will be no need for environments such as sleep laboratories [1,2]. There are many studies in the literature on the detection of OSA from ECG using methods. In the study conducted by Yildiz [3], obstructive sleep apnea data from ECG recordings were classified. Twelve features were obtained using wavelet transform and they achieved the highest success rate of 98.3% with the support vector machine/artificial neural network classifier algorithms. In the study by Faal et al. [4], they presented a new feature generation method using autoregressive integrated moving average and exponential generalized autoregressive conditional heteroscedasticity model in the time domain from ECG signals. ECG signals were analyzed in one-minute segments. The results were evaluated using five different classifiers (support vector machine, neural network, quadratic separation analysis, linear separation analysis and k-nearest neighbor). As a result of the classification, a success rate of 81.43% was achieved. Tyagi et al. [5] proposed a new approach to cascade two different types of restricted boltzmann machines in the deep belief networks method for sleep apnea classification using electrocardiogram signals. They achieved a success rate of 89.11% from the ECG data examined in one-minute epochs. Yang et al. [6] proposed a one-dimensional compression and excitation residual group network for sleep apnea detection. With the proposed method, an accuracy rate of 90.3% was achieved. Thus, they argued that cheap and useful sleep apnea detectors can be integrated with wearable devices.

The aim of this study is to present an automatic machine-learning method that can detect OSA from ECG recordings. In the proposed method, a wavelet transform-based algorithm is proposed. Unlike the studies in the literature, it is the examination of the effect of different features on apnea data instead of the features frequently used in the literature.

2. Materials and methods

2.1 Data set

The ECG recordings used in the study were taken from the PhysioNet Apnea-ECG dataset. There are 70 ECG recordings in total. Recordings can take up to 10 hours in length. All of the sleep recordings were taken from 32 subjects. The age range of the subjects was between 27 and 63 years. The standard V2 lead was used for the placement of the electrodes on the body surface during recording. ECGs were digitized by sampling at 16 bits per sample and 100 Hz. ECG signals with 16-bit resolution. Evaluation of whether the ECG recordings belong to people with obstructive sleep apnea was made according to the sleep study technique [7]. In this study, 18 ECG recordings of 10 randomly selected patients (a01, a02, a03, a04, a05, a06, a07, a08, a09, a10) were used. The randomly selected apnea and normal ECG data signal form is given in **Figure 1**. In **Figure 1(a)**, heart rate variability is visually striking after the 4000th sample. In **Figure 1(b)**, the normal one-minute ECG signal form is given.

2.2 Feature selection

Discrete wavelet transform

The discrete wavelet transform aims to solve the fixed width window source problem of the fourier transform by using a scalable wavelet function. Thus, optimum time-frequency resolution is provided in different frequency ranges for the biomedical signals to be analyzed. With the discrete wavelet transform, it is aimed to eliminate the excessive computational load. Since an efficient algorithm based on filters has been developed in the discrete wavelet transform, the calculation of the wavelet coefficients is made for discrete values at certain points. This algorithm, called multiple resolution, consists of sequential high-pass and low-pass filter pairs [8,9]. The lower fre-

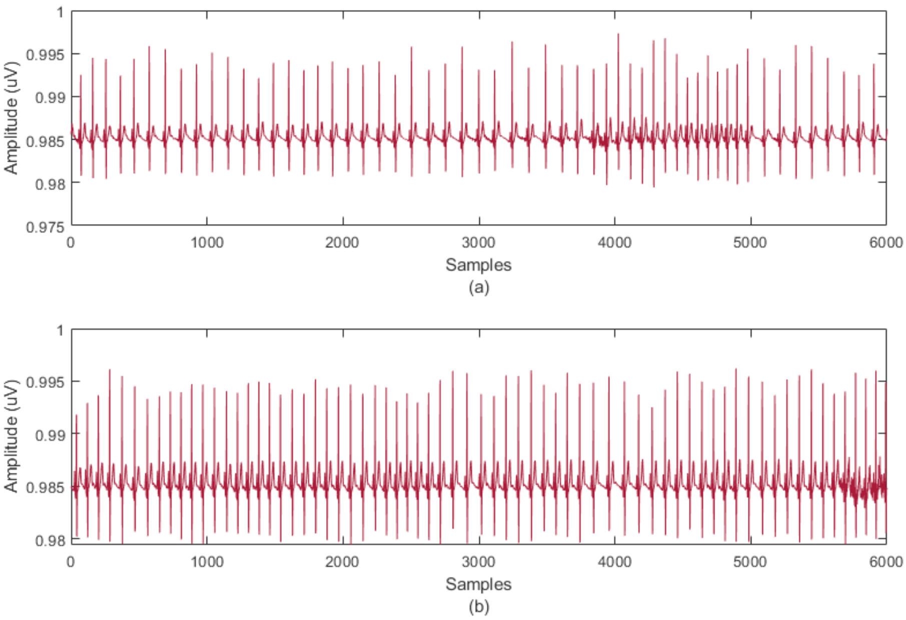


Figure 1. (a) ECG sign with apnea, (b) Normal ECG sign.

quency bands of the ECG data used in the study are given in **Table 1**. As shown in **Table 1**, a six-level wavelet transform is used.

Table 1. Ranges of frequency bands in wavelet transform decomposition of ECG signal.

Sub-bands	Frequency ranges (Hz)
D1	25-50
D2	12.5-50
D3	6.25-12.5
D4	3.125-6.25
D5	1.5625-3.125
D6	0.78125-1.5625
A6	0-0.78125

Dynamic time warping algorithm

Dynamic time warping algorithm is a classification algorithm that uses similarity measurement of time series. Biomedical signals sampled over a period of time form a time series. The similarity between the series can be calculated by finding the sum of the Euclidean distances between the elements of each element of two discrete time series. The closer the Euclidean distance sum is to zero, the more similar the time series are. Today, the dynamic time-warping algorithm is used in many areas from image processing to audio processing [10,11].

$$Q = q1, q2, …., qn\text{-}1, qn \quad (1)$$

$$C = c1, c2, …., cm\text{-}1, cm \quad (2)$$

Q and C in Equation (1) and Equation (2) represent two different signals or data; n and m indicate the lengths of these signals. The similarity ratio between the Q and C signals is calculated using the Euclidean length as in Equation (3).

$$d(q_i, c_j) = (q_i, c_j)^2 \quad (3)$$

After obtaining the (i, j) matrix for Q and C, the accumulated distance matrix is calculated using this matrix. d represents the accumulated cost matrix and is calculated recursively [12].

Median frequency

Power spectral density is the frequency domain equivalent of the power content of the signal. It is used to characterize broadband random signals. The median frequency represents the midpoint of the power spectral density distribution and is the name given to the frequencies above and below that make-up 50% of the total power in the ECG [13,14].

2.3 Classification

Random forest

Random Forest (RF) is a very popular learning algorithm for classification and regression problems.

The RF algorithm is to generate a large number of unbiased decision trees where each tree votes for a class. The Gini index is used to construct the decision trees and determine the last class in each tree. Therefore, the Gini index Gini (v) at node v measures the purity of v. It is expressed by the formula in Equation (4) [15,16].

$$\text{Gini (v)} = \sum_{i=1}^{k} f_i(1 - f_i) \tag{4}$$

Here fi is the fraction of class i recorded at node v.

Long short-term memory

Introduced by Hochreiter and Schmidhuber, Long Short-Term Memory (LSTM) is an advanced variant of the Recurrent Neural Network (RNN) architecture. The basic structure of LSTM is that it uses a memory cell to remember and explicitly span unit outputs at different time steps. The memory cell of LSTM uses cell states to remember the information of temporal contexts. It has a forget gate, an entry gate and an exit gate to control the flow of information between different time steps. The three gates of LSTM make it easy to organize long-term memory. LSTM models can learn the temporal dependence between data. Due to its ability to learn long-term correlations in a sequence, LSTM networks are capable of accurately modeling complex multivariate sequences such as the ECG signal [17,18].

2.4 Evaluation of classification models

One of the performance metrics for the machine learning classification problem is the confusion matrix. **Table 2** contains four different combinations of the value to be estimated and the actual values are called the confusion matrix [19].

Table 2. Confusion matrix.

	Predicted: No	Predicted: Yes
Actual: No	True Negative	False Positive
Actual: Yes	False Negative	True Positive

Here, TP: True positive, TN: True negative, FP: False positive, FN: False negative. Some of the metrics we can calculate with the terms in **Table 2** are accuracy, precision and recall. Their mathematical equations are given in Equations (5), (6) and (7).

$$\text{Accuracy} = \text{TP} + \text{TN}/\text{TP} + \text{FP} + \text{TN} + \text{FN} \tag{5}$$

$$\text{Precision} = \text{TP}/\text{TP} + \text{FP} \tag{6}$$

$$\text{Recall} = \text{TP}/\text{TP} + \text{FN} \tag{7}$$

In addition, the test performance of each classifier was evaluated by calculating statistical parameters.

3. Results

In this study, the machine learning method that can predict the automatic detection of OSA disease, which is time-consuming and costly to diagnose, from single-channel ECG recordings is presented. The flow chart of the proposed method is shown in **Figure 2**.

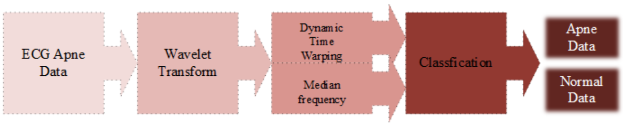


Figure 2. Flow chart of the proposed model.

In the presented method, ECG data were analyzed in one-minute windows. The coefficients of the lower frequency bands were obtained from each window data by using the wavelet transform (6-level Symlet2 wavelet). After applying the dynamic time-warping algorithm to the wavelet coefficients in different frequency bands, the results obtained are recorded in the feature matrix. The relationship of the A6 coefficients with the other coefficients was evaluated with the dynamic time-warping algorithm. Another parameter calculated as a feature is the median frequency. The median frequency values of the wavelet coefficients obtained from all lower frequency bands were calculated. As shown in **Table 3**, a total of 13 features were extracted and given as input to the classifier algorithms.

In this study, two different classifier algorithms were evaluated. One is the deep learning architecture LSTM and the other is the traditional learning algorithm RF. The architecture of the model created in the LSTM classifier is shown in **Figure 3**. LSTM architecture layers are composed of input layer, LSTM layer, dropout layer, LSTM layer, dropout layer and output layer, respectively. The LSTM layer contains 50 units per layer. These units use the Corrected

Table 3. Feature list.

No	Feature name	Difference between wavelet coefficients	No	Feature name	Wavelet coefficient
1	Dynamic Time Warping	A6, D1	7	Median frequency	D1
2	Dynamic Time Warping	A6, D5	8	Median frequency	D2
3	Dynamic Time Warping	A6, D4	9	Median frequency	D3
4	Dynamic Time Warping	A6, D3	10	Median frequency	D4
5	Dynamic Time Warping	A6, D2	11	Median frequency	D5
6	Dynamic Time Warping	A6, D1	12	Median frequency	D6
			13	Median frequency	A6

Linear Unit (ReLU) activation function and give a different output for each time step. The reason for using ReLU is that it is generally less costly to train the model in terms of computational load and can achieve better performance than other models. In addition, ReLU can avoid the vanishing gradient problem, which is an advantage over the tanh function.

After the first LSTM layer, the dropout layer (with a value of 0.2) is applied to reduce overfitting. The next layer is a new LSTM layer containing 50 units and ReLU activation functions, followed by the dropout layer. Finally, the value containing the classification result is estimated after a sigmoid activation function is used to estimate the result with the output layer.

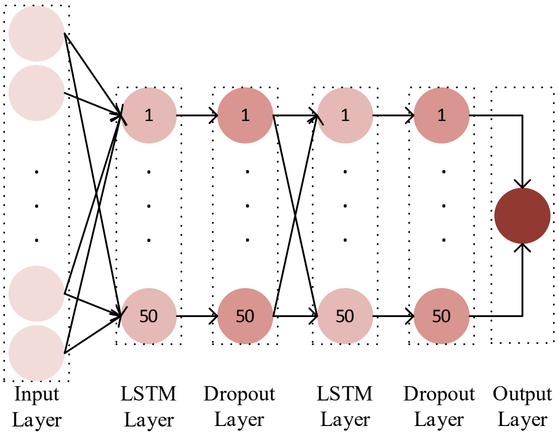


Figure 3. Diagram of the LSTM architecture.

The classification accuracy of the RF method depends on user-defined parameters such as the number of trees and the number of parameters. Therefore, the selection of the most appropriate parameter for the data increases the classification accuracy. In the study, multiple combinations were tested to find the optimum parameters of tree and parameter numbers. The success rates obtained for combinations of different tree and parameter numbers are shown in **Table 4**. As can be seen in **Table 4**, the number of trees with the highest success rate was selected as 250 and the number of parameters as two for the classification of apnea data. Since increasing the number of trees does not increase the performance of the model, the model with the highest performance with the least number of trees was selected.

Table 4. RF algorithm success results by parameters.

Number of trees	Number of parameters	Accuracy rate (%)
10	2	78.15
20	2	80.13
30	2	80.46
40	2	80.57
50	2	80.79
70	2	81.22
100	2	81.66
150	2	81.99
200	2	81.88
250	2	82.43
500	2	82.43

The success rates obtained as a result of LSTM architecture and RF architecture are given in **Table 5**. As can be seen from **Table 5**, the optimized RF algorithm performed better than the LSTM architecture. Therefore, the LSTM architecture has the best performance.

Table 5. Classifier performances.

Dataset	Accuracy (%)		Precision (%)		Recall (%)	
	RF	LSTM	RF	LSTM	RF	LSTM
ECG Apnea	82.43	77.60	82.10	76.70	82.40	77.50

4. Discussion

The analyzed results show that it gives the highest accuracy with 82.43% accuracy with the RF algorithm. High classification performance was achieved with thirteen features obtained by using two features from ECG data. When the studies in the literature were examined, the norm entropy values of each wavelet level were calculated by using the twelve-level wavelet transform of the obstructive sleep apnea data from the ECG recordings in the study conducted by Yildiz [3]. The obtained features were applied to the support vector machine/artificial neural network classifier algorithms and the highest success rate of 98.3% was obtained. In the study by Faal et al. [4], they presented a new feature generation method. As a result of five different classifier algorithms, a success rate of 81.43% was achieved. Tyagi et al. [5] proposed a new approach and achieved a success rate of 89.11%. Yang et al. [6] proposed a one-dimensional compression and excitation residual group network and 90.3% accuracy was achieved with the proposed method. In the study by Razi et al. [20], ten-time domain features were extracted and reduced to five features. Principal component analysis and discriminant linear analysis were used for size reduction. RF algorithm is proposed for classification and the results are compared with other classifier algorithms. The highest success rate detected is 95.01%.

When the studies in the literature are examined, it is observed that the success rates are generally higher than the study of this article. Most of the studies aimed to reach a higher success rate by using similar methods and techniques. However, in the field of machine learning, the goal is not only to increase classification success but also to develop different features and method techniques. From this point of view, our article differs from the studies in the literature. A previously unused feature set is suggested on ECG apnea data. At the same time, the change in success rates with the optimization of the classifier algorithms was examined. It is possible to reach higher success rates by diversifying and optimizing the parameters of the machine learning model.

5. Conclusions

This article discusses the estimation of apnea diagnosis from ECG data. We propose a binary classification machine learning method to support physicians' decisions in clinical practice. For decision support applications, modeling using the RF algorithm as a classifier and classification of patients' apnea data are recommended. It has been seen that the feature method selected with the RF algorithm is successful. In the classification made with the used feature set and RF algorithm optimization, a successful prediction was made with 13 features with an accuracy rate of 82.43%. The feature set and method we used in our study give hope for higher future success rates. In further studies, it is aimed to evaluate the efficiency of the feature set by expanding the dataset.

Conflict of Interest

The author has no conflicts of interest to declare.

Funding

This research received no external funding.

References

[1] Wiegand, L., Zwillich, C.W., 1994. Obstructive sleep apnea. Disease-a-Month. 40(4), 202-252. DOI: https://doi.org/10.1016/0011-5029(94)90013-2

[2] Paiva, T., Attarian, H., 2014. Obstructive sleep apnea and other sleep-related syndromes. Handbook of clinical neurology. Elsevier: Amsterdam. pp. 251-271.

[3] Yildiz, A., 2017. Tek kanallı EKG kayıtları analizinden uyku apne tespiti (Turkish) [Detection of sleep apnea from analysis of single-channel ECG recordings]. Dicle Üniversitesi Mühendislik Fakültesi Mühendislik Dergisi. 8(1), 111-122.

[4] Faal, M., Almasganj, F., 2021. Obstructive sleep apnea screening from unprocessed ECG signals using statistical modelling. Biomedical Signal Processing and Control. 68, 102685.
DOI: https://doi.org/10.1016/j.bspc.2021.102685

[5] Tyagi, P.K., Agrawal, D., 2023. Automatic detection of sleep apnea from single-lead ECG signal using enhanced-deep belief network model. Biomedical Signal Processing and Control. 80(2), 104401.
DOI: https://doi.org/10.1016/j.bspc.2022.104401

[6] Yang, Q., Zou, L., Wei, K., et al., 2022. Obstructive sleep apnea detection from single-lead electrocardiogram signals using one-dimensional squeeze-and-excitation residual group network. Computers in Biology and Medicine. 140, 105124.
DOI: https://doi.org/10.1016/j.compbiomed.2021.105124

[7] Penzel, T., Moody, G.B., Mark, R.G., et al. (editors), 2000. The Apnea-ECG database. Computers in Cardiology. 2000 September 24-27; Cambridge. USA: IEEE. p. 255-258.

[8] Tuncer, E., Bolat, E.D., 2022. Destek Vektör Makinaları ile EEG Sinyallerinden Epileptik Nöbet Sınıflandırması (Turkish) [Epileptic seizure classification from EEG signals with support vector machines]. Politeknik Dergisi. 25(1), 239-249.
DOI: https://doi.org/10.2339/politeknik.672077

[9] Mallat, S.G., 1989. A theory for multiresolution signal decomposition: The wavelet representation. IEEE Transactions on Pattern Analysis and Machine Intelligence. 11(7), 674-693.
DOI: http://dx.doi.org/10.1109/34.192463

[10] Zhang, Z., Tavenard, R., Bailly, A., et al., 2017. Dynamic time warping under limited warping path length. Information Sciences. 393, 91-107.
DOI: https://doi.org/10.1016/j.ins.2017.02.018

[11] Jeong, Y.S., Jeong, M.K., Omitaomu, O.A., 2011. Weighted dynamic time warping for time series classification. Pattern Recognition. 44(9), 2231-2240.
DOI: https://doi.org/10.1016/j.patcog.2010.09.022

[12] Bakir, C., 2016. Automatic speaker gender identification for the German language. Balkan Journal of Electrical and Computer Engineering. 4(2), 79-83.
DOI: https://doi.org/10.17694/bajece.43067

[13] Brown, C.G., Griffith, R.F., Ligten, P.V., et al., 1991. Median frequency—a new parameter for predicting defibrillation success rate. Annals of Emergency Medicine. 20(7), 787-789.
DOI: https://doi.org/10.1016/S0196-0644(05)80843-1

[14] Tonner, P.H., Bein, B., 2006. Classic electroencephalographic parameters: Median frequency, spectral edge frequency etc. Best Practice & Research Clinical Anaesthesiology. 20(1), 147-159.
DOI: https://doi.org/10.1016/j.bpa.2005.08.008

[15] Masetic, Z., Subasi, A., 2016. Congestive heart failure detection using random forest classifier. Computer Methods and Programs in Biomedicine. 130, 54-64.
DOI: https://doi.org/10.1016/j.cmpb.2016.03.020

[16] Coskun, G., Aytekin, I., 2021. Early detection of mastitis by using infrared thermography in holstein-friesian dairy cows via classification and regression tree (CART) Analysis. Selcuk Journal of Agriculture and Food Sciences. 35(2), 118-127.

[17] Tuncer, E., Bolat, E.D., 2022. Classification of epileptic seizures from electroencephalogram (EEG) data using bidirectional short-term memory (Bi-LSTM) network architecture. Biomedical Signal Processing and Control. 73, 103462.
DOI: https://doi.org/10.1016/j.bspc.2021.103462

[18] Sowmya, S., Jose, D., 2022. Contemplate on ECG signals and classification of arrhythmia signals using CNN-LSTM deep learning model. Measurement: Sensors. 24, 100558.
DOI: https://doi.org/10.1016/j.measen.2022.100558

[19] Dağli, E., Büber, M., Taspinar, Y.S., 2022. Detection of accident situation by machine learning methods using traffic announcements: The case of metropol Istanbul. International journal of applied mathematics electronics and computers. 10(3), 61-67.

[20] Razi, A.P., Einalou, Z., Manthouri, M., 2021. Sleep Apnea classification using random forest via ECG. Sleep and Vigilance. 5, 141-146.
DOI: https://doi.org/10.1007/s41782-021-00138-4

BILINGUAL PUBLISHING GROUP

Journal of Computer Science Research

https://journals.bilpubgroup.com/index.php/jcsr

ARTICLE

Enhancing Human-Machine Interaction: Real-Time Emotion Recognition through Speech Analysis

Dominik Esteves de Andrade **,** ***Rüdiger Buchkremer****

Institute of IT Management and Digitization Research (IFID), FOM University of Applied Sciences, Dusseldorf, 40476, Germany

ABSTRACT

Humans, as intricate beings driven by a multitude of emotions, possess a remarkable ability to decipher and respond to socio-affective cues. However, many individuals and machines struggle to interpret such nuanced signals, including variations in tone of voice. This paper explores the potential of intelligent technologies to bridge this gap and improve the quality of conversations. In particular, the authors propose a real-time processing method that captures and evaluates emotions in speech, utilizing a terminal device like the Raspberry Pi computer. Furthermore, the authors provide an overview of the current research landscape surrounding speech emotional recognition and delve into our methodology, which involves analyzing audio files from renowned emotional speech databases. To aid incomprehension, the authors present visualizations of these audio files in situ, employing dB-scaled Mel spectrograms generated through TensorFlow and Matplotlib. The authors use a support vector machine kernel and a Convolutional Neural Network with transfer learning to classify emotions. Notably, the classification accuracies achieved are 70% and 77%, respectively, demonstrating the efficacy of our approach when executed on an edge device rather than relying on a server. The system can evaluate pure emotion in speech and provide corresponding visualizations to depict the speaker's emotional state in less than one second on a Raspberry Pi. These findings pave the way for more effective and emotionally intelligent human-machine interactions in various domains.

Keywords: Speech emotion recognition; Edge computing; Real-time computing; Raspberry Pi

*CORRESPONDING AUTHOR:

Rüdiger Buchkremer, Institute of IT Management and Digitization Research (IFID), FOM University of Applied Sciences, Dusseldorf, 40476, Germany; Email: ruediger.buchkremer@fom.de

ARTICLE INFO

Received: 7 June 2023 | Revised: 7 July 2023 | Accepted: 10 July 2023 | Published Online: 21 July 2023

DOI: https://doi.org/10.30564/jcsr.v5i3.5768

CITATION

Esteves de Andrade, D., Buchkremer, R., 2023. Enhancing Human-Machine Interaction: Real-Time Emotion Recognition through Speech Analysis. Journal of Computer Science Research. 5(3): 22-45. DOI: https://doi.org/10.30564/jcsr.v5i3.5768

1. Introduction

A phenomenon that transcends both professional and personal domains is the growing amalgamation of machines, which aims to foster human connection. Numerous individuals cannot decipher socio-affective cues, such as nuances in tone of voice. The challenge is to ensure that gestures, facial expressions, and paralinguistic information such as volume, frequency, and intonation that make up communication are not lost. Therefore, incorporating emotions into interface design becomes indispensable, as people tend to exhibit social behaviors during interactions with machines. Nonverbal communication often carries pivotal information in a typical conversation, revealing the speaker's intentions. Apart from the semantic content conveyed through text, how words are expressed imparts significant nonverbal cues. The precise delivery of spoken words, accompanied by appropriate emotions, can bring a special message altogether.

