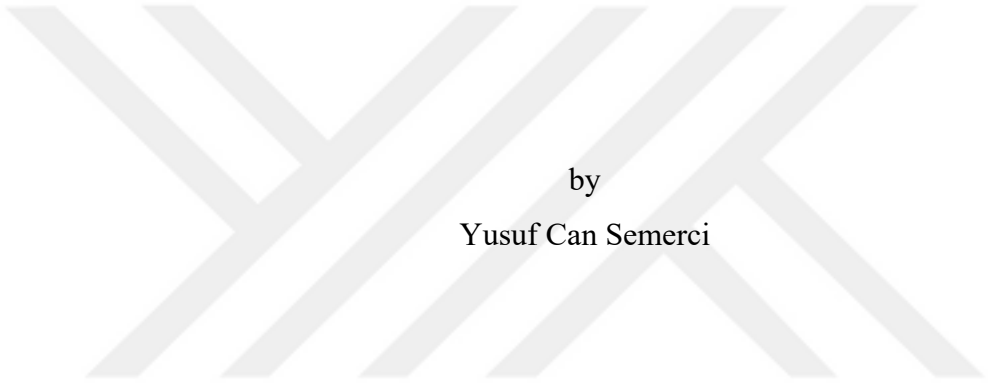


STUDENT LEARNING ANALYSIS IN E-LEARNING USING DEEP NEURAL
NETWORKS



by
Yusuf Can Semerci

Submitted to Graduate School of Natural and Applied Sciences
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy in
Computer Engineering

Yeditepe University
2020

STUDENT LEARNING ANALYSIS IN E-LEARNING USING DEEP NEURAL
NETWORKS

APPROVED BY:

Assist. Prof. Dr. Dionysis Goularas
(Thesis Supervisor)
(Yeditepe University)

Prof. Dr. Duygun Erol Barkana
(Yeditepe University)

Prof. Dr. Emin Erkan Korkmaz
(Yeditepe University)

Prof. Dr. Semih Bilgen
(Istanbul Okan University)

Assist. Prof. Dr. Kerem Altun
(Izmir Institute of Technology)

DATE OF APPROVAL:/...../2020

ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to my supervisor Assist. Prof. Dr. Dionysis Goularas. My transformation from a clueless student to a Ph.D. candidate would not be possible without his guidance and support. I learned a lot from him both in academia and in life. Thank you for encouraging me when I felt down and frustrated. I will remember my years in Ph.D. studies as fun times thanks to you.

This study is partially supported by the Erasmus+ Programme of the European Union [2017-1- TR01-KA203- 046818]. I would like to thank all the members of the project.

I would also like to thank Assoc. Prof. Dr. Gökçe Bayındır Goularas for her help in organizing a course with the students from Political Science and International Relations department. My thesis would lack a crucial experimental study if it was not for her help.

To my mother Yaşar Semerci and my father Selahattin Semerci, your unconditional love and support made my life easier in every way. You listened, comforted, and advised me throughout my life. You are the reason I am where I am today. To my brother Cem Semerci, you are the best brother anyone could ask for.

Most importantly, this thesis could not be completed if it wasn't for my wife Filiz Tezcan Semerci. This thesis is yours as much as mine. You encouraged me to start working on it when I felt unmotivated. You helped me to clear my head after spending hours working on this thesis. You read every sentence I wrote in every version of the thesis just to help me fix my mistakes. You gave me the final push whenever I needed it. During your Ph.D., I will be everything you were to me throughout this thesis.

ABSTRACT

STUDENT LEARNING ANALYSIS IN E-LEARNING USING DEEP NEURAL NETWORKS

In a period where demand for web-based education systems is rapidly increasing, estimating whether a student is learning or not and how a course can be adapted to the needs of its students becoming crucial. Flow theory provides an explanation for how the emotional state of an individual can have an impact on their learning. Hence, in the scope of e-learning, estimating the flow states of students can provide useful information that will lead to the estimation of their learning. The challenge in flow state estimation in e-learning platforms is to extract parameters that reflect the effort, the activity, and the performance of students. In this scope, this thesis proposes a deep neural network and flow theory-based method aiming to estimate student learning in a course taught with e-learning. First, the activities of the students are analyzed using the interaction of students with an e-learning platform that comprises classical e-learning pages and a timeline tool. This analysis contains activity heatmaps and deep neural network techniques. Then, the performances of the students are obtained through quizzes presented with the e-learning interface. The activities and performances are then validated with a statistical analysis using student surveys. Using the activities and the performances the method generates an estimation about the flow state of students in a course. The proposed deep neural network model acts as a dimension reduction method that extracts patterns from the activity heatmaps. The student profile biases are eliminated by conducting experiments in two different courses from two different disciplines. The proposed model presents superior performance over the dimension reduction methods used in the literature in terms of correlation with the surveys.

ÖZET

DERİN SINIR AĞLARI İLE E-ÖĞRENMEDE ÖĞRENCİ ÖĞRENME ANALİZİ

Web tabanlı eğitim sistemlerine olan talebin hızla arttığı bir dönemde, bir öğrencinin öğrenip öğrenmediğini ve bir dersin öğrencilerinin ihtiyaçlarına göre nasıl uyarlanabileceğini tahmin etmek çok önemli hale geliyor. Akış teorisi, bir bireyin duygusal durumunun öğrenmeleri üzerinde nasıl bir etkiye sahip olabileceğine dair bir açıklama sağlar. Bu nedenle, e-öğrenme kapsamında, öğrencilerin akış durumlarının tahmin edilmesi, öğrenmeleri hakkında bir tahmine yol açacak faydalı bilgiler sağlayabilir. E-öğrenme platformlarında akış durumu tahminindeki zorluk, öğrencilerin çabalarını, aktivitelerini ve performansını yansıtan parametreleri çıkarmaktır. Bu kapsamda bu tez, e-öğrenme ile öğretilen bir derste öğrenci öğrenmesini tahmin etmeyi amaçlayan derin bir sinir ağı ve akış teorisi temelli bir yöntem önermektedir. İlk olarak, klasik e-öğrenme sayfaları ve bir zaman çizelgesi aracından oluşan bir e-öğrenme platformu ile öğrencilerin etkileşimi kullanılarak öğrencilerin etkinlikleri analiz edilir. Bu analiz, aktivite ısı haritalarını ve derin sinir ağı tekniklerini içerir. Daha sonra e-öğrenme arayüzü ile sunulan quizler ile öğrencilerin performansları elde edilir. Etkinlikler ve performanslar daha sonra öğrenci anketleri kullanılarak istatistiksel bir analizle doğrulanır. Yöntem, etkinlikleri ve performansları kullanarak bir derste öğrencilerin akış durumu hakkında bir tahmin üretir. Önerilen derin sinir ağı modeli, etkinlik ısı haritalarından kalıpları çıkaran bir boyut azaltma yöntemi olarak işlev görür. Öğrenci profili yanlılıkları, iki farklı disiplinden iki farklı derste deneyler yapılarak ortadan kaldırılmaktadır. Önerilen model, anketlerle korelasyon açısından literatürde kullanılan boyut küçültme yöntemlerine göre üstün performans göstermektedir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	iii
ABSTRACT	iv
ÖZET	v
LIST OF FIGURES	ix
LIST OF TABLES.....	xiii
LIST OF SYMBOLS/ABBREVIATIONS.....	xiv
1. INTRODUCTION.....	1
2. THEORETICAL BACKGROUND	9
2.1. FLOW THEORY	9
2.2. HEATMAPS	11
2.2.1. Grayscale Coloring Scheme.....	11
2.2.2. Rainbow Coloring Scheme	12
2.3. DIMENSION REDUCTION METHODS.....	12
2.3.1. Principal Component Analysis	13
2.3.2. Factor Analysis	14
2.3.3. Multidimensional Scaling.....	14
2.3.4. Independent Component Analysis.....	15
2.3.5. Gaussian Random Projection.....	16
2.3.6. Isometric Feature Mapping.....	16
2.3.7. t-Distributed Stochastic Neighbor Embedding.....	17
2.4. COMPUTATIONAL FRAMEWORK	18
2.4.1. Convolution	18
2.4.2. Pooling and Up-Sampling.....	20
2.4.3. Machine Learning Techniques.....	21
2.5. STATISTICAL ANALYSIS.....	28
2.5.1. Normality Test	28
2.5.2. Correlation Analysis	29
3. METHODOLOGY	31

3.1.	DATA COLLECTION	39
3.1.1.	Activity	39
3.1.2.	Performance	45
3.2.	DATA PROCESSING	45
3.2.1.	Activity	45
3.2.2.	Performance	56
3.3.	THE E-LEARNING ENVIRONMENT	56
3.3.1.	Timeline Tool	58
3.3.2.	E-learning Web Pages.....	60
3.3.3.	Data Collection Tool.....	61
3.4.	DATASET PREPARATION	62
3.4.1.	Activity Vector Generation using Interaction Statistics	63
3.4.2.	Activity Heatmaps Generation.....	65
3.4.3.	Activity Vector Generation using Activity Heatmaps.....	67
3.5.	COURSES.....	72
3.6.	SURVEY.....	72
4.	IMPLEMENTATION	73
4.1.	THE E-LEARNING ENVIRONMENT	75
4.1.1.	Timeline	76
4.1.2.	Map	77
4.1.3.	Infobox.....	78
4.1.4.	Editor Panel	79
4.2.	DATA COLLECTION	80
4.3.	DATASET PREPARATION	82
4.4.	COURSES.....	83
5.	RESULTS.....	87
5.1.	PARTICIPANTS	88
5.2.	CONVOLUTIONAL AUTOENCODER MODEL SELECTION	89
5.3.	PAIRWISE CORRELATION ANALYSIS OF COURSE 1: HISTORY OF PEOPLE IN COMPUTATION.....	91
5.4.	PAIRWISE CORRELATION ANALYSIS OF COURSE 2: REGIONAL STUDIES	93

5.5. FLOW STATE DIAGRAMS OF THE TWO COURSES.....	94
5.6. DISCUSSION	96
6. LIMITATION	100
7. CONCLUSION	101
REFERENCES	103
APPENDIX A.....	112
APPENDIX B	113
APPENDIX C	114



LIST OF FIGURES

Figure 1.1. The general overview of the proposed approach.....	7
Figure 2.1. The flow state diagram proposed by Csikszentmihalyi.....	10
Figure 2.2. A grayscale heatmap	11
Figure 2.3. A heatmap using rainbow color scheme.....	12
Figure 2.4. A padding example (a) Original 5×5 matrix (b) Matrix with 1 padding.....	19
Figure 2.5. A pooling example with kernel size of 2×2 and stride=2 (a) Original 4×4 matrix (b) Maximum Pooling (c) Average Pooling.....	20
Figure 2.6. An up-sampling example with kernel size of 2×2 (a) Original 2×2 matrix (b) Uniform Sampling (c) Zero Sampling.....	21
Figure 2.7. An Artificial Neural Network architecture.....	23
Figure 2.8. A Convolutional Neural Network architecture	24
Figure 2.9. An Autoencoder architecture.....	26
Figure 2.10. A Convolutional Autoencoder architecture.....	27
Figure 3.1. The conceptual relationship between challenge difficulty and <i>Activity</i>	32
Figure 3.2. The conceptual relationship between skill and <i>Performance</i>	33
Figure 3.3. Proposed concept relations.....	34

Figure 3.4. The relation between emotional states, measurement concepts, and constructs of flow theory	35
Figure 3.5. Relation between concepts and Expert.....	36
Figure 3.6. General overview of the research method.....	39
Figure 3.7. Convolutional Autoencoder model architecture for colored heatmaps.....	48
Figure 3.8. Convolutional Autoencoder model architecture for grayscale and weighted scale heatmaps	50
Figure 3.9. Activity value generation through interaction statistics	53
Figure 3.10. Heatmap generation procedure.....	54
Figure 3.11. Activity value generation using the concatenation procedure	55
Figure 3.12. Activity value generation using total heatmaps	56
Figure 3.13. Timeline Tool interface for Students.....	58
Figure 3.14. Timeline Tool interface for Teachers.....	59
Figure 3.15. E-learning Web Pages interface for Students.....	60
Figure 3.16. E-learning Web Pages interface for Teachers	60
Figure 3.17. Entity-relationship diagram of the e-learning platform.....	61
Figure 3.18. Entity-relationship diagram of the raw statistics collection tool.....	62
Figure 3.19. The flowchart of Activity dataset preparation procedure.....	63

Figure 4.1. Overview of the software structure	74
Figure 4.2. Presentation of a course in E-learning Web Pages.....	75
Figure 4.3. An example timeline which presents a map, a timeline, and an infobox	75
Figure 4.4. An example timeline with different intervals a) 10-year b) 50-year c) 100-year	76
Figure 4.5. Examples of map views a) Road b) Satellite c) Terrain.....	77
Figure 4.6. An example infobox that contains information about Alan Turing	78
Figure 4.7. The content creation page of the editor panel	79
Figure 4.8. The timeline management page of the editor panel	80
Figure 4.9. Example entries in Click table of Open Web Analytics.....	81
Figure 4.10. Example entries in Stream table of Open Web Analytics.....	81
Figure 4.11. The content of an example log file.....	82
Figure 4.12. Example activity heatmaps generated from a student's interactions a) Color b) Grayscale c) Weighted scale.....	83
Figure 4.13. Example heatmaps generated by the proposed convolutional autoencoder model in different stages a) Original b) First Epoch c) Tenth Epoch d) Final Prediction...	83
Figure 4.14. History of People in Computation timeline.....	85
Figure 4.15. Political figures in the Middle East timeline.....	85

Figure 5.1. Flow state diagrams for a) Course 1: History of People in Computation b)
Course 2: Regional Studies..... 95



LIST OF TABLES

Table 3.1. Flow state validation techniques used in the literature.....	37
Table 3.2. Measurements calculated based on interaction statistics.....	40
Table 3.3. The configurations for Convolutional Autoencoder model selection.....	49
Table 3.4. Number of reduced dimensions chosen for each method.....	51
Table 3.5. The actions that can be performed by the actors of the platform	57
Table 3.6. Survey questions on a 5-point Likert scale.....	72
Table 4.1. Server configuration	73
Table 4.2. Course details.....	84
Table 5.1. Participants in the e-learning platform courses	88
Table 5.2. The loss values obtained for different types of heatmaps in CAE model selection	90
Table 5.3. Evaluation of the analyses for the History of People in Computing course.....	91
Table 5.4. Evaluation of the analyses for the Regional Studies course.....	93
Table 5.5. Average student survey responses in 5-point likert scale.....	96

LIST OF SYMBOLS/ABBREVIATIONS

AE	Autoencoder
ANN	Artificial neural network
CAE	Convolutional autoencoder
CMS	Content management system
CNN	Convolutional neural network
EEG	Electroencephalogram
FA	Factor analysis
HSV	Hue-saturation-value
ICA	Independent component analysis
Isomap	Isometric feature mapping
MC	Number of mouse clicks
MDS	Multidimensional scaling
MM	Mouse movement paths
PCA	Principal component analysis
ReLU	Rectified linear unit
RGB	Red-green-blue
RP	Random projection
SGD	Stochastic gradient descent
t-SNE	T-distributed stochastic neighbor embedding
TA	Time spent actively
TT	Total spent time

1. INTRODUCTION

The e-learning systems provide computer-enhanced educational environments where the combination of software and hardware usage facilitates learning. The number of e-learning systems is rapidly increasing due to students' increased access to computer systems and the convenience of distance learning. The distance learning systems offer an environment where the course materials can be accessed without any time and place restrictions and they provide content for a large selection of fields. These systems are used by a large number of people simultaneously whose profiles can be different from each other. An emerging challenge arising from e-learning systems that provides education to different student profiles is the design of a system that can interpret the behaviors of its users in order to distinguish between different student profiles and adapt its content according to the needs of these profiles. Hence, in e-learning systems, the generation of student models based on user behaviors is especially important in order to improve the educational impact.

Student modeling is one of the main topics of Learning Analytics. Although there is not a universally accepted definition of Learning Analytics, the most popular definition is given by the field's one of the most known researchers, George Siemens. He defines Learning Analytics as "the measurement, collection, analysis, and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs." [1]. This description gives the main definition of the field but the researchers tend to define Learning Analytics specifically for the area that the term will be used. Van Barneveld et al. gave the definitions of the term in a wide range of fields [2]. In the case of this thesis and the field of education, the above definition can be expressed in terms of understanding the students and their behaviors through their interactions in a learning environment and taking actions based on the analysis of these behaviors.

Learning Analytics has been the subject of a lot of disciplines but the major studies can be categorized into five fields: Social Network Analysis, User Modelling, Cognitive Modelling, Data Mining, and E-Learning [3]. User Modelling studies [4, 5] contributed to the field by trying to understand how users interact with a computerized system and provided the concept of treating each user individually. Cognitive Modelling [6, 7] asked

the question “How learners develop knowledge?” and trying to answer this question, led to an increase in the development of intelligent tutor systems. Intelligent tutor systems [8] has three levels of goals in terms of intelligence: domain knowledge, learner knowledge evaluation, and pedagogical intervention. Learner Knowledge evaluation and pedagogical intervention have become the subject of E-learning systems. E-learning platforms [9, 10] have contributed to the advancements in Learning Analytics studies because they can provide data about students in both performance and their interaction patterns.

According to John Self, a student model should be able to analyze the performance of students, identify their prior and acquired knowledge, and describe their personality characteristics [11]. Therefore, evaluating the behaviors of students and their performances in an e-learning system to obtain their learning states is critical in student modeling. The learning state identification was studied by educational scientists for years and one of the most widely used theories is the Flow Theory.

Flow theory was proposed by Mihaly Csikszentmihalyi in 1975 and being in “flow” was defined as a feeling experienced during an activity in which an individual is completely engaged and has high levels of focus, enjoyment, and fulfillment [12]. Csikszentmihalyi suggests that learning occurs when an individual is in the flow state and flow state is achieved when there is a balance between the difficulty of the presented challenge and the skill of the individual.

dos Santos et al. published a review on the usage of flow theory of Csikszentmihalyi in computerized education systems [13]. According to their analyses, most of the studies that use flow theory present game-based techniques or analyze an educational game. In these studies, the flow state identification is performed using surveys [14, 15, 16, 17], user data logs [18, 19], interviews [19, 20] or face video recordings [21]. The goals of these studies include investigating the correlation between the subjective flow measures, investigating the correlations between the components of the flow measurements, and utilizing flow-based rules to alter the difficulty of the activity or to alter the content of the proposed activity. The researchers also state that there is a lack of computer-based learning activity design in the flow theory-based studies.

Recent studies in computerized education systems perform flow state identification with surveys, face video recordings, and emotion recognition based on physiological signals. Erhel and Jamet investigated the effects of flow experience in educational computer games where they used self-report questionnaires to measure flow experience [22]. In a gamified learning environment, Özhan and Kocadere found relations between flow, engagement, and motivation [23]. They used a flow experience scale to identify the flow state in students. Ha and Im used questionnaires to measure control, focus, curiosity, and interest to study the connections between flow experience and the use of interactive online learning tools in e-learning environments [24]. Bitrian et al. explored the connection between learning outcomes and flow through challenge and skill balance in educational business simulation games [25]. The flow experience of each student is obtained with surveys. Rodríguez-Ardura and Meseguer-Artola investigated the connection between gender, academic performance, and flow experience where they used surveys for flow identification [26]. Verma et al. used facial expressions of the students in educational video games to detect their flow-related emotional states which are validated with self-reports [27]. Standen et al. proposed an adaptive learning system where educational activities are personalized according to the affective states of the learners [28]. The study used facial expressions and physiological signals to detect engagement, frustration, and boredom.

A recent study attempts to detect students' flow state through Electroencephalogram (EEG) [29]. The authors proposed to investigate the presence of EEG indicators in identifying real-time flow states. Their experiment was conducted with 30 elementary school students using a computerized test which contains 30 questions. Before the experiments, students were presented with a survey to identify their skill levels prior to the presented challenge. The authors propose that the computerized test indicates the challenge difficulty and the survey responses present the skill level of the students. The authors categorized the students based on their responses to the test and the survey in a 2-by-2 design. These categories are obtained through labeling the students with high-low skills or perceived challenge. Hence, the labels of the balance of challenge-skill measurement the authors propose are High-Challenge/High-Skill, High-Challenge/Low-Skill, Low-Challenge/High-Skill, and Low-Challenge/Low-Skill.

The EEG data were collected using NeuroSky EEG headsets in order to measure "Attention" and "Engagement" measurements. The attention was obtained through the

software provided by NeuroSky which indicates attentive students with high values and distracted students with low values. On the other hand, the engagement was obtained by converting the EEG data into alpha, beta, and theta frequency bands. For all students, the engagement value was calculated with $\text{beta}/(\text{alpha} + \text{theta})$ where high values indicate high engagement and low values present low engagement. Then, students were labeled as experiencing flow if they have high attention and high engagement, otherwise, they are labeled as not in flow.

The authors found that the flow states generated from Attention and Engagement measurements which are obtained using EEG data show a significant correlation with the balance of challenge and skill measurement. Although the findings are promising, the application of the approach is not practical in an e-learning environment that provides materials for distance learning because it relies on the use of expensive EEG equipment for each student.

In another study, Georgiou and Demiris propose to use flow theory for user modeling in adaptive car racing simulators [30]. The authors propose a method that is based on flow theory to identify the emotional states of learners. The aim of the study is to utilize an adaptive simulator that can react to the emotional states of its users to improve their driving skills. To achieve this goal, the authors suggest three measurements: Exploration, Experience, and Attention. Exploration is defined as the variation of a user's actions by changing their behaviors in the simulation and can be an indication of the difficulty of the challenge presented. Experience presents the skill level of the user that can be generated through their performance in the simulation. Finally, Attention is the measurement of the engagement of the individual throughout a session of a racing simulation.

The authors collected behavioral, physiological, and performance data through a racing simulator called "rFactor" from 52 subjects. The behavioral information includes steering, brake, and throttle times, time of collisions, and speed, while physiological information consists of eye-gaze and user presence collected from eye-trackers and information collected through cameras. The performance is collected from the time to finish a lap in the simulated racing environment and the paths taken for each lap. The subjects were asked to evaluate their skills both prior to and after the data collection sessions and their behaviors in each lap after the sessions have ended.

The raw behavioral data are converted to low-level primitives to be used in Exploration metric calculations. These low-level primitives include the number of collisions, average brake, throttle and steering, and position-speed pairs. The performance data collected from each subject presents their skill levels for each trial. The authors call this metric the Experience and use it to indicate the skill of each individual. Furthermore, the physiological information is also converted into low-level primitives such as head pose, eye fixations, and the number of blinks. These primitives are used to calculate the Attention of each individual.

In order to calculate single values for each subject, the authors suggest the concept of an Expert which can be defined as the individual who finishes the racing challenge in minimum time with minimum actions. The single values are calculated using the average Euclidean distances between primitive values of each individual with an Expert's values. The authors use these Exploration and Experience values to draw flow state diagrams where Exploration is used instead of challenge difficulty and Experience is used to indicate the skill of an individual.

The Experience and Exploration values obtained from the proposed methods present significant correlations with the subjective reports of skill and exploration. The alteration method is based on the emotional states obtained through the flow state diagrams and the achieved improvements show significant correlations with the self-reports where the subjects evaluate their improvements after the challenges were altered after each lap. Although the correlation analysis presents a satisfying indication of using flow theory for adaptive systems, the proposed primitives can be considered as a limitation in transferring the method to other domains. This limitation occurs due to the method's need for specific information in the racing simulator domain.

In summary, the literature lacks objective, transferable, generalizable, and state of the art methods for learning state identification. To the best of my knowledge, a learning state identification method that provides objective measurements for e-learning systems which are based on information that can be transferred to other domains, present similar evaluations for different student profiles, and can be extracted with state of the art methods does not exist. Although the previous studies provide methods that satisfy one of these aspects, they usually focus on game-based activities or educational games rather than e-learning environments. Most of these studies use self-reports to identify learning states

using flow theory rather than focusing on objectivity. The studies that provide measurements without relying on subjective reports lose the transferability of their methods due to the use of information that can not be collected in other domains or modalities that can not be expected from every user to have access to. Furthermore, the experiments in these studies are generally conducted with subjects in similar profiles, or the experiments are designed to eliminate the different profile types.

According to the lack in the literature, there are several aspects that a new approach needs to consider in order to fill the gap. First of all, the new approach needs to propose a valid method that would be able to express the way a student faces the challenges in an online learning environment. This expression needs to be able to extract meaning from the student's behavioral patterns generated through interactions with the environment without depending on subjective measurements. Secondly, the proposed approach should be able to present consistent evaluations for different student profiles. This suggests that the method should be able to be utilized in different course materials for different disciplines. Finally, the approach should show promise in its ability to be used in other learning environments without any need for additional external equipment or without relying on any domain-specific information.

In summary, the research conducted for the learning state identification should ask the following questions:

- Is there a way to extract meaning for learning state evaluation from behavioral interactions of a student in e-learning?
- If a method is able to analyze interaction patterns, to what extent it can be generalized and applied to different student profiles?
- Can this method be transferred to other e-learning environments?

To tackle these questions, this thesis proposes a learning state identification approach that is based on flow theory, activity heatmaps, and deep learning techniques. The proposed environment comprises a classical e-learning platform that is accompanied by an interactive tool. It provides a medium where students interact with the content of a lecture through mouse actions only. The system works as follows: the interaction patterns generated by the students through mouse actions are collected and represented as activity heatmaps. The proposed deep learning model provides the mechanism that reduces the

dimensions of the activity heatmaps in order to extract interaction patterns. After students explore the content, they are presented with exams based on the content of the lectures to collect their grades. The learning state identification is handled by the flow theory mechanisms using these patterns and the corresponding grades.

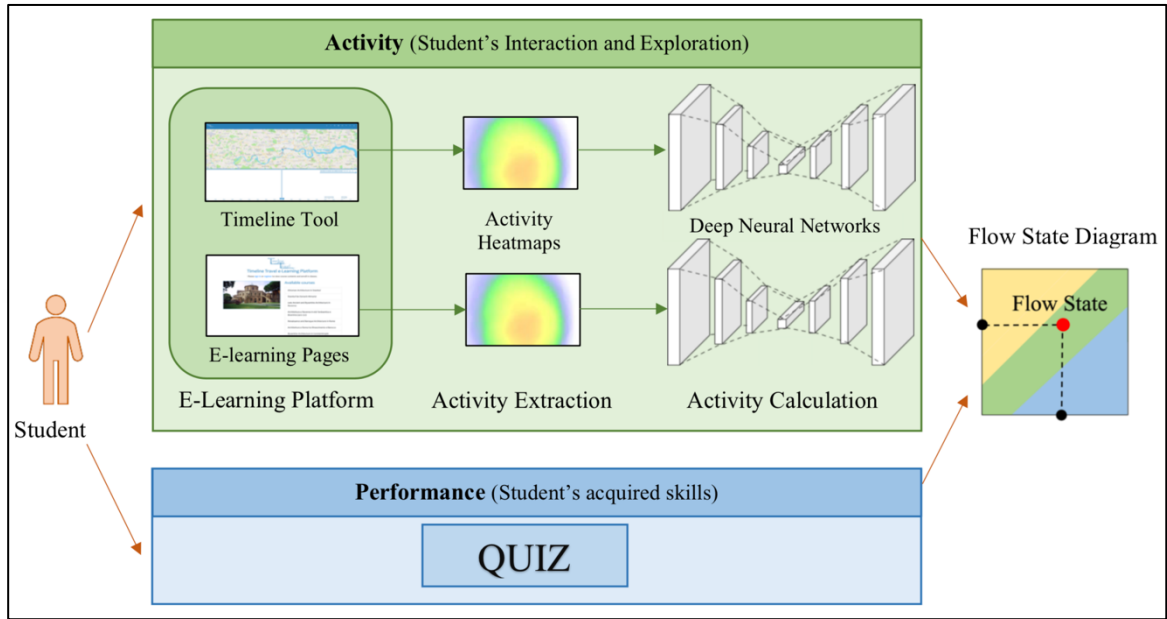


Figure 1.1. The general overview of the proposed approach

Figure 1.1 presents a general overview of the approach presented in the thesis. The students will follow a course the content of which is presented in an e-learning platform that has two interfaces: a timeline tool and classical e-learning pages. Then, an “*Activity*” value which represents the behavioral activity of a student will be generated using a deep neural network model and activity heatmaps obtained through the patterns generated by the exploration of the student. After each lecture in the course, a “*Performance*” value which indicates the acquired skills of a student, will be extracted using a quiz. Finally, according to the *Activity* and *Performance* values, a flow state diagram will be drawn to illustrate the state of the student. This approach, in this way, ensures a graphical representation that can estimate the collective learning states of the students of the course for the instructor. This graphical representation can also be observed by the students themselves to adjust their behaviors according to their flow states.

This thesis is organized as follows: theoretical information about flow theory, the techniques used in the proposed method, and the methods used to compare the correlation analysis performances with are presented in the next chapter. In Chapter 3, the introduced concepts and the methodology of the proposed methods are explained. The experimental design and its implementation are given in Chapter 4. Chapter 5 consists of findings and discussions. Chapter 6 presents the limitations and the last chapter contains the conclusions for the study.



2. THEORETICAL BACKGROUND

This section presents the concept of flow theory, heatmaps, dimension reduction methods and deep learning techniques used in this thesis.

2.1. FLOW THEORY

Mihaly Csikszentmihalyi introduced the concept of flow in 1975 and defined “flow” as a feeling experienced during an activity in which an individual is completely engaged and has high levels of focus, enjoyment, and fulfillment [12]. In other words, when an individual is in the flow state, he or she shows full involvement in the activity with full focus while enjoying the experience and feels fulfilled throughout the activity. Csikszentmihalyi states that people have the ability to choose where their attention will be focused on. This ability is discarded when an individual is in the flow state. People who are in flow show full engagement to the task presented to them without any conscious decision to divert their focus to that task. People also lose awareness of time and distractions. According to Csikszentmihalyi, this phenomenon occurs because the individual is fully engaged in the task at hand and there is no attention left to allocate to other things.

The activities that produce this feeling are identified as self-motivating and there is a balance between the difficulty of the challenge presented in the activity and the personal skill of the individual. These activities should have three conditions met in order to induce flow in people. The activities should have clear goals so that the individual can have a sense of direction and progress. They also should provide immediate feedback in order that the individuals can adjust themselves in order to stay in the flow state. Finally, these activities should have appropriate tasks that provide a sense of balance between the perceived difficulty of the challenge and the perceived skill of the individual. This condition generates confidence in people that they have the ability to complete the task presented. Since the balance between challenge and skill requires the individuals to know what they need to do and how they are doing in the activity, Csikszentmihalyi defined the flow state based on the balance between challenge and skill.

According to the flow theory, the feelings the individuals experience in an activity can be divided into three states: *Anxiety*, *Boredom*, and *Flow*. If the individual is highly skilled in the task and he or she perceives the challenge difficulty as being high, the individual would show interest in the task, the awareness of other things would be lost and the individual would divert all attention to the task. This state is the optimal state for being in flow and is defined as the Flow State. If the individual is highly skilled and the difficulty of the challenge presented is low, overcoming the challenge would be easy for the individual. After the individuals overcome the challenge, they would leave the flow state, start to get bored with the task, and divert their attention to things other than the task at hand. This state is defined as the Boredom State. On the other hand, if the individual is not skilled enough and if the challenge is difficult, the individual would get frustrated in the challenge since their attempts to overcome the challenge would not produce the intended outcome. Since this would lead the individual to anxiety, this state is defined as the Anxiety State. Csikszentmihalyi based on the challenge and skill levels proposed the flow state diagram that presents these states. Figure 2.1 presents the original three-channel flow state diagram proposed by Csikszentmihalyi.

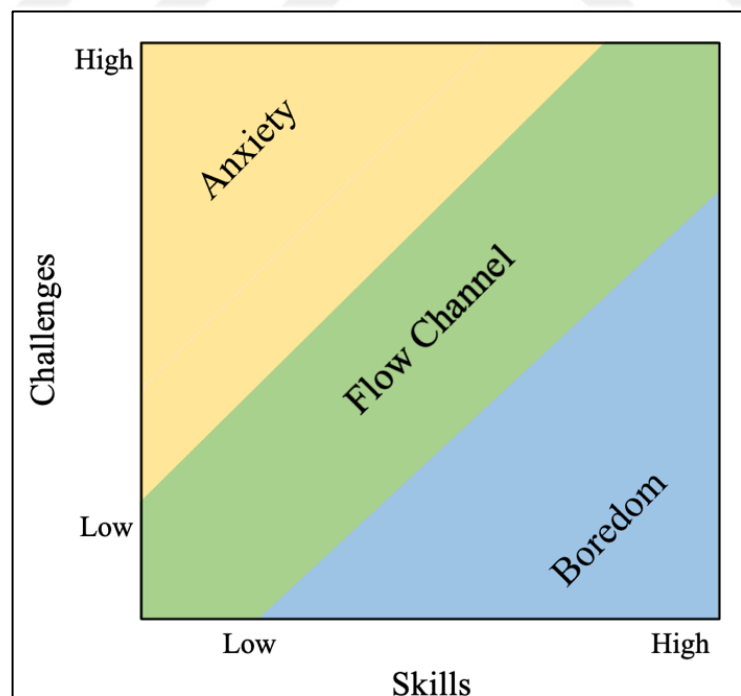


Figure 2.1. The flow state diagram proposed by Csikszentmihalyi

2.2. HEATMAPS

A heatmap is a graphical representation of matrix data where the values in each cell are represented with colors. The heatmaps are useful when the density and the spatial information are important for the data represented. In order to create this graphical representation, a matrix with the same size as the observed space is created. The density or the number of occurrences in a location are presented as numbers in the corresponding cell of this matrix. These numbers are then distributed to the neighbor cells using a propagation mechanism to obtain a smooth transition between cells with stored information. Finally, the values in the matrix are converted into colors based on a coloring scheme such as a grayscale coloring scheme or rainbow coloring scheme.

2.2.1. Grayscale Coloring Scheme

The grayscale coloring scheme converts the value of a cell to a shade of gray. The large values are represented with a color close to black and small values with a color close to white. In other words, if the shade of gray is getting darker the value in that cell becomes larger and if the shade is getting lighter the value gets smaller. Black colored cells are the largest values in the matrix while white values indicate that there are no data in the cell. The final heatmap image representation has one color channel in which the range of values is between 0 and 255. An example grayscale heatmap is presented in Figure 2.2.

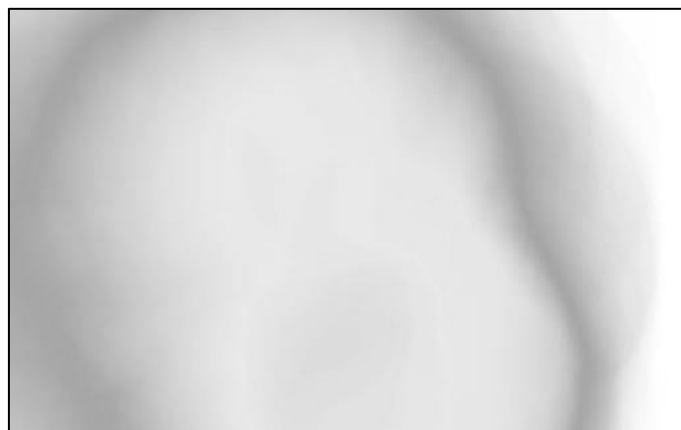


Figure 2.2. A grayscale heatmap

2.2.2. Rainbow Coloring Scheme

In the rainbow coloring scheme, the value in each cell is converted into a shade of red, green, or blue. The large values are converted into shades of red, the middle values into shades of green, and the small values into shades of blue. If there is no value in a cell, the cell is represented as white. The final heatmap image representation has three color channels where each channel has a range of values between 0 and 255. These channels, representing the red, green, and blue color channels, when combined correspond to an RGB image. An example heatmap which uses a rainbow color scheme is presented in Figure 2.3.

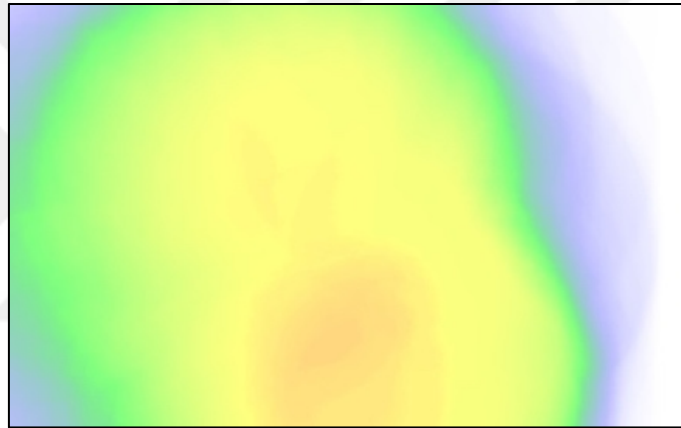


Figure 2.3. A heatmap using rainbow color scheme

2.3. DIMENSION REDUCTION METHODS

Dimension reduction is the process of reducing the number of attributes that represent information with the aim that the information can be understood or visualized with fewer features. The goal is to summarize n-dimensional data space in the form of lower 1-dimensional subspace. To fulfill this purpose, dimension reduction methods try to find similarities or correlations between the dimensions in order to represent them together or find dissimilarities so that the dimensions can be categorized into separate features.