Consequently, el Ayadi et al. [1] elucidate that humanity remains distant from achieving natural interaction with machines, particularly in comprehending the emotional states of counterparts. At this juncture, the emergence of emotion recognition technologies becomes pivotal, encompassing various methods and technologies that enable the recognition of emotions beyond human perception. The primary objective of emotion recognition is to allow a system to adapt its response when certain emotions, such as frustration or anger, are detected. In 2001, Corvie et al. [2] expound on the two channels of communication present in every human interaction: the explicit and the implicit. While the explicit channel conveys messages, the implicit channel reveals the speaker's underlying feelings and moods. They explain that extensive research has been conducted to comprehend the explicit channel, whereas the implicit channel, though less explored, holds great significance in understanding speakers and their emotional states.

Machines typically exhibit neutral behavior, which humans may perceive as indifference. Hence, devices need to recognize emotions conveyed through speech to interact effectively. This ability is commonly referred to as Speech Emotional Recognition (SER). Schuller [3] states that even animals can perceive the tonality of human speech, suggesting that the time has come for machines to possess this capability. Kraus [4] asserts that distinguishing pure voice communication from visual or audiovisual communication is vital in determining a person's empathy.

In 2020, Akçay and Oğuz [5] commented that although real-time emotion recognition through SER systems is technically feasible, it has not yet become a ubiquitous part of daily life, unlike speech recognition systems. Implementing a computer system necessitates considering both economic and ecological factors. Energy supply and usage issues are intertwined with global warming and environmental concerns [6]. Thus, an edge emotion recognition machine must consume minimal energy while operating in real time.

To achieve near real-time operation, a machine requires a highly optimized processor performance, short transmission paths, and low latency times. These three conditions constitute essential components of edge computing. Unlike cloud computing, which transfers data to a centralized location for processing, edge computing brings computational power closer to the data. Mao et al. [7] argue that cloud computing is unsuitable for latency-critical mobile applications due to the distance between the user and the data center, resulting in significant delays. Abbas et al. [8] explain that cloud computing is unsuitable for real-time applications such as augmented reality or car-to-car communication and thus supports the edge computing approach. Cao et al. [9] report that over 50 billion end devices are connected to the Internet and thus to each other, producing a data volume of 40 zettabytes. It includes mobile and ambient end devices such as smartphones, smart speakers, or Raspberry Pis. For all these network participants to act and communicate with each other in near real-time, computing power must be shifted closer to the data.

2. Related work

The introduction of cloud computing was a milestone in the early 2000s, enabling new business models and innovations. However, the era of cloud computing seems to be ending as the edge computing paradigm is increasingly replacing the cloud computing paradigm due to new requirements. Edge computing can support the new requirements for low latency, increased data security, mobility support, and real-time processing. The literature divides edge computing into the sub-areas of fog computing, cloudlet, and mobile edge computing (MEC). While the first two approaches mentioned are hardly found in practice, MEC is ubiquitous. In MEC, computationally intensive cloud servers are stationed in mobile base stations at the network's edge and thus close to the end devices, ensuring daily use of this technology. As Shi et al. [10] stated, MEC means data processing immediately to the end device and on it. In addition to MEC, mobile cloud computing (MCC) is based on the principle that end devices perform the processing and only send the result or partial result to the MEC server or the MCC server. However, none of these approaches can be found in pure form in practice. Instead, cloud and edge computing techniques are combined to cover various use cases and exploit their advantages.

The topic of speech emotion recognition (SER) and its feature extraction and pattern recognition are a constant part of current research. Thus, the recent literature review shows that in SER, especially the continuous and the spectral features of speech are used since these reflect the characteristics of emotions most appropriately. Priority is given to the course of the primary speech frequency or loudness, the temporal ratios, pauses, and spectral features such as the Mel frequency cepstral coefficient (MFCC) and the Mel spectrograms [3]. The most common classification techniques used in speech recognition in recent years are the Gaussian Mixture Model (GMM) in combination with the Hidden Markov Model (HMM), the support vector machine (SVM) (Cortes and Vapnik 1995), and more recently, neural networks [11,12]. Consequently, the successes achieved in this regard also inspired using these techniques in SER, but with a focus on neural networks, SVM, or any combination of these two. Studies reveal that even pure emotion determination by humans is not accurate in all cases, so the focus is on the use and further development of neural networks [13]. In the field of neural networks, recurrent neural networks (RNNs) such as Long Short-Term Memory Hochreiter and Schmidhuber [14] were initially used because their feedback loops make them more suitable for processing continuous inputs such as speech signals [15,16]. RNNs have been superseded by convolutional neural networks (CNNs) such as AlexNet, VGG16, ResNet, or MobileNetV2 due to their high resource and memory requirements and continued success. Furthermore, MFCC or Mel spectrograms were launched using a Convolutional Neural Network (CNN). Moreover, the everyday use of transfer learning and Multitask Learning methods makes the CNN deployment even more efficient [17].

Every pattern recognition is based on previously extracted features in considerable quantity and quality. Due to this given diversity, selecting suitable parts is relevant in classification. The method generally used in machine learning for feature extraction is the use of the framework open-source Speech and Music Interpretation by Large-space Extraction (openSMILE) [18], which in turn includes the datasets extended Geneva Minimalistic Acoustic Parameter Set (eGeMAPS) and ComParE. In deep learning, recent literature has increasingly used CNN for this purpose. In this approach, the output layer is either preserved as a classifier or replaced by, for example, an SVM.

In the phase of emotion classification, diverse sets of emotions diverge, which in turn harbor a different number of emotions. The settings can vary from five to 20 other emotions. The most common set of emotions in the literature refers to the six basic emotions, according to Ekman [19], which are happiness, sadness, anger, fear, disgust, and surprise, including a seventh neutral emotion.

In the mainstream literature, descriptions of the hardware on which a neural network is trained or executed are scarce. However, Tariq et al. [15] describe that neural networks—especially deep neural networks run in cloud-like data centers. The locally collected data is transferred to these servers, deleted on the local device, processed on the servers, and only the result is sent back to the end device. Thus, applying neural networks in the context of MEC and real-time capability represents a novelty. Despite the intensive research on these topics, no everyday emotion recognition products currently exist.

3. Materials and methods

We employ labeled emotional speech data for the prototypical implementation. Audio files with a minimum length of one second but a maximum length of 20 seconds are considered for use. Most emotion databases refer to six basic emotions [20]. Considering arousal and valence dimensions is not part of our work, which is why these criteria are neglected in data acquisition. Since part of this work is the emotion recognition in speech, but human speech is divided into sentences from which the emotions emerge, the audio length of one to 20 seconds is subjectively chosen since most sentences are spoken within this period. Thus, the audio files must still contain spoken sentences without singing, noise, or the like. However, the native language is not a selection criterion since emotions are expressed in any language. Even though the speaking gender is not an immediate selection criterion, the totality of all databases must contain both male and female spoken sentences to allow for the generalization of the data. In addition, the stored channel number or sampling rate is irrelevant in data acquisition, as these are standardized in the training process. Finally, the audio files and databases must be freely accessible and identified by labels. Thus, the following audio databases are placed that meet the given quality criteria:

1) Ryerson Audiovisual Database of Emotional Speech and Song (RAVDESS) [21]

2) Berlin Database of Emotional Speech (Emo-DB) [22]

3) Toronto Emotional Speech Set (TESS) [23]

4) EMOVO [24]

5) eNTERFACE'05 [25]

The eNTERFACE'05 database, in particular, holds data in an audiovisual format, whereas the remaining databases are in pure auditory waveform format (WAV). The audio part of the database eNTERFACE'05 is extracted from the audiovisual files to use the required audio data.

In **Table 1**, a detailed overview of databases is presented. While RAVDESS, eNTERFACE'05, TESS, and EMOVO reflect the six basic emotions, Emo-DB contains only five. Except for eNTERFACE'05, all other databases have a neutral emotion. Only RAVDESS and Emo-DB include any other emotions beyond these seven listed. However, since most databases contain the six primary and neutral emotions, the prototype will classify only those.

Hence, our dataset comprises 6656 audio files from 140 different references, amounting to a cumulative playback time of 5.14 hours. On average, each file has a duration of 2.97 seconds. Within the file names, emotions are encoded either as complete textual representations, abbreviations, or numerical values. The number of files and their total length per database vary considerably. **Figure 1** illustrates the distribution of emotions across all the acquired databases, demonstrating a relatively balanced allocation of emotion labels. While the neutral emotion category is slightly underrepresented, this discrepancy is compensated for during model training by appropriately adjusting the hyperparameters.

Moreover, the distribution of audio file durations per database is depicted in **Figure 2** using boxplots that exclude outliers. Notably, the eNTERFACE'05 database exhibits six outliers surpassing the upper threshold of 20 seconds. To preserve the readability of the boxplot representation, these outliers are omitted. However, it is worth mentioning that most files in the eNTERFACE'05 database have a maximum length of less than 20 seconds. Consequently, this database remains a foundational component for the prototype.

Table 1. Overview of the speech audio databases employed.

Database	Number of files	Min. length in sec.	Max. Length in sec.	Average length in sec.	Total length in minutes	Language	Emotions
RAVDESS	1440	2.94	5.27	3.7	88.82	English	Neutral, calm, joy, sadness, anger, fear, disgust, surprise
Emo-DB	535	1.23	8.98	2.78	24.79	German	Neutral, joy, sadness, anger, fear, disgust, boredom
TESS	2800	1.25	2.98	2.06	95.91	English	Neutral, joy, sadness, anger, fear, disgust, surprise
EMOVO	588	1.29	13.99	3.12	30.59	Italian	Neutral, joy, sadness, anger, fear, disgust, surprise
eNTERFACE'05	1293	1.12	106.92	3.17	68.37	English	Joy, sadness, anger, fear, disgust, surprise

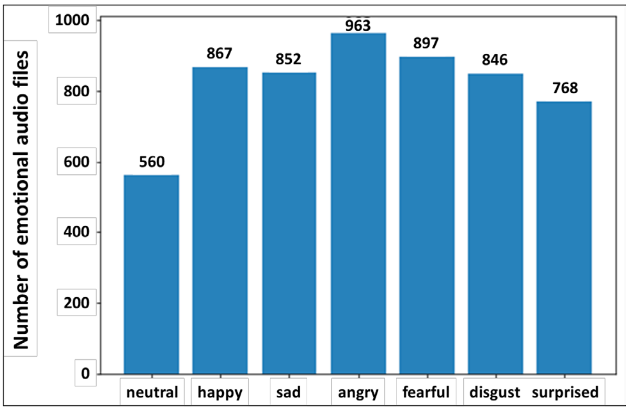


Figure 1. The overall distribution of the seven emotions across the databases was visualized with Matplotlib.

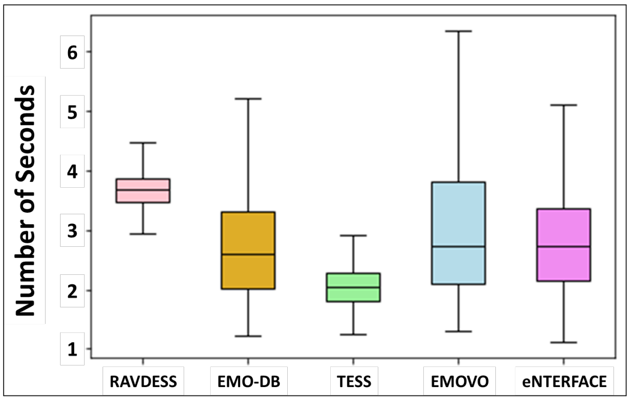


Figure 2. Boxplot representation of the databases used without outliers visualized with Matplotlib.

In recent studies, audio files with a sampling rate of 16000 hertz and mono tracks are primarily used [11,26-28]. As a result, the audio files of our data corpus are transformed to that uniform format.

A model is developed through algorithms that employ pattern recognition to classify recorded audio inputs into specific activities or emotions. Specifically, two distinct models are utilized—one using a machine learning algorithm, while the other utilizes a deep learning algorithm. It is important to note that, as indicated by Shinde and Shah [29], these algorithms are not synonymous; instead, the latter is a subtype of the former. A noteworthy distinction between the two lies in the requirement for specifying hyperparameters in the machine learning algorithm, whereas the deep learning algorithm automatically determines and optimizes these hyperparameters. Once trained, the model is deployed on ambient devices to assess the feasibility of implementing such an application using edge computing. The real-time capability of the prototype is determined by measuring its processing time.

The implementation of the prototypes in this research is carried out using Python [30]. As mentioned earlier, an essential aspect of Speech Emotion Recognition (SER) is the availability of suitable language data within the system. Consequently, implementing SER necessitates an initial filtering process that distinguishes between speech and non-speech audio. For this purpose, Hershey et al. [31] describe a neural network called YAMNet in their paper, which is specifically trained on audio classification using the AudioSet database [32]. YAMNet can distinguish between 521 audio classes from human, animal, machine, and natural sources. This publicly available Convolutional Neural Network (CNN) YAMNet serves as the upstream filter for SER and is utilized in both methods described in this work. However, since YAMNet does not impact the main SER algorithm developed in this research, further details regarding its functionality or structure are not provided here.

Distinct terminal configurations are employed during the creation and execution of the prototypes. The machine learning and deep learning models are trained on a Windows server. This step focuses on processing the five databases using the respective method, necessitating hardware utilization with appropriate performance capabilities. It should be noted that servers do not possess microphone inputs due to their general structure, broad physical localization, and clustering, which are also irrelevant for training purposes. **Table 2** lists the hardware components used in the training process.

Once the functional models are prepared, they are transferred to ambient end devices where real-time classification occurs. Typically, these end devices have lower computational power and memory than servers, rendering them unsuitable for training machine learning or deep learning models. However, these devices' internal processors and microphones are well-suited for executing such models. The performance achieved is contingent upon the specific hardware components of each device. **Table 3** presents the ambient terminals employed in this study and their respective specifications. The table encompasses two distinct types of devices chosen to represent each category.

The selection of end devices encompasses various categories, encompassing multiple operating systems, performance levels, and storage capacities. As a result, the chosen range serves as a representative cross-section of the available ambient end devices.

Due to these end devices' distinct architectures and operating systems, specific methods and requirements are necessary for utilizing the trained models. The fundamental prerequisites for deploying the ported models on the end devices include the frame-

Table 2. Server hardware used for model training.

Hardware component	Designation
Rack Server	HPE ProLiant DL380 Gen10
Operating system name	Microsoft Windows Server 2016 Standard
Processor	Intel® Xeon® Gold 6226 CPU @ 2.70GHz
Installed memory	256 GB
System type	64-bit operating system, x64-based processor

Table 3. Terminal devices and their components.

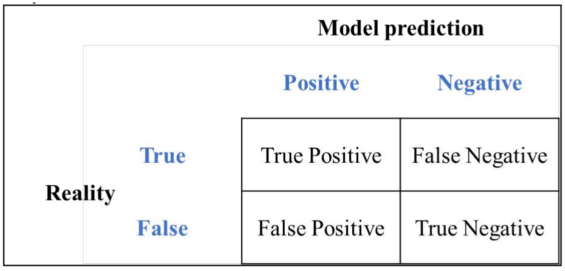
ID	Category	Terminal	Operating system	Processor	Working memory	Battery
1	Notebook	HP Envy x360 Convertible 15-cn0xxx	64-bit Windows 10 Home version 21H1	Intel® Core™ i7-8550U CPU @ 1.80 GHz	16 GB	3-cell, 52-Wh, 4.55-Ah, 11.55V, Li-ion battery
2	Raspberry PI	Raspberry Pi 4 Model B including an additional USB mini microphone	32-bit Raspbian GNU/Linux 11	Broadcom BCM2711 (Cortex-A72, ARM v8), 4-core CPU with 1.5 GHz	4 GB	External power supply

works openSMILE and TensorFlow/TensorFlow Lite alongside the Python programming language.

To measure and evaluate the performance of the prototypes, appropriate metrics are employed, focusing on real-time capability and classification success rate. Real-time capacity is assessed by measuring the response times of the prototypes in seconds. These measurements are conducted on the end devices, commencing immediately after the recording and storing of speech and concluding after classification. It is important to note that the model training and recording time are not considered during this evaluation. However, the exact time frame within which the measured response time must satisfy the criteria for real-time capability in machine processes is not explicitly defined in the literature.

In contrast, the ISO/IEC 2382: 2015 standard defines real-time as the "processing of data by a computer in connection with another process outside the computer according to time requirements imposed by the outside process" (ISO/IEC JTC 2015). Thus, it is apparent from this definition that specifying an exact time in seconds or milliseconds is not feasible. Instead, the external process defines the real-time capability, which may include human perception. Human perception is susceptible to linguistic communication, as pauses of a few milliseconds can be subjectively perceived as interruptions. Vogt et al. [33] suggest that subjective interruption is perceived after 1000 milliseconds. Zhang et al. [34] report that neural networks for image classification require a range of 15.2 to 184 milliseconds for processing, with input dimensions similar to the Deep Learning method utilized in this study (224 × 224 × 3). Furthermore, Liu et al. [35] state that compressed neural networks require only 103 to 189 milliseconds for processing on ambient devices such as smartphones. Consequently, in this prototyping without employing compressed methods, a measured duration of fewer than 1000 milliseconds is considered real-time.

Confusion matrices are commonly employed for representing and evaluating classification problems in machine learning. These matrices juxtapose the model's predictions with the actual states. **Figure 3** illustrates an example of a confusion matrix, depicting the four potential outcomes. The primary distinction lies in whether the model's prediction aligns with reality or deviates from it.



		Model prediction	
		Positive	Negative
Reality	True	True Positive	False Negative
	False	False Positive	True Negative

Figure 3. Example of a confusion matrix based on Davis and Goadrich [36].

The confusion matrix, also named the four-field matrix, does not represent a key figure in the narrower sense but provides the basis for its creation. Thus, the overall accuracy of the respective machine learning system is calculated from the confusion matrix and represented in decimal numbers, where the value 1.0 represents the maximum, and the value 0.0 is the minimum. The accuracy indicates the total number of correct predictions of the model and is determined using the following formula:

$$Accuracy = \frac{True\ Positives + True\ Negalives}{True\ Positives + False\ Pastitves + True\ Negatives + False\ Negatives}$$

For comparison, the previously mentioned CNNs are used, whereby the highest accuracy achieved in each case, as shown in **Table 4**, is deposited. The CNNs mentioned are sorted chronologically by publication date within the table. Besides MobileNetV2, the cited papers do not specify the machine used to generate results. Therefore, for the time being, it is assumed that the results of the CNNs were generated on cloud-like servers, similar to what is described by Tariq et al. (2019). Since a direct comparison of server-generated results with terminal device-generated results is not possible, the subsequent interpretation of the results of this study is limited.

Additional metrics are utilized for neural networks to measure and evaluate training results. These include training and validation accuracy and the duration of training and validation losses. Training and validation accuracy are represented as decimal values, ranging between 0.0 and 1.0, where

1.0 signifies the highest accuracy. Similarly, training and validation losses are expressed as decimal numbers, with no upper limit but a minimum value of 0.0 representing the optimal loss. Consequently, **Table 4** can also be applied to assess the accurate measurement of neural networks in this context. However, in evaluating CNNs, the focus is primarily on accuracy, rendering an evaluation or classification of training losses unnecessary. Accuracy measurement and the creation of confusion matrices occur immediately after training on the server, unlike the measure of classification time.

Table 4. Comparison of prediction accuracy of known CNN models.

CNN	Accuracy	Source
LeNet	82%	LeCun et al. 1998 [37]
AlexNet	84.6%	Krizhevsky et al. 2017 [38]
VGG	93.2%	Simonyan and Zisserman 2015 [39]
ResNet-152	96.43%	He et al. 2016 [40]
MobileNetV2	75.32%	Sandler et al. 2018 [41]

However, assessing the accuracy and elapsed time alone is insufficient to achieve the objectives. It is also essential to determine whether the described methods can be executed on ambient end devices and whether the results are comparable. While the training of the models does not occur on the end devices, the classification process does. To evaluate the performance of a general machine learning method on an end device, Liu et al. proposed criteria such as accuracy, delay, memory requirements, and power requirements. Accuracy is assessed using the confusion matrix and the resulting accuracy score, as mentioned earlier. As explained previously, the delay or temporal duration is determined by the model itself and is expressed in seconds, indicating the time required for one classification cycle. Memory requirements are measured in gigabytes, representing the average memory allocation needed for a cycle, calculated by comparing the working memory usage before and during classification. However, measuring the energy consumption of an application directly is not feasible since an application does not exclusively run on a single system. As a result, the strict identification of the energy demand for a specific application is challenging. Energy consumption can be estimated indirectly by measuring processor utilization. Thus, the difference in measured processor utilization, represented as a percentage, before and during classification is utilized as a metric in this context.

The overall accuracy metric is related to the model and, therefore, independent of the hardware utilized. However, metrics such as time, memory consumption, energy consumption, and processor utilization are hardware-dependent. The metrics mentioned above are evaluated using the hardware listed in **Table 3** in the subsequent analysis.

4. Results

To implement Speech Emotion Recognition (SER), an upstream filter is required to distinguish between speech and non-speech. Hershey et al. [31] introduced a neural network called YAMNet, trained on the AudioSet database [32], which classifies audio into 521 different audio classes, including human, animal, machine, and natural sounds. YAMNet, a freely available Convolutional Neural Network (CNN), serves as the upstream filter for SER in this work without affecting the main SER algorithm. As YAMNet's functionality and structure have been described elsewhere, further details are not provided in this study.

The traditional machine learning algorithm employed in this research follows a supervised learning approach using a data corpus of the five mentioned databases. The objective is to generate a model that can be ported to an end device for classification purposes. The openSMILE framework is utilized for feature extraction in this method, while a Support Vector Machine (SVM) is employed as the classifier. The SVM is trained on the eGeMAPS features extracted from the audio files using openSMILE. The extraction and training processes are applied to the entire data corpus rather than individual databases. After extraction, the parameter dataset is normalized by removing the mean and scaling it to unit variance. Subsequently, the normalized dataset is divided into training and test partitions in an 80:20 ratio. The training partition is used for model training, while the test partition validates the training results.

The relevant hyperparameters for training are optimized and determined by the algorithm itself. Initially, four hyperparameters with specified value ranges are provided. These include the selection of available SVM kernels (polynomial, linear, sigmoidal, and radial basis function), a regulation parameter ranging from 10^{-3} to 10^{2}, and a degree parameter ranging from zero to nine for the polynomial kernel. The algorithm optimizes and applies various combinations of hyperparameters during training on the training partition. Following the training phase, validation is performed using the training dataset.

Upon completion of training, the machine learning system can classify new, unknown data based on the learned generalization. The system is connected to a microphone, which records human speech at 1024 frames per buffer every three seconds. The recorded audio is stored locally in a 16-bit WAV format with a sampling rate of 16000 Hz and a mono channel. The stored file is then read by the machine learning system and processed using YAMNet. If the classification result from YAMNet indicates "human speech", the file is further processed using openSMILE to obtain eGeMAPS features. Similar to the training phase, this dataset is normalized and passed to the SVM for emotion classification. The classification is performed immediately, and the process continues in a continuous three-second cycle, processing the following files until manually terminated.

Figure 4 provides a schematic overview of the mentioned processing steps, indicating the sequence of individual actions. Not all processing steps are executed on the same hardware, and the figure specifies which steps are performed on the server and the end device.

The deep learning algorithm is based on the same data corpus to ensure a subsequent parity comparison of both approaches. The goal is to generate an executable model for subsequent porting to the end devices. As an alternative to machine learning, CNN acts as a feature extractor and classifier. In this context, the creation and training of the CNN are based on TensorFlow [42]. Since a CNN expects image files instead of audio files as input, it is first required to generate corresponding representative spectrograms from audio data.

Input for the CNNs is Mel spectrograms derived from the spectrogram audio representations. Therefore, speech recordings of different lengths also result in spectrograms of various sizes. However, since it is necessary to always use identically sized spectrograms for training the CNN, the audio files must be read in and processed with a fixed window. To ensure a subsequent comparison, the first three seconds of the audio files are read in, of which only two seconds are processed with an offset of half a

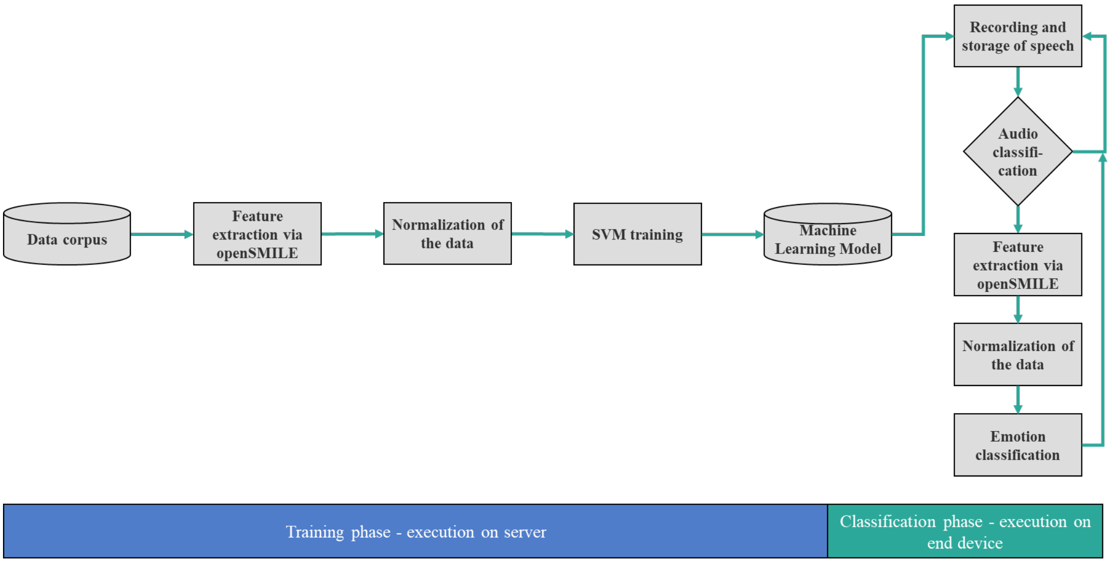

Recording and storage of speech
Audio classifi-cation
Data corpus
Feature extraction via openSMILE
Normalization of the data
SVM training
Machine Learning Model
Feature extraction via openSMILE
Normalization of the data
Emotion classification
Training phase - execution on server
Classification phase - execution on end device

Figure 4. Schematic representation of the processing steps of the machine learning method.

second. If an audio file is smaller than three seconds, the content of the file is duplicated until the minimum size is reached. To generate the spectrograms, the final two-second audio file is transformed using Fast Fourier Transform with a window size of 512 milliseconds and a jump size of 256 milliseconds between windows. From this spectrogram, the Mel spectrogram is derived with 128 Mel filters, a minimum frequency of 0 Hertz, and a maximum frequency of 8000 Hertz. Finally, this Mel spectrogram is plotted on a dB scale with 80 dB and the magma color scheme and is available for subsequent classification. The generated dB-scaled Mel spectrogram, including its intermediate stages, is visually presented in **Figure 5**. This way, the entire data corpus is preprocessed and then split again into a training and a test data set in a ratio of 80 to 20.

They must be normalized before the Mel spectrograms can serve as input data to the CNN. In this method, normalization consists of importing the image files with fixed dimensions of 224 × 224 × 3 pixels and then dividing each pixel value by a factor of 255. The dimensions of 224 × 224 × 3 pixels have been proven in image recognition by CNN since AlexNet, which is why they are also used here. The division by 255 is necessary because neural networks are known to operate from zero to one, and thus the pixel values are normalized.

Training a neural network from scratch is computationally intensive, time-consuming, and involves significant data, so transfer learning is used now. Transfer learning for neural networks consists of removing the output layer of a pre-trained neural network and replacing it with new output layers of its own, which act as classifiers. MobileNetV2, which is pre-trained on ImageNet[1], is this method's base model for transfer learning. MobileNetV2, the training base, was initially designed for object recognition and execution on mobile devices. Three separate output layers then augment the base model with a GlobalAveragePooling2D, a dropout of 0.2, and a fully connected layer including a softmax activation function, which is used when the number of classes is more significant than two. The neural network is then optimized using the Adam optimization algorithm [43] and an initial learning rate dropout of 10^{-5}.

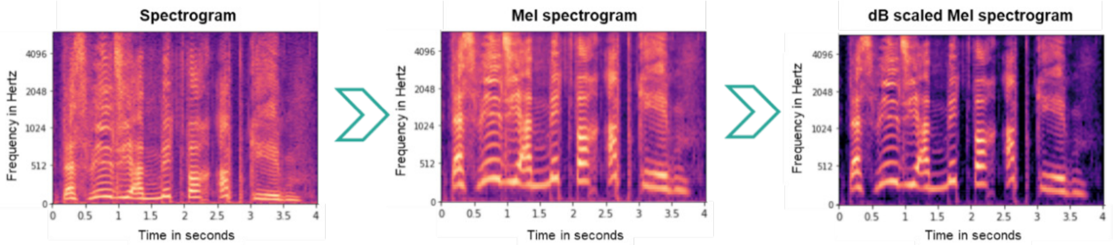


Figure 5. Mel spectrogram generation in individual steps visualized with TensorFlow and Matplotlib.

Furthermore, categorical cross-entropy is used as a loss function, which is used to quantify the differences between two probability distributions in prediction. Finally, the model is trained in two phases of equal size to apply transfer learning. First, the training of the new model operates with 50 initial epochs on the 154 untrainable layers and weights, which is used to transfer the experience of the base model to the task. Only the three newly added layers are trainable in this phase. Subsequently, the model training operates another time with 50 epochs, this time with 54 trainable and 100 untrainable layers, which is called fine-tuning in the corresponding literature. Each epoch is run with 100 training steps and ten validation steps. The training and validation data are read into the model training with a batch size of 16.

Figure 6 schematically visualizes the described sequence of the deep learning process. In addition to the individual processing steps and their arrangement, it is also apparent here which processing steps are executed on the server or the end device. The

[1] ImageNet is an image database consisting of over 15 million labeled and high resolution natural images with approximately 22000 categories.

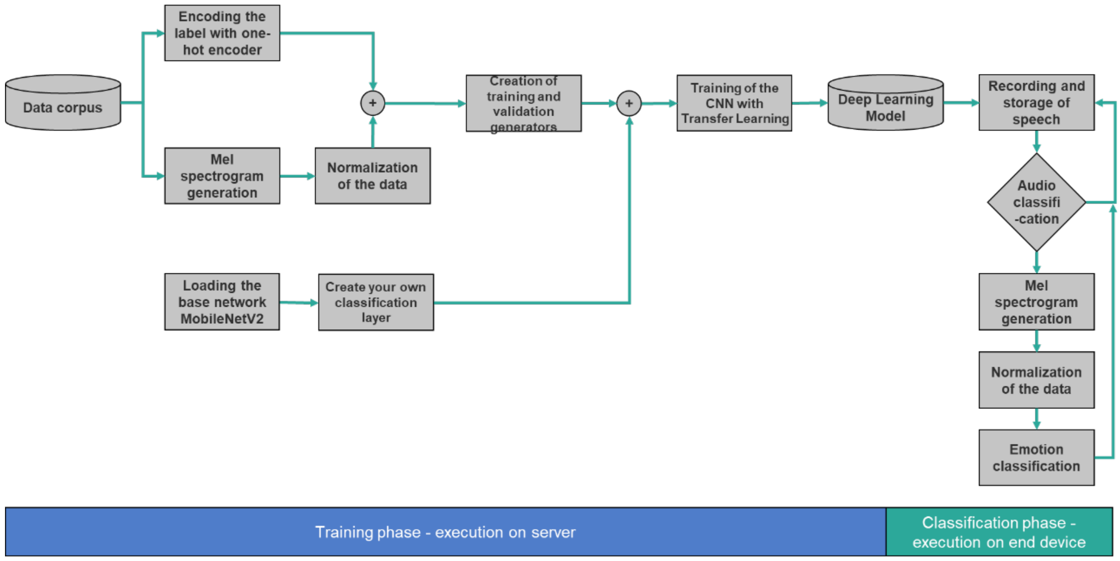

Data corpus
Encoding the label with one-hot encoder
Mel spectrogram generation
Normalization of the data
+
Creation of training and validation generators
+
Loading the base network MobileNetV2
Create your own classification layer
Training of the CNN with Transfer Learning
Deep Learning Model
Recording and storage of speech
Audio classifi -cation
Mel spectrogram generation
Normalization of the data
Emotion classification
Training phase - execution on server
Classification phase - execution on end device

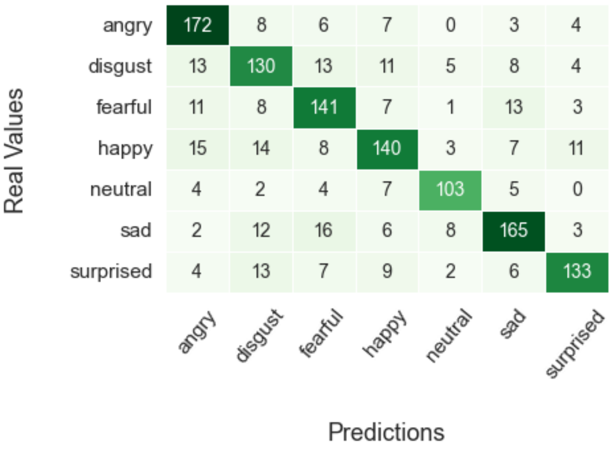
Figure 6. Processing steps of the deep learning method.

similarities and differences between this graphic and the similarities and differences in **Figure 4** become apparent. The diagrams outline the appropriate processing steps, sequence, and execution location. Furthermore, the optimization of the hyperparameters and the general parameterization of the models are part of the training and are, therefore, not listed in both diagrams. Furthermore, it can be seen from the comparison that additional work steps are necessary for the deep learning method before the neural network training is started. The effects of the extra steps on the result will be discussed later.

While describing the results of both prototypes, a distinction is made between the generation of the executable model, including its training, and the real-time classification by the same model.

The prototype is a supervised machine learning method using a support vector machine as a classifier for emotion determination. The algorithm is trained on the five databases to generate an executable and portable model. The training of this algorithm, including the optimization of the hyperparameters, is about 96 hours. The hyperparameters selected and optimized by the algorithm are the radial basis function kernel, the regulation parameter with a value of 101, the gamma with 10^{-2}, and the degree parameter with a zero value.

The accuracy after the model training is indicated by the confusion matrix, shown in **Figure 7**, for the test partition of the trained model and the classification report based on it. In the former, the list of the seven considered emotions can be lined up vertically on the left edge as absolute values on the one hand and horizontally on the bottom edge as values predicted by the model on the other. Furthermore, it can be seen from the marked green fields that the model's prediction agrees with the actual values in most cases. Those correct predictions represent the true positives. The remaining whitish areas represent the False Positives since the predicted emotion classes do not match the real-world conditions.

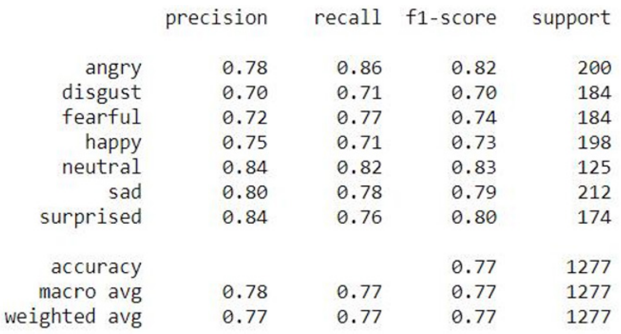


	angry	disgust	fearful	happy	neutral	sad	surprised
angry	172	8	6	7	0	3	4
disgust	13	130	13	11	5	8	4
fearful	11	8	141	7	1	13	3
happy	15	14	8	140	3	7	11
neutral	4	2	4	7	103	5	0
sad	2	12	16	6	8	165	3
surprised	4	13	7	9	2	6	133

Figure 7. Confusion matrix of the machine learning process.

The confusion matrix calculates the model's overall accuracy using the above formula, which is already included in the classification report and visualized in **Figure 8**, together with other metrics. For example, in addition to the overall accuracy rates, accuracy rates for individual emotions are also present. **Figure 8** shows that the overall accuracy of this procedure is 0.77. With an achieved value of 0.77 and 77%, respectively, the model is ranked between the CNN MobileNetV2 and LeNet based on **Table 4**.

The confusion matrix and the classification report are valid for the machine learning model and thus independent of the end device used.



```
              precision    recall  f1-score   support

       angry       0.78      0.86      0.82       200
     disgust       0.70      0.71      0.70       184
     fearful       0.72      0.77      0.74       184
       happy       0.75      0.71      0.73       198
     neutral       0.84      0.82      0.83       125
         sad       0.80      0.78      0.79       212
   surprised       0.84      0.76      0.80       174

    accuracy                           0.77      1277
   macro avg       0.78      0.77      0.77      1277
weighted avg       0.77      0.77      0.77      1277
```

Figure 8. Classification report of the machine learning method.

An exemplary metrics measurement is performed on the notebook mentioned in **Table 3**. The estimated time is measured within the model between two cycles. A cycle consists of a speech recording, an audio classification, and an emotion classification, depending on this result and its output. The average estimated time for 15 observed cycles is 0.799 seconds. For comparison purposes, processes without audio classification are also performed, where emotion classification is applied to each incoming audio signal. The average time required here is 0.114 seconds, calculated from 15 observed cycles.

In conclusion, based on the criteria set, 144 milliseconds for emotion classification alone and 799 milliseconds for emotion classification, including previous audio classification, are declared to be real-time capable. The memory requirement increases from 9.8 gigabytes to 10.1 gigabytes after starting the classification, which is derivatively an increase from 62% to 64% utilization. On the other hand, the processor utilization increases by 17% points after starting the classification, from an average of 11% to around 28%.

Furthermore, measurement is also performed on the Raspberry Pi. Here, the arithmetic means of 15 observed cycles is 4.33 seconds for audio and emotion classification. Also, at this point, the result is compared with a pure emotion classification without prior audio classification. This cycle takes an average of 0.337 seconds, calculated from 15 observed cycles. In conclusion, based on the set benchmarks, the emotion-only classification is declared real-time capable, but the combined audio and emotion classification with a time of 4330 milliseconds is not. Further memory measurement indicates an increase in memory usage after starting classification from 415 megabytes to 586 megabytes, a relative increase of 4.4% for a total availability of 3838 megabytes, from 10.8% utilization for the first time to 15.2% utilization now. On the other hand, processor utilization increases by 26.4% points during execution, from 0.7% utilization for the first time to 27.1%.

The second method described in 4.3, a CNN, is used based on the pre-trained MobileNetV2 network. The CNN developed in this method is also trained with the same intention on the five databases. The training time of the neural network with a total of 100 epochs is about six hours. As described above, the training proceeds in two identical phases of 50 epochs each, one step for initial learning and one stage for finetuning the model. Following each training epoch, the training and validation accuracy and the training and validation loss are reported. The preliminary result after the first 50 epochs is graphically visualized in **Figure 9**. On the left side is the training and validation accuracy course. The training and validation loss, each for 50 epochs, is shown on the right side. Each of these four parameters is shown as a separate curve.

The accuracy curve shows that the training accuracy starts at around 0.16 and increases to approximately 0.42 by the 50th epoch. The validation accuracy also starts at about 0.16 and reaches an accuracy of about 0.5 at the 50th epoch. It is noticeable here that the validation accuracy is always above the training accuracy. This phenomenon is due to the peculiarity of transfer learning. When training a neural

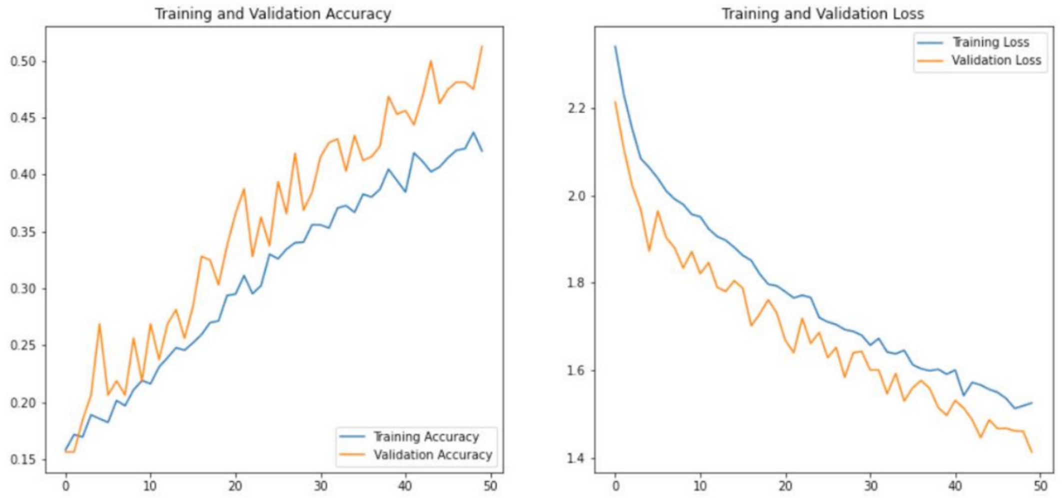


Figure 9. Training result of the CNN after initial 50 epochs with transfer learning.

network without transfer learning, the training accuracy is always above the validation accuracy.

Similar behavior can be observed in the course of the loss curve. The training loss curve starts at a loss of around 2.3 and drops to about 1.55 by completing the 50th epoch. On the other hand, the validation loss curve begins at 2.2 and drops to about 1.4 by the 50th epoch. Once again, it is characteristic of transfer learning that the validation curve always lies below the training curve.

During the subsequent fine-tuning of the CNN, more trainable layers and, thus, more trainable weights are available. The model also has more possibilities to optimize performance. The result of the fine-tuning is illustrated in **Figure 10**.

Fifty other fine-tuning epochs supplement the result in the previous **Figure 9**. A vertical straight line in the 50th epoch shows at which point the fine-tuning phase starts. Thus, after the beginning of the fine-tuning stage, the training accuracy curve drops to 0.35 but then takes a steeper course than before and reaches the maximum of 1.0 from the 90th epoch, at which point the curve stagnates until the 100th epoch. On the other hand, the validation accuracy curve initially drops to around 0.45 after the start of the finetuning phase. Still, it rises again and reaches an accuracy rate of about 0.7 by the 100th epoch.

A change can also be observed in the loss curves after the start of fine-tuning. For example, the train-

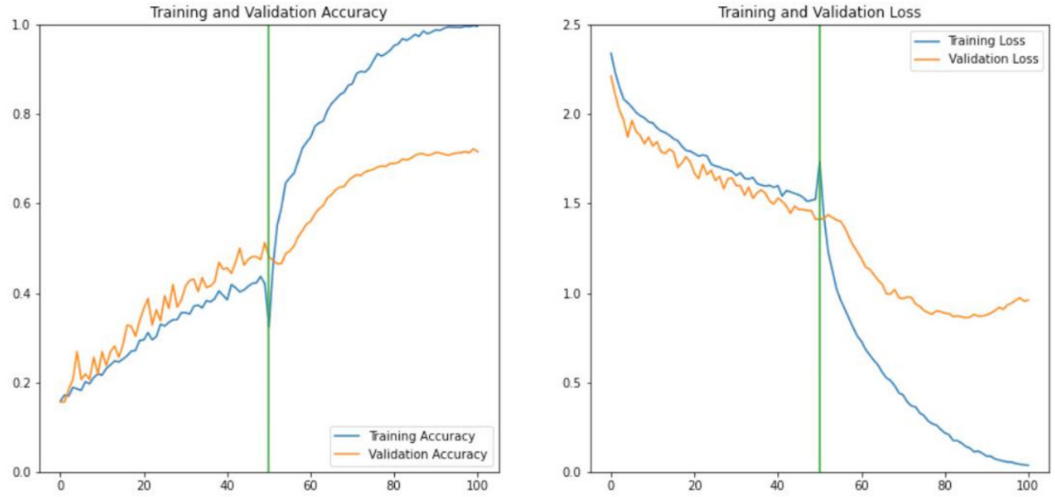


Figure 10. Training result of the CNN after 100 epochs using transfer learning.

ing loss curve rises to 1.75 after the start of fine-tuning but then drops more sharply than before, reaching a value of 0.1 at the 100th epoch. The validation loss curve does not rise after the start of fine-tuning but drops to a value of 0.8 by the 80th epoch, where the local minimum of the curve is located. By completing the 100th epoch, the curve rises to about 1.0. Ultimately, it is not the training accuracy but the validation accuracy that is decisive for the correct classification. With an accuracy of 0.7 and 70 %, respectively, this result is based on **Table 4** ranks below MobileNetV2.

Analogously, the CNN of this method is trained a second time on the five databases but without applying transfer learning. In this training, the CNN is also qualified with 100 epochs, but in only one phase and with the absolute number of trainable layers. With its 100 epochs, this training has a running time of about six hours, as before. The result of this training of 100 epochs without transfer learning is visualized in **Figure 11**. Here, it can be seen that the training accuracy curve starts at a value of 0.22 and rises to the maximum of 1.0 by the 50th epoch. There the curve remains until the 100th epoch. The validation accuracy curve also starts at a value of 0.22 and increases to 0.7 by the 50th epoch, which remains with fluctuations until the 100th epoch. The training loss curve begins at the value of 2.0 and steadily decreases until it reaches the minimum of 0.0 at about the 70th epoch and remains there.

On the other hand, the validation loss curve starts with a value of about 2.2 and falls with a fluctuating downward trend until the 35th epoch. There, the curve has reached its local minimum of 0.9. However, the curve rises again until the 100th epoch to about 1.2.

The advantages of transfer learning become apparent when comparing **Figure 10** with **Figure 11**, and the benefits of transfer learning become evident. Starting from the start of fine-tuning, it can be stated that the validation accuracy curve has already reached the value of 0.7 after 20 epochs. In contrast, the validation accuracy curve without transfer learning has only reached this value after about 50 epochs. The advantage can also be seen in that the validation accuracy curve for the method with transfer learning has a higher slope than the validation accuracy curve without transfer learning. Finally, it can be seen that the start of the validation accuracy curve for the process with transfer learning starts higher on the Y axis with a value of 0.45 than the curve without transfer learning with a value of 0.22.

In contrast to an SVM, the classification result in a neural network does not output a single value but a value range with seven entries corresponding to the number of classes present. The entries in this value range represent the probabilities with which the model predicts one class each. The individual entries

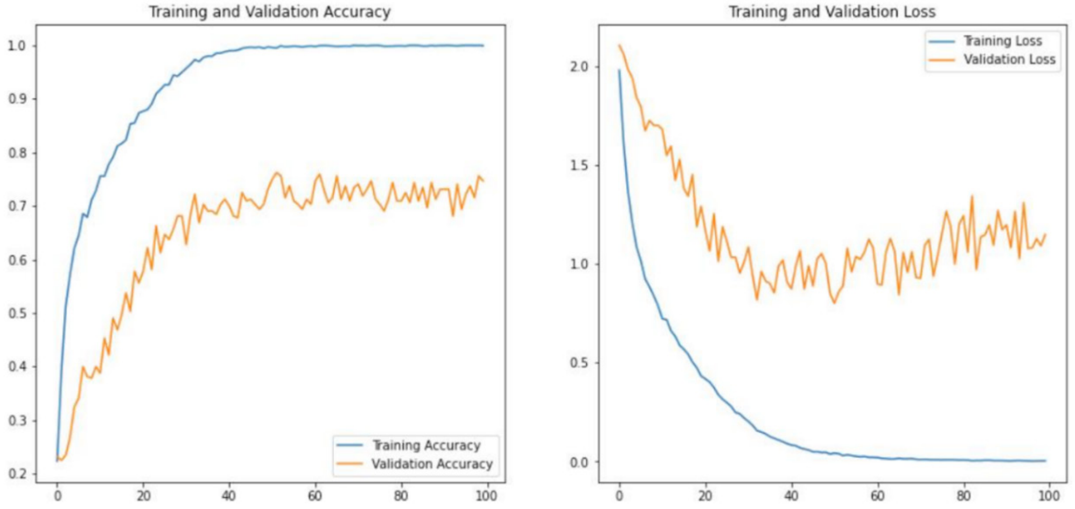


Figure 11. Training result of the CNN after 100 epochs without using transfer learning.

can assume a value between 0.0 and 1.0, with the sum of all entries in the value range again resulting in 1.0. The emotion class with the highest probability value is the classified emotion.