2.3.1. Principal Component Analysis

Principal Component Analysis (PCA), invented by Karl Pearson in 1901 [31], is a mathematical technique that aims to reduce the dimensions of a dataset while increasing interpretability with minimum information loss. The dimension reduction is achieved by finding a set of uncorrelated lines that maximize variance in the dataset. These lines are called principal components. The maximum number of principal components is either the number of data or the number of dimensions whichever is the smallest.

Let S be the dataset, each data x_k is defined as $x_k = \{a_1, a_2, a_3, \dots, a_n\}$ where each a_n is an attribute (dimension), n is the number of attributes, and m is the number of data. Initially, for each dimension, the mean value is calculated using the values only in that dimension using equation 2.1.

$$mean_n = \frac{\sum_{i=1}^m a_i}{m} \quad (2.1)$$

The point x_{mean} is defined by the mean values of each dimension and denoted as $x_{mean} = \{mean_1, mean_2, \dots, mean_n\}$. Then, all the data are adjusted according to this point with x_{mean} placed at the origin of the new coordinate system where each data are denoted as x'_k .

$$x'_k = x_k - x_{mean} \quad (2.2)$$

After the coordinate system is adjusted to the new shifted value, a randomly chosen line that goes through the origin point is placed on the coordinate system. Each data x'_k in the dataset S is projected onto this randomly chosen line where each projected point is denoted as p_k .

The PCA chooses the line which fits the dataset best according to the distance between p_k and the new origin point. The random line is changed to search for the best fit. The best-fitting line is found by maximizing the sum of squared distances between p_k and origin point. This value is called the eigenvalue and the line that fits the data best is called the principal component.

For the other principal components, the new lines are randomly chosen lines that go through the origin point and are perpendicular to the previously calculated principal components. Then the same process is applied to find the best fitting line.

2.3.2. Factor Analysis

In 1904, Charles Spearman discussed a method which revealed that in the data he was working on there was a single unobserved variable that could explain the observed variables with a weighted representation [32]. Louis Thurstone, in 1935, published his book “The Vector of Mind” which summarizes two of his previous studies [33]. In this book, Thurstone built on the idea proposed by Spearman and showed that there can be multiple unobserved variables to explain observed variables and proposed the Factor Analysis (FA).

A factor in FA is the weighted representation of a set of dimensions that show similar patterns in explaining the data. The factors are not directly measured but obtained as a linear function of a set of the observed variables or dimensions.

Initially, the procedures of PCA is applied to the data to produce a set of factors. Then a correlation analysis is performed to extract the corresponding dimensions of the factors. The correlation values calculated for this procedure is called factor loadings. Ideally, every dimension should correspond to a single factor. If there is a dimension that corresponds to more than one factor and has multiple factor loadings, these dimensions are called cross-loadings. If all the dimensions have single factor loadings, the factors are chosen to be the lower dimension representation of the dataset. In the event of the presence of cross-loadings, a rotation algorithm is applied to the factor loadings matrix in order to achieve the desired structure with the dimensions and the factors.

2.3.3. Multidimensional Scaling

Multidimensional Scaling (MDS), also called Principal Coordinates Analysis, is a dimension reduction method proposed by Warren S. Torgerson in 1958 [34] that aims to map an n-dimensional dataset to lower dimensions preserving the pairwise distances between each object in the dataset. Hence, the dimension reduction is performed using a pairwise distance matrix.

Let S be the dataset, each data x_k is defined as $x_k = \{a_1, a_2, a_3, \dots, a_n\}$ where each a_n is an attribute (dimension), n is the number of attributes, and m is the number of data. Initially, the data needs to be converted to a distance matrix. For each pair in the dataset S , x_i and x_{ij} , a distance value d_{ij} is calculated using Euclidean distance. The matrix D created with all d_{ij} values is called the dissimilarity matrix.

After the dissimilarity matrix is created, a randomly chosen line that goes through the origin point is placed on the coordinate system created using two of the dimensions of the original data. Each data x_k in the dataset S is projected onto this randomly chosen line where each projected point is denoted as p_k . Then, the dissimilarity matrix D' is generated using the new p_k values and the values in the remaining dimensions. The goal is to minimize the distance between two matrices D and D' . The line that has the minimum distance between matrices is chosen to be one of the dimensions for the lower dimension representation. This process is applied one by one until the desired number of dimensions is achieved. Hence, the maximum number of dimensions in the lower dimension representation is $n - 1$.

2.3.4. Independent Component Analysis

The early concepts of Independent Component Analysis (ICA) have been introduced by Jeanny Hérault and Bernard Ans in 1984 [35] and refined by Pierre Comon in 1994 [36]. ICA aims to maximize the independence of the attributes that define the dataset provided by separating independent information from data with mixed sources.

In ICA, the data D is first whitened to produce the matrix X . The whitening is a linear transformation that transforms the values of a vector to a new set of values so that the covariance matrix of the new vector is the identity matrix. This means that the new values are uncorrelated. Then, the Gaussianities of the distributions of all axes are reduced by rotating the matrix X . The components that represent the data with fewer dimensions are the values in the transformation matrix T which transform the original matrix D to the final rotated matrix X .

2.3.5. Gaussian Random Projection

Random Projection (RP) relies on the assumption made by William B. Johnson and Joram Lindenstrauss in 1984 [37] which is if points in a vector space have a sufficient amount of dimensions, they can be projected into a lower dimension space preserving approximate distances between points. In 2001 Ella Bingham and Heikki Mannila proposed to use Gaussian matrices for random projection [38].

Suppose that the dataset is represented as $X_{m \times n}$ where m is the number of data in the dataset, n is the number of dimensions, and the desired new dimensions is p . According to the assumption of Johnson and Lindenstrauss, the minimum number of dimensions that the data can be projected with high probability is calculated with the number of data m and the desired error ε using equation 2.3.

$$p > \frac{\ln m}{\varepsilon^2} \quad (2.3)$$

If the desired new dimensions satisfy the assumption then a random matrix $R_{n \times p}$ is generated. The values in the random matrix should be standard normal random values. This means the values should be in Gaussian distribution with a mean of 0 and the standard deviation should be 1. Then, the new values in the lower dimension space are calculated with the following formula where $Y_{m \times p}$ is the representation of the dataset in p-dimension space.

$$Y_{m \times p} = \frac{1}{\sqrt{p}} X_{m \times n} R_{n \times p} \quad (2.4)$$

2.3.6. Isometric Feature Mapping

Isometric Feature Mapping (Isomap) is a non-linear dimension reduction method proposed by Tenenbaum et al. in 2000 [39]. Isomap is an extension of MDS where unlike MDS which uses Euclidean distances, the pairwise distances are preserved using geodesic distances. Geodesic distance is calculated between two vertices in a graph by the number of edges in the shortest path that connects these vertices.

Initially, the data are converted into graphs where each point having a predetermined number of neighbors. The distances between points in different neighbor groups are defined as infinite. Then MDS is applied on the dissimilarity matrix D where the distances are calculated using geodesic distances between the points.

2.3.7. t-Distributed Stochastic Neighbor Embedding

The t-Distributed Stochastic Neighbor Embedding (t-SNE) introduced by Maaten and Hinton in 2008 [40] is a technique, primarily used for data visualizing by reducing high dimensional data into low dimensions. This technique aims to preserve only the local similarities or the small pairwise distances.

The t-SNE is performed in four steps: similarity probability in the high-dimension space, similarity probability in the low-dimension space, similarity measurement between the probability distributions, and update process. Let D be the dataset, each data x_k is defined as $x_k = \{a_1, a_2, a_3, \dots, a_n\}$ where each a_n is an attribute (dimension), n is the number of attributes, and m is the number of data.

In the first step, the distances between all pairs of data are calculated using Euclidean distance in the high-dimension space. For calculating the similarity probabilities of all the values with x_i , a Gaussian distribution is centered over x_i and a point d_{ji} is created next to x_i in the same distance as the measured distance between x_i and x_j . The distance between the distribution curve and the d_{ji} gives P_{ji} , the similarity probability of x_j to x_i . The P_{ji} and P_{ij} might be different. Therefore, t-SNE calculates the average value with these two values in order to find the actual similarity probability between x_i and x_j .

For the second step, the data are randomly placed on a low-dimension space and the procedures in the first step are applied exactly as they are. The only exception is to use t-distribution instead of Gaussian distribution. This procedure will generate another set of similarity probabilities denoted as Q_{ij} .

The third and fourth steps aim to move the data in the low-dimension space so that the Q_{ij} gets close to P_{ij} as much as possible. In order to achieve this purpose, in the third step t-SNE calculates the differences between these two probabilistic distributions with

Kullback-Liebler divergence [41] and in the fourth step changes the positions of these randomly placed points according to the cost calculated in the third step. The updating process, calculation of Q_{ij} , and measuring the differences between Q_{ij} and P_{ij} is repeated until the cost is minimized.

2.4. COMPUTATIONAL FRAMEWORK

This section presents the computational techniques that the proposed method uses. The convolution, pooling, and upsampling operations are described in the following sections. Then the classical Artificial Neural Networks and Convolutional Neural Networks which use convolution operations are explained. Finally, the details of the Autoencoders and its convolution-based form Convolutional Autoencoders are presented.

2.4.1. Convolution

Convolution is a mathematical operation which is defined as the operation that finds the effect of function g on another function f and is denoted as $f * g$. In image processing, the convolution operation defines the effect of a kernel matrix K on the data matrix D which is the input image. The image convolution is denoted as $D * K$ and the result is calculated with the following formula where n is the number of rows and m is the number of columns in the kernel matrix, k_{ij} is the value in the i, j position of the kernel matrix, and d_{ij} is the value at the i, j position of the data matrix. If the sum of the values in the kernel matrix is 0, it is set to 1.

$$D * K = \left\lfloor \frac{\sum_{i=1}^n \sum_{j=1}^m d_{ij} k_{ij}}{\sum_{i=1}^n \sum_{j=1}^m k_{ij}} \right\rfloor \quad (2.5)$$

The image convolution is usually used for filtering operations where the kernel matrix is defined as the filter. In image filtering, the center of the filter is placed on the top-left pixel in the original image and the convolution operation is performed. In order to obtain convolution results for all pixels, the center of the filter is shifted to each pixel. The convolution operation is always done with the filter and the original image in order to ensure the integrity of the effect value calculated with each operation.

The pixels at the edges of the image should be investigated differently in image convolution because these pixels would need values outside of the image boundaries for this operation. These pixels can be handled in two ways: by adding padding to the original image or performing the operation starting from the first pixel that fits the kernel. Padding would ensure the size of the original image to be preserved while performing the operation without padding would shrink the resulting image size. Figure 2.4 presents a 5×5 matrix and its padding added version.

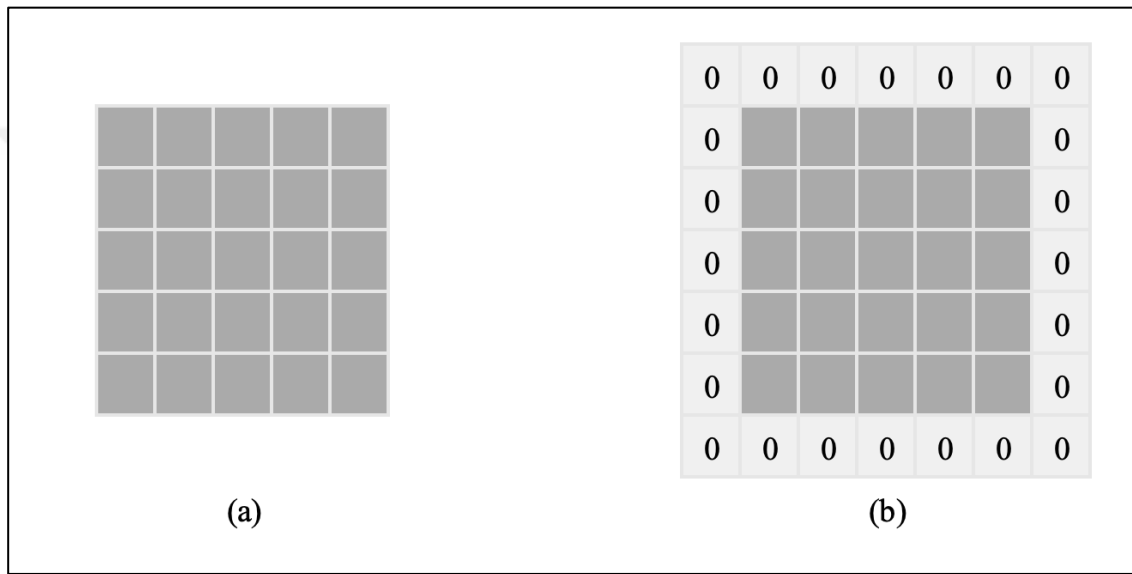


Figure 2.4. A padding example (a) Original 5×5 matrix (b) Matrix with 1 padding

The padding operation adds layers of zeroes to the image, in order that the edge pixels can be considered in the convolution operation. The number of zero layers to add, p , can be calculated as $p = (m - 1)/2$ where m is the kernel size. The size of the output matrix n' can be calculated with the equation 2.6 where n is the size of the data matrix.

$$n' = n + 2p - m + 1 \quad (2.6)$$

In convolution operation, the kernel slides through the image vertically and horizontally. The number of pixels the kernel shifts in each iteration is called stride. The default stride value for convolution is 1 where the kernel is shifted one pixel vertically or horizontally. This means that the kernel will perform convolution on each pixel of the image. If the stride is 2, the kernel is shifted two pixels in order to perform convolution for one of every two pixels. The size of the output matrix n' can be calculated with equation 2.7 where s is the stride value.

$$n' = \frac{n + 2p - m}{s} + 1 \quad (2.7)$$

2.4.2. Pooling and Up-Sampling

Pooling is a matrix operation that aims to downsample a data matrix in order to decrease its size and obtain an abstract representation. The pooling operation represents a set of values in the data matrix as a single value in order to decrease the size of the matrix. The boundaries that a single value will be generated from is defined as the pooling kernel size and usually, it is chosen as a square matrix. This kernel slides through the data matrix to obtain single values from each window defined with the kernel size. The amount of cells the kernel shifts is called stride. There are three types of pooling operations: maximum, average, and global.

The maximum pooling (MaxPooling) represents a kernel window with the maximum value inside, while average pooling calculates the mean value. On the other hand, global pooling generates a single value for the whole image and can be considered as having a kernel size exactly as the data matrix. The global pooling can be used with maximum or the average operations. Figure 2.5 presents maximum pooling and average pooling examples with a kernel size of 2×2 and the kernel shifting two cells.

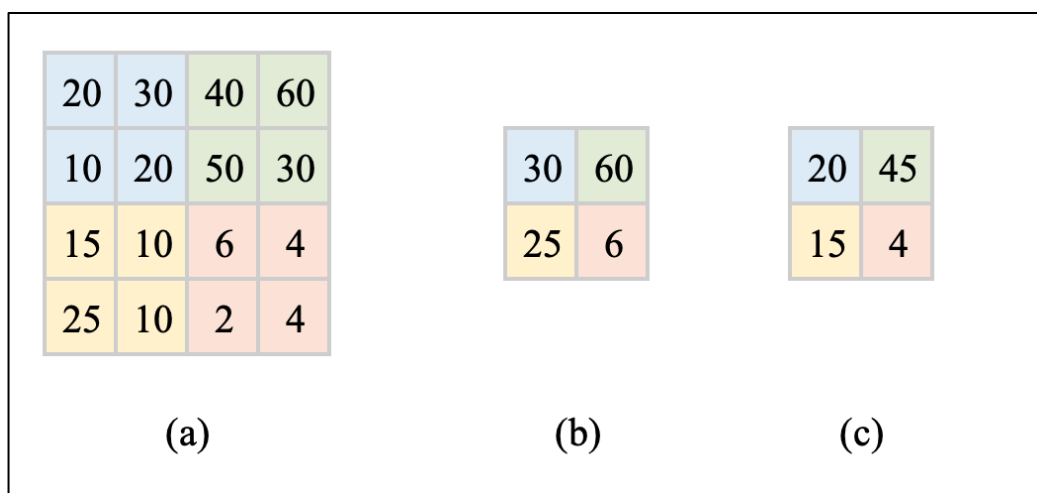


Figure 2.5. A pooling example with kernel size of 2×2 and $stride = 2$

(a) Original 4×4 matrix (b) Maximum Pooling (c) Average Pooling

The aim of the up-sampling operation is to increase the size of the matrix. This operation is performed in two ways: uniform sampling and zero sampling. In uniform sampling, the value of a pixel is copied to its neighbors specified in the sampling kernel while in zero sampling, the neighbor values are set to 0. Figure 2.6 presents an example for each method with a 2×2 kernel size.

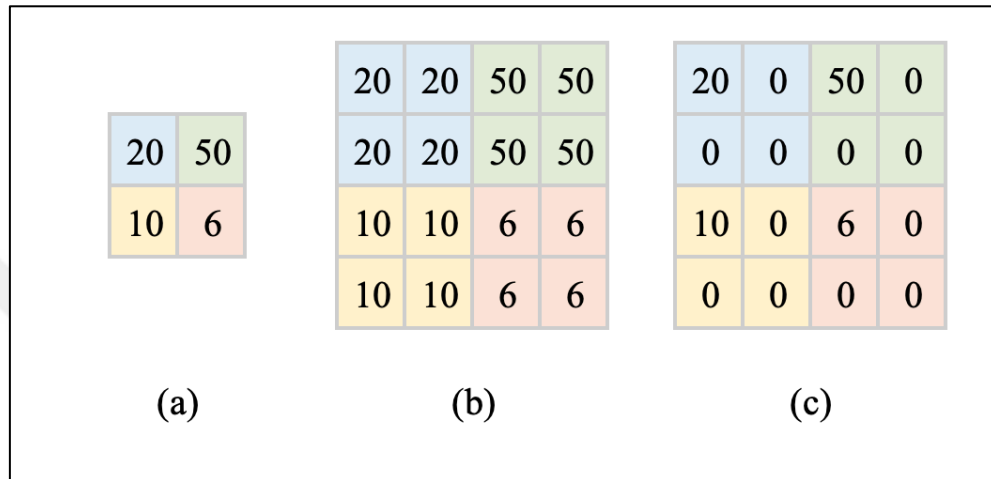


Figure 2.6. An up-sampling example with kernel size of 2×2
(a) Original 2×2 matrix (b) Uniform Sampling (c) Zero Sampling

2.4.3. Machine Learning Techniques

Machine Learning is the field of computer science studies which conducts research on algorithms that can improve through experience automatically. Tom Mitchell gives the formal definition of Machine Learning algorithms as “A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P , if its performance at tasks in T , as measured by P , improves with experience E .” [42].