An exemplary metrics measurement is also performed on the notebook mentioned in **Table 3**. Here, the time is also measured between two cycles, whereby one consists of the audio and emotion classification, including the output. The arithmetic mean of the estimated time is 0.856 seconds for 15 observed processes. At this point, a comparison is also made to a cycle without prior audio classification. The time counted for this cycle is 0.119 seconds, also calculated from 15 observed cycles. With a time value of 119 milliseconds for a cycle without audio classification and 856 milliseconds for a cycle with audio classification, respectively, the result is below the set benchmarks and is therefore considered real-time capable. The memory requirement increases from 9.9 gigabytes to 10.6 gigabytes from the start of classification. Relative to the total available memory, this is an increase of 4% points, from 63% utilization for the first time to 67%. The processor utilization also shows an increase of 16% points, from 15% to 31% for the first time.

Based on the implementation of the prototype on the Raspberry Pi, the average time required for a cycle, including audio and emotion classification, is around 4.43 seconds, calculated from 15 observed cycles. In comparison, emotion recognition without prior audio classification requires an arithmetic mean of only 0.393 seconds, again calculated from 15 practical cycles. With a needed time of 393 milliseconds, the latter result is below the set benchmarks, but the previous mark with 4427 milliseconds is not. Thus, implementing the prototype on the Raspberry Pi lacks real-time capability. The memory used increases from 563 megabytes to 675 megabytes during runtime, a rise of 2.9% points for a total availability of 3838 megabytes, from 14.7% utilization for the first time to 17.6% now. On the other hand, processor utilization increases by 22% points during execution, from 2.9% utilization for the first time to 24.9%.

In this study, four core elements are to be noted as findings. First, a tabular comparison of the results of the two methods used is provided in **Table 5**, where the metrics listed here represent the arithmetic mean across all measured metrics.

It can be deduced from the previous table that an SER system can distinguish between speech, non-speech, and silence. To this end, the YAMNet neural network, which is not a primary component of this work and was not developed within this research, is used within the prototypes. Nevertheless, the YAMNet neural network is part of both prototypes, which are thus able to classify audio inputs into different categories, such as music, meowing, barking, silence, or even speech.

Concerning the databases used in this research, it was shown that they contain various emotional audio files, including the six basic emotions mentioned by Ekman (1971), plus further emotional stimuli such as tiredness or boredom. Neutral emotion can also be found in the majority of the databases. The prototypes trained on these databases are thus able to distinguish between the seven emotions. Therefore, an SER system can distinguish between positive, nega-

Table 5. Comparison of the results of the two methods.

Metrics	Machine learning method	Deep learning method
Training duration	96 hours	6 hours
Accuracy	77%	70%
Working memory requirement increase	10.7% points	3.45% points
Processor load increase	21.7% points	19% points
Time consumption emotion classification	225.5 milliseconds	256 milliseconds
Time consumption audio and emotion classification	2565 milliseconds	2642 milliseconds

tive, and neutral emotions but is not limited to this. Instead, such a system can perform a more detailed categorization of speech input into individual emotions with an accuracy of 77% for machine learning and 70% for deep learning.

The third finding relates to the feasibility of an SER system on ambient terminals but is distinguished between the phases and the ambient end devices used in **Table 3**. Due to the intensive computing power and high runtime, the one-time model training step must be executed on a server. Therefore, therefore not feasible on an end device. The subsequent real-time classification phase is based on the trained model and can be performed multiple times on a terminal device. The prototype porting to a notebook is feasible since notebooks generally support corresponding Python runtime environments. Thus, running the emotion classification is possible on a notebook regardless of the method used. Porting the prototypes to a Raspberry Pi, on the other hand, is more complex since, on the one hand, Python runtime environments are supported in principle. Still, on the other hand, the necessary frameworks, openSMILE and TensorFlow, are not available for Raspberry Pi's. Alternatively, for TensorFlow, the porting of the Deep Learning procedure is done with TensorFlow Lite, which runs the trained model on the end device. In the absence of openSMILE compatibility with 32-bit operating systems and a lack of a qualitative alternative, the porting of the machine learning procedure is omitted at this point. In summary, it can be stated that realizing an SER system using edge computing is only possible to a limited extent. While they assist in executing deep learning approaches and neural networks on the end devices, this does not always apply to the machine learning method.

Regarding the real-time capability of the classification system, it is necessary to differentiate which method is used and whether only SER or SER plus prior audio classification is considered. Concerning the machine learning method, the SER system requires an average of 114 milliseconds for pure SER without prior audio classification and is thus below the set index of 1000 milliseconds. Additionally, the SER, including initial audio classification with an average time of 799 milliseconds, is below the set of 1000 milliseconds. Related to Deep Learning, the average time for a cycle without an audio classification is 256 milliseconds. Meanwhile, the average time for a cycle with audio and emotion classification is 2642 milliseconds. According to the results, the fourth finding is that the choice of the method determines whether the real-time capability is given or not. However, since there is no porting and, therefore, no results regarding the machine learning method, there is the possibility that this last finding is falsified.

5. Discussion

Both machine learning and deep learning approaches in this study rely on a shared data corpus, which is obtained and selected based on predefined criteria outlined in the existing literature. These criteria encompass several factors, including a minimum audio duration of one second and a maximum duration of 20 seconds. The choice of one second as the minimum duration is justified because shorter audio files generally lack sufficient information. Conversely, selecting a maximum period of 20 seconds is somewhat arbitrary, as durations of 10, 15, or 30 seconds could have also been considered. However, as depicted in **Figure 2**, most audio files in the chosen databases have durations of less than 10 seconds.

Another criterion is the exclusion of non-spoken sentences since the prototypes focus on speech-emotion recognition (SER) rather than general audio classification. Therefore, the audio files must exclusively contain speech, even though some music may include spoken segments accompanied by instruments.

Furthermore, it is essential to note that this study does not address emotion recognition in music, although it could be a potential avenue for future research. Including audio files with background noise is essential, as real-life communication often occurs in noisy environments. While background noises are prominent in music, they play a secondary role in speech-related scenarios. Therefore, incorporating databases containing audio data with such background noise would be valuable for enhancing the research.

The criteria specify that the files must be in an auditory or audiovisual format, but there are no restrictions regarding file type, sampling rate, or dubbing. Although limiting the selection to purely auditive file formats may influence the choice of databases, it would not impact the subsequent procedures, as all audio files are transformed to a standardized format and file type before model training.

The native language used in the audio files is also not a selection criterion, as indicated in **Table 1**, which demonstrates the inclusion of German, English, and Italian data in the selected databases and audio files in other languages such as Turkish, Danish, or Chinese. Since the six basic emotions described by Darwin (1873) and Ekman (1971) are expressed similarly across cultures, the spoken language does not significantly affect the Mel spectrograms, model training, or results. However, it is crucial to include both male and female voices when selecting databases. Failing to meet this criterion could impede data generalization and lead to overfitting or underfitting of the model.

Open accessibility and availability of labeled data are mandatory for data collection. Without open access to the databases, it would be impossible for third parties to reproduce the procedures and results of this study. Moreover, the absence of labeled data would render supervised machine-learning algorithms infeasible. Investigating the impact of different database quantities or compositions, including language variations, on the outcomes of this research can be pursued in future investigations.

Other methodologies that equally impact both procedures involve dividing the data corpus into training and validation sets using an 80 to 20 ratio. Preprocessing the audio files commonly entails transforming them to a 16,000-hertz format with a mono channel, as frequently reported in the literature.

In the machine learning method, the hyperparameters and their value ranges are defined at the start of the training process. The algorithm then independently determines the optimal values based on these predefined hyperparameters. The selection of these hyperparameters is based on the research conducted by Mao et al. [27]. However, alternative parameters or value ranges described by Wang et al. [44] or Cummins et al. [45] could also be considered. Feature extraction utilizes the openSMILE framework, particularly the eGeMAPS, which aligns with its usage in Cummins et al.'s work.

In contrast, the deep learning method employs explicit hyperparameters. The training process consists of two phases, each comprising 50 epochs, as established by Tan et al.. Alternatively, Zhang et al. utilized a batch size of 30, SGD as the optimization algorithm, and a learning rate of 10-3 as hyperparameters. However, standard hyperparameters employed by Lim et al. include SGD with a learning rate of 10-2, a dropout of 0.25, and a Rectified Linear Unit activation function. Discrepancies also exist in the generation of Mel spectrograms, as mentioned in section 2.3.3 and the relevant literature. For instance, Zhang et al. used 64 Mel filters for audio classification within a frequency range of 20 to 8000 hertz, utilizing a 25-millisecond Hamming window with a ten-millisecond overlap for each window. The variation in Mel spectrogram generation can be justified since speech emotion recognition (SER) and speech recognition are distinct processes, as noted by Zhang et al. Nonetheless, the use of Mel spectrograms aligns with the current state of research. Alternatively, MFCC can also be applied within the deep learning procedure.

The base model utilized in this study is MobileNetV2 when employing transfer learning. However, the literature suggests considering CNN ResNet50 or SqueezeNet [46]. In this study, only the last 54 layers out of 154 are fine-tuned. Optionally, different numbers of trainable layers, such as 32 or 16, can be considered. Exploring the impact of modifications to these hyperparameters on the results can be a subject of future research.

The other cannot be drawn solely from comparing the approaches and their results. **Table 5** does not provide conclusive evidence to support the dominance of either procedure. Examining the training time reveals that the Machine Learning approach requires approximately 16 times the duration compared to the Deep Learning approach. However, the Machine Learning model exhibits higher accuracy, faster classification, and lower increases in processor load and memory requirements.

The speed advantage of the machine learning method in real-time classification arises from the utilization of distinct emotion recognition algorithms. In this case, audio classification is not considered, as it is identical in both approaches and precedes emotion recognition. In the Machine Learning model, speech input undergoes openSMILE processing, normalization, and subsequent classification using SVM. Conversely, in the Deep Learning model, the speech input is initially transformed into a spectrogram, stored, normalized, and processed through all neural network layers. The storage and retrieval of spectrograms involve additional read-and-write transactions that are not required in the machine learning method, thereby impacting the speed of the Deep Learning model. However, the difference in speed is marginal and invisible to humans, as such disparities occur in milliseconds.

Furthermore, **Table 5** highlights that the pure emotion classification alone operates, on average, ten times faster than the combined audio and emotion classification, regardless of the chosen method. This discrepancy significantly influences the declaration of real-time capability, particularly concerning porting to the Raspberry Pi. While pure emotion classification can be deemed real-time capable, the same cannot be said for the combined type due to extended runtime. The disparity may likely stem from the CNN YAMNet employed for audio classification and its external development, which falls outside the scope of this study. Consequently, a comprehensive analysis of the time difference and its origin cannot be provided. Therefore, optimizing speech classification, which is not extensively examined in this paper, could considerably enhance the overall process latency.

When examining the individual process steps in **Figures 4 and 6**, no direct conclusion regarding training duration is apparent. However, the Deep Learning method necessitates more process steps than the Machine Learning method. As mentioned earlier, the models' parameterization is not depicted in these figures, as it is part of the mapped training. Specifically, the choice between fixed hyperparameters and ranges of hyperparameter values significantly impacts training time. In the Deep Learning method, training duration also varies based on parameters such as batch size, number of epochs, and number of steps per epoch. The Machine Learning method determines training duration by the number of hyperparameters and their value ranges. The resulting hyperparameters are determined through the algorithm's processing and optimization of potential combinations. In contrast to explicit parameterization, processing all conceivable combinations is likely responsible for the disparity in training duration. Consequently, this implies that the overall accuracy of the machine learning model surpasses that of the deep learning model due to the processing of all combinations.

Moreover, the results in **Figure 10** reveal that the validation loss curve increases again after epoch 80. A similar phenomenon is observed in **Figure 11** from epoch 50 onwards. However, the validation accuracies in these figures do not exhibit the same increase. The rising course of these curves may indicate overfitting, which warrants further investigation in future research.

The higher memory requirement in the Deep Learning model is attributed to the necessity of storing the generated spectrogram, in addition to the primary audio file, for emotion recognition. Another contributing factor is the higher number of parameters in the CNN than in an SVM, which are also stored in memory.

Contrary to our expectations, the Machine Learning method exhibits higher processor utilization than the Deep Learning method. This phenomenon is likely due to the improper timing of the recording. However, considering the marginal variances, the

21.7% difference in processor utilization between the two ways is negligible.

It should be noted that both models consider the presence of a microphone as an essential prerequisite. Unlike multimodal emotion recognition, a unimodal SER system does not require additional provisions such as cameras. Most ambient terminals are equipped with native microphones but lack native cameras, as seen in smart speakers or smart TVs. Therefore, the prototypes developed in this study are suitable for porting to such devices.

Regarding the theoretical foundations of this research, SER plays an increasingly crucial role in human-computer interaction (HCI). HCI occurs within the context of remote participation, which is a component of the growing computer-supported hybrid communication in everyday life. Consequently, SER holds greater significance in everyday life and is the subject of ongoing research. Similar to this study, there are investigations into real-time SER [33] and edge computing. However, no research on SER applications on edge devices exists, as Shi et al. (2016) defined. Thus, the combination of SER, edge computing, and real-time processing explored in this study represents a novel research extension. To maintain the focus of this work, restrictions are deliberately made. However, other external limitations also limit this work, which will be explained in more detail below. According to Ekman (1971), only the six basic emotions, including a seventh neutral emotion, are considered, which is why emotions such as tiredness or boredom are excluded in this work. Accordingly, the data acquisition is made with the mentioned seven emotions, further limiting the selection of suitable databases.

Furthermore, the dimensions of arousal and valence are also omitted. These dimensions can be considered in continuing work but do not play a role in the mere emotion recognition in this research. Therefore, it is pinpointed that these dimensions exist, but it does not address them in the further course of the study.

A technical limitation, however, is the mapping of processor and memory utilization. Since the system constantly updates these two indicators, it is impossible to identify the exact utilization. Thus, the processor and RAM utilization documentation only represents a snapshot, not a calculated average value. Furthermore, the maximum number of simultaneously recognizable emotions is another technical limitation. This paper assumes that only one emotion is contained in a sentence or voice recording. As the sentences and audio file length increases, the probability that multiple emotions are controlled increases. However, the machine learning method using the SVM can only classify one emotion, which is why the length of the voice recording is limited to three seconds.

On the other hand, the CNN in the Deep Learning method calculates a probability for each of the seven emotions. For this reason, this method can potentially identify multiple emotions within one speech recording. Another technical limitation is the applicability of the prototypes to only one person. The model training is based on emotional content in the audio files of the acquired databases. Individuals can be heard in each audio file so that the prototypes can apply emotion recognition only to individuals. When multiple individuals speak simultaneously, the prototypes cannot distinguish between individuals and their emotions. The extension to multi-person recognition goes beyond the definition of these prototypes and therefore needs to be investigated in further work.

Furthermore, the porting of prototypes is also limited. For example, only two device categories were selected since porting to more devices would exceed the scope of this paper. For this reason, porting to smartphones or tablets, for example, is not included.

6. Conclusions

The outcomes and interpretations presented in this study provide compelling evidence that the developed prototypes are functional and well-suited for practical applications. This Speech Emotion Recognition (SER) systems have the potential for various use cases and can offer extensions to existing products and services. Here are some examples of application areas and their resulting benefits.

(i) Universal Application: SER applications can be utilized wherever speech plays a central role, such as call centers, radio broadcasts, podcasts, and television shows. New business models can be developed that merge physical and virtual presence. For instance, implementing an SER system in a smart speaker can detect vocal activity and emotions in a home environment. These emotions can be visually presented to the user and, with their consent, transmitted to the producer for product optimization, offering the user a premium in return. Similarly, SER can automate the editing of highlights in a broadcast sports game based on detected emotions. Such scenarios can be extended to internet-based broadcasts like Twitch or Netflix.

(ii) Real-time Audience Mood Capture: SER applications can capture the current mood of an audience in real time. Unlike the previous use case, where emotions are summarized over time, this approach focuses on determining emotion levels at precise moments. This can be valuable in political talks or product presentations, where immediate feedback on expressed emotions is crucial. By providing unbiased input to speakers, SER enables them to gauge audience response accurately. These techniques apply to physical, virtual, or hybrid forms of communication, further emphasizing the increasing trend of remote participation.

(iii) Individual-focused Applications: SER applications can cater to individual users, tailoring experiences based on their detected emotions. For example, a smart speaker or automobile with an SER system can adjust music or lighting according to the user's emotional state. In gaming, the algorithm can offer in-game relief when anger is detected. Individualized advertising in social media or e-commerce platforms, varying prices dynamically based on emotional states (e.g., increasing costs for joyful emotions), is also a potential application.

This research investigated the ability of an SER system to distinguish speech, non-speech, and silence, as well as classify different emotions. The study involved a systematic literature review, developing two prototypes using machine learning and deep learning approaches, and training the models using a data corpus comprising five audio databases. Before being used for model training, the audio files underwent preprocessing, including conversion to a sampling rate of 16000 Hz and a mono channel.

In the machine learning approach, the openSMILE framework was employed for feature extraction, generating eGeMAPS features that were normalized and used for classification. Support Vector Machine (SVM) served as the classifier. The model training took approximately 96 hours on a server, and while successful porting to a notebook was achieved, porting to a Raspberry Pi was unsuccessful. The prototype demonstrated the capability to identify different sounds in under 1000 milliseconds and classify seven emotions in the case of speech.

In the Deep Learning Model approach, audio files were transformed into Mel spectrograms, normalized, and used as input for a CNN implemented using TensorFlow. The CNN performed feature extraction and classification. The model training, including transfer learning, took around six hours on the server. The completed model was successfully ported to a notebook and a Raspberry Pi. The notebook achieved classification below 1000 milliseconds, while the Raspberry Pi required approximately 4427 milliseconds. The models' computation time and classification accuracy were evaluated using the provided formula.

SER systems are embedded in Human-Computer Interaction (HCI) systems and can potentially be applied in everyday scenarios. The results of this study demonstrate that the technical feasibility of practical implementation is achievable, and several use cases described in this research can find real-world applications. Moreover, these findings highlight the growing relevance of SER in everyday communication, where remote participation is increasingly combined with a physical presence. Such SER systems have the potential to enhance human-machine interaction, making communication more human-like and intuitive. Based on this research, the acceptance and utilization of SER-enabled remote participation applications can be considered an extension of the fourth criterion of emotion recognition.

The results and discussions presented in this study can be further enriched and expanded through future research. Additional investigations could explore other emotions or broaden the scope of the utilized databases. Furthermore, examining arousal and valence dimensions would be valuable. Investigating the machine's subsequent actions linked to recognized emotions within the SER system is another avenue worth exploring. For example, studying the most suitable lighting settings, color combinations, or music choices to support or counteract specific emotions based on the determined emotions could be interesting. This could involve studying music's genre, volume, and beat rate and its relation to emotion recognition within songs. Combining both approaches, selecting songs based on identified emotions and playing them in response to human emotions, could provide an intriguing direction for further exploration.

Further research could also focus on the Deep Learning method, exploring different hyperparameters for model training and investigating modified transfer learning techniques. Multitask or semi-supervised learning could offer new perspectives in advancing SER research. Additionally, the limitations identified in this study open up opportunities for independent research and raise further questions. For instance, investigating whether an SER system can differentiate between multiple individuals based on speech or identify numerous emotions within a sentence could be explored. Exploring subjective user perception and experience could also be valuable. Lastly, the prototypes developed in this study were ported to two device categories, prompting whether they can be extended to other device categories, such as smartphones or tablets, each with its diverse range of devices and operating systems.

Regarding the real-time capability of the prototype, it would be worthwhile to explore the execution of digital signal processors and their potential for optimizing runtimes. Utilizing digital signal processors optimized for real-time functions like Fast Fourier Transform in mobile devices like the Raspberry Pi could further enhance the prototypes' real-time capability and overall performance.

In conclusion, this study demonstrates the theoretical and practical feasibility of real-time speech-based emotion recognition through edge computing. The implications of this research extend to practical applications and provide a foundation for future investigations.

Author Contributions

D.E.d.A. conceived the idea of researching a real-time processing method that captures and evaluates emotions in speech. R.B. and D.E.d.A. conceived the study. R.B. served as D.E.d.A.'s graduate advisor on his graduate thesis at the FOM University of Applied Sciences. All authors reviewed and approved the final manuscript.

Conflict of Interest

There is no conflict of interest.

Funding

This research received no external funding.

References

[1] El Ayadi, M., Kamel, M.S., Karray, F., 2011. Survey on speech emotion recognition: Features, classification schemes, and databases. Pattern Recognition. 44(3), 572-587. DOI: https://doi.org/10.1016/j.patcog.2010.09.020

[2] Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., et al., 2001. Emotion recognition in human-computer interaction. IEEE Signal Processing Magazine. 18(1), 32-80.

DOI: https://doi.org/10.1109/79.911197

[3] Schuller, B.W., 2018. Speech emotion recognition: Two decades in a nutshell, benchmarks, and ongoing trends. Communications of the ACM. 61(5), 90-99.
DOI: https://doi.org/10.1145/3129340

[4] Kraus, M.W., 2017. Voice-only communication enhances empathic accuracy. American Psychologist. 72(7), 644.
DOI: https://doi.org/10.1037/amp0000147

[5] Akçay, M.B., Oğuz, K., 2020. Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers. Speech Communication. 116, 56-76.
DOI: https://doi.org/10.1016/j.specom.2019.12.001

[6] Dincer, I., 2000. Renewable energy and sustainable development: A crucial review. Renewable and Sustainable Energy Reviews. 4(2), 157-175.
DOI: https://doi.org/10.1016/S1364-0321(99)00011-8

[7] Chao, K.M., Hardison, R.C., Miller, W., 1994. Recent developments in linear-space alignment methods: A survey. Journal of Computational Biology. 1(4), 271-291.
DOI: https://doi.org/10.1089/cmb.1994.1.271

[8] Abbas, N., Zhang, Y., Taherkordi, A., et al., 2017. Mobile edge computing: A survey. IEEE Internet of Things Journal. 5(1), 450-465.
DOI: https://doi.org/10.1109/JIOT.2017.2750180

[9] Cao, K., Liu, Y., Meng, G., et al., 2020. An overview on edge computing research. IEEE Access. 8, 85714-85728.
DOI: https://doi.org/10.1109/ACCESS.2020.2991734

[10] Shi, W., Cao, J., Zhang, Q., et al., 2016. Edge computing: Vision and challenges. IEEE Internet of Things Journal. 3(5), 637-646.
DOI: https://doi.org/10.1109/JIOT.2016.2579198

[11] Lin, Y.L., Wei, G. (editors), 2005. Speech emotion recognition based on HMM and SVM. 2005 International Conference on Machine Learning and Cybernetics; 2005 Aug 18-21; Guangzhou, China. New York: IEEE.
DOI: https://doi.org/10.1109/icmlc.2005.1527805

[12] Nassif, A.B., Shahin, I., Attili, I., et al., 2019. Speech recognition using deep neural networks: A systematic review. IEEE Access. 7, 19143-19165.
DOI: https://doi.org/10.1109/ACCESS.2019.2896880

[13] Schuller, B., Batliner, A., Steidl, S., et al., 2011. Recognising realistic emotions and affect in speech: State of the art and lessons learnt from the first challenge. Speech Communication. 53(9-10), 1062-1087.
DOI: https://doi.org/10.1016/j.specom.2011.01.011

[14] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Computation. 9(8), 1735-1780.
DOI: https://doi.org/10.1162/neco.1997.9.8.1735

[15] Khalil, R.A., Jones, E., Babar, M.I., et al., 2019. Speech emotion recognition using deep learning techniques: A review. IEEE Access. 7, 117327-117345.
DOI: https://doi.org/10.1109/ACCESS.2019.2936124

[16] Hinton, G., Deng, L., Yu, D., et al., 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine. 29(6), 82-97.
DOI: https://doi.org/10.1109/MSP.2012.2205597

[17] Torrey, L., Shavlik, J., Walker, T., et al., 2010. Transfer learning via advice taking. Advances in machine learning. Springer: Berlin.
DOI: https://doi.org/10.1007/978-3-642-05177-7_7

[18] Eyben, F., Scherer, K.R., Schuller, B.W., et al., 2015. The Geneva minimalistic acoustic parameter set (GeMAPS) for voice research and affective computing. IEEE Transactions on Affective Computing. 7(2), 190-202.
DOI: https://doi.org/10.1109/TAFFC.2015.2457417

[19] Ekman, P., 1971. Universals and cultural differences in facial expressions of emotion. Nebraska Symposium on Motivation. University of Nebraska Press: Nebraska.