The aim of machine learning techniques is to build a mathematical model based on a set of training data where a machine can perform tasks using the built model without a need for that specific task to be programmed using a set of rules. Machine learning approaches can be divided into three major categories: supervised learning, unsupervised learning, and reinforcement learning.

Supervised Learning

Supervised learning approaches aim to associate input-output pairs through a mathematical model. These approaches are trained using a set of inputs and their corresponding outputs in order to either perform classification or regression. Classification categorizes data into a set of discrete values or classes while regression finds a prediction among continuous values. The mathematical model built at the end of the training procedure ideally can predict the desired outputs of new inputs that it has never seen before.

Artificial Neural Networks: Artificial Neural Network (ANN) is a supervised machine learning technique which is inspired by biological neurons and generally used for classification and regression problems. ANNs can be simply described as a network that contains a set of nodes and the connections between these nodes.

The simplest form of ANNs is the Feed Forward Neural Networks in which a neuron in a layer is connected only with the neurons of the next layer. A standard feed-forward ANN is composed of three layers: the input layer, the hidden layer, and the output layer. The input layer handles the input data and the output layer produces the desired output. The hidden layer transforms the input into the output form and acts as the transformation layer. The number of hidden layers depends on the data and the problem the network is needed for.

Each node in the layers is called neurons and it stores an activation function and a weight. The activation function decides the way, the output generated based on the weighted inputs is transferred to the next node. The number of neurons in each layer is chosen according to the data and the problem.

In the training process, an update algorithm changes the weights based on the error measured in the output layer. The error is calculated using the labels of the training data and the predicted output of the network. The goal of this process is to minimize the error so that the network can generate predictions as close to the actual outputs as possible. Figure 2.7 presents an example of an ANN architecture.

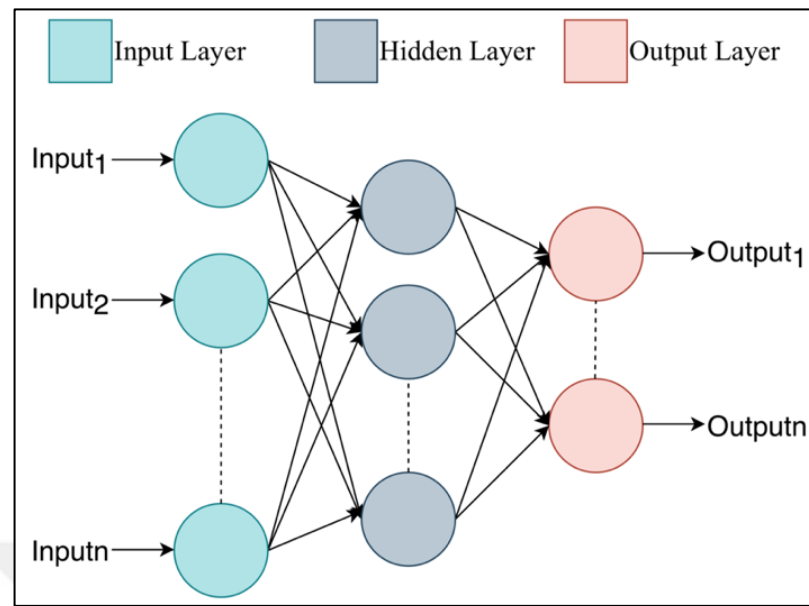


Figure 2.7. An Artificial Neural Network architecture

Convolutional Neural Networks: Convolutional Neural Network (CNN) is one of the most successful recent forms of neural networks. Recent CNN models report a better accuracy in classification problems, mostly because of their ability to find local patterns. These networks that use convolutional operations were proposed by LeCun et al. [43]. Classical ANNs are modeled to be fully connected in-between the network layers. On the other hand, CNNs use kernel structures (filters) to connect the consecutive layers which reduce the number of connections. The abstraction ability of convolution operation takes the place of weighted activation of ANNs. CNNs comprise of three layers: a convolutional layer, a pooling layer, and a classification layer.

In the convolutional layer, the input data act as the data matrix for the convolution operations. The layer has a series of filters with a fixed kernel size. The number of filters in each convolutional layer and the kernel size of these filters are set empirically based on the data and the problem. With each filter, a convolution operation is applied to the input data matrix to generate the effect of the kernel on the data. This result then becomes the input to the next layer whether it is another convolution layer or a pooling layer. The kernel values are set randomly at the beginning of the training procedure. The update algorithms of CNNs updates the kernel values based on the measured error at the end of the network.

The pooling layer is used in the CNNs to decrease the size of the input and perform an abstraction of the data. Both the maximum and the average pooling methods could be used for this purpose. The output of the convolutional layer is transferred to the pooling layer, where data are resampled by using a maximum or average function. The kernel size is usually chosen as 2×2 but can be set empirically based on the expected output and the problem.

The classification layer is used only if the network aims to perform classification and it can be a fully connected neural network. After a certain number of convolutional and pooling layers, the representation of the data matrix becomes smaller. This final representation then can be used directly as the abstract representation or it could be the input to the classification layer. Figure 2.8 presents a CNN architecture with a fully connected classification layer at the end.

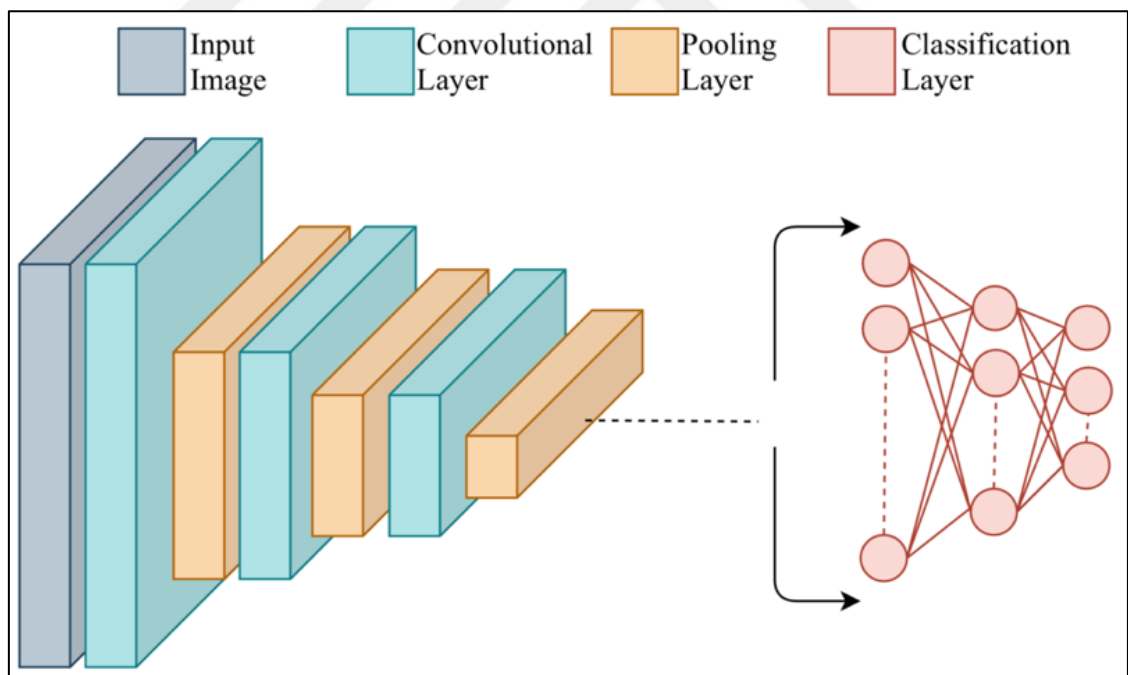


Figure 2.8. A Convolutional Neural Network architecture

Unsupervised Learning

Unsupervised learning approaches aim to find undetected patterns in a dataset without needing any human supervision on the data. This means that these methods can find patterns in a dataset even if the data are not labeled, classified, or categorized. Unsupervised learning methods can be divided into two categories: clustering and dimension reduction. Clustering searches for commonalities in order to separate the data into groups. On the other hand, dimension reduction, tries to find dissimilarities or commonalities in the dataset and consequently generates a representation that has fewer attributes.

Autoencoders: Autoencoder (AE) is an unsupervised learning method that was originally introduced by Mark Kramer [44]. Essentially, Autoencoder is a dimension reduction method in which the abstract representations of a dataset are obtained by the reconstruction of the input data through a set of fully connected feed-forward networks. An ideal autoencoder should be sensitive to the input in such a way that the input could be reproduced accurately and be insensitive enough in order that the network does not memorize the training data. When there are structures, patterns, or abstract features among the data, autoencoders try to learn these structures in order to extract compressed representations that comply with the structures found in each input data.

Autoencoders are modeled to have several networks which consist of an input layer, an output layer, and a hidden layer. Each network is trained individually in an unsupervised manner. In the training process, the goal is to minimize the error between the input and the predicted output. At the end of each network's training, the hidden layer, also called the bottleneck, represents the input data with fewer attributes. The results of the last bottleneck layer are the abstract representations of the input data and they are usually called the *Features*. Figure 2.9 presents the architecture for an autoencoder with a single fully connected network.

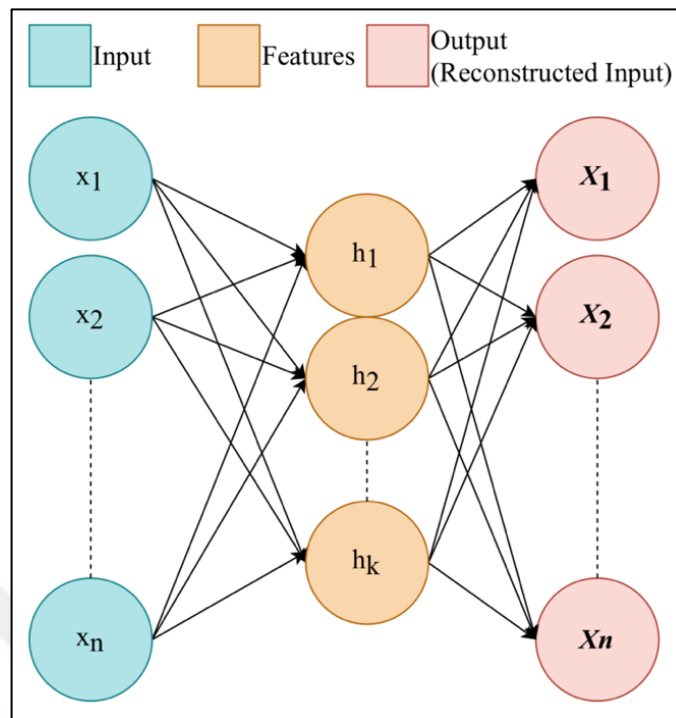


Figure 2.9. An Autoencoder architecture

Convolutional Autoencoders: In 2011 Masci et al. introduced the Convolutional Autoencoder (CAE), which is a dimension reduction method that takes advantage of convolution operations and unsupervised approach of autoencoders [45]. The CAEs use convolution operations to reduce the number of parameters to be trained by discarding the fully connected structure of the classical autoencoders. The convolution operation also enhances the pattern extraction power of autoencoders. CAEs use inputs in multidimensional matrix structures, generally images, because of the convolution operation.

In CAE training, the goal is to reproduce the input image as an output image through a convolutional network. This network comprises five layers: the input layer, the encoder layer, the feature layer, the decoder layer, and the output layer.

The encoder layer comprises a set of convolutional and pooling layers that find the patterns in the dataset and generate the compressed representations of the inputs. This compressed abstract representation is called the features of the input data and the layer which contains the features is called the feature layer. The decoder layer takes the features generated by

the encoder and by using up-sampling and convolution operations attempts to increase the size and reproduce the original input. The goal of the training process is to minimize the error measured between the input layer and the output layer.

The compressed representation of an input x in the k th layer h^k is obtained as:

$$h^k = \sigma(x * W^k + b^k) \quad (2.8)$$

where the operator $*$ is the convolution operation, W^k represents the weights, b^k is the bias and σ denotes the activation function. These feature representations are then used to reconstruct the input x in the output y using

$$y = \sigma\left(\sum_k h^k * \tilde{W}^k + b\right) \quad (2.9)$$

where $\tilde{W}^k$ represents the flipped weights in all directions. The convolution operation using the flipped weights is called deconvolution. Figure 2.10 presents a convolutional autoencoder architecture.

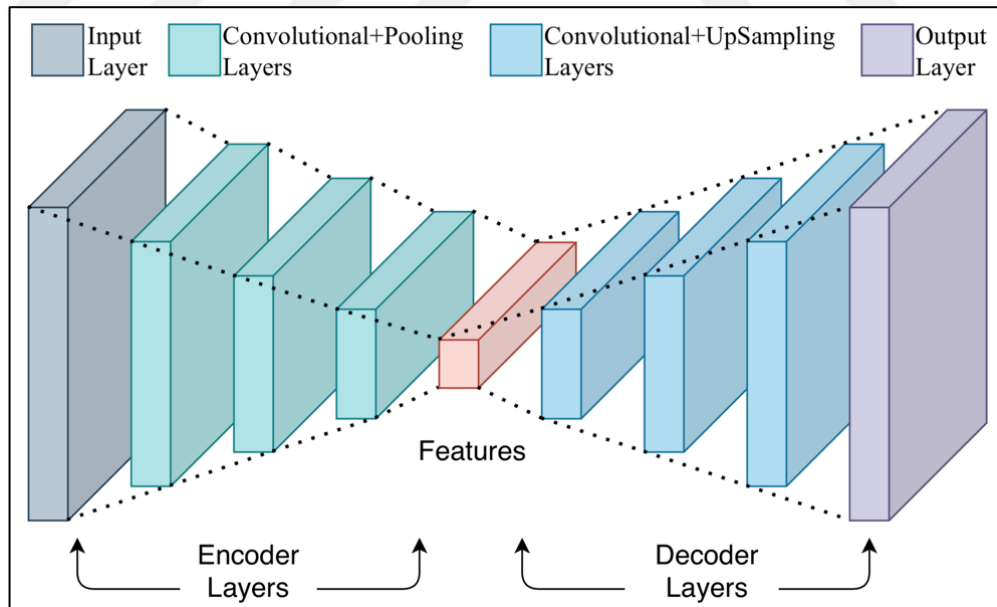


Figure 2.10. A Convolutional Autoencoder architecture

2.5. STATISTICAL ANALYSIS

A null hypothesis is a statement that there is no relation between two sets of variables. The null-hypothesis is assumed to be true unless proven otherwise. In statistical analysis, if the null-hypothesis is tested and accepted according to the p -value calculated in the analysis and the statistical significance set for the experiment, the two phenomena being tested are not linearly related to each other. Usually, an alternative hypothesis is chosen against the null-hypothesis so that when the null-hypothesis is rejected the alternative hypothesis can be accepted instead.

Statistical significance, α -value, is the probability that the analysis performed would reject the hypothesis proposed in a study. This value is chosen before the data collection process starts and is usually set as 5 percent ($\alpha = 0.05$). The p -value generated with the performed analysis is compared with the α -value to determine if the hypothesis is accepted or not.

p -value is the probability of finding values at least as extreme as the observed data. A small p -value means that the probability of finding values outside the observed data is small so the null-hypothesis that there is no relation between the observed values and the tested distribution can be rejected. On the other hand, a large p -value means that the probability of finding values outside the range of observed values is high so the observed values do not cover the distribution well enough and there is no relation between the observed values and the distribution. The hypothesis is said to be accepted if p -value is smaller than the α -value specified for the experiment the data are collected.

2.5.1. Normality Test

In statistics, normality tests are used to identify if the collected sample is normally distributed or not. The data are tested against the null-hypothesis that the distribution of the collected values is normal. One of the most powerful and widely used normality tests is the Shapiro-Wilk normality test proposed by Samuel Sanford Shapiro and Martin Wilk in 1965 [46]. The normality tests are used in correlation analysis to identify which algorithm will be chosen based on the normality assumptions of the algorithms.

Suppose that the values observed for an experiment are denoted as x_n and the number of values is n . Initially, the data are ordered from the smallest to the largest with the intention that $x_1 \leq \dots \leq x_n$. Then the test statistic W is calculated using the equation 2.10 where $\bar{x}$ is the mean of the observed values and a_i are the coefficient values chosen from the table proposed by Shapiro and Wilk in their original publication. If n is even, m is $n/2$, otherwise, it is $(n - 1)/2$.

$$W = \frac{(\sum_{i=1}^m a_i (x_{n+1-i} - x_i))^2}{\sum_{i=1}^n (x_i - \bar{x})^2} \quad (2.10)$$

The p -value for this normality test is obtained from another table proposed in the original publication of Shapiro and Wilk. The p -value is chosen based on the test statistic W and the number of values collected in the experiment n . As described before, if the obtained p -value is less than the α -value then null-hypothesis is rejected and the observed sample is accepted as normally distributed.

2.5.2. Correlation Analysis

Correlation analysis is a statistical method that evaluates the strength of the relation between two sets of observed variables. It should be noted that the existence of a correlation between variables does not mean causality hence, this analysis can not be used to prove the existence of cause-effect relation. The correlation analysis proves that there is a relation between the variables and evaluates the strength of this relation. The most used correlation analysis methods are Pearson correlation coefficient and Spearman's rank correlation.

Pearson Correlation Coefficient

In 1895 Karl Pearson proposed the Pearson correlation coefficient which tries to find linear relations between two variables [47]. The assumption for applying this method is that both of the variables should be normally distributed.

Suppose that there are two variables x_n and y_n where n is the number of observed values for the x, y pairs. The Pearson correlation coefficient r_{xy} is calculated using equation 2.11

where $\bar{x}$ and $\bar{y}$ are the mean values. The r_{xy} value indicates the power of relation between x_n and y_n variables.

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (2.11)$$

In order to prove if the found correlation coefficient can be accepted or not, the p -value for this analysis should be compared with an α -value. The p -value is obtained from a t -distribution table using a calculated t -value and degree of freedom value which is $n - 2$. The t -value is calculated with equation 2.12.

$$t = \frac{r \sqrt{n-2}}{\sqrt{1-r^2}} \quad (2.12)$$

Spearman's Rank Correlation

Charles Spearman proposed a rank correlation which evaluates the existence and the power of the relation between the rankings of two variables [48]. This method proposed in 1961 was named Spearman's rank correlation and it tries to describe the relation between two variables using a monotonic function. This method is developed to be more sensitive to nonlinear relations compared to the Pearson correlation. Usually, if the assumption of normal distribution in Pearson correlation is not satisfied, the relation is evaluated using Spearman's rank correlation.

Suppose that there are two variables x_n and y_n where n is the number of observed values for the x, y pairs. Initially, both of the variables are sorted in ascending order and they are given a rank, starting from 1. The coefficient r_{xy} is found using equation 2.10 where d_i is the difference between the rank values of x_i and y_i .

$$r = 1 - \frac{\sum_i^n d_i^2}{n(n^2 - 1)} \quad (2.10)$$

The p -value which is needed to prove the significance of the found relation is obtained from a t -distribution table using the same procedure and the equations used in the Pearson correlation p -value calculations.

3. METHODOLOGY

This research aims to provide an objective measurement of student learning state through concepts designed to reflect “challenge” and “skill” proposed by Mihaly Csikszentmihalyi for the Flow Theory. The research contributes to the e-learning field by proposing objective, transferable and generalizable measurements, and an analysis method for learning state evaluation.

According to Csikszentmihalyi, for an individual to experience flow, there must be a balance between the difficulty of the challenge presented and the skill of the individual. These concepts need to be measured so that the existence of a balance between the measurements can be analyzed. In the scope of e-learning, challenge difficulty and the skill level of a student can only be obtained through subjective self-reports that measure the perceived difficulty of a course and the perceived skill of a student.

This thesis proposes two concepts that reflect challenge difficulty and the skill of a student: *Activity* and *Performance*. *Activity* is defined as the value of a student’s interactions throughout a lecture obtained from their mouse click and movement operations while *Performance* is defined as the evaluation value of a student’s knowledge through examination.

Activity is proposed to be used in this thesis instead of challenge difficulty since these concepts affect each other directly. Let us assume that the difficulty of a lecture is high. This would lead to an intensive interaction from the student in the lecture content since the knowledge to be acquired from the lecture would require the student to search for critical information. Having an intense interaction with the content would then provoke high activity for the student. If the lecture difficulty is assumed to be lower, the student would not need to search for critical information as much. This would produce weak interaction, hence low activity. The same conclusion can be obtained when the concepts are analyzed in the opposite direction. When the activity of a student is assumed to be high, the presented challenge would be perceived by the student as difficult. Then in the scope of e-learning, *Activity* can be used instead of challenge difficulty. Figure 3.1 presents the conceptual relationship between challenge difficulty and *Activity*.

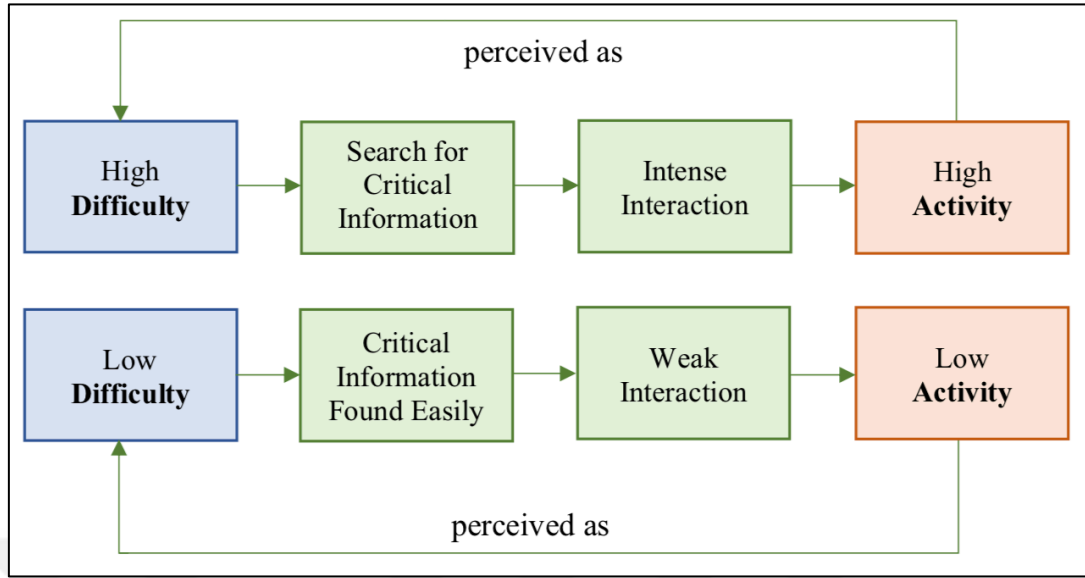


Figure 3.1. The conceptual relationship between challenge difficulty and *Activity*

The *Activity* of a student is defined as a vector that contains a set of features. The features correspond to a set of particular features which are either defined to reflect high-level representations of the activity or representations extracted through feature extraction techniques. To define the *Activity* vector let V be a vector space and the set of all possible student interactions. The set of interactions in a course is defined as:

$$A = \{ a_1, a_2, \dots, a_m \}, A \in V$$

where each $a_k \in [1, m]$ represents the *Activity* of a particular student and m is the number of students in that particular course. The *Activity* vector of each student a_k , then, is defined as:

$$a_k = \{ f_{k_1}, f_{k_1}, \dots, f_{k_n} \}, a_k \in A$$

where each $f_{k_i}, i \in [1, n]$ represents a feature of interaction and n is the number of features defined by the method used to generate the *Activity* vector.

The concept of *Performance* is proposed to be used instead of the skill of the student. The examination is usually described as the formal test of an individual's knowledge or skill and the student's grade in an examination is called their performance. Hence, using *Performance* instead of skill is logical. Let us assume that a student is highly skilled considering the content of a lecture. This student is either already familiar with the subject or would acquire a sufficient amount of knowledge from the lecture which would lead to

high points in the exam for the student. Then, that student would be described to show high performance. If the student is assumed to be not skilled enough for the lecture, then the student would acquire insufficient knowledge from the content of the lecture. This would lead to low points in the exams hence the student will show low performance. While analyzing the concepts in the opposite direction, if the student is assumed to have high performance in the exams, the student's skill concerning the content of the course is expected to be high. Then, in the scope of e-learning, *Performance* can be used instead of the skill of an individual. Figure 3.2 presents the relation between the skill of an individual and *Performance*.

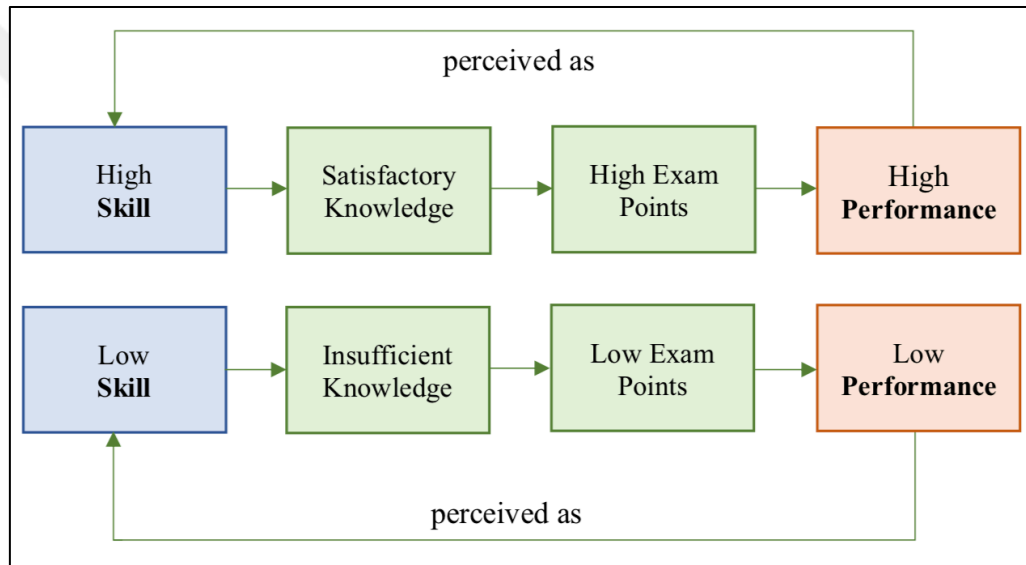


Figure 3.2. The conceptual relationship between skill and *Performance*

The *Performance* of a student is defined as a vector that contains a set of grades. Grades can be obtained from quizzes, exams, reports, or exercises. Hence, let W be the vector space and all possible student performances. The set of performances in a particular course is defined as:

$$P = \{ p_1, p_2, \dots, p_m \}, P \in W$$

where each $p_k \in [1, m]$ represents the *Performance* of a particular student and m is the number of students in that particular course. The *Performance* vector of each student p_k , then, is defined as:

$$p_k = \{ g_{k_1}, g_{k_1}, \dots, g_{k_n} \}, p_k \in P$$

where each $g_{k_i}, i \in [1, n]$ represents a grade obtained from one of the examination methods and n is the number of examinations presented in that particular course.

According to Flow Theory, if a student experiences flow, that student is said to be learning. This means that for a student to learn, that student should be in flow, hence there must be a balance between the difficulty of the challenge presented and the skill of the student. Therefore, if the balance between challenge difficulty and skill produces flow experience then, the balance between *Activity* and *Performance* should produce the same feeling. Then the balance between *Activity* and *Performance* can be considered as the indicator of whether a student is learning or not. Figure 3.3 illustrates the relation between the concepts proposed to reflect the concepts of Flow Theory.

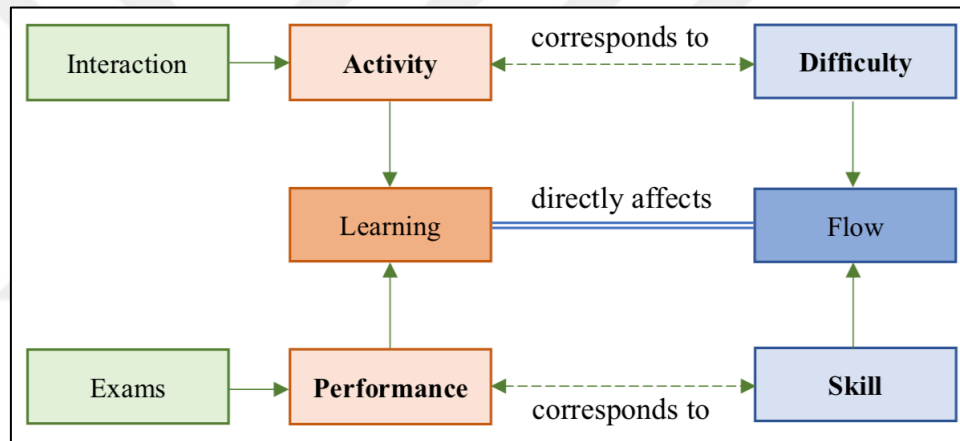


Figure 3.3. Proposed concept relations

It should be noted that the relation between skill and challenge difficulty also provides two other emotional states a student can experience: *Anxiety* and *Boredom*. To use the proposed concepts to reflect difficulty and skill, they also must explore these emotional states. According to Flow Theory, *Anxiety* occurs when the difficulty is high and the skill of the individual is low. When a lecture content is perceived as difficult, the students are expected to search for critical information through high activity in the environment. If the performance of a student is low, even after showing high activity, the student would start to show signs of anxiety. This emotional state occurs because there is a lack of perceived accomplishment for the student. Decreasing the difficulty of the lecture content would produce better performances for the students who experience *Anxiety*. Therefore, the student would experience flow since the performance and the activity would start to be in balance.

On the other hand, an individual experiences *Boredom* when the difficulty of a challenge is low and the student is highly skilled. When a student is highly skilled in the content of a lecture, the student is expected to either find the required information quickly or have the knowledge they were going to acquire. This student is, then, expected to show low activity in the environment. Since learning is acquiring new knowledge or skill, even though the performance of the student is high, this student is not learning. If the level of the course does not increase or the student does not encounter new information, this student would start to get bored. Figure 3.4 presents the relation between emotional states and concepts.

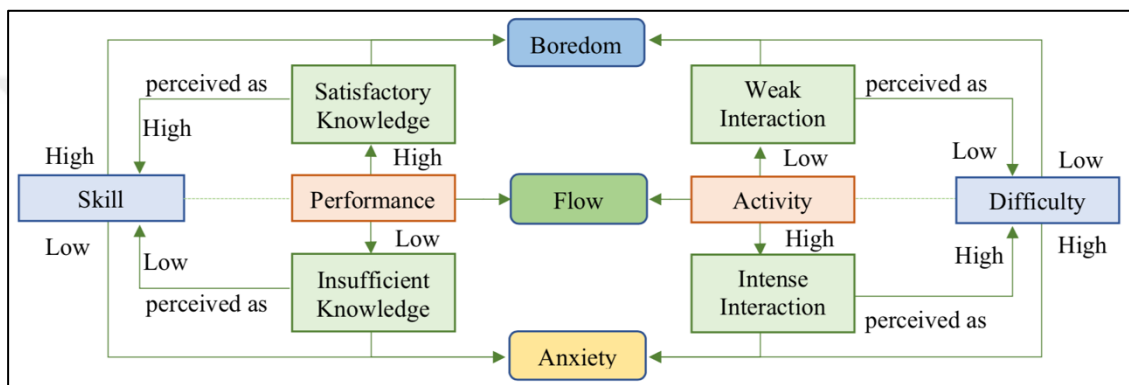


Figure 3.4. The relation between emotional states, measurement concepts, and constructs of flow theory

In the scope of e-learning, *Performance* can be obtained numerically through the exam points of the students, but there is no simple way of obtaining a single value for the *Activity*. This thesis proposes methods to generate *Activity* through data collected from mouse interactions of the students but then again, there is no simple way of knowing which value is the optimal value for interaction. To produce a single *Activity* value for each student that is grounded by an optimal value, the thesis proposes the concept of an *Expert* which is inspired by the article published by Georgiou and Demiris [30].

According to the article published by Hong and Liu where the authors examined the thinking strategies of several participants based on a sliding blocks game called Klotzi, an expert user can be determined by observing the minimum number of actions in minimum time [49]. Their findings indicate that an expert would present analogical thinking where a learner would behave randomly. Georgiou and Demiris based their Expert concept on the proposition made by Hong and Liu and suggested that in a car racing game, an Expert would finish a track in a minimum time using fewer actions such as the number of brakes,

turns, and crashes. Furthermore, they proposed that the goal of a learner is to imitate the activity patterns and achieve the performance of an Expert who is assumed to perform on the task presented without hesitation and with full attention and motivation.

Similarly, this thesis develops the Expert concept based on the findings of Hong and Liu and the indications about the flow state of an Expert proposed by Georgiou and Demiris. Let us assume that a student presents the best performance in a lecture, acquires the required knowledge through minimum activity, and spends minimum time to complete the lecture. This student can be considered to complete the lecture requirements without hesitation and distractions and with full attention and motivation. These qualifications indicate that the student is experiencing flow. This student is defined as the *Expert*.

The activities of all the other students in a lecture can be compared to the *Expert* student to obtain a relative value for the *Activity*. Based on the definitions of concepts, *Activity* and *Expert*, if the *Activity* value of a student is higher than the *Expert* student's *Activity* value, then that student can be considered to show higher activity than what is necessary for acquiring the required knowledge from the lecture content. Figure 3.5 presents the relation between the Expert and its relation to the proposed concepts.

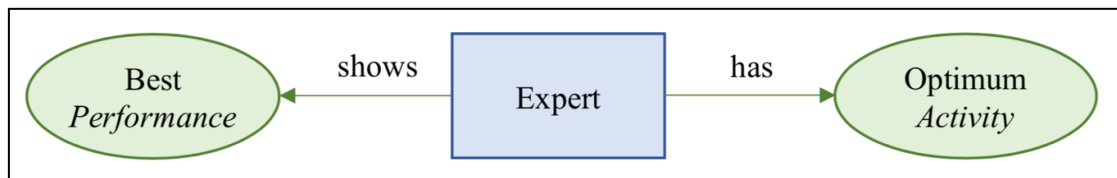


Figure 3.5. Relation between concepts and *Expert*

Although using *Activity* and *Performance* concepts instead of the challenge difficulty and the student skill is logical, this claim must be validated to prove the findings of the thesis. In the existing studies, the flow states are validated using three types of flow state identification techniques: interview questionnaires, experience sampling, and self-report questionnaires. In the interview questionnaires technique, the researchers conduct interviews where the participants are asked to answer predetermined questions in an open-ended manner and the researchers extract the flow states from the narrative explanations [19, 50]. This technique could also include observatory analysis where the researchers combine their observations about the emotional states of the subjects with the interviews to extract flow experiences [51]. In the experience sampling, the participants keep a diary

where the flow experiences are recorded as they happen [52, 53, 54]. The researchers then extract the flow experiences from the diaries manually. In the self-report questionnaires, the participants are asked to fill a survey where each question reflects a construct of flow state such as attention, concentration, skill-challenge difficulty balance, and loss of time [14, 15, 16, 17]. The majority of the existing researches on Flow Theory and education use a flow state scale to obtain flow experiences of students through self-report questionnaires [13]. Using the same technique the validation of the *Activity* and the *Performance* concepts can be performed. If these concepts present a correlation with the self-reports, it can be claimed that they reflect the concepts of Flow Theory. Table 3.1 presents a summary of the validation techniques used in the literature with example studies.

Table 3.1. Flow state validation techniques used in the literature

Validation Technique	Studies	Year	Subject	Reference
Interview Questionnaire	Kiili	2005	Educational Games	[19]
	Seifert and Hedderson	2009	Sports	[51]
Experience Sampling	Csikszentmihalyi et al.	1977	Adolescent Activity	[52]
	Jackson et al.	2008	Sports	[53]
	Magyaródi et al.	2013	Interactive Games	[54]
Self-report Questionnaire	Liu et al.	2009	E-learning	[14]
	Katuk and Ryu	2011	E-learning	[15]
	Van Schaik et al.	2012	Virtual Environment	[16]
	Hou	2015	Educational Game	[17]
	Erhel and Jamet	2019	Educational Game	[22]
	Özhan and Kocadere	2020	Gamified Learning Environment	[23]
	Ha and Im	2020	E-learning	[24]
	Verma et al.	2020	Educational Game	[27]
	Standen et al.	2020	Adaptive Learning System	[28]
	Current study	2020	E-learning	

Activity is proposed to measure student interactions and behaviors. This measurement is used to generate the flow state diagram which illustrates the emotional states of each student in a course. The emotional states are suggested to be directly related with a user's behaviors and physiological responses [55, 56, 57]. There are several studies that suggest the interaction patterns collected with mouse tracking can reflect an individual's behavioral patterns, personalities, and cognitive and emotional states. Hinbarji et al. proposed a user behavioral model by monitoring mouse patterns [58]. Khan et al. presented correlations between mouse usage and personality tests [59]. Azcarraga and Suarez suggested that an

unusual mouse interaction pattern implies the change of emotions [60]. Furthermore, several studies presented a strong relation between student performance and time spent actively on an online educational environment [61, 62]. Based on the findings of these studies, this thesis proposes to utilize mouse interaction patterns in order to measure the behavioral responses of students and environment access times to enhance the *Activity* measurement.