[20] Siedlecka, E., Denson, T.F., 2019. Experimental methods for inducing basic emotions: A qualitative review. Emotion Review. 11(1), 87-97.

DOI: https://doi.org/10.1177/1754073917749016

[21] Livingstone, S.R., Russo, F.A., 2018. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. PloS One. 13(5), e0196391. DOI: https://doi.org/10.1371/journal.pone.0196391

[22] Burkhardt, F., Paeschke, A., Rolfes, M., et al. (editors), 2005. A database of German emotional speech. 9th European Conference on Speech Communication and Technology; 2005 Sep 4-8; Lisbon, Portugal. DOI: https://doi.org/10.21437/interspeech.2005-446

[23] Choudhury, A.R., Ghosh, A., Pandey, R., et al. (editors), 2018. Emotion recognition from speech signals using excitation source and spectral features. 2018 IEEE Applied Signal Processing Conference (ASPCON); 2018 Dec 7-9; Kolkata, India. New York: IEEE. DOI: https://doi.org/10.1109/ASPCON.2018.8748626

[24] Costantini, G., Iadarola, I., Paoloni, A., et al. (editors), 2014. EMOVO corpus: An Italian emotional speech database. Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14); 2014 May; Reykjavik, Iceland.

[25] Martin, O., Kotsia, I., Macq, B., et al. (editors), 2006. The eNTERFACE'05 Audio-Visual emotion database. 22nd International Conference on Data Engineering Workshops (ICDEW'06); 2006 Apr 3-7; Atlanta, GA, USA. New York: IEEE. DOI: https://doi.org/10.1109/ICDEW.2006.145

[26] Lim, W., Jang, D., Lee, T. (editors), 2016. Speech emotion recognition using convolutional and Recurrent Neural Networks. 2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA); 2016 Dec 13-16; Jeju, Korea (South). New York: IEEE. DOI: https://doi.org/10.1109/APSIPA.2016.7820699

[27] Mao, Q., Dong, M., Huang, Z., et al., 2014. Learning salient features for speech emotion recognition using convolutional neural networks. IEEE Transactions on Multimedia. 16(8), 2203-2213. DOI: https://doi.org/10.1109/TMM.2014.2360798

[28] Tzirakis, P., Zhang, J., Schuller, B.W. (editors), 2018. End-to-end speech emotion recognition using deep neural networks. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2018 Apr 15-20; Calgary, AB, Canada. New York: IEEE. DOI: https://doi.org/10.1109/ICASSP.2018.8462677

[29] Shinde, P.P., Shah, S. (editors), 2018. A review of machine learning and deep learning applications. 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA); 2018 Aug 16-18; Pune, India. New York: IEEE. DOI: https://doi.org/10.1109/ICCUBEA.2018.8697857

[30] Adetiba, E., Adeyemi-Kayode, T.M., Akinrinmade, A.A., et al., 2021. Evolution of artificial intelligence programming languages-a systematic literature review. Journal of Computer Science. 17(11), 1157-1171. DOI: https://doi.org/10.3844/JCSSP.2021.1157.1171

[31] Hershey, S., Chaudhuri, S., Ellis, D.P.W., et al. (editors), 2017. CNN architectures for large-scale audio classification. 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2017 Mar 5-9; New Orleans, LA, USA. New York: IEEE. DOI: https://doi.org/10.1109/ICASSP.2017.7952132

[32] Gemmeke, J.F., Ellis, D.P.W., Freedman, D. et al. (editors), 2017. Audio set: An ontology and human-labeled dataset for audio events. 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2017 Mar 5-9; New Orleans, LA, USA. New York: IEEE. DOI: https://doi.org/10.1109/ICASSP.2017.7952261

[33] Vogt, T., André, E., Wagner, J., 2008. Automatic recognition of emotions from speech: A review of the literature and recommendations for practical realisation. Affect and emotion in human-computer interaction. Springer: Berlin. pp. 75-91.

DOI: https://doi.org/10.1007/978-3-540-85099-1_7

[34] Zhang, S., Zhang, S., Huang, T., et al., 2017. Speech emotion recognition using deep convolutional neural network and discriminant temporal pyramid matching. IEEE Transactions on Multimedia. 20(6), 1576-1590.
DOI: https://doi.org/10.1109/TMM.2017.2766843

[35] Liu, S., Nan, K., Lin, Y., et al. (editors), 2018. On-demand deep model compression for mobile devices: A usage-driven model selection framework. Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services; 2018 Jun 10-15; Munich Germany.
DOI: https://doi.org/10.1145/3210240.3210337

[36] Davis, J., Goadrich, M. (editors), 2006. The relationship between precision-recall and ROC curves. Proceedings of the 23rd International Conference on Machine Learning; 2006 Jun 25-29; Pittsburgh Pennsylvania USA.
DOI: https://doi.org/10.1145/1143844.1143874

[37] LeCun, Y., Bottou, L., Bengio, Y., et al., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 86(11), 2278-2324.
DOI: https://doi.org/10.1109/5.726791

[38] Krizhevsky, A., Sutskever, I., Hinton, G.E. (editors), 2012. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems; 2012 Dec 3-6; Lake Tahoe, Nevada, United States.

[39] Simonyan, K., Zisserman, A. (editors), 2015. Very deep convolutional networks for large-scale image recognition. The 3rd International Conference on Learning Representations (ICLR2015); 2015 May 7-9; San Diego, CA, USA.

[40] He, K., Zhang, X., Ren, S. et al. (editors), 2016. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016 Jun 27-30; Las Vegas, NV, USA. New York: IEEE.
DOI: https://doi.org/10.1109/CVPR.2016.90

[41] Sandler, M., Howard, A., Zhu, M., et al. (editors), 2018. MobileNetV2: Inverted residuals and linear bottlenecks. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018 Jun 18-23; Salt Lake City, UT, USA. New York: IEEE.
DOI: https://doi.org/10.1109/CVPR.2018.00474

[42] Abadi, M., Barham, P., Chen, J. et al. (editors), 2016. TensorFlow: A system for large-scale machine learning. Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI'16); 2016 Nov 2-4; Savannah, GA, USA.

[43] Kingma, D.P., Ba, J.L. (editors), 2015. Adam: A method for stochastic optimization. 3rd International Conference for Learning Representations; 2015 May 7-9; San Diego, CA, USA.

[44] Wang, X., Han, Y., Leung, V.C., et al., 2020. Convergence of edge computing and deep learning: A comprehensive survey. IEEE Communications Surveys & Tutorials. 22(2), 869-904.
DOI: https://doi.org/10.1109/COMST.2020.2970550

[45] Cummins, N., Amiriparian, S., Hagerer, G., et al. (editors), 2017. An image-based deep spectrum feature representation for the recognition of emotional speech. Proceedings of the 25th ACM international conference on Multimedia; 2017 Oct 23-27; Mountain View California USA.
DOI: https://doi.org/10.1145/3123266.3123371

[46] Ottl, S., Amiriparian, S., Gerczuk, M., et al. (editors), 2020. Group-level speech emotion recognition utilising deep spectrum features. Proceedings of the 2020 International Conference on Multimodal Interaction; 2020 Oct 25-29; Virtual Event Netherlands.
DOI: https://doi.org/10.1145/3382507.3417964

BILINGUAL PUBLISHING GROUP

Journal of Computer Science Research

https://journals.bilpubgroup.com/index.php/jcsr

REVIEW

Innovating Pedagogical Practices through Professional Development in Computer Science Education

Xiaoxue Du[1*], Ellen B Meier[2]

[1] *MIT Media Lab, MIT, Cambridge, MA 02139, USA*

[2] *Teachers College, Columbia University, New York City, NY 10027, USA*

ABSTRACT

Recent advancements in technology have opened up new avenues for educators to facilitate teaching and leverage more learning access in the digital age. As the demand for computational skills continues to grow in preparation for future careers, both teachers and students face the challenge of developing problem-solving, critical thinking, communication, and collaboration skills within an emerging digital landscape. Technology adoption, big data, cloud computing and artificial intelligence pose ongoing challenges for both teachers and students in adapting to the changing workforce development landscape. To tackle these challenges, the paper highlights the importance of exploring the implications of learning sciences in classroom teaching, developing a holistic vision for professional development in education, and understanding the complexities of teacher change. To effectively implement these components, it is crucial to adopt design approaches that prioritize student ownership in education and embrace the principles of inclusive education to reconceptualize the teaching practices in education and technology.

Keywords: Education; Computational thinking; Teacher education; Professional development; Design; Equity

1. Introduction

As computers continue to automate our routine and complex tasks, equity in technology access, content, and use becomes a key barrier to future learning opportunities. In particular, a recent emphasis on education requires the development of intellect, ethical judgment, societal understanding, and creativity [1]. The technological challenges raise the critical question of how to prepare teachers to face

*CORRESPONDING AUTHOR:
Xiaoxue Du, MIT Media Lab, MIT, Cambridge, MA 02139, USA; Email: xiaoxued@media.mit.edu

ARTICLE INFO
Received: 30 May 2023 | Revised: 9 July 2023 | Accepted: 13 July 2023 | Published Online: 20 July 2023
DOI: https://doi.org/10.30564/jcsr.v5i3.5757

CITATION
Du, X.X., Meier, E.B., 2023. Innovating Pedagogical Practices through Professional Development in Computer Science Education. Journal of Computer Science Research. 5(3): 46-56. DOI: https://doi.org/10.30564/jcsr.v5i3.5757

the unprecedented changes in the immediate future of education [2]. Teachers must acquire fundamental knowledge and adopt innovative teaching methods in order to effectively incorporate technology into their instruction and meet both the academic and social-emotional requirements of students in the realm of technology [3,4]. Drawing upon insights from the learning sciences and teacher education literature, technology possesses the capability to pave the way for groundbreaking teaching approaches within classrooms. By harnessing the power of technology, educators can unlock diverse learning opportunities that cater to the needs of all students, including those from diverse cultural and language backgrounds [5]. Emerging technologies could serve as cognitive tutors, peer learners, and conversational agents, in order to introduce students to novel methods of reflection, reasoning, and learning in their everyday lives [6,7].

The growing demands in computer science (CS) education among schools and educational entities have shown the need to strengthen students' knowledge and skills in problem-solving and analytical thinking [8,9]. Therefore, establishing meaningful pedagogical practices and fostering a culture of lifelong learning are crucial aspects when it comes to computer science education. The basic premise of the paper is that integrating CS is a complex process, which requires much more than simply "shoehorning" a new curriculum into the school day. Teachers need to cultivate the skills to design and establish a student-centered learning environment, purposefully enabling the effective integration of computer science education. Moreover, it is essential to foster a rigorous community space that encourages learners to make connections between their acquired knowledge and other areas within the computing field. This approach helps foster a sense of belonging to the wider computing community. In this paper, we will first synthesize key challenges and opportunities identified in computer science education. Then, we will introduce the key components of a research-based, systematic professional development approach to build teachers' capacity to design a student-centered learning environment. Finally, we will discuss the implications of strengthening teachers' professional development in the computing science education community.

2. Key challenges and opportunities

Implementing in school settings presents new challenges for educators for at least five key reasons. First, there is a lack of "shared meaning" [10] for computer science as an academic discipline in K-12 education. Teachers should strive to develop a shared comprehension of both content knowledge and effective pedagogical practices in order to seamlessly integrate them into their curriculum planning. Second, computational thinking is increasingly considered a foundational skill in the 21st century, but is not systematically addressed in the curriculum. It serves as a process for recognizing aspects of computation in the surroundings and introduces techniques from CS to understand both natural and artificial systems and processes [11]. Third, there is not a clear scope and sequence for standards in each grade, which creates challenges for educators interested in developing student learning plans to integrate CS across disciplines. Because of the lack of a scope and sequence, there is insufficient empirical evidence for student learning and a lack of clear assessment objectives to support content definition and sequencing. Fourth, teachers' professional development in computer science is a new process, which requires more empirical evidence and research to identify the core professional development content material and resources needed to prepare educators for designing student-centered learning experiences in education [12]. Lastly, recent research has emphasized the significance of nurturing young people's capacity to create through the acquisition of computing skills. The development of these skills holds substantial implications for their personal lives and the betterment of their communities. To ensure that students develop essential computing design skills, it is imperative to create an inclusive, motivating, and empowering learning environment. This could provide students with greater autonomy to code, break down complex problems, and apply their learning across various contexts to solve real-world problems. However, in many cases, the curriculum may focus exclusively on technical challenges and entry points, requiring students to have prior programming experience. This limits opportunities for students to engage critically with a broader curriculum and participate in a larger computing community. To address the challenge, it is important to create a wide and deep learning space for learners, allowing them to connect what they have learned to other computing and content fields and fostering a sense of belonging to the greater computing community [13].

One way to address the challenges faced by both educators and students in computer science education is to provide more professional development opportunities for educators that position teachers as designers and effectively integrate computational-oriented curriculum into daily learning and teaching. Teachers need the essential knowledge and skills to plan enriched lessons, select the most relevant user cases, and design curricula to develop students' skills in problem-solving, computational thinking, and critical awareness in education. In addition, in the design process, teachers can develop student-centered learning environments that allow students with different background knowledge to engage curriculum, develop their interests, and build confidence that empowers them to learn and grow in the computer science curriculum. Finally, more research should be conducted to develop high-quality professional development, which could, in turn, prepare a cohort of change leaders to innovate pedagogical practices in computer science beyond programming, while building local community networks to sustain the change and innovation in daily teaching.

3. Applying the innovating instruction model in education

The Innovating Instruction © model has been developed by the Center for Technology and School Change, Teachers College, Columbia University. The model is developed and built upon the theory of change, learning science, professional development theories, and the emerging capabilities of technology. In this model, teachers take on the role of designers collaborating with facilitators to co-design projects that can be implemented in their real school settings. Technology plays a pivotal role as a catalyst for driving pedagogical innovation and motivates teachers as changemakers to advocate and sustain change in classroom teaching [14,15]. (see **Table 1**, CTSC Professional Development model: Innovating Instruction model).

The model comprises three fundamental components: Design, Situate, and Lead, all aimed at assisting teachers in transforming their teaching and learning approaches. It is imperative for teachers to grasp effective teaching practices aligned with principles derived from the learning sciences. This understanding enables them to design environments that foster meaningful learning experiences and facilitate opportunities for students to deepen their understanding of the subject. Equipping teachers with the capacity to design curriculum goals, employ formative assessments, and engage diverse students in inquiry-based learning environments holds great promise in supporting their professional growth [15].

The Situate component plays a crucial role in customizing the learning experience for each teacher's classroom and their students. It not only showcases engaging pedagogical practices through a hands-on approach but also offers personalized support to teachers. Incorporating insights from the learning sciences, it establishes a foundation for comprehending the intricacies of learning and thinking. The science of learning and development (SoLD) approach has been utilized to expound upon the "whole child model", which underscores the necessity of addressing various aspects of students' academic, cognitive, ethical, physical, psychological, and socio-emotional well-being. Specifically, creating a supportive environment fosters strong relationships and a sense of community among students. The situated approach encourages teachers to position students as active "knowledge-builders" within an inquiry-based learning environment [16,17].

Finally, the Lead component of the model focuses on preparing teachers to become leaders and col-

Table 1. Innovating instruction model.

DESIGN—Engage teachers as designers of student-centered, authentic learning experiences	
1. Embrace a Design Approach	Model and support a backwards design approach to project planning that creates meaningful learning experiences for students.
2. Enrich Content Knowledge	Provide opportunities for deepening teachers' understanding of content, including cross-curricular connections, learning standards, and student misconceptions.
3. Integrate Assessment Practices	Facilitate the design of authentic assessment and data use to identify and respond to student needs.
4. Leverage Digital Tools	Teach the integration of digital tools as part of the design process to facilitate interactive student learning and to enrich content.
SITUATE—Provide learning experiences for teachers that respects them as professionals and adapts the learning for their particular school and situation	
1. Contextualize Teacher Learning	Situate the design work in the professional lives of teachers in order to connect deeply to the realities of teachers' classrooms and their students.
2. Model Effective Practice	Provide interactive, hands-on professional development that engages teachers and models project-based learning with available tools and resources.
3. Individualize Support	Co-construct project plans based on student and curricular needs, provide ongoing support for classroom implementation, and facilitate reflection on teaching and learning.
LEAD—Support leaders in guiding and sustaining change initiatives, while positioning teachers as agents of change	
1. Envision Change	Prioritize instructional leadership and develop actionable goals to promote change in self-identified areas of need.
2. Empower Leadership at All Levels	Provide a forum for identifying leaders--administrators, teachers, and community members--who can spearhead efforts that contribute to the common vision.
3. Sustain A Culture for Innovation	Scaffold educators' efforts toward instructional innovation to realize goals beyond the immediate scope of the professional development.
4. Research	Lead research that informs the transformative use of technology in existing and emerging practices in schools, while contributing to evolving scholarship on innovations for teaching and learning

Figure. Situate. Design. Lead. © CTSC's Professional Development Model for Innovating Instruction, Detailed

laborates with building administrators to empower individual leadership and foster a culture of change and innovation. The guiding theoretical framework is the theory of change, which recognizes the complexity of the learning environment and the essential components required to facilitate transformative shifts in education. Strategic planning, the effective implementation of new teaching approaches, and the continuous development of teachers' beliefs about teaching and learning are emphasized as pivotal factors for driving meaningful change within the school system. The model also underscores the importance of creating "shared meaning" among key stakeholders, considering the institutional, historical, and cultural perspectives that shape relationships and language in the field of education [10,17].

Two recent National Science Foundation (NSF) grants—the Systemic Transformation of Inquiry Learning Environments (STILE 1.0) for STEM (Exploratory Award No. DRL-1238643) and STILE 2.0 (Early-Stage Design and Development Award No. DRL-1621387)—have established the model's positive impact on teachers' ability to design projects, to shift from disciplinary to transdisciplinary project design, and to shift instructional thinking to include inquiry-based approaches. The research findings from the STILE initiatives demonstrate the positive impacts of the model on teachers' pedagogical change as defined by shifts in STEM perspectives, STEM design practices, and STEM classroom practices [11]. The research in thirteen diverse school contexts, included 169 classroom visitations, 372 planning meetings, and over 51 hours of administrator interaction. The total average dosage was estimated at 61 hours per teacher, supporting 169 teachers in the New York City public school systems, cumulatively ultimately serving over 7536 students. The research identified positive changes that teachers have made under the STILE 2.0 program, specifically in teachers' ability to design projects, shift from a disciplinary/subject-orientation to a more sophisticated transdisciplinary focus, and broaden their instructional thinking to include more inquiry-based approaches [11].

Two recent grants from the National Science Foundation (NSF), namely STILE 1.0 (Systemic Transformation of Inquiry Learning Environments for STEM) and STILE 2.0, have demonstrated the positive impact of the model on teachers' capacity to design projects, transition from disciplinary to transdisciplinary project design, and adopt inquiry-based approaches in their instruction. The research findings from these initiatives highlight the constructive influence of the model on teachers' pedagogical change, as evidenced by shifts in STEM perspectives, STEM design practices, and STEM classroom practices. The research study took place in thirteen different school settings, comprising 169 classroom observations, 372 planning meetings, and over 51 hours of engagement with administrators. On average, each teacher received around 61 hours of support from the program, benefiting a total of 169 teachers in the New York City public school systems and ultimately influencing over 7536 students. The research findings revealed significant improvements in teachers' abilities to design projects, shift from a narrow disciplinary focus to a broader transdisciplinary perspective, and enhance their instructional thinking by integrating more inquiry-based methods. These positive changes were observed as a result of the STILE 2.0 program [11].

4. Visionary goals in professional development and CS education

4.1 A grand vision for professional development in CS education

To effectively introduce computer science curriculum in schools, professional development plays a crucial role for teachers to adopt a broader vision for the teaching profession that prioritizes learning and the needs of learners from diverse communities [18]. For example, professional development programs can provide teachers with training and resources to enhance their pedagogical skills in computer science education. This could include workshops on integrating technology into lessons, designing engaging coding activities, or implementing project-based learning in the computer science classroom. By equipping teachers with the necessary knowledge and tools, professional development empowers them to create meaningful learning experiences and cater to the diverse needs of their students in the realm of computer science [19,20].

With access to resources in computer science (CS) education, teachers receive valuable support in designing student-centered learning experiences that foster the development of students' identity and their willingness to actively engage in the broader computing community [21]. For instance, through professional development, teachers can learn about various tools, platforms, and instructional strategies that enable them to create interactive coding projects, collaborative programming exercises, or real-world CS applications. By incorporating these resources into their teaching, teachers can empower students to explore their interests, develop problem-solving skills, and cultivate a sense of belonging within the computing field [22]. This not only enhances students' learning experiences but also nurtures their enthusiasm and motivation to actively participate in the wider CS community beyond the classroom [23]. As a result, students will have more ownership and responsibility to explore concepts beyond essential programming ideas (e.g., loops, arrays, conditional statements). They will utilize their skills to build a deeper understanding of how these concepts apply to broader social and cultural contexts. This expanded perspective encourages students to consider the practical applications of computer science in various domains, such as healthcare, environmental sustainability, or social justice [24,25]. By connecting programming skills to real-world contexts, students develop a more comprehensive understanding of the societal impact and significance of computer science, empowering them to become critical thinkers and active contributors to their communities [26,27].

Finally, CS professional development should deepen teachers' content knowledge and provide curriculum resources, encouraging teachers to utilize different means, access, and representations to simulate abstract concepts in order to develop students' interests and curiosity in the field [28,29]. Specifically, ongoing research in professional development should prepare teachers to continuously innovate pedagogical practices to design, pilot and implement computer science curriculum in classroom settings [30]. It should invite educators to consider a broader, culturally relevant approach that designs curriculum situated for a range of learners, especially students who have been traditionally under-represented in the computing fields [31]. The CS literature has shown that women and students of color have been overlooked and excluded by the wider computing community [8], therefore, it is critical for teachers to identify effective approaches for engaging all students in the field of computer science education [32,33].