Performance is proposed to represent the acquired skills of the students. The acquired skill can be measured through educational assessment which is defined as the systematic process of collecting, analysing and documenting empirical data that reflects the skills, knowledge, or attitudes of individuals in order to improve the impact of the educational contents and student learning [63]. The educational assessment can be examined in four categories: placement, summative, diagnostic, and formative. The definitions of these categories of assessment have been provided by Madaus and Airasian in 1970 [64]. Placement assessment is the evaluation of the students prior to an educational context such as placing a student in different levels of a class according to a prior test. Summative assessment is used to evaluate a student's overall performance in a relatively long course or a project. This assessment technique is generally used for certification purposes. Diagnostic assessment evaluates the external factors in a student's learning. In other words, it is used to assess the difficulties students face outside the educational content provided by the instructors. Formative assessment is the evaluation of students in relatively small units of instruction. This technique provides quick feedbacks to instructors and the students about the level of learning. In the scope of e-learning, the use of formative assessments such as quizzes and multiple question exams is proven to be successful [65, 66, 67]. Considering these studies, this thesis proposes to measure *Performance* of students with formative assessment techniques such as quizzes, exams, reports, and exercises.

To define a learning evaluation model, the research introduces *Activity* and *Performance* concepts, an interactive e-learning environment, and a data processing policy. In the research, the participant students enroll in an online course provided through an e-learning environment. Then, the enrolled students participate in online lectures designed explicitly with online and interactive materials. The e-learning environment consists of an interactive tool that is designed based on a timeline and a map structure and classical e-learning web pages. The mouse interactions of the students are collected to generate *Activity* values,

while the grades obtained by the students from the exams produce the *Performance* values. Surveys that report the emotional states of the students are used to validate the *Activity* and *Performance* values. Finally, these values are used to draw flow state diagrams that present the learning states of the students in an online course. The general overview of the procedure is presented in Figure 3.6.

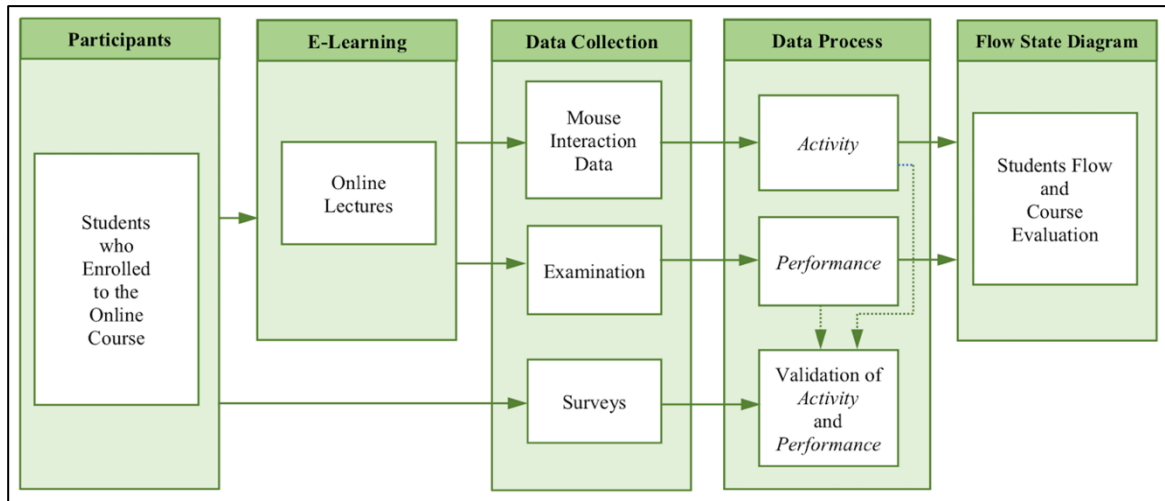


Figure 3.6. General overview of the research method

3.1. DATA COLLECTION

The e-learning platform developed in the scope of this research is equipped with a web analytics tool called Open Web Analytics [68] that provides raw statistics for web-based applications. These statistics are divided into two measures: time and screen coordinate. The time measures consist of access times of each element in a page and page view durations which can be either active page view duration or total page view duration. The coordinate measures are mouse click coordinates and the coordinates of locations visited with the mouse. The e-learning platform also stores the exam grades of all students for each lecture in each course.

3.1.1. Activity

Activity value for each student is generated based on their interactions with the e-learning environment through mouse actions. The data collected with mouse interactions are raw

information. The unprocessed data are converted into meaningful information through two methods: interaction statistics and activity heatmaps.

Interaction Statistics

The raw information is stored as log files with information about the coordinates of the mouse click and visit locations, how much time has passed since the student logged in to the environment, date and time of the log file, the interface the log is collected, and student user profile. These log files are inserted into a database every three seconds if there is any interaction from the student in the interface. If the student abandons the interface or there is no action taken with the mouse in three seconds there is no insertion. Four critical information can be measured using the log files in order to explain the activity of a student: total spent time (TT), time spent actively (TA), number of mouse clicks (MC), and mouse movement paths (MM). The short descriptions of these measurements are presented in Table 3.2.

Table 3.2. Measurements calculated based on interaction statistics

Measurement	Description
Total Spent Time (TT)	The total amount of time a student has spent after logging in
Time Spent Actively (TA)	The total amount of time a student has performed mouse actions
Number of Mouse Clicks (MC)	The number of mouse click actions taken by a student
Mouse Movement Paths (MM)	The number of locations visited through mouse by a student

Total Spent Time (TT) measure can be described as the total session duration of a student in a lecture from the time of login until the completion of the lecture requirements. This measure comprises of the time spent in the e-learning environment and in other applications that are not related to the lecture. Considering the time outside the environment is important since the time spent in other web applications while participating in a lecture is an indication of the student's lack of attention. TT is calculated with the equation 3.1 where d , for each log file, presents how much time has passed from the time a

student logged in to the environment until the time the log file is stored and n is the number of log files.

$$TT = \max(d_1, d_2, \dots, d_n) \quad (3.1)$$

Time Spent Actively (TA) can be described as the duration of performing actions in the e-learning environment. This measure presents how much time a student spends in the interface performing mouse movement and mouse click operations. When compared with the TT measure, if TA is closer to the TT value then the student can be considered to have high attention and high activity. On the other hand, if the TA is small related to the TT, then the student is not attentive to the lecture and less likely to be experiencing flow. TA is calculated using equation 3.2 since each log file is stored every three seconds while there is an activity with a mouse.

$$TA = \text{number of log files} \times 3 \text{ (sec)} \quad (3.2)$$

The number of Mouse Clicks (MC) and Mouse Movement Paths (MM) are obtained from the coordinate statistics. In each three-second log file, all the mouse operations performed by the student in that three seconds are stored. The stored information is the x,y coordinates and the type of operation, either click or movement. MC is the number of mouse click operations, while MM is the number of locations visited. If a location is clicked or visited more than once, that location is counted as much as it is clicked or visited since an object can be clicked more than once for different effects and a location can be visited in different directions of movements.

Activity Heatmaps

The interaction statistics, even though they are converted into meaningful measurements do not consider the coordinate information for mouse interactions. This information is worthy of being examined since the location of the mouse actions are affected directly from the place of information the students are interacting with. The coordinates of mouse actions can be investigated through heatmaps.

Initially, an image with the size of the interface needs to be created. All the cells of this initial image are considered to be empty. The empty image E is defined as:

$$E = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, a_{ij} = 0$$

where each a_{ij} represents the mouse action values in each location. Then, for each cell, the number of mouse movement and mouse click actions performed from the time of login until the time of lecture completion are counted. These values represent the density value for each cell. The density image D is defined as:

$$D = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, a_{ij} = click_{ij} + movement_{ij}$$

where $click_{ij}$ is the number of mouse clicks that occurred at the i, j location and $movement_{ij}$ is the number of times the mouse cursor has passed through the i, j location. In order to achieve a smooth transition between cells, these density values are distributed to their neighbors using weights calculated using Algorithm 3.1. The algorithm is for Gaussian weight distribution and the operation is called heat distribution.

Algorithm 3.1. Gaussian weight distribution algorithm

```

size ← 7
sigma ← 2
radius ← 3

for rows from 0 to size
  for cols from 0 to size
    r ← (row − radius) / sigma
    c ← (col − radius) / sigma

    rWeight ←  $e^{-(r \times r)/2}$ 
    cWeight ←  $e^{-(c \times c)/2}$ 

    MATRIX[row][col] ← rWeight × cWeight
  end for
end for

```

The result of the Gaussian weight distribution algorithm is a weight matrix, W which is defined as:

$$W = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{17} \\ w_{21} & w_{22} & \dots & w_{27} \\ \vdots & \vdots & \dots & \vdots \\ w_{71} & w_{m2} & \dots & w_{77} \end{bmatrix}, w_{ij} \in \mathbb{R}$$

This matrix is multiplied with the values in each cell to obtain temporary distributed matrices T_{ij} which are defined as:

$$T_{ij} = \begin{bmatrix} t_{11} & t_{12} & \dots & t_{17} \\ t_{21} & t_{22} & \dots & t_{27} \\ \vdots & \vdots & \dots & \vdots \\ t_{71} & t_{m2} & \dots & t_{77} \end{bmatrix} = a_{ij} \times W, a_{ij} \in D$$

Then each T_{ij} should be converted into a bigger matrix with the size of the empty image by placing the t_{33} value at the i, j cell and the other t values to their corresponding cells. The remaining cells should be 0. In order to generate this bigger matrix, two empty matrices E_x and E_y are defined as:

$$E_x = \begin{bmatrix} a_{11} & a_{12} & \dots & x_{1t} \\ a_{21} & a_{22} & \dots & a_{2t} \\ \vdots & \vdots & \dots & \vdots \\ a_{71} & a_{72} & \dots & a_{7t} \end{bmatrix}, E_y = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \dots & \vdots \\ a_{k1} & a_{k2} & \dots & a_{kn} \end{bmatrix}, t = n - 7, k = m - 7, a_{ij} = 0$$

The temporary distribution matrix T_{ij} is reshaped by concatenating it with E_x and E_y to obtain the resized representation R_{ij} as follows:

$$C_{ij} = [T_{ij} \quad E_x]_{7 \times n}$$

$$R_{ij} = \begin{bmatrix} C_{ij} \\ E_y \end{bmatrix}_{m \times n}$$

Then, the values in the resized matrix R_{ij} are shifted to obtain the heat distribution matrix G_{ij} . The shift operation is performed by the upper shift matrix U and the lower shift matrix L . If a matrix A is premultiplied with U , the values in A are shifted one position upward and the bottom cells become 0. On the other hand, if A is postmultiplied with U , the values in A are shifted one position right and the left most cells become 0. The direction of the shift is inverted when the same multiplication operations are performed with lower shift matrix. U and L are defined as:

$$U = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}, L = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$

The number of shift operations and the direction of the shift are determined by subtracting the i and j values from 3 since the center of the weight matrix is at the cell 3, 3. If $3 - i > 0$ the matrix is shifted left, if $3 - i < 0$ the matrix is shifted right, and otherwise there is no shift operation on the x-axis. If $3 - j > 0$ the matrix is shifted upward, if $3 - j < 0$ the matrix is shifted downward, and otherwise there is no shift operation on the y-axis. The final image is obtained by the summation of all heat distribution matrices G_{ij} .

After the heat distribution, according to the maximum intensity value, all values are normalized to be between 0 and 1. Then, these normalized values are used to generate colors using equation 3.3 where h is the Hue of the color. The saturation, s , and Value, v , are chosen to be 1. As a result of this operation, an HSV (Hue-Saturation-Value) representation of the heatmap image is created.

$$h = (1.0 - \text{cell value}) \times 240 \quad (3.3)$$

From the final HSV heatmap image representation, three types of heatmaps are created: Colored, Grayscale and Weighted Scale. The colored heatmap is generated as an RGB representation of the HSV heatmap. The grayscale heatmap is obtained from the RGB image using equation 3.4 where R is the value in the R channel of the image, G is from the green channel and B is from the blue channel [69].

$$I = 0.299R + 0.587G + 0.114B \quad (3.4)$$

The weighted scale heatmap is another grayscale heatmap where the weights of the color channel information are chosen to reflect the importance of channels in terms of conversion. The weights of R , G , and B in the equation are defined with this research. The weight of red is 3, green is 2 and blue is 1. In order to have a uniform weight distribution, the weights are normalized using the lowest common denominator, 6. The final equation obtained through this process is presented in equation 3.5.

$$I = 0.500R + 0.330G + 0.170B \quad (3.5)$$

3.1.2. Performance

Performance value for each student depends on the grades they obtained from exams of each lecture. The performance of a student in a lecture can be measured with one or more examination methods such as quizzes, exams, reports, or exercises. The overall performance of a student in a lecture is determined by considering all the examinations the student is subjected to. The e-learning platform developed for the scope of this research can store test scores, exam grades, and report and exercise points. These grades are converted into scaled values by normalizing them with the maximum grade that can be obtained by a student from the examination method.

3.2. DATA PROCESSING

The data collected for *Activity* and *Performance* are converted into meaningful measurements such as activity heatmaps and interaction statistics, to eliminate redundant information. These measurements need to be further processed to generate single values for each concept. This thesis proposes methods to convert the calculated measures which are the high-level representations of the data into vectors and finally, single values for *Activity* and *Performance*.

3.2.1. Activity

The *Activity* vector for a student is generated based on a set of features. While generating the vector for a student, choosing appropriate features that are capable of representing the interaction of the student best is not a trivial task. To tackle this problem, this thesis proposes methods that can be divided into three categories based on which high-level representation the method uses and which technique it utilizes. The vector can be obtained using the interaction statistics, with Convolutional Autoencoders using activity heatmaps, and with other dimension reduction techniques that use the same heatmaps.

Activity Vector from Interaction Statistics

The log data, collected from the interaction of a student is converted into four measurements: total spent time (TT), time spent actively (TA), number of mouse clicks (MC), and mouse movement paths (MM). These measurements are defined to represent the interaction statistics of a student in an e-learning platform. The e-learning platform developed for this thesis contains two separate interfaces: timeline tool and e-learning web pages. Since these interfaces provide distinct information about the interaction of a student, calculating the measurements individually for each interface would provide a better representation of the student's interaction.

A student who spent a great amount of time in a lecture could present similar activity to a student who finished the lecture quickly. To analyze whether eliminating the total amount of time spent in a lecture would provide a better representation or not, this thesis proposes two methods that utilize the statistical measurements: using the measurements as they are and using the measurements normalized with total spent time.

In the interaction statistics only method, the activity vector of a student a_k is defined as:

$$a_k = \{f_{tt}^T, f_{ta}^T, f_{mc}^T, f_{mm}^T, f_{tt}^E, f_{ta}^E, f_{mc}^E, f_{mm}^E\}$$

where the first four features ($f_{measure}^T$) are the measurements obtained from the timeline tool and the last four features ($f_{measure}^E$) are from the e-learning web pages. The features $f_{tt}^{interface}$, $f_{ta}^{interface}$, $f_{mc}^{interface}$, and $f_{mm}^{interface}$ correspond to total spent time (tt), time spent actively (ta), number of mouse clicks (mc), and mouse movement paths (mm) respectively.

In the normalized interaction statistics method, the activity vector of a student a_k is defined as:

$$a_k = \{f_{ta}^T, f_{mc}^T, f_{mm}^T, f_{ta}^E, f_{mc}^E, f_{mm}^E\}$$

where the first three features ($f_{measure}^T$) are the measurements obtained from the timeline tool and the last three features ($f_{measure}^E$) are from the e-learning web pages. Each feature is normalized with the total spent time measurement obtained in the corresponding interface.

Activity Vector from Heatmaps

The interaction statistics do not consider the locations of the mouse interactions and only focus on time and the number of actions taken with the mouse. The spatial information for the mouse is a factor worthy of being examined. To create a model that considers the spatial information, this thesis proposes to generate activity heatmaps. The heatmaps are created in three color schemes: rainbow color, grayscale, and weighted scale. The e-learning environment developed for this thesis has two distinct interfaces. Since the locations in these interfaces differ based on the content they provide, generating heatmaps individually for each interface would provide a better representation of the student's interaction. To investigate this claim, a heatmap from each interface and a singular heatmap that contains all the locations of the interfaces are created in all three color schemes. The size of each rainbow color heatmap is $24 \times 32 \times 3$ since this type of heatmap has three color channels, R, G, and B. On the other hand, the grayscale and weighted scale heatmaps have a size of $24 \times 32 \times 1$ because they represent a heatmap image with a singular channel.

The critical behavior patterns in the activity heatmaps need to be extracted to generate an *Activity* vector for each student. For this task, this thesis proposes to use Convolutional Autoencoder (CAE) models. The reasoning behind this proposition rests on the recent successes in feature extraction and finding local patterns of multidimensional data presented with CAEs. Nishio et al. proposed a CAE-based approach for image denoising in computerized tomography [70]. The method outperforms the fully connected denoising AEs. Turchenko et al. presented a CAE model that performs better than classical AEs in dimension reduction using MNIST [71] dataset of handwritten digits [72]. Seyfioğlu et al. also showed that using features obtained from CAE networks is better at classifying radar-based activities when compared to fully connected AEs [73]. The data used in these studies heavily rely on local patterns. Similarly, the local patterns of mouse actions in heatmaps are required to be extracted for the student activity. Hence, it is worthy to consider exploiting the power of these models.

CAE Model using Colored Heatmaps: The proposed model for rainbow color heatmaps comprises an encoder and a decoder. The encoder is composed of three convolutional layers that have filters with a size of 3×3 and three max-pooling layers where the

maximum value in a 2×2 window is taken to represent each window. Each convolutional layer is followed by a max-pooling layer. Hence, after every convolution-pooling pair, the size of the input image is decreased to its half. The number of filters in each convolutional layer are 16, 8, and 4 respectively. The decoder contains three convolutional layers with 3×3 sized filters. The number of filters in each convolutional layer are 4, 8, and 16. After every convolutional layer, the obtained values are passed to an up-sampling layer in which the uniform sampling with a size of 2×2 is performed. The size of the input doubles with each convolution-up-sampling pair. The size of the input and output layers is $24 \times 32 \times 3$ and the model aims to reproduce the input image in the output layer. The output of the encoder is considered to be the abstract representation of the input heatmaps and this layer is called the *Features* layer. The *Features* layer which has the size of $3 \times 4 \times 4$ is converted into a single vector that contains 48 values that represent the interaction patterns of a student. Figure 3.7 presents the architecture of this model with detailed information about the output sizes of each layer.

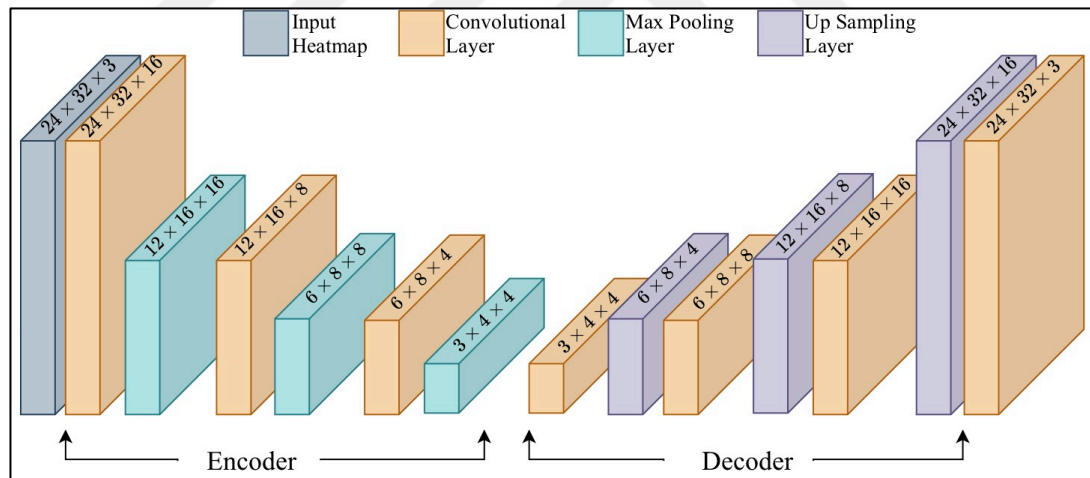


Figure 3.7. Convolutional Autoencoder model architecture for colored heatmaps

The proposed model is empirically found to be the best performing model in terms of loss obtained in the reproduction process. The model structure is obtained after an investigation with several other CAE models with different configurations. The parameters of these different configurations are selected to investigate the effects of the number of layers, filter size, the number of filters, and the pooling method. Table 3.3 presents the different parameters for the investigated configurations. The models are trained individually with the color heatmaps obtained from each interface of the e-learning environment and the color heatmap obtained considering the environment as a single interface.

Table 3.3. The configurations for Convolutional Autoencoder model selection

Model*	Layers	Number of Filters		Filter Size	Pooling
		Encoder	Decoder		
L1-E4D4-F3-MP	1 Layer	4	4	3×3	Max
L1-E4D4-F3-AP					Average
L1-E4D4-F5-MP				5×5	Max
L1-E4D4-F5-AP					Average
L1-E8D8-F3-MP		8	8	3×3	Max
L1-E8D8-F3-AP					Average
L1-E8D8-F5-MP				5×5	Max
L1-E8D8-F5-AP					Average
L1-E16D16-F3-MP		16	16	3×3	Max
L1-E16D16-F3-AP					Average
L1-E16D16-F5-MP				5×5	Max
L1-E16D16-F5-AP					Average
L2-E4D4-F3-MP	2 Layers	4-4	4-4	3×3	Max
L2-E4D4-F3-AP					Average
L2-E4D4-F5-MP				5×5	Max
L2-E4D4-F5-AP					Average
L2-E8D8-F3-MP		8-8	8-8	3×3	Max
L2-E8D8-F3-AP					Average
L2-E8D8-F5-MP				5×5	Max
L2-E8D8-F5-AP					Average
L2-E16D16-F3-MP		16-16	16-16	3×3	Max
L2-E16D16-F3-AP					Average
L2-E16D16-F5-MP				5×5	Max
L2-E16D16-F5-AP					Average
L2-E84D48-F3-MP		8-4	4-8	3×3	Max
L2-E84D48-F3-AP					Average
L2-E84D48-F5-MP				5×5	Max
L2-E84D48-F5-AP					Average
L2-E168D816-F3-MP		16-8	8-16	3×3	Max
L2-E168D816-F3-AP					Average
L2-E168D816-F5-MP				5×5	Max
L2-E168D816-F5-AP					Average
L3-E4D4-F3-MP	3 Layers	4-4-4	4-4-4	3×3	Max
L3-E4D4-F3-AP					Average
L3-E4D4-F5-MP				5×5	Max
L3-E4D4-F5-AP					Average
L3-E8D8-F3-MP		8-8-8	8-8-8	3×3	Max
L3-E8D8-F3-AP					Average
L3-E8D8-F5-MP				5×5	Max
L3-E8D8-F5-AP					Average
L3-E16D16-F3-MP		16-16-16	16-16-16	3×3	Max
L3-E16D16-F3-AP					Average
L3-E16D16-F5-MP				5×5	Max
L3-E16D16-F5-AP					Average
L3-E1684D4816-F3-MP**		16-8-4	4-8-16	3×3	Max
L3-E1684D4816-F3-AP					Average
L3-E1684D4816-F5-MP				5×5	Max
L3-E1684D4816-F5-AP					Average
* The models are named according to the parameters. L denotes the number of layers, E and D denote the number of filters in the encoder and decoder respectively, F denotes the filter size, and MP and AP denote Maximum and Average pooling respectively.					
** Proposed Model					

CAE Model using Grayscale and Weighted Scale Heatmaps: The proposed model for grayscale and weighted scale heatmaps also comprises an encoder and a decoder. The encoder is composed of three convolutional layers that have filters with a size of 3×3 and three max-pooling layers where the window size is 2×2 . Each convolutional layer is followed by a max-pooling layer to reduce the size of the input to its half. The number of filters in each convolutional layer are 16, 8, and 4 respectively. The decoder contains three convolutional layers with 3×3 sized filters. The number of filters in each convolutional layer are 4, 8, and 16. Every convolutional layer is followed by an up-sampling layer which performs 2×2 uniform sampling to double the size of the input. The size of the input and output layers is $24 \times 32 \times 1$ and the model aims to reproduce the input image in the output layer. The *Features* layer has a size of $3 \times 4 \times 4$ and it is converted into a single vector which contains 48 values that represent the interaction patterns of a student. Figure 3.8 presents the architecture of this model with detailed information about the output sizes of each layer.

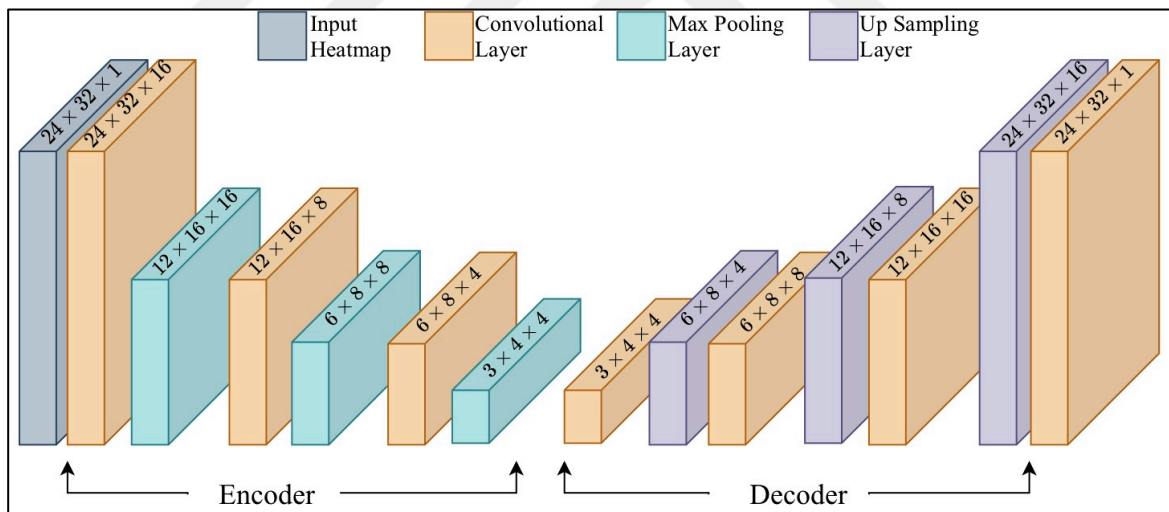


Figure 3.8. Convolutional Autoencoder model architecture for grayscale and weighted scale heatmaps

Other Dimenson Reduction Methods: The proposed Convolutional Autoencoder based method acts as a dimension reduction method. To assess whether the proposed method presents a contribution to the literature, the interaction patterns from activity heatmaps are also extracted with several dimension reduction methods. These methods are Principal Component Analysis, Factor Analysis, Multidimensional Scaling, Independent Component Analysis, Gaussian Random Projection, Isometric Feature Mapping, and t-distributed

Stochastic Neighbor Embedding. The numbers of reduced dimensions are empirically chosen based on the requirements of the method, the size of the heatmaps, and the number of data available for a dataset. For each method, *Activity* vectors using all four types of activity heatmaps with different number of reduced dimensions are created. Table 3.4 presents the number of selected dimensions for each vector.

For each dimension reduction method, 2 or 3 vectors are created with different dimensions. In Principal Component Analysis, the data are reduced to 20 dimensions for the first vector, because more than 95 percent of the study data can be expressed with 20 dimensions according to the cumulative explained variance method. A second vector is created with 32 dimensions where 97 percent of the data can be expressed. For this reason, a third vector was not required. In t-distributed Stochastic Neighbor Embedding, only two vectors were utilized with 2 and 3 dimensions since the method is strictly used to reduce the dimensions for data visualization. For all other methods, three vectors were created with a maximum, minimum, and average number of dimensions in order to evaluate the performance of different dimension ranges.

Table 3.4. Number of reduced dimensions chosen for each method

Method	Reduced Dimensions		
	Vector 1	Vector 2	Vector 3
Principal Component Analysis	20	32	Not utilized
t-distributed Stochastic Neighbor Embedding	2	3	Not utilized
Factor Analysis	2	20	32
Multi-dimensional Scaling	2	20	32
Independent Component Analysis	2	20	32
Gaussian Random Projection	2	20	32
Isometric Feature Mapping	2	10	16

Single Activity Value Through Expert Similarity

To produce a single *Activity* value for each student that is grounded by an optimal value, this thesis proposes the concept of an *Expert* which was inspired by the concept introduced by Georgiou and Demiris. The Expert student presents maximum engagement, best performance, and the most effective time in a lecture. The most effective time is defined as having the minimum total amount of time spent in the environment after the login until the end of the lecture while having the maximum total amount of time performing actions through mouse and performing required operations in a lecture with minimum mouse actions. Although it is a rare case since one of the constraints of expert selection criteria is time, more than one student would present the same values in a lecture. In this case, one of these students is selected randomly as the Expert. The algorithm to find the *Expert* in a lecture is presented in Algorithm 3.2.

Algorithm 3.2. Expert selection algorithm

```

for all students do
  if student has maximum grade then
    ADD student to expert candidates
  end if
end for
for all expert candidates do
  if student has more than minimum total time then
    REMOVE student from expert candidates
  end if
  if student has less than maximum active time then
    REMOVE student from expert candidates
  end if
  if student has more than minimum mouse activity then
    REMOVE student from expert candidates
  end if
end for
if only 1 student in expert candidates then
  Expert ← expert candidate
else
  Expert ← random(expert candidates)
end if

```

Since the *Expert* student expresses the optimal behavior in a lecture with maximum performance, the comparison of all students with this student would provide a relative

measurement for *Activity*. If the *Activity* vector of a student is similar to the *Expert*'s, that student can be considered to show optimal activity. Therefore, calculating the similarity between two vectors would provide the desired relative *Activity* value. To perform this comparison, cosine similarity which is a measure of similarity between non-zero vectors is used. The cosine similarity is calculated using the Euclidean dot product operation. Hence, the *Activity* value of a student A_k is calculated using equation 3.6 where a_k is the activity vector of the student, a_E is the vector of the *Expert*, n is the number of features, and each f_k and f_E correspond to a feature in the *Activity* vector of the student and the expert respectively.

$$A_k = a_k \cdot a_E = \sum_{i=0}^n f_{k_i} \times f_{E_i} \quad (3.6)$$

The *Activity* values are generated through *Expert* similarity with interaction statistics for each student using two methods: interaction statistics only and normalized interaction statistics. First, the data are collected from both interfaces of the developed e-learning environment separately. Then, the data are converted into *Activity* vectors for each student. The *Expert* student is determined based on the provided algorithm. Finally, the similarity between the vector of each student with the vector of the *Expert* student determines the *Activity* value for each method. Figure 3.9 presents the *Activity* generation using proposed statistical methods where A_{ISO} is the *Activity* value with interaction statistics only and A_{NIS} is the *Activity* value with normalized interaction statistics.

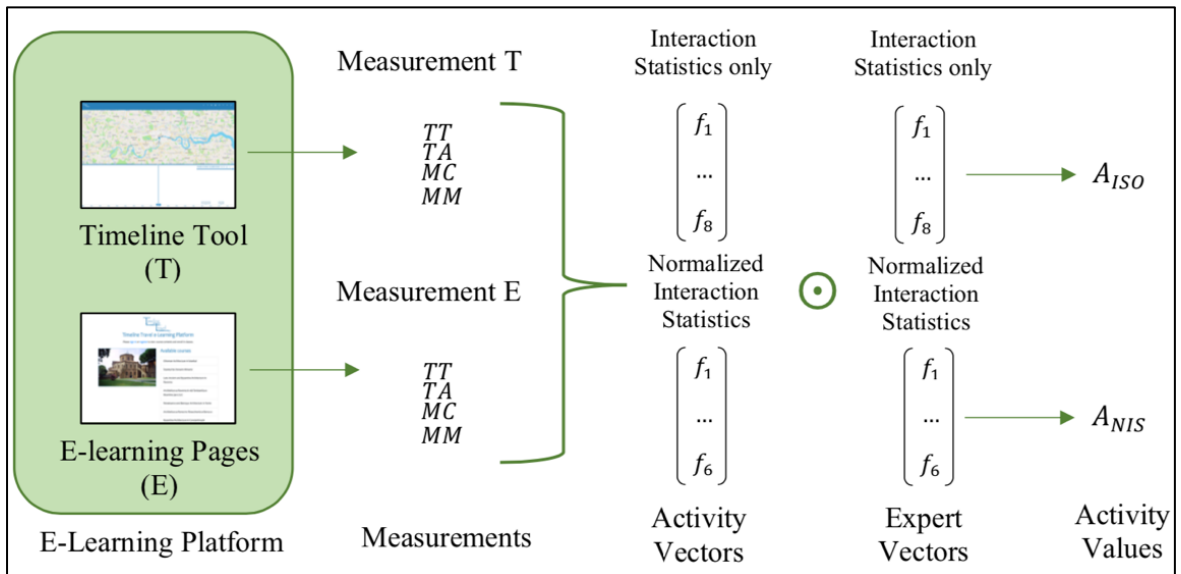


Figure 3.9. *Activity* value generation through interaction statistics

The Activity value generation with heatmaps consists of different types of heatmaps and several dimension reduction methods. Two rainbow color heatmaps are generated from each interface separately and a single color heatmap is generated from the interface using all the information from both interfaces. Then these heatmaps are converted into grayscale heatmaps and weighted scale heatmaps. The grayscale heatmaps are created using equation 3.4 and the weighted scale heatmaps are generated using equation 3.5. Figure 3.10 illustrates the procedure to generate these heatmaps where color heatmaps, grayscale heatmaps, and weighted heatmaps are represented as $Color - H_{interface}$, $Gray - H_{interface}$, and $Weighted - H_{interface}$ respectively, H_T is the heatmap from Timeline Tool, H_E is the heatmap from e-learning web pages, and H_{Total} is the heatmap generated using the information from both interfaces.

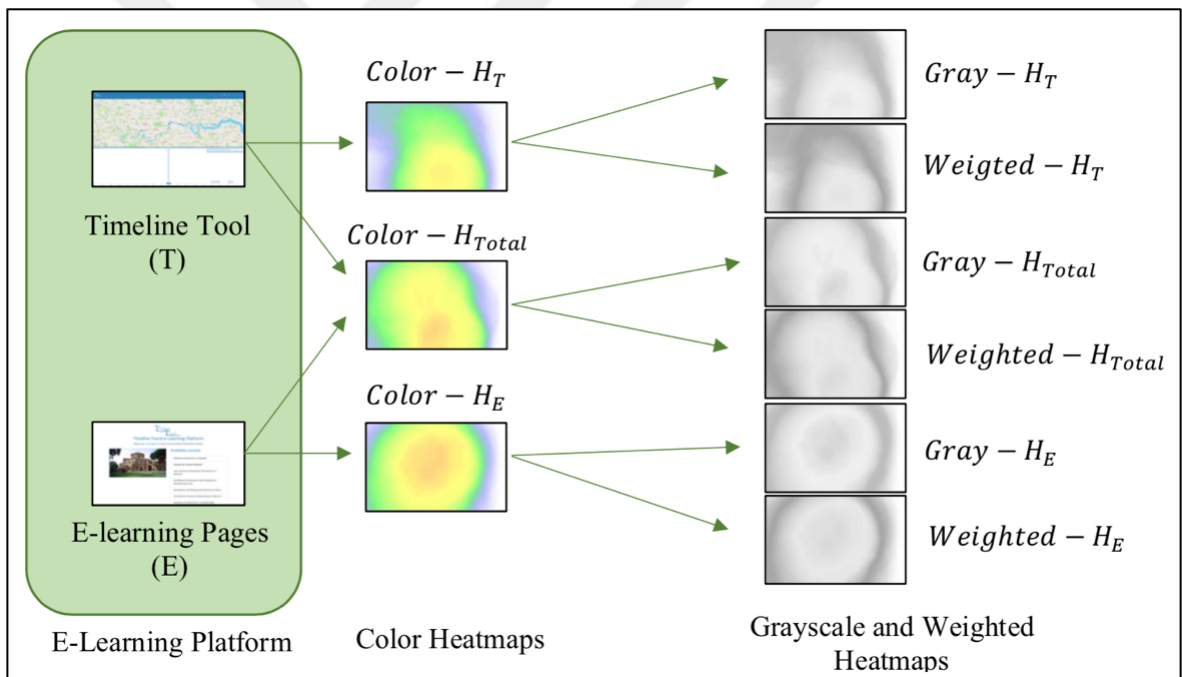


Figure 3.10. Heatmap generation procedure

The Expert similarity procedure is categorized into two groups: concatenated activity generation and total activity generation. In concatenated activity generation, $Color - H_T - Color - H_E$, $Gray - H_T - Gray - H_E$, and $Weighted - H_T - Weighted - H_E$ pairs are processed together to obtain a single Activity vector for the interaction of the student from the e-learning environment. Each heatmap is converted into an *Activity* vector using either the appropriate proposed CAE method or a dimension reduction method where the number of features in each vector, n is determined by the method utilized. Then, to obtain a single

Activity vector for each student, these vectors are appended onto one another and the number of features in the final vector becomes $2n$. Hence, the procedure is named concatenated activity generation. Finally, the single *Activity* value is obtained using the similarity measure between the activity vectors of each student and the vector of the *Expert*. Figure 3.11 illustrates this procedure where H_T is the heatmap obtained from the Timeline Tool, H_E is the heatmap from e-learning web pages, and A represents the single *Activity* value. The *Activity* value depends on the heatmap color scheme and the method utilized.