4.2 The complexity of the teacher changes in CS education

Research findings demonstrate that thoughtful professional development can significantly impact teachers' ability to incorporate technology into their classroom practices [34]. Studies indicate that teachers' beliefs and practices can evolve when provided with clear and specific instructions during professional development sessions in CS education [35]. In the context of computer science education, ongoing research should place a strong emphasis on exposing teachers to the design process and enabling them to explore the integration of technology in projects that promote a shift in instructional thinking [36]. For example, professional development programs can provide teachers with opportunities to engage in coding and computational thinking activities themselves, allowing them to experience firsthand the creative process involved in designing and developing computer programs. These programs can also offer teachers access to various technology tools and platforms specifically designed for computer science education. This includes platforms for creating interactive programming projects, virtual environments for exploring computer science concepts, and collaborative coding platforms. Through hands-on workshops and training sessions, teachers can gain practical knowledge of these tools and learn effective strategies for incorporating them into their teaching. To summarize, professional development initiatives offer teachers increased opportunities to gain familiarity with a variety of technology tools, demonstrate their use in classroom instruction, and provide checkpoints for reflection on the implementation process [37,38]. By successfully employing new practices and research-based methods, teachers receive further support in assimilating innovative approaches into their existing belief systems [39,40].

4.3 Reconceptualizing teaching practices in CS education

In the realm of computer science education, it is crucial to critically examine practices and emerging research in general education [41]. For example, teacher education programs can incorporate pedagogical strategies that promote hands-on learning experiences, such as coding workshops or robotics projects. By engaging students in these practical activities, they can develop a deeper understanding of programming concepts and gain valuable problem-solving skills. Additionally, project-based approaches can be implemented in computer science education [42]. For instance, students can work on real-world projects like designing a mobile application or creating a website for a local business. These projects not only reinforce technical skills but also encourage critical thinking and creativity as students navigate challenges and make design decisions [43].

Collaborative problem-solving is another important strategy to consider. An example of this could be organizing group activities where students collaborate to solve complex coding problems or develop a software solution together [44]. Through teamwork, students learn how to communicate effectively, share ideas, and leverage each other's strengths to achieve a common goal [45]. To stay up-to-date with the latest advancements, it is important for educators to explore research and developments in educational technology, computational thinking, and computer science instruction [46]. For instance, they can learn about new tools and platforms that facilitate interactive learning experiences or discover innovative teaching methods that enhance student engagement and understanding [47].

By reconceptualizing teacher education in computer science, educators can better prepare future teachers to design and deliver engaging and meaningful learning experiences [48]. For example, they can develop new curricular materials that incorporate coding exercises, multimedia resources, and interactive simulations to make learning more interactive and dynamic [49]. Furthermore, assessment approaches can be adapted to evaluate students' computational thinking skills, creativity, and problem-solving abilities [50]. This could involve designing coding challenges or projects that require students to apply their knowledge in practical contexts, as opposed to traditional exams or quizzes that solely focus on theoretical concepts [51]. By focusing on these aspects, students not only gain technical knowledge, but also develop the necessary skills to thrive in an ever-changing digital landscape. They become equipped with computational thinking skills, creativity, and problem-solving abilities, which are highly sought-after in the industry [52,53].

4.4 Embracing computer science education for all students with real-world connections

Specifically, effective professional development should provide teachers with more accessible resources that reduce barriers for teachers to learn, adopt, and integrate into the daily curriculum [54]. For instance, providing teachers sample curriculum that allows teachers to adapt and integrate into current lesson plans, could be effective for teachers to develop capacity in computer science disciplines, develop students' interests in exploring CS topics and encourage educators to understand the value of designing situated, culturally relevant computer science curriculum [55]. In addition, professional development should provide teachers with an easy-to-use platform that could encourage students to quickly build prototype, implementation solutions without creating complicated programming syntax [56]. For instance, the growing usage of block-based programming languages (e.g., PoseBlocks, App Inventor) has shown the value for students to build solutions, implement design, and create functional mobile applications without complex debugging and programming process [57,58]. The ongoing research study could further explore core processes and components that prepare teachers to use, adapt, and implement computer science curriculum with technology integration across diverse classroom settings domestically and internationally in computer science education [59,60].

5. Conclusions

To enhance teacher education in the field of computer science, it is crucial to equip teachers with the skills to strategically utilize technology in designing engaging curriculum that promotes deep learning in computer science education. Given the complexity of school systems, collaborative efforts involving researchers, scientists, and professionals are necessary to drive these transformative shifts. To prepare for the change, the *Innovating Instruction* model has shown an effective model to incorporate interactive and hands-on activities, project-based learning, and real-world applications of computer science concepts, tailor their instruction to meet the diverse needs and interests of their students, and constantly refine their instructional techniques and become more effective educators. By nurturing a community of practice and fostering collaboration among teachers, researchers, and professionals, the field can collectively drive the adoption of innovative technology and create a more engaging and impactful learning environment for students.

Author Contributions

Both authors made equal contributions to the manuscript.

Conflict of Interest

The authors have no conflicts of interest to declare.

Data Availability Statement

Situate. Design. Lead. © CTSC's Professional Development Model for Innovating Instruction is available at the website from Center for Technology and School Change, Teachers College, Columbia University. https://ctsc.tc.columbia.edu/

References

[1] Breazeal, C., 2022. AI Literacy for All with Prof. Cynthia Breazeal [Internet]. Available from: https://openlearning.mit.edu/news/ai-literacy-all-prof-cynthia-breazeal

[2] Darling-Hamond, L., Oakes, J., 2019. Preparing teachers for deeper learning. Harvard Education Press: Cambridge, MA.

[3] Podolsky, A., Kini, T., Darling-Hammond, L., 2019. Does teaching experience increase teacher effectiveness? A review of US research. Journal of Professional Capital and Community. 4(4), 286-308.

[4] Sutcher, L., Darling-Hammond, L., Carver-Thomas, D., 2019. Understanding teacher shortages: An analysis of teacher supply and demand in the United States. Education Policy Analysis Archives. 27(35).

[5] National Academies of Sciences, Engineering, and Medicine, 2018. How people learn II: Learners, contexts, and cultures. National Academies Press: Washington, D.C.

[6] Papadopoulos, I., Lazzarino, R., Miah, S., et al., 2020. A systematic review of the literature regarding socially assistive robots in pre-tertiary education. Computers & Education. 155, 103924.

[7] Rosenberg-Kima, R.B., Koren, Y., Gordon, G., 2020. Robot-supported collaborative learning (RSCL): Social robots as teaching assistants for higher education small group facilitation. Frontiers in Robotics and AI. 6, 148.

[8] Grover, S., Pea, R., Cooper, S., 2015. Designing for deeper learning in a blended computer science course for middle school students. Computer Science Education. 25(2), 199-237.

[9] Hsu, T.C., Chang, S.C., Hung, Y.T., 2018. How to learn and how to teach computational thinking: Suggestions based on a review of the literature. Computers & Education. 126, 296-310.

[10] Fullan, M., 2016. The new meaning of educational change. Teachers College Press: New York.

[11] Grover, S., Pea, R., 2013. Computational thinking in K-12: A review of the state of the field. Educational Researcher. 42(1), 38-43.

[12] Webb, M., Bell, T., Davis, N., et al. (editors), 2017. Computer science in the school curriculum: Issues and challenges. Tomorrow's Learning: Involving Everyone. Learning with and about Technologies and Computing: 11th IFIP TC 3 World Conference on Computers in Education, WCCE 2017; 2017 Jul 3-6; Dublin, Ireland. p. 421-431.

[13] Du, X., Parks, R., Tezel, S., et al. (editors), 2023. Designing a computational action program to tackle global challenges. SIGCSE 2023: Proceedings of the 54th ACM Technical Symposium on Education; 2023 Mar 15-18; Toronto ON, Canada. New York: Association for Computing Machinery. p. 1320-1320.

[14] Meier, E.B., Mineo, C., 2021. Pedagogical challenges during COVID: Opportunities for transformative shifts. Handbook of research on transforming teachers' online pedagogical reasoning for engaging K-12 students in virtual learning. IGI Global: Hershey. pp. 86-108.

[15] Meier, E.B., 2021. Designing and using digital platforms for 21st century learning. Educational Technology Research and Development. 69(1), 217-220.

[16] Darling-Hammond, L., Flook, L., Cook-Harvey, C., et al., 2020. Implications for educational

practice of the science of learning and development. Applied Developmental Science. 24(2), 97-140.

[17] Scardamalia, M.B., 2014. Knowledge building and knowledge creation. Cambridge handbook of the learning sciences. Cambridge University Press: Cambridge. pp. 297-417.

[18] Harju, V., Niemi, H., 202). Newly qualified teachers' support needs in developing professional competences: The principal's viewpoint. Teacher Development. 24(1), 52-70.

[19] Ng, D.T.K., Lee, M., Tan, R.J.Y., et al., 2022. A review of AI teaching and learning from 2000 to 2020. Education and Information Technologies. 1-57.

[20] Dash, B.B., 2022. Digital tools for teaching and learning English language in 21 st century. International Journal Of English and Studies. 4(2), 8-13.

[21] Biswas, S., Benabentos, R., Brewe, E., et al., 2022. Institutionalizing evidence-based STEM reform through faculty professional development and support structures. International Journal of STEM Education. 9(1), 1-23.

[22] McGill, M.M., Reinking, A., 2022. Early findings on the impacts of developing evidence-based practice briefs on middle school computer science teachers. ACM Transactions on Computing Education. 22(4), 1-29.

[23] Apiola, M., Sutinen, E., 2021. Design science research for learning software engineering and computational thinking: Four cases. Computer Applications in Engineering Education. 29(1), 83-101.

[24] Casey, E., Jocz, J., Peterson, K.A., et al., 2023. Motivating youth to learn STEM through a gender inclusive digital forensic science program. Smart Learning Environments. 10(1), 2.

[25] Tissenbaum, M., Weintrop, D., Holbert, N., et al., 2021. The case for alternative endpoints in computing education. British Journal of Educational Technology. 52(3), 1164-1177.

[26] Schaper, M.M., Smith, R.C., Tamashiro, M.A., et al., 2022. Computational empowerment in practice: Scaffolding teenagers' learning about emerging technologies and their ethical and societal impact. International Journal of Child-Computer Interaction. 34, 100537.

[27] Tsortanidou, X., Daradoumis, T., Barberá, E., 2019. Connecting moments of creativity, computational thinking, collaboration and new media literacy skills. Information and Learning Sciences. 120(11/12), 704-722.

[28] Alfaro-Ponce, B., Patiño, A., Sanabria-Z, J., 2023. Components of computational thinking in citizen science games and its contribution to reasoning for complexity through digital game-based learning: A framework proposal. Cogent Education. 10(1), 2191751.

[29] Ketelhut, D.J., Mills, K., Hestness, E., et al., 2020. Teacher change following a professional development experience in integrating computational thinking into elementary science. Journal of Science Education and Technology. 29, 174-188.

[30] Bragg, L.A., Walsh, C., Heyeres, M., 2021. Successful design and delivery of online professional development for teachers: A systematic review of the literature. Computers & Education. 166, 104158.

[31] Mystakidis, S., Fragkaki, M., Filippousis, G., 2021. Ready teacher one: Virtual and augmented reality online professional development for K-12 school teachers. Computers. 10(10), 134.

[32] Li, M., 2020. Multimodal pedagogy in TESOL teacher education: Students' perspectives. System. 94, 102337.

[33] Kafai, Y.B., Baskin, J., Fields, D., et al. (editors), 2020. Looking ahead: Professional development needs for experienced CS teachers. SIGCSE'20: Proceedings of the 51st ACM Technical Symposium on Computer Science Education; 2020 Mar 11-14; Portland OR, USA. New York: Association for Computing Machinery. p. 1118-1119.

[34] Tshukudu, E., Cutts, Q., Goletti, O., et al. (editors), 2021. Teachers' views and experiences on teaching second and subsequent programming languages. ICER 2021: Proceedings of the 17th

ACM Conference on International Computing Education Research; 2021 Aug 16-19; Virtual Event, USA. New York: Association for Computing Machinery. p. 294-305.

[35] Rich, P.J., Larsen, R.A., Mason, S.L., 2021. Measuring teacher beliefs about coding and computational thinking. Journal of Research on Technology in Education. 53(3), 296-316.

[36] Bereczki, E.O., Kárpáti, A., 2021. Technology-enhanced creativity: A multiple case study of digital technology-integration expert teachers' beliefs and practices. Thinking Skills and Creativity. 39, 100791.

[37] Griful-Freixenet, J., Struyven, K., Vantieghem, W., 2021. Exploring pre-service teachers' beliefs and practices about two inclusive frameworks: Universal design for learning and differentiated instruction. Teaching and Teacher Education. 107, 103503.

[38] Dignath, C., Rimm-Kaufman, S., van Ewijk, R., et al., 2022. Teachers' beliefs about inclusive education and insights on what contributes to those beliefs: a meta-analytical study. Educational Psychology Review. 34(4), 2609-2660.

[39] Almazroa, H., Alotaibi, W., 2023. Teaching 21st century skills: Understanding the depth and width of the challenges to shape proactive teacher education programmes. Sustainability. 15(9), 7365.

[40] Bhutoria, A., 2022. Personalized education and artificial intelligence in the United States, China, and India: A systematic review using a human-in-the-loop model. Computers and Education: Artificial Intelligence. 3, 100068.

[41] Bozkurt, A., 2020. Educational technology research patterns in the realm of the digital knowledge age. Journal of Interactive Media in Education. (1).

[42] Çiftci, S., Bildiren, A., 2020. The effect of coding courses on the cognitive abilities and problem-solving skills of preschool children. Computer Science Education. 30(1), 3-21.

[43] Saad, A., Zainudin, S., 2022. A review of Project-Based Learning (PBL) and Computational Thinking (CT) in teaching and learning. Learning and Motivation. 78, 101802.

[44] Wang, Y., 2023. The role of computer supported project-based learning in students' computational thinking and engagement in robotics courses. Thinking Skills and Creativity. 48, 101269.

[45] Bers, M.U., Blake-West, J., Kapoor, M.G., et al., 2023. Coding as another language: Research-based curriculum for early childhood computer science. Early Childhood Research Quarterly. 64, 394-404.

[46] Huang, W., Looi, C.K., 2021. A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. Computer Science Education. 31(1), 83-111.

[47] Yildiz Durak, H., Atman Uslu, N., Canbazoğlu Bilici, S., et al., 2022. Examining the predictors of TPACK for integrated STEM: Science teaching self-efficacy, computational thinking, and design thinking. Education and Information Technologies. 1-28.

[48] Lee, S.W.Y., Liang, J.C., Hsu, C.Y., et al., 2023. Students' beliefs about computer programming predict their computational thinking and computer programming self-efficacy. Interactive Learning Environments. 1-21.

[49] Lee, S.J., Francom, G.M., Nuatomue, J., 2022. Computer science education and K-12 students' computational thinking: A systematic review. International Journal of Educational Research. 114, 102008.

[50] Ung, L.L., Labadin, J., Mohamad, F.S., 2022. Computational thinking for teachers: Development of a localised E-learning system. Computers & Education. 177, 104379.

[51] Kallia, M., van Borkulo, S.P., Drijvers, P., et al., 2021. Characterising computational thinking in mathematics education: A literature-informed Delphi study. Research in Mathematics Education. 23(2), 159-187.

[52] Ogegbo, A.A., Ramnarain, U., 2022. A systematic review of computational thinking in science classrooms. Studies in Science Education. 58(2),

203-230.

[53] Chen, C.H., Liu, T.K., Huang, K., 2023. Scaffolding vocational high school students' computational thinking with cognitive and metacognitive prompts in learning about programmable logic controllers. Journal of Research on Technology in Education. 55(3), 527-544.

[54] Gao, X., Li, P., Shen, J., et al., 2020. Reviewing assessment of student learning in interdisciplinary STEM education. International Journal of STEM Education. 7(1), 1-14.

[55] Madkins, T.C., Martin, A., Ryoo, J., et al. (editors), 2019. Culturally relevant computer science pedagogy: From theory to practice. 2019 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT); 2019 Feb 27; Minneapolis, MN, USA. New York: IEEE. p. 1-4.

[56] Guskey, T.R., 2002. Does it make a difference? Evaluating professional development. Educational Leadership. 59(6), 45-51.

[57] Li, L., Ruppar, A., 2021. Conceptualizing teacher agency for inclusive education: A systematic and international review. Teacher Education and Special Education. 44(1), 42-59.

[58] Tissenbaum, M., Sheldon, J., Abelson, H., 2019. From computational thinking to computational action. Communications of the ACM. 62(3), 34-36.

[59] Madkins, T.C., Howard, N.R., Freed, N., 2020. Engaging equity pedagogies in computer science learning environments. Journal of Computer Science Integration. 3(2).

[60] Morales-Chicas, J., Castillo, M., Bernal, I., et al., 2019. Computing with relevance and purpose: A review of culturally relevant education in computing. International Journal of Multicultural Education. 21(1), 125-155.

Journal of Computer Science Research

https://journals.bilpubgroup.com/index.php/jcsr

ARTICLE

Expert Review on Usefulness of an Integrated Checklist-based Mobile Usability Evaluation Framework

Hazura Zulzalil[1*]***, Hazwani Rahmat***[2]***, Abdul Azim Abd Ghani***[1]***, Azrina Kamaruddin***[1]

[1] *Department of Software Engineering and Information System, Universiti Putra Malaysia, UPM Serdang, Selangor, 43400, Malaysia*

[2] *Department of Information Technology, Centre for Diploma Studies, Universiti Tun Hussein Onn Malaysia, Pagoh Higher Education Hub, Pagoh, Johor, 84600, Malaysia*

ABSTRACT

Previous mobile usability studies are only pertinent in the context of ergonomics, physical user interface, and mobility aspects. In addition, much of the previous mobile usability conception was built on desktop computing measurements, such as desktop and web application checklists, or scarcely addressed the mobile user interface. Moreover, the studies focus mainly on interface features for desktop applications and do not reflect comprehensive mobile interface features such as navigation drawers and spinners. Therefore, conducting usability evaluation using conventional usability measurement would result in irrelevant results. In addition, the resulting works are tailored for usability testing, which requires highly skilled evaluators and usability specialists (e.g., usability testers and user experience designers), who are rarely integrated into a development team. The lack of expertise could lead to unreliable usability evaluations. This paper presents a review from industrial experts on a comprehensive and feasible usability evaluation framework developed in our previous work. The framework is dedicated to smartphone apps, which integrate evaluator skills and design concerns. However, there is no evidence of its usefulness in practice. Therefore, the usefulness of the framework measurement for evaluating apps' usability in the eyes of non-usability specialists is empirically assessed in this paper through an expert review. The expert review involved eleven industrial developers and was complemented by a semi-structured interview. The method is replicated in comparison with a framework from another study. The findings show that the formulated framework significantly outperformed the framework ($p = 0.0286$) from other studies with large effect sizes ($r = 1.81$) in terms of usefulness.

Keywords: Usability framework; Mobile usability; Usability evaluation; Expert review; Heuristic walkthrough

*CORRESPONDING AUTHOR:

Hazura Zulzalil, Department of Software Engineering and Information System, Universiti Putra Malaysia, UPM Serdang, Selangor, 43400, Malaysia; Email: hazura@upm.edu.my

ARTICLE INFO

Received: 28 June 2023 | Revised: 24 July 2023 | Accepted: 26 July 2023 | Published Online: 31 July 2023

DOI: https://doi.org/10.30564/jcsr.v5i3.5816

CITATION

Zulzalil, H., Rahmat, H., Ghani, A.A.A., et al., 2023. Expert Review on Usefulness of an Integrated Checklist-based Mobile Usability Evaluation Framework. Journal of Computer Science Research. 5(3): 57-73. DOI: https://doi.org/10.30564/jcsr.v5i3.5816

1. Introduction

The user interface serves as a conduit between human and computer interactions. The evolution of the user interface has progressed from the command line interface (CLI) of a console to the graphical user interface (GUI), and later to the web user interface and mobile user interface. The evolution of the user interface has characterized usability dimensions differently for mobile applications (apps). CLI requires high memorability for competence and knowledge of entering massive commands. On the other hand, a GUI adopts a graphical representation for user interaction. Thus, learnability, effectiveness, and efficiency come first in a usable application. The Web user interface operates mostly on hypertext and multimedia elements, forming the navigational system and interconnected content. Consequently, consistency, simplicity, and information architecture play an important role in the effectiveness, efficiency, and navigability of web applications.

Likewise, a mobile user interface is shaped by its technological features. Physical device features and limitations, an integrated sensor such as proximity, tactile, or image recognition sensor, and its context of use has introduced emergent usability properties in characterizing the mobile usability dimension. Additionally, the unique data entry model, such as the use of a stylus, gestures, and the virtual keyboard, has taken place as the input device instead of a mouse and keyboard.

Mobile applications (apps) are used on-the-go, thus opening them to divided attention while used in different mobility conditions (e.g., sitting, walking, and driving). Apps offer support for a broad range of tasks (e.g., streaming online movies, browsing information, and performing online transactions) without the need for a computer. However, as mobile operating systems advanced, the user interface of mobile applications was rapidly enhanced through software updates. The update involves logical user interfaces (LUI) and graphical user interfaces (GUI), which affect apps' usability, rather than physical user interfaces (PUI), which affect the device's usability. New features and functionalities are introduced to enhance the mobile user interface with constant version updates. Interface features such as the navigation drawer and expansion panel are used to maximize the limited screen size. Meanwhile, snack boxes and toasts are used to deliver a prompt visual response when handling divided attention and mobility conditions, which could result in an accidental activation. Consequently, usability criteria such as connectivity, relevance, and responsiveness are the highlights of conceptualizing app usability. However, the smartphone is used by users of all ages with various levels of computing background. Thus, usability criteria such as familiarity, flexibility, and appropriateness are also highlighted in denoting mobile usability.

The conceptualization of the mobile usability dimension has been widely studied [1,2]. Numerous attempts have been made to characterize usability in view of performance-based measures [3], the physical user interface [4,5], mobile device concerns [6], usability principles [7], usability criteria [8], and interface features [9-12].

This study presents a measurement for evaluating the usability of mobile applications in the context of integrated evaluator skills. The measurements are developed by capturing the interface features, usability criteria, and design pattern, which augment the evaluation basis from multiple evaluators' viewpoints. The integration involves a comprehensive bridging of the semantic gap between different abstraction levels of usability constructs; interface features, usability features, and usability criteria into one integrated framework.

The remainder of this article is divided into eight sections. Section 2 reviews the existing mobile usability evaluation framework. Section 3 describes how the framework measurement is formulated. The resulting framework is presented in Section 4. The evaluation of the framework's usefulness is described in Section 5. Meanwhile, the results and discussions, and threats to validity are discussed in Sections 6 and 7. Finally, in Section 8, conclusions and future studies are presented.

2. Related works

Mobile usability evaluation methods such as field testing and lab experiments have been introduced for evaluating mobile applications. However, the limitations and difficulties resulting from these methods favor traditional usability evaluation methods such as usability testing and inspection methods. Inspection methods such as heuristic evaluation (HE) gain wide acceptance in industrial practice due to their simplicity, low cost, and short time, with no additional equipment required. Hence, a wide variety of heuristic evaluation methods besides Nielsen and Molich's ten heuristics have been developed, such as checklist-based heuristics [6]. Consequently, the use of a checklist was extended to frameworks in an effort to characterize usability. This has benefits for usability conception, design, and evaluation purposes.

This section discusses the checklist-based frameworks in four categories. The categories are attribute-based frameworks, integrated-based frameworks, theoretical-based frameworks, and decision-based frameworks. The first and third categories serve to conceptualize the usability dimension. Meanwhile, the second approach focuses on the interface feature, and the last category is specific to decision-making and prioritizing usability constructs. The framework's structural base, evaluator viewpoint, intended platform, and scope of evaluation item for each framework concerning the aim of each literature's work were compared.

2.1 Attribute-based frameworks

The increasing capabilities of mobile phones have encouraged several usability investigations to characterize the usability dimension of mobile phones. Initially, as smartphones started to emerge in 2009, Hussain and Kutar adopted a Goal Question Metric (GQM) approach for their framework in conceptualizing usability dimensions [3]. Based on ISO 9241-11 usability criteria (effectiveness, efficiency, and satisfaction) as usability parameters for the goal, a set of questions was associated with each goal, in a checklist form. The questions were further used to derive metrics as a performance indicator for each goal and question. However, since mobile-specific interface features for smartphones were just being launched at the time, the outcome focuses more on the logical user interface.

Further, another study by Saleh et al. [8] adopted the same approach, GQM, in constructing their framework. In contrast to Hussain and Kutar's approach [3], they developed a more comprehensive set of usability criteria denoting mobile applications (apps) by extending the PACMAD model [5] as the base of their framework structure.

Though both studies managed to conceptualize usability, the use of the GQM approach resulted in a metrics-oriented performance-based checklist that scarcely acknowledges the characteristics of smartphones, such as screen size and interaction method, which reflect the interface features of apps in detail.

2.2 Integrated-based frameworks

Insufficient literature on mobile phone characteristics concerning its interface feature has inspired the effort of an in-depth comprehension of the mobile interface feature. As a result, the abstraction levels for this type of framework are realized as an organization of mobile interface features.

In addressing the comprehensive aspect of usability issues on a mobile phone, Mugisha et al. articulate their framework in view of mobile phone UI practitioners [13]. Based on a review of usability principles, they defined five categories of UIs tailored for a feature phone. A pairwise comparison approach was used in mapping the UIs to usability principles. A checklist relevant to the UIs was developed to match the usability principles.

As a continuation, Xu and Jonsson [14] devised their framework by determining common interface features for tablet applications. The identified interface features were grouped into three categories: UI input, UI components, and UI characteristics. Each UI, which was paired with a developed checklist, was mapped to the usability principles based on their effect and relationship. Though tailored for tablet applications, and acknowledging smartphone characteristics in their work, the UIs mainly reflect desktop and web application UIs and features such as input, hardware, bookmarks, and headers.

2.3 Theoretical-based frameworks

The usability framework, which was developed based on usability conceptions of principles and criteria, mainly revolves around the effectiveness of existing usability measurements for evaluating apps. For example, Dubey and Rana [9] acknowledged the characteristics and features of mobile devices. They doubt the effectiveness of existing usability measurements on mobile phones. By hierarchically organizing usability indicators (principles), criteria, and properties based on a goal-mean relationship between the parameters, they formulated a framework for usability specialists to conduct an analytical evaluation of mobile phones. While focusing on the parameters of each abstraction level and all three categories of UIs (PUI, LUI, and GUI), their checklist suffers from redundancy, ambiguity, confusion, and indirectly measurable issues.

Pursuing a different approach, Gómez, Caballero, and Sevillano formed their framework by formulating a structure of heuristics and sub-heuristics, paired with a checklist based on their semantic relations [6]. They achieved excellent results in addressing mobile-specific usability issues while focusing on LUI and GUI. Unfortunately, though they argue for the effectiveness of a desktop-centered checklist for evaluating apps, a portion of their checklist stems from a web-based checklist that appears irrelevant for apps.

Judging by the limitations of mobile devices, Fatih Nayebi developed a heuristic-based framework for app evaluation [7]. A set of usability criteria established based on his review of academic and industrial heuristics, theories, and guidelines were assigned to the most relevant logical groups of the reviewed bibliographic references. Although he managed to address the characteristics of mobile devices, the proposed criteria were ambiguous and hardly addressed mobile interface features.

Further, arguing for the effectiveness of current usability measurement for mobile applications, Hoehle, Aljafari, and Venkatesh proposed a set of measurements for mobile applications in view of interface features based on measurement theory [12]. A content analysis approach was used to relate the constructs and variables. Though their work explicitly focused on apps, the measurements were tailored for Microsoft-based apps, and mobile interface features are not well addressed in their work. Instead, they emphasize aspects such as usability principles, aesthetics, and navigation.

2.4 Decision-based framework

The primary purpose of adopting a decision-based framework is to determine a usable mobile application. Lachgar and Abdelmounaim pursued an analytic hierarchy process in developing their framework. Grounded in measurement theory, he developed usability constructs and variables to facilitate the selection of usable mobile phones [15]. **Table 1** summarizes the literature review.

Earlier mobile usability studies emphasized the physical user interface. While the logical user interface persists across most computing platforms, rapid updates in smartphone technologies highlight the importance of the graphical user interface, particularly interface features. The coverage of IU studied previously conforms to the scope of UI covered in the reviewed framework from the age of feature phones to handhelds until smartphones, where PUI is scarcely studied in recent works.

3. Formulation of framework measurement

Representative definitions of usability by the industry (i.e., ISO 9241-11 [16], ISO 9126 [17]) and academia [5,18-22] are usually referred to most studies. In the context of mobile usability studies, Harrison et al. [5] work, which extends Nielsen's usability conception in view of the ISO 9241-11 context, is deemed as a comprehensive reference [23]. However, neither metrics nor checklists are associated with their work, thus leaving little support for usability

Table 1. Literature review summary.

Types of Framework	**Attribute-based frameworks**		**Integrated-based frameworks**		**Theoretical-based frameworks**				**Decision-based framework**
Authors	Hussain and Kutar [3]	Saleh, Ismail, and Fabil [8]	Mugisha et al. [13]	Xu and Jonsson [14]	Dubey et al. [9]	Gómez, Caballero and Sevillano [6]	Fatih Nayebi [7]	Hoehle, Aljafari and Venkatesh [12]	Lachgar and Abdelmounaim [15]
Viewpoint	developer	developer	developer	developer	Usability specialist	Non-expert	Usability specialist	Usability specialist	Non-expert
Aims of research	Conceptualise usability dimension	Conceptualise usability dimension	Bridging different groups of usability constructs	Bridging different groups of usability constructs	Bridging different groups of usability constructs	Addressing mobile usability issues	Conceptualise usability dimension	Bridging different groups of usability constructs	Select best alternatives among available usability criteria
Base structure	ISO 9241-11	PACMAD	Usability principles	Usability principles	Usability principles	Mobile constraints	Usability principles	Usability principles	ISO 9421-11
Mapping of abstraction levels components	Analytic Hierarchy Process	Goal Question Metric	Goal Question Metric	Pairwise-comparison	Pairwise-comparison	Content analysis	Content analysis	Content analysis	Content analysis
Context of use	Understanding measurement	Understanding measurement	Prioritizing constructs	Prioritizing constructs	Correlating constructs	Correlating constructs	Correlating constructs	Correlating constructs	Decision making
Benefit	Explicit measurement interpretation	Explicit measurement interpretation	Consistent judgement	Consistent judgement	Thorough construct classification	Thorough construct classification	Thorough construct classification	Thorough construct classification	Consistent judgement
Drawback	Tunnel vision bias	Tunnel vision bias	Large number of evaluations	Large number of evaluations	Tunnel vision bias	Tunnel vision bias	Tunnel vision bias	Tunnel vision bias	Large number of evaluations
Countermeasure implemented	Not applicable	Not applicable	Establishing selection criteria	Establishing selection criteria	Expert review	Not applicable	Not applicable	Expert review	Content rating
Platform	smartphone	smartphone	Feature phone	Tablet	Feature phone	Smartphone, tablet	smartphone	smartphone	Handhelds
Scope	LUI	LUI	LUI, GUI	LUI, GUI	PUI, LUI, GUI	LUI, GUI	LUI, GUI	GUI	PUI, LUI, GUI

inspection.

Consequently, this study formulates a framework for usability conception by reviewing checklist-related bibliographic references. This approach has been demonstrated in previous studies by conducting a bibliographic search when constructing their checklist-based framework [6,9,24]. They restricted their search scope by covering only the relevant and most influential references. In contrast, this study exhaustively examined relevant bibliographic sources for possible quality criteria denoting usability, such as standards, guidelines, and requirements in their work. This study reviewed requirements up to the evaluation life cycle to obtain a comprehensive description of apps' usability [25]. This process, however, is restricted to mobile and app-related sources.

Eleven relevant bibliographic references were reviewed for possible usability criteria. As a result, a collection of 572 measures was compiled. Measures irrelevant in the context of app usability were excluded to ensure mobile-specific measurements. **Table 2** highlights the distribution of redacted measures.

Measures referring to desktop-based input devices such as mouse and keyboard; web-related user interfaces such as a link to related content, breadcrumb, and splash screen; physical user interfaces such as a widget, soft keys, and notification drawer; and shared devices concern; performance-based checklists such as task completion time, loading time, download speed, and installation are removed from the collection. In addition, cross-domain concerns such as user experience and interaction design are excluded from the collection of candidate checklists. Technical and design aspects, such as naming convention and image size, which require coding inspection, are also removed. Any game-specific measure was removed due to the exceptional design objective, which distinguishes them from general apps such as banking, utilities, etc. [26,27].

Measures specifically for the impaired user, such as blind users, are also removed due to their exceptional design concerns. Conflicting measures within the same bibliographic reference were both excluded due to no concrete design decision. Measures referring to application purpose, e.g., "Application's pur-

Table 2. Distribution of redacted measures.

Exclusion criteria	No. of excluded measures
Design-related measures (Ex: image size)	34
Web-specific design elements (Ex: refresh button, wish list)	15
User-impaired related measures (Ex: visually impaired)	13
Game-based measures	10
Physical user interface related measure (Ex: widget, soft keys, notification drawer)	11
Programming related measures (Ex: naming convention)	9
Performance-based measures (Ex: time taken, number of successful task)	9
Input and output devices (Ex: Desktop based input hardware, wearable)	8
Miscellaneous (Ex: conflicting measure between the same bibliographic reference, application purpose, design statement or fact)	8
Cross-domain concern (Ex: user experience, interaction design)	4
Technical-related measures (Ex: installation, system resource)	3
Device specific features (Ex: shared device, tablet specific)	3
No. of redacted measures	127

pose is understandable at first sight", were excluded, although they refer to usability criteria of understandability. The rationale is that the measure does not contribute towards achieving user goals in operating an app, thus irrelevant in representing usability for apps.

Additionally, given that the measures consist of different forms such as usability requirements, heuristics, checklists, guidelines, recommendations, and usability problems, it is not possible to review the bibliography in terms of quality criteria that share a similar meaning, the same name, or both. Instead, regardless of their original form, the measures were rephrased into a checklist.

These measures were reviewed using a content analysis technique to develop usability constructs for apps. Content analysis of the measures developed relevant emergent quality attributes and interface features, which later resulted in a paired usability checklist. Initially, a conceptual definition for each usability criterion and interface feature is established. The conceptual definitions are made as unambiguous as possible in the context of apps. Conceptually similar items and repeated items referring to the same usability criteria are grouped together and rephrased to homogenize the resulting usability checklist. In the case of conflicting items, items that coincide with other items are retained, and the conflicting items are excluded from the checklist pool. Finally, the usability criteria are examined for similarities and differences in terms of their design patterns. The usability criteria are then grouped under conceptual units of similar apps' design patterns and usability features.

4. An integrated usability evaluation framework

Characterizing usability solely on usability principles or usability attributes suffers from a lack of reflection on interface features in detail such as notification and interaction method, which is another aspect influencing mobile usability. On the other hand, depending solely on the UI component for the evaluation would be inappropriate for measuring the usability factor. In addition, considering apps' short time-to-market, where usability specialists are rarely involved during the usability evaluation, there is a need to support non-usability specialists in conducting reliable usability evaluations from their point of view. These suggest a mobile usability framework that integrates multiple evaluator viewpoints. However, this would result in different evaluation criteria, such as interface features and usability features, in contrast to usability specialists and developers, who mostly view usability in terms of usability heuristics and quality criteria. **Figure 1** illustrates the conceptual framework.

The usability constructs are abstracted into three tiers of abstraction levels: usability feature level, usability criteria level, and interface feature level. Each abstraction level of the framework denotes a construct that consists of a group of framework components. The framework components are paired with the usability checklists for usability inspection.

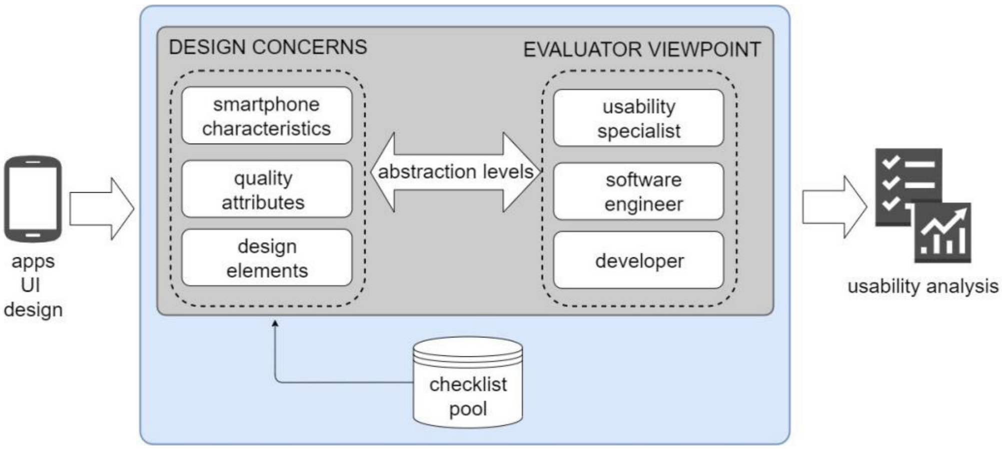


Figure 1. The conceptual framework.

Each framework tier reflects the different viewpoints of the usability evaluator and their level of expertise. The identified interface feature serves as the framework measurement, which formed the interface feature component for the lowest abstraction levels in the framework. The usability criteria are tied to the middle tier of the framework, the usability criteria component. Components for the top abstraction levels and usability features were identified by formulating conceptual units with similar usability criteria. **Figure 2** illustrates the framework abstraction level.

4.1 Usability features

Usability is commonly viewed by specialists in terms of constructs such as heuristics, principles, and guidelines, which are generally abstract. However, the mobile context of use, such as the interaction and operating environment, of the application on the intended platform has been regarded as an emergent property that affects usability [9,10,26]. Functional features of technology have been addressed in usability studies through design patterns [28-31]. The design pattern of app functionalities demonstrates the interaction complexities of smartphone apps. In this study, the design patterns are formulated as usability features to meet the viewpoint of usability specialists in conceptualizing usability as an emergent property of app interaction complexities. **Table 3** presents the elicited usability features in this study.

The usability features level denotes a collection of smartphone characteristics. These features are characterized by the attributes in the usability criteria level. It is formulated to meet the usability specialist's viewpoint in conceptualizing usability as an emergent property of app interaction complexities. It serves as an evaluation basis for both 1) specialists who view usability in terms of design patterns and 2) non-usability specialists, such as developers and designers, who could benefit from understanding usability in terms of design functionalities, in conducting usability evaluation.

4.2 Usability criteria

Characterizing usability solely by either usability principles or usability attributes suffers from a lack of reflection on interface features in detail such as notification and interaction methods, which is another aspect influencing mobile usability. However, on

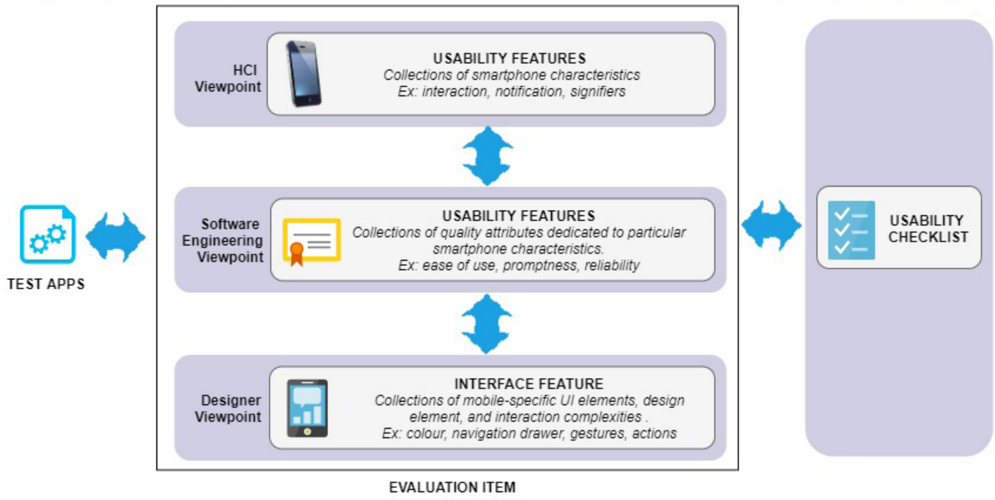

TEST APPS
HCI Viewpoint
USABILITY FEATURES
Collections of smartphone characteristics
Ex: interaction, notification, signifiers
Software Engineering Viewpoint
USABILITY FEATURES
Collections of quality attributes dedicated to particular smartphone characteristics.
Ex: ease of use, promptness, reliability
Designer Viewpoint
INTERFACE FEATURE
Collections of mobile-specific UI elements, design element, and interaction complexities .
Ex: colour, navigation drawer, gestures, actions
EVALUATION ITEM
USABILITY CHECKLIST

Figure 2. Framework abstraction level.

Table 3. Components of the usability features.

- F01 Interaction
- F02 Notification
- F03 Permission
- F04 Signifiers
- F05 Aesthetic
- F06 Presentation
- F07 Navigation
- F08 Information Architecture
- F09 Search
- F10 Data Entry
- F11 Workflow
- F12 Selection

the other hand, depending solely on the UI component for the evaluation would be inappropriate for measuring the usability factor. Therefore, this study bridged usability constructs, usability criteria, and interface features together with usability features.

The usability criteria level consists of a collection of usability attributes addressing the corresponding usability feature in the top tier. It emphasizes usability evaluation from a software engineering perspective. **Table 4** lists the components of the usability criteria.

The label next to each usability criterion denotes the usability features they are associated with. The usability criteria and interface features in the next tier facilitate usability evaluation and the perception of the evaluator in the domain of software engineering and development.

4.3 Interface features

The interface feature level defines components that are tied to the usability criteria in the middle tier. This level facilitates technical evaluators (e.g., analysts, designers, etc.) who perceive usability in view of the design context approach. It evaluates usability in view of design elements. Table 5 lists the components of the interface features.

In the formulated framework, each usability feature is decomposed into several usability criteria (as in a one-to-many relationship). However, a usability criterion is tied to more than one checklist, assessing different UI elements. Likewise, it is also possible for the UI elements to be associated with more than one usability criteria (as in a many-to-many relationship). **Table 6** exhibits the partial list of the paired usability checklist.

Table 4. Components of the usability criteria.

- Responsiveness (F01)
- Interactivity (F01)
- Playability (F01)
- Ease of Use (F01)
- Safety (F01)
- Completeness (F02)
- Promptness (F02)
- Reliability (F03)
- Connectivity (F03)
- Flexibility (F03)
- Security (F03)
- Visibility (F04)
- Discoverability (F04)
- Consistency (F05)
- Appropriateness (F05)
- Familiarity (F05)
- Readability (F06)
- Relevance (F06)
- Accessibility (F06)
- Trustworthy (F06)
- Navigability (F07)
- Complexity (F07)
- Linkage (F07)
- Understandability (F08)
- Conciseness (F08)
- Structuredness (F08)
- Formality (F08)
- Effectiveness (F09)
- Accuracy (F10)
- Customisation (F10)
- Operability (F11)
- Efficiency (F12)

Table 5. Components of the interface features.

- Action
- Content
- Menus
- Layout
- Steppers
- Media
- Action bar
- Activity bar / circle
- Button
- Color
- Default
- Dialogue
- Expansion panels
- Gestures
- Grid list
- Indicator
- Icons
- Layout
- List
- Navigation drawer
- Picker
- Progress bar
- Slider sub-headers
- Sub-screen
- Spinner
- Snackbars
- Switches
- System bar
- Tabs
- Text fields
- Typography

Table 6. Partial list of the paired usability checklist.

Usability features	Usability criteria	Interface features	Checklist items
Navigation	Navigability	Action bar	Tabs or spinner in the top bar is used for quick view change
Navigation	Linkage	Action bar	Shortcut to most frequent task is provided
Notification	Completeness	Dialogue	There are at most 3 possible actions in a notification
Notification	Promptness	Dialogue	Notification is not created if it is possible for the app to recover from the error without user action
Presentation	Accessibility	Content	Content is structurally separated from navigational elements
Presentation	Relevance	Action bar	Unavailable action in the current context is hidden instead of disabled
Permission	Flexibility	Snackbars	The app allows to revert accidental activation
Permission	Security	Content	User's data are kept private and safe (encrypted in the event of loss or malfunction)
Signifiers	Visibility	Button	The UI Buttons are visible
Signifiers	Discoverability	Gestures	The user interface gives visual clues if something can be used with Pinch-To-Zoom gesture

5. Evaluating the frameworks usefulness

In our previous work, we validated the comprehensiveness of the framework components among academicians in Malaysia's public universities [32]. The components were refined based on the survey responses. Subsequently, the components were evaluated for their feasibility in real practice among software engineering practitioners in Malaysia and refined once again based on the survey response [33]. In this paper, we conducted an expert review and a semi-structured interview to evaluate the framework's usefulness in comparison to existing usability evaluation frameworks.

Usefulness is characterized in most usability studies as a composition of usability and as is utility [34,35]. Likewise, available usefulness questionnaires (e.g., USE and TAM) measure usefulness in the same dimension. The dimension includes a composition of several usability criteria, such as ease of use, learnability, and satisfaction, in addition to as-is utility. This section demonstrates the framework's evaluation in terms of its usefulness in comparison to the selected study.

The usability evaluation framework to be compared to the one from this study was selected through an exhaustive search of existing work on online databases subscribed by Universiti Putra Malaysia (UPM) and accessed publications. The search was performed using Google Scholar to review the recently proposed checklist-based framework published during the development of the framework in this study. The query returned 424 results in the English language. Any matching results that have been adopted in developing the framework in this study are omitted to avoid bias. Subsequently, publication on the checklist-based framework was filtered for selection. The process ends with two relevant search results. Since the work of Joseph [36] is more about usability heuristics, we have selected the work of Thitichaimongkhol and Senivongse [37] as a comparison against the formulated framework.

Methods and material

Prior to the evaluation, the participants were given a demographic form to record their background experience, the specifications of the smartphone used during the evaluation, such as brand and operating system, and their experience using apps in the dominant category in the marketplace.

Evaluating the entire framework measurement (373 checklists) from this study in comparison with the previous work is inefficient in terms of time and resources. Therefore, the evaluation scope covers usability measurements from both sets that match the

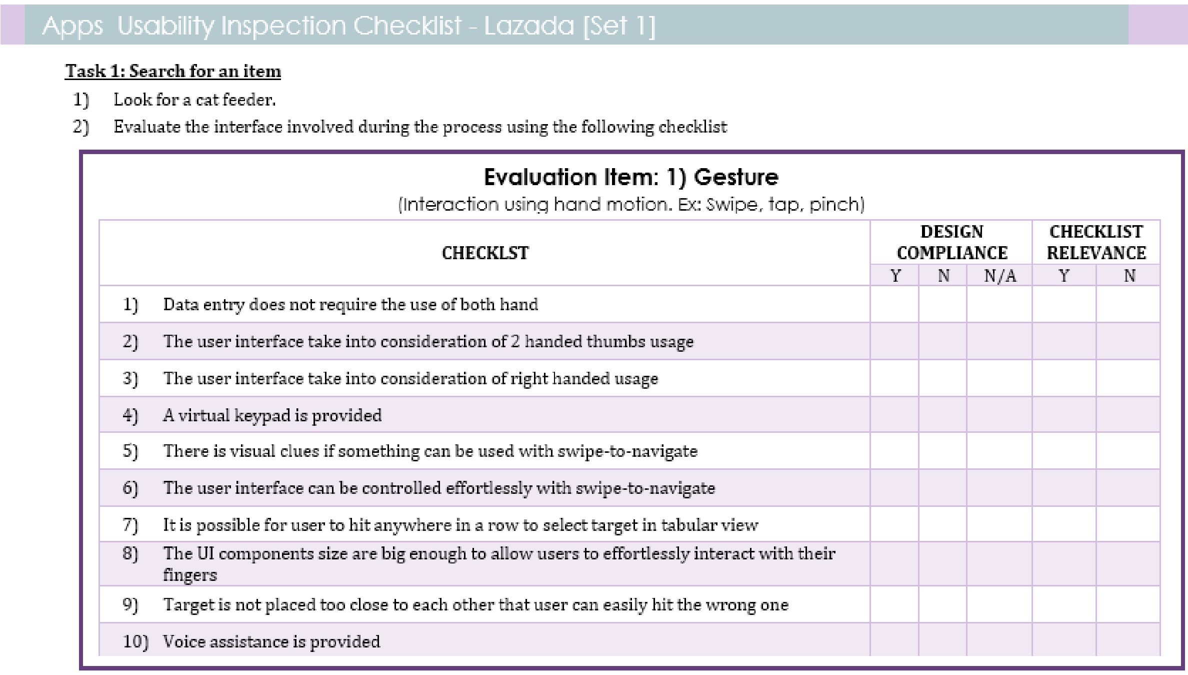
ISO/IEC 25010 product quality model. Although the usability criteria corresponding to both checklist sets for the evaluation have a different name compared to the ISO/IEC 25010 quality criteria, the conceptual definition for the corresponding usability criteria shares the same description as the ISO/IEC 25010 usability criteria.

Three apps from different categories commonly used by Malaysians (from survey responses in our previous study) are selected from the Play Store. Task analysis is performed on the apps to identify the primary task and the interface feature associated with each task. Usability criteria from both studies (this study and the other) corresponding to the interface features associated with the primary task are selected for the heuristic walkthrough. **Figure 3** exhibits an excerpt from the identified checklist from this study, corresponding to the usability criteria from ISO/IEC 25010 and interface features of the primary task for Lazada apps.

Likewise, the checklist for Set 2 is prepared using the same task and interface features of the same apps, corresponding to the same usability criteria as in Set 1. **Figure 4** exhibits an excerpt from the identified checklist from another study, corresponding to the usability criteria from ISO/IEC 25010 and interface features of the primary task for Lazada apps.

The participants were given two sets of checklists (76 items from the formulated framework and 39 items from the other framework), which correspond to the ISO/IEC 25010 usability criteria and interface features of primary tasks from selected apps. The evaluators were required to perform a heuristic walkthrough on three apps (Google+, Viber, and Lazada) using both checklist sets. Subsequently, they are required to review both checklist sets. The checklist sets are given in random order. The first evaluator is given Set 1, followed by Set 2. Meanwhile, the next evaluator is given Set 2, followed by Set 1.

Finally, both frameworks were rated for their usefulness using the USE questionnaires. The evaluators were given two sets of USE questionnaires, one for each framework. The questionnaire includes 30 checklist items on a 7-point Likert scale. The scale ranges from 1 (strongly disagree) to 7 (strongly agree). The resulting USE score was analyzed using paired t-test to determine if there was any difference between the compared frameworks. A post hoc test, Cohen's D, is used to investigate the effect size on the significance of the compared framework. Equation (1) explains Cohen's D measure of effect size.



Apps Usability Inspection Checklist - Lazada [Set 1]

Task 1: Search for an item

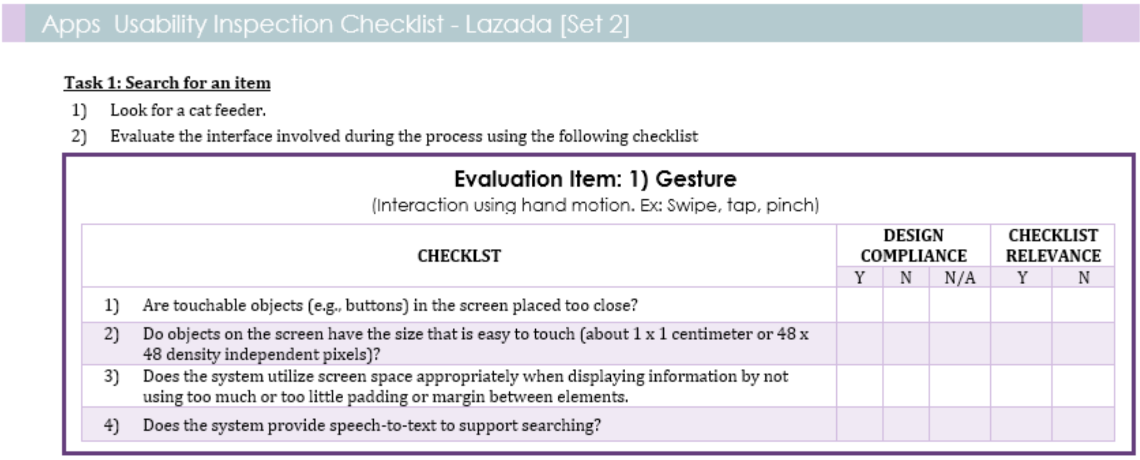
1) Look for a cat feeder.
2) Evaluate the interface involved during the process using the following checklist

Evaluation Item: 1) Gesture

(Interaction using hand motion. Ex: Swipe, tap, pinch)

CHECKLST		DESIGN COMPLIANCE			CHECKLIST RELEVANCE	
		Y	N	N/A	Y	N
1)	Data entry does not require the use of both hand					
2)	The user interface take into consideration of 2 handed thumbs usage					
3)	The user interface take into consideration of right handed usage					
4)	A virtual keypad is provided					
5)	There is visual clues if something can be used with swipe-to-navigate					
6)	The user interface can be controlled effortlessly with swipe-to-navigate					
7)	It is possible for user to hit anywhere in a row to select target in tabular view					
8)	The UI components size are big enough to allow users to effortlessly interact with their fingers					
9)	Target is not placed too close to each other that user can easily hit the wrong one					
10)	Voice assistance is provided					

Figure 3. Checklist for Set 1.



Apps Usability Inspection Checklist - Lazada [Set 2]

Task 1: Search for an item

1) Look for a cat feeder.
2) Evaluate the interface involved during the process using the following checklist

Evaluation Item: 1) Gesture
(Interaction using hand motion. Ex: Swipe, tap, pinch)

CHECKLST		DESIGN COMPLIANCE			CHECKLIST RELEVANCE	
		Y	N	N/A	Y	N
1)	Are touchable objects (e.g., buttons) in the screen placed too close?					
2)	Do objects on the screen have the size that is easy to touch (about 1 x 1 centimeter or 48 x 48 density independent pixels)?					
3)	Does the system utilize screen space appropriately when displaying information by not using too much or too little padding or margin between elements.					
4)	Does the system provide speech-to-text to support searching?					

Figure 4. Checklist for Set 2.

$$\delta = \frac{\overline{X}_2 - \overline{X}_1}{\sqrt{\frac{S_1{}^2 + S_2{}^2}{2}}} \qquad (1)$$

where, $\overline{X}$ = sample mean, S = sample standard deviation.

Effect size is categorized into small, medium, large, and very large impacts [38]. An effect size of 0.2 indicates a small magnitude of the effect. Medium effect size ranges from 0.5 vicinities. Large effect size ranges from 0.8. Meanwhile, a very large effect size is indicated by values larger than, or equal to 1.3.

A semi-structured interview is conducted after the experiment to clarify why the experts rated one framework better than the other. The identity of each treatment, which framework was the one from this study, and which framework was from the previous study were not revealed until the end. The rationale is to have an expert's honest opinion on the framework.

6. Results and discussions

We approached eleven industrial experts, ranging from mobile developers to mobile testers and UX designers. However, five of them repeatedly rescheduled the dateline to complete the evaluation and failed to complete the requested evaluation even after more than three follow-up reminders.

Only six of the experts managed to complete the experiment. A difference in the overall USE scores rated by the six experts for both frameworks was computed. However, one of them gives an unreliable rating, even after reviewing the score given. It is not feasible to set up an upfront meeting with that expert. The expert's background profile showed that this expert is the only participant to select gaming apps as the most frequently used apps. Since gaming apps have different designs and purposes compared to other categories of apps, the expert's perception of usability might skew away from the other five participants, who were not familiar with mobile gaming. Thus, it is reasonable that the expert gave a contradictory score compared to the other participants. Therefore, the response by this expert was excluded from the analysis. The overall USE score collected from each expert is analyzed to evaluate the usefulness of the frameworks. **Table 7** exhibits the mean differences in the overall USE score for both treatments.

A paired t-test is used to determine the significance of the usefulness score with a p-value of 0.05. The distribution of the USE score differences between both groups of experts is normally distributed, thus making it appropriate for conducting a paired t-test. The mean indicates that the five experts (N = 5) gave a larger USE score for the formulated framework (mean = 180.60) compared to the other framework (mean = 156.60). In addition, a smaller standard deviation compared to the other framework indicates that the USE scores among the experts were more consistent in the formulated framework. **Table 8** exhibits the results of the paired t-test.

USE scores for the formulated framework are

Table 7. Overall USE score for both frameworks.

Paired Samples Statistics					
		Mean	N	Std. deviation	Std. error mean
Pair 1	Formulated framework USE score	180.60	5	10.991	4.915
	Previous framework USE score	156.60	5	15.143	6.772

Table 8. Results of the paired t-test.

Paired samples test							
		Paired differences			t	df	Sig. (1-tailed)
		Mean	Std. deviation	Std. error mean			
Pair 1	Formulated framework USE score – Previous framework USE score	24.000	20.273	9.066	2.647	4	.0286

24 points higher (mean paired difference = 24) than USE scores for the previous framework. There is enough evidence to claim that the mean USE score given by the experts for the formulated framework is greater than the previous framework, t (4) = 2.647, p = 0.0286. Thus, the null hypothesis is rejected since the p-value is less than 0.05. **Figure 5** illustrates the position of the calculated t statistic (within the H_0 rejection region), t-value, and p-value in a graph.

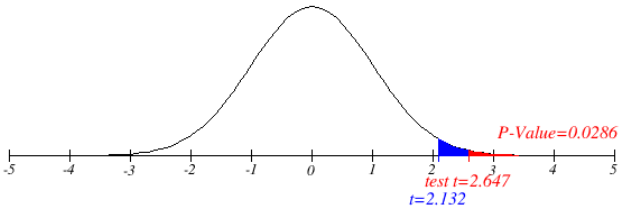


Figure 5. Position of calculated t statistics.

The learning experience gained in conducting the heuristic walkthrough using the checklist facilitates comprehending the framework measurement, thus ensuring reliable scoring of the framework's usefulness. However, although the result presented indicates that the USE scores of both frameworks are unlikely to occur by chance, the magnitude of the effect of the treatment (the formulated framework) over the other framework is unexplained. A post hoc test, Cohen's D, is used to investigate the effect size on the significance of the compared framework. The result of 1.81 indicates a very large effect size, implying a meaningful difference in the USE score between both frameworks.

Discussions

The demographic distribution of the experts indicates that they were the appropriate participants to evaluate the formulated framework measurement from the perspective of developers. Only two of the experts are experienced usability practitioners with seven years or more of experience in the field. Both of them were of different genders and were using different mobile OS. Additionally, all of the experts are of different ages, ranging from the twenties to the thirties and forties.

The formulated framework proved to be more useful than the previously proposed framework. The semi-structured interview revealed that the experts came to a consensus, agreeing that the formulated framework is more useful for usability evaluation compared to the other framework. The reason lies in the fact that the formulated framework measurement is much simpler, UI-oriented, and less ambiguous for experts, both developers and usability practitioners.

Both frameworks were compared using the same baseline measurement: the ISO/IEC 25010 product quality model (usability component) in conjunction with a common interface feature for the primary task in the evaluated apps. Nevertheless, the learning gained from experiencing the framework measurement exhibits the usefulness of each framework.

The formulated framework is criticized for its large number of checklists. However, it is not practical to inspect every available criterion. Usability evaluation is commonly conducted based on an evaluation plan established beforehand, which determines the criteria to be evaluated during an inspection. The formulated framework came in handy due to its features in supporting different backgrounds of usability evaluators through the abstraction level. In fact, restricting usability evaluation to usability criteria of interest will eventually reduce the number of checklists to be used during the usability evaluation.

7. Threats to validity

Threats are inevitable yet manageable in research. In this section, threats to internal, external, conclusion, and construct validity are discussed. The selection of associated usability criteria tied to UI elements was determined by adopting the ISO/IEC 25010 product quality model (usability component) as a benchmark criterion in comparing the formulated framework over previously proposed frameworks. However, the experiment is still vulnerable to the order effect. Thus, in replicating the experiment over the other framework in comparison, two sets of evaluation plans representing the formulated framework and the previously proposed framework were given to the evaluator in random order.

Regarding the external threat, the respondent's expertise and experience in using the evaluated app might affect the validity of the result. The respondents consist of field experts from various branches of software engineering disciplines and app development stages with a different range of years of experience. In addition, they might use their experience of using a particular type of app, e.g., transactional, communication, or games, as a benchmark in scoring UI elements. Altogether, the respondent might perceive usability differently based on their background of expertise and experience with the app, thus affecting their subjective judgement. These threats are controlled through two countermeasures. Firstly, a conceptual definition of the evaluated interface feature was established. There is a possibility that an interface feature is recognized by a different name in academia and industry. For example, a drop-down menu is well-known in desktop computing. On some occasions, it is used as a jump menu. However, in mobile computing, the drop-down menu is recognized as a list. In addition, the jump menu is known as a spinner in mobile computing. Furthermore, elements such as sub-screen and gesture are interface features that are absent in desktop computing and could be misinterpreted differently by individuals. This necessitates a further description of a UI element's operation or behavior in view of desktop computing to facilitate an inexperienced evaluator in this case. Secondly, the experiment is designed as a repeated measure to reduce variability across participants.

The main threat to the conclusion validity of the result is statistical power. This threat is alleviated by applying the most common statistical test, appropriate for the research design of within-subject design. Moreover, the significance level was 5%. Hence, the chance of a Type I error is small.

A checklist from the previous study is used in comparison to the checklist in this study to manage construct validity. The scores of both checklists in measuring the ISO/IEC 25010 product quality model (usability component) were correlated in conjunction with the use of an established questionnaire to measure the framework's usefulness. In addition, a well-established usability questionnaire was carefully selected for this study to measure usefulness appropriately.

8. Conclusions

This study empirically evaluates the usefulness of an integrated usability evaluation framework through an expert review. The framework measurement is reviewed and compared against a framework from another study. Both frameworks were compared based on the ISO/EIC 25010 product quality model (usability component). Hypothesis testing was conducted to investigate the significance and effect size of the response from the expert review. The results of the statistical test proved that the formulated framework had a significant and large effect size and was more useful compared to the other framework. In the future, we plan to improve the effectiveness of this framework by comparing the results of using it in usability testing against usability inspection. The rationale is to alleviate a possible false alarm in the formulated framework measurement and capture the true usability problem. Consequently, an additional checklist could be proposed based on the usability testing result to complement the developed usability measurement.

Author Contributions

H. R. conceived the idea and study of proposing a usability evaluation framework for mobile apps that incorporates the usability criteria and interface features in conjunction with different evaluator viewpoints into a framework abstraction level. H. Z., A. K. and the late A. A. A. A. served as H.R.'s supervisor and co-supervisors on her Ph.D. thesis at the Universiti Putra Malaysia. All authors reviewed and approved the final manuscript.

Conflict of Interest

There is no conflict of interest.

Funding

This research was partially funded by the Research University Grant Scheme (RUGS), Universiti Putra Malaysia (UPM).

References

[1] Moumane, K., Idri, A., Abran, A., 2016. Usability evaluation of mobile applications using ISO 9241 and ISO 25062 standards. SpringerPlus. 5, 1-15.

[2] Hussain, A., Abubakar, H.I., Hashim, N.B. (editors), 2014. Evaluating mobile banking application: Usability dimensions and measurements. Proceedings of the 6th international Conference on Information Technology and Multimedia; 2014 Nov 18-20; Putrajaya, Malaysia. New York: IEEE. p. 136-140.

[3] Hussain, A., Kutar, M. (editors), 2009. Usability metric for mobile application. 2008 Proceedings of the 10th International Conference on Information Integration and Web-Based Applications and Services (IiWAS'08); 2008 Nov 24-26; Linz, Austria. New York: Association for Computing Machinery. p. 567-570.

[4] Liu, N., Yu, R., 2017. Identifying design feature factors critical to acceptance and usage behavior of smartphones. Computers in Human Behavior. 70, 131-142.

[5] Harrison, R., Flood, D., Duce, D., 2013. Usability of mobile applications: literature review and rationale for a new usability model. Journal of Interaction Science. 1, 1-16.

[6] Yáñez Gómez, R., Cascado Caballero, D., Sevillano, J.L., 2014. Heuristic evaluation on mobile interfaces: A new checklist. The Scientific World Journal. 1-19.

[7] Nayebi, F., 2015. iOS application user rating prediction using usability evaluation and machine learning [Ph.D. thesis]. Quebec: University of Quebec.

[8] Saleh, A., Ismail, R., Fabil, N. (editors), 2017. Evaluating usability for mobile application: A MAUEM approach. ICSEB 2017 Proceedings of the 2017 International Conference on Software and E-Business; 2017 Dec 28-30; Hong Kong. New York: Association for Computing Machinery. p. 71-77.

[9] Dubey, S.K., Gulati, A., Rana, A., 2012. Integrated model for software usability. International Journal on Computer Science and Engineering. 4(3), 429.

[10] Elsantil, Y., 2020. User perceptions of the security of mobile applications. International Journal of E-Services and Mobile Applications (IJESMA). 12(4), 24-41.
DOI: https://doi.org/10.4018/IJESMA.2020100102

[11] Malatini, S., Bogliolo, A. (editors), 2015. Gamification in mobile applications usability evaluation: A New Approach. MobileHCI'15: Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services; 2015 Aug 24-27; Copenhagen, Denmark. New York: Association for Computing Machinery. p. 897-899.

[12] Hoehle, H., Aljafari, R., Venkatesh, V., 2016.

Leveraging Microsoft's mobile usability guidelines: Conceptualizing and developing scales for mobile application usability. International Journal of Human-Computer Studies. 89, 35-53.

[13] Mugisha, A., Nankabirwa, V., Tylleskär, T., et al., 2019. A usability design checklist for Mobile electronic data capturing forms: the validation process. BMC Medical Informatics and Decision Making. 19(1), 1-11.
DOI: https://doi.org/10.1186/S12911-018-0718-3

[14] Xu, H., Jonsson, M., 2012. Tablet application GUI usability checklist—Creation of a user interface usability checklist for tablet applications [Master's thesis]. Huddinge: Södertörns University College.

[15] Lachgar, M., Abdelmounaim, A., 2017. Decision framework for mobile development methods. International Journal of Advanced Computer Science and Applications. 8(2), 110-118.
DOI: https://doi.org/10.14569/IJACSA.2017.080215

[16] ISO 9241-11:1998 Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)—Part 11: Guidance on Usability [Internet]. International Organization for Standardization; 1998 [cited 2018 Dec 21]. Available from: https://www.iso.org/standard/16883.html

[17] ISO/IEC 9126-1:2001 Software Engineering—Product Quality—Part 1: Quality Model [Internet]. International Standard for Standardization; 2001 [cited 2018 Dec 21]. Available from: https://www.iso.org/standard/22749.html

[18] Nielsen, J., Budiu, R., 2012. Mobile usability. New Riders Press: Berkeley CA.

[19] Constantine, L.L., Lockwood, L.A., 1999. Software for use: A practical guide to the models and methods of usage-centered design. Addison-Wesley Publishing Co.: Boston.

[20] Böckle, M., Rühmkorf, J. (editors), 2019. Towards a framework for the classification of usability issues. Human-Computer Interaction-INTERACT 2019: 17th IFIP TC 13 International Conference; 2019 Sep 2-6; Paphos, Cyprus. p. 610-614.

[21] Kronbauer, A.H., Santos, C.A., Vieira, V. (editors), 2012. Smartphone applications usability evaluation: A hybrid model and its implementation. Human-Centered Software Engineering: 4th International Conference, HCSE 2012; 2012 Oct 29-31; Toulouse, France. p. 146-163.

[22] Olsina, L., Santos, L., Lew, P., 2014. Evaluating mobileapp usability: A holistic quality approach. Lecture notes in computer science. Springer, Cham.: New York. pp. 111-129.

[23] Fabil, N.B., Saleh, A., Isamil, R.B., 2015. Extension of pacmad model for usability evaluation metrics using goal question metrics (Gqm) approach. Journal of Theoretical and Applied Information Technology. 79(1), 90-100.

[24] Inostroza, R., Rusu, C., Roncagliolo, S., et al. (editors), 2012. Usability heuristics for touchscreen-based mobile devices. 2012 Ninth International Conference on Information Technology—New Generations; 2012 Apr 16-18; Las Vegas, NV, USA. New York: IEEE.

[25] Mi, N., Cavuoto, L.A., Benson, K., et al., 2014. A heuristic checklist for an accessible smartphone interface design. Universal Access in the Information Society. 13, 351-365.

[26] Soomro, S., Ahmad, W.F.W., Sulaiman, S. (editors), 2012. A preliminary study on heuristics for mobile games. 2012 International Conference on Computer and Information Science; 2012 Jun 12-14; Kuala Lumpur, Malaysia. New York: IEEE. p. 1030-1035.

[27] Zahra, F., Hussain, A., Mohd, H. (editors), 2017. Usability evaluation of mobile applications; Where do we stand? The 2nd International Conference on Applied Science and Technology 2017 (ICAST'17); 2017 Apr 3-5; Kedah, Malaysia.

[28] Zamfiroiu, A., 2014. Factors influencing the quality of mobile applications. Informatica Economica. 18(1), 131.

[29] Homann, M., Wittges, H., Krcmar, H., 2013. Towards user interface patterns for ERP applications on smartphones. Business Information Systems. 157, 14-25.

[30] Roder, H. (editor), 2012. Specifying usability features with patterns and templates. 2012 First

International Workshop on Usability and Accessibility Focused Requirements Engineering (UsARE); 2012 Jun 28; Zurich, Switzerland. New York: IEEE. p. 6-11.

[31] Punchoojit, L., Hongwarittorrn, N., 2017. Usability studies on mobile user interface design patterns: A systematic literature review. Advances in Human-Computer Interaction. 1-22.
DOI: https://doi.org/10.1155/2017/6787504

[32] Rahmat, H., Zulzalil, H., Ghani, A.A.A., et al., 2018. A comprehensive usability model for evaluating smartphone apps. Advanced Science Letters. 24(3), 1633-1637.

[33] Zulzalil, H., Rahmat, H., Ghani, A.A.A., et al., 2019. Conceptualising mobile apps usability dimension: A feasibility assessment of Malaysian industrial practitioners. International Journal of Engineering and Advanced Technology. 9(1), 1708-1713.

[34] Tarkkanen, K., Harkke, V., Reijonen, P., 2015. Are we testing utility? Analysis of usability problem types. Design, user experience, and usability: Design discourse. Springer: Cham. pp. 269-280.
DOI: https://doi.org/10.1007/978-3-319-20886-2_26

[35] MacDonald, C.M., Atwood, M.E. (editors), 2014. What does it mean for a system to be useful? Proceedings of the 2014 Conference on Designing Interactive Systems; 2014 Jun 21-25; Vancouver BC, Canada. New York: Association for Computing Machinery. p. 885-894.

[36] Joseph, V., 2017. User experience guidelines for improving retention rate in mobile apps [Master's thesis]. Madrid: Universidad Politécnica de Madrid.

[37] Thitichaimongkhol, K., Senivongse, T., 2016. Enhancing usability heuristics for android applications on mobile devices. Proceedings of the World Congress on Engineering and Computer Science. 1, 19-21.

[38] Sullivan, G.M., Feinn, R., 2012. Using effect size—or why the P value is not enough. Journal of Graduate Medical Education. 4(3), 279-282.

BILINGUAL
PUBLISHING
GROUP

Tel:+65 65881289
E-mail: contact@bilpublishing.com
Website:https://journals.bilpubgroup.com

2630-5151 03