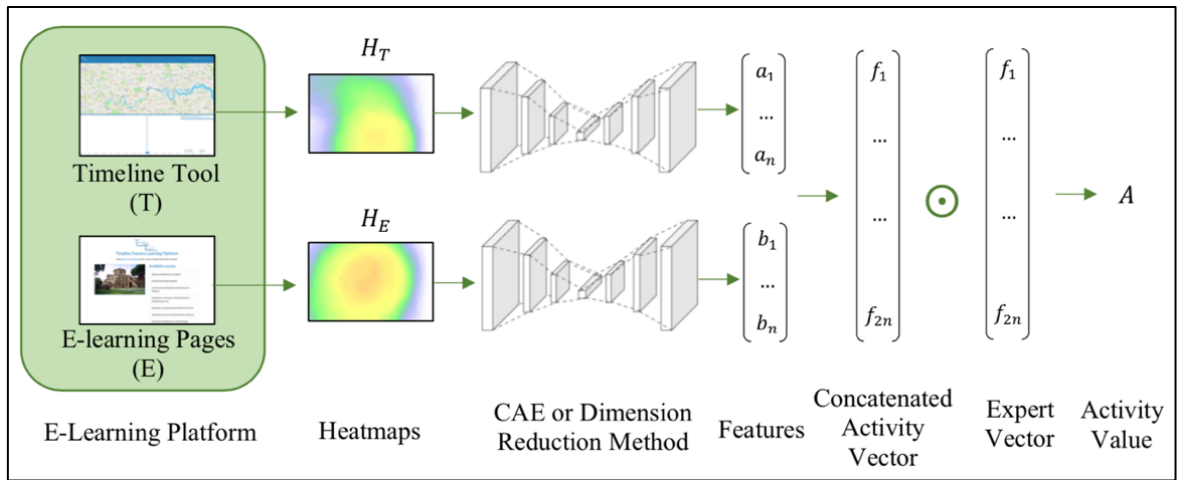


Figure 3.11. *Activity* value generation using the concatenation procedure

In total activity generation, *Color* – H_{Total} , *Gray* – H_{Total} , and *Weighted* – H_{Total} heatmaps are converted into an *Activity* vector using either the proposed CAE method or a dimension reduction method where the number of features in each vector, n is determined by the method utilized. Then, the single *Activity* value is obtained using the similarity measure between the activity vectors of each student and the vector of the *Expert*. Figure 3.12 presents this procedure where H_{Total} is the heatmap obtained from the e-learning environment as if it had a single interface and A represents the single *Activity* value. The *Activity* value depends on the heatmap color scheme and the method utilized.

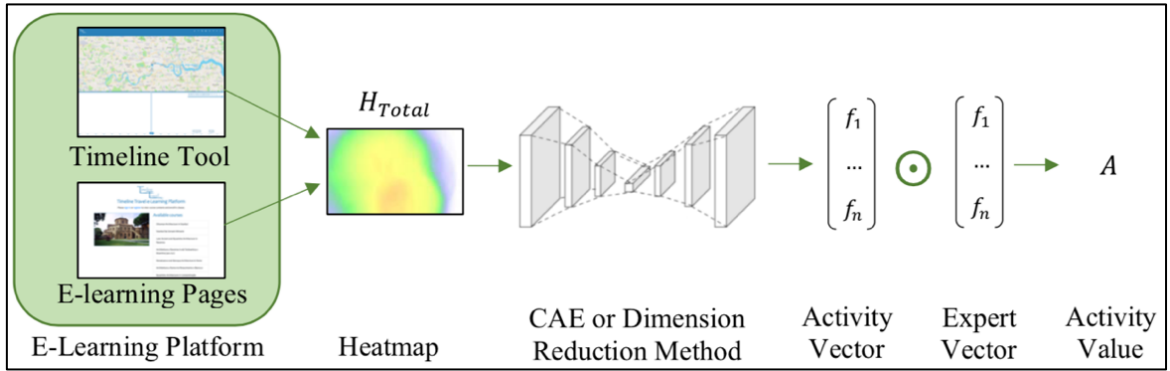


Figure 3.12. Activity value generation using total heatmaps

3.2.2. Performance

Performance value is a straightforward measure that is calculated using the grades of the students. The grades can be from quizzes, exams, reports, or assignments. The *Performance* value of a student in a lecture is calculated using equation 3.7 where g_i presents the normalized grade from an exam and N is the number of exams used to evaluate the performance of a student in that lecture.

$$Performance = \frac{\sum_{i=1}^N g_i}{N} \quad (3.7)$$

3.3. THE E-LEARNING ENVIRONMENT

The platform has two types of actors: *Teachers* and *Students*. *Teachers* are the instructors who have the capability of creating courses, exams, and timelines, evaluating the performance and progress of their students, and altering their course content through the evaluation of the students. On the other hand, the *Students* are the users of the platform who can enroll in the available courses, take exams, observe their performance and progress in a course, and compare their performance with other students who are in the same courses the student is enrolled in. Table 3.5 presents all the actions the actors can take on the platform.

Table 3.5. The actions that can be performed by the actors of the platform

Actor	Feature	Action
Teacher	Content Management	1. Create/Modify/Delete Course
		2. Add/Modify/Delete Lecture
		3. Add/Modify/Delete Electronic Document
		4. Create/Modify/Delete Timeline
		5. Add/Modify/Delete Timeline Content
		6. Add/Modify/Delete Timeline Connection
		7. Add/Modify/Delete Video Content
		8. Create/Modify/Delete Exam
	Evaluation	1. List Enrolled Students in a Course
		2. View Student Performance and Progress
		3. View Evaluation for a Course
Student	Course Enrollment	1. Search Courses
		2. Enroll in/Leave a Course
		3. Interact with the Content of an Enrolled Course
		4. Take an Exam
	Performance Evaluation	1. View Performance and Progress in a Course
		2. View Performance Comparison with other Students

The platform has two user interfaces, a classical e-learning interface and an interactive tool interface. The e-learning interface is considered to be a classical e-learning webpage that contains materials such as videos and electrical documents. On the other hand, the interactive tool comprises of a map and a timeline in order to provide a tool that can present spatial and chronological information at the same time. The interactions generated by the users of these interfaces are stored in a single data storage that is managed by a data collection tool.

In the timeline tool interface, a *Timeline* contains a set of *Items* which consist of coordinates and textual and time-based information. Furthermore, an *Item* can have a set of *Images* that are defined by captions. The main element in the e-learning web pages interface is a *Course* that consists of a set of *Lectures*. Each *Lecture* contains various amounts of *Content* which could be an electronic document, a video link, an external web

for specific events and to filter the content according to some predefined criteria. The infoboxes present detailed information about the events and the animation module allows the changes in the map and timeline to be animated in a period of time. Figure 3.14 illustrates the actions for a *Student* in the timeline tool interface.

The timeline tool is quite essential in extracting the interactions of a student since it ensures that the student would need to interact with the platform in order to obtain knowledge. The students can manipulate the timeline in order to see information in a specific time, open infoboxes for detailed information, and explore the map to see the surrounding events happening in the same area. Hence, the students would generate behavioral patterns through their mouse activity.

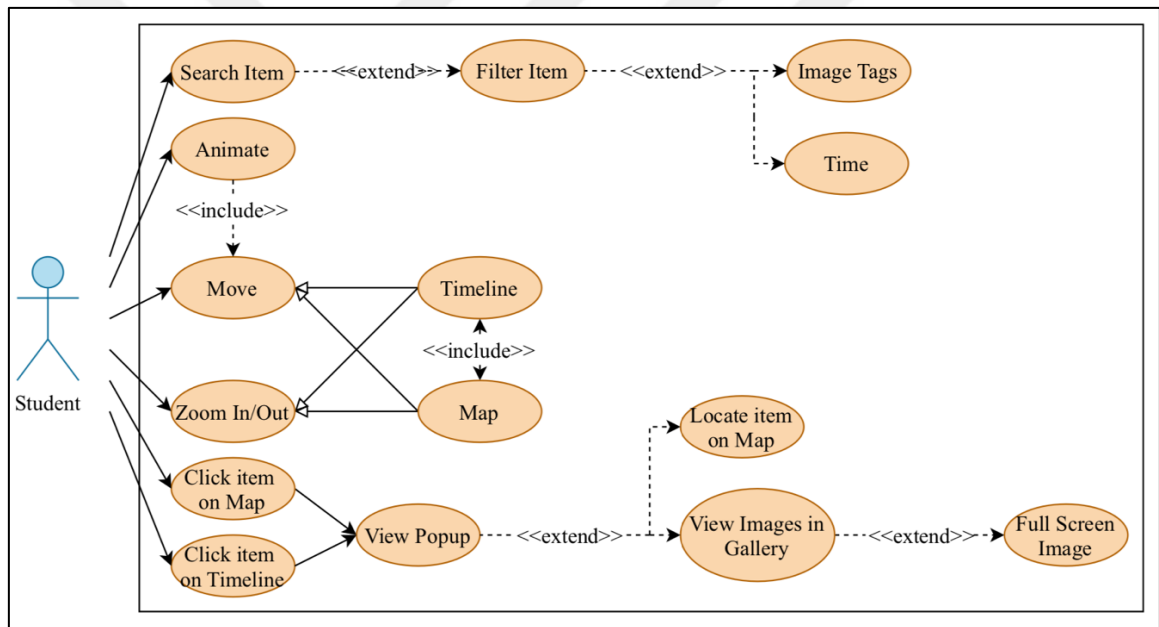
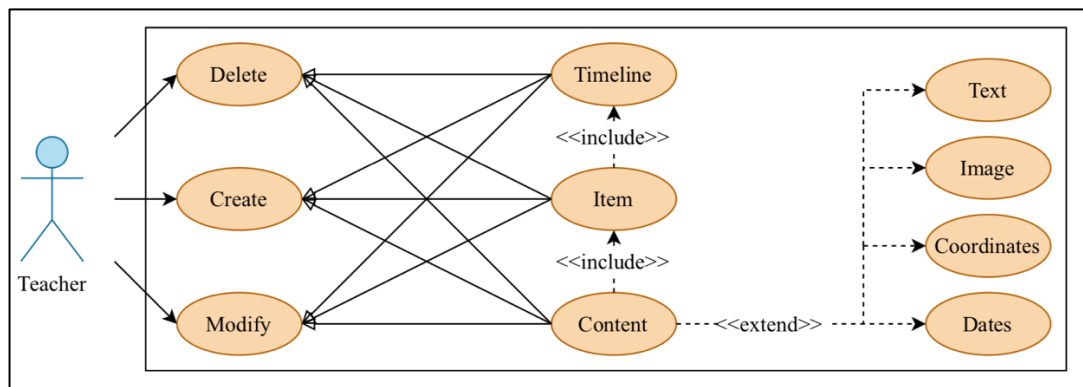


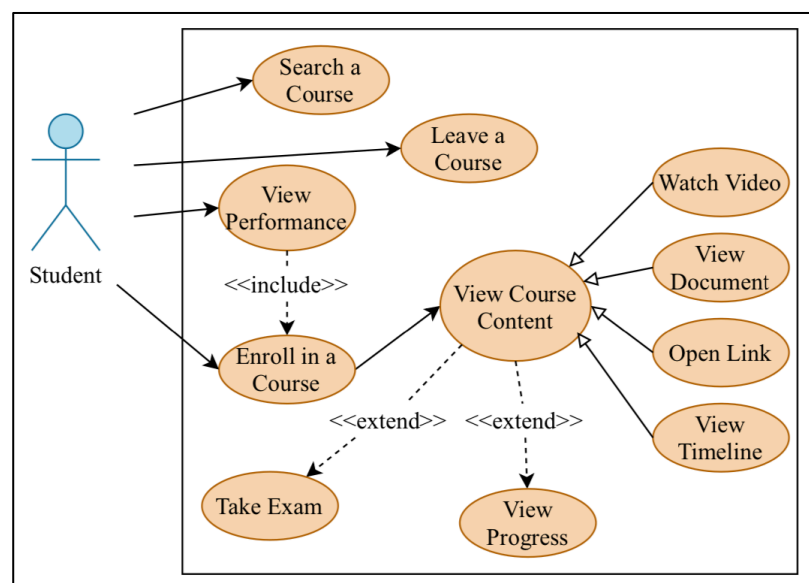
Figure 3.14. Timeline Tool interface for *Students*

A content management system (CMS) is needed to generate and modify the content of the timeline tool. The CMS interface allows *Teachers* to create timelines that store the spatial, textual, and chronological information in data storage and provide the necessary conversions that allow the information to be presented in the timeline tool. In this interface, *Teachers* can create, modify, and delete timelines, in a timeline add, modify, and delete items, and for each item add, modify, and delete textual information and images. Figure 3.15 presents the relations between the actions taken by the *Teachers* and the CMS interface of the timeline tool.

Figure 3.15. Timeline Tool interface for *Teachers*

3.3.2. E-learning Web Pages

For the classical e-learning interface, four types of lecture materials can be created: electronic documents, videos, timeline links, and quizzes. The electronic documents contain lecture notes, slides, or any textual or visual information that can be presented as a document. The videos are embedded links from online video sharing platforms and allow *Students* to watch them inside the platform. The quizzes, exams, or exercise tasks provide means to measure the performances or the signs of progress of the *Students* in a lecture. The e-learning interface also provides direct links to the timeline tool interface for the course's timeline content. Figure 3.16 illustrates the actions for a *Student* in the e-learning web pages interface.

Figure 3.16. E-learning Web Pages interface for *Students*

A CMS is also needed to generate and modify the content of the e-learning web pages. In this interface, *Teachers* can create, modify, and delete a course, in a course add, modify, and delete lectures, and for each lecture add, modify, and delete course contents such as electronic documents, video or timeline links, and exams. In each exam, a Teacher can add as many questions as possible and for each question add as many options as desired. Furthermore, a *Teacher* can observe the signs progress and the performances of the *Students* who are enrolled in their courses. Figure 3.17 presents the relations between the actions taken by the *Teachers* and the CMS for E-learning Web Pages interface.

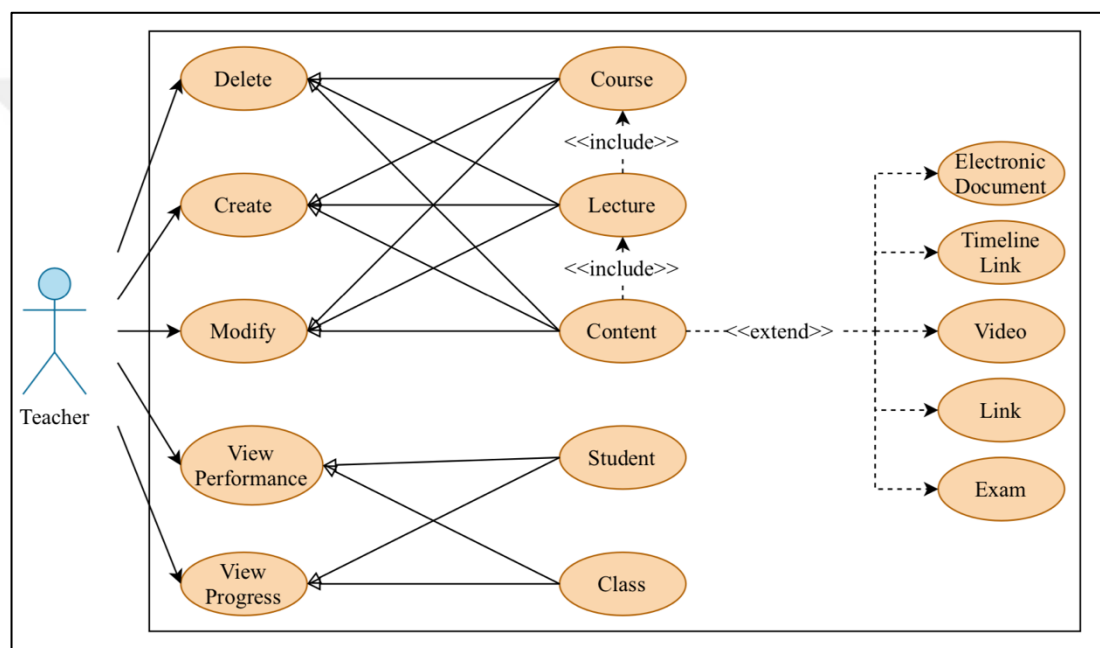


Figure 3.17. E-learning Web Pages interface for *Teachers*

3.3.3. Data Collection Tool

There are two data entities in the data collection tool: *Click* and *Stream*. *Click* stores the time and coordinate information about mouse click actions while *Stream* collects the equivalent information about the mouse movements. If a student, who is a *User* of the system, clicks on a location with the mouse, the screen coordinates of that location, the screen size at the time the action occurred, the exact time of the action, the interface that this action is performed, and how much time has passed since the start of the *User*'s session until this action is stored in the *Click* data entity. On the other hand, a *Log* file is created at *Stream* entity every three seconds where how much time has passed since the

start of the session until the time the *Log* file is recorded, the interface of the movements, and the screen resolution at the time of the entry. Each *Log* contains the exact time and screen coordinates of all locations visited by the *User*'s mouse movements. The relations between the data entities are presented in Figure 3.18.

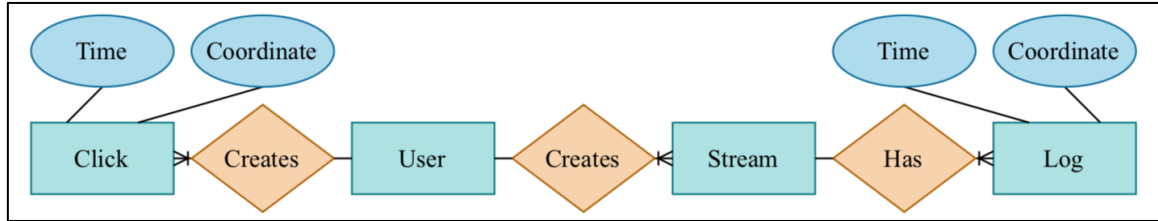


Figure 3.18. Entity-relationship diagram of the raw statistics collection tool

3.4. DATASET PREPARATION

The *Activity* dataset for analysis is prepared using the procedure illustrated in Figure 3.19. First, the number of students is determined for each course and a list of all students is obtained. Then, for each student in the course, the raw statistics are converted into high-level interaction metrics and heatmaps. The heatmaps are converted into appropriate structures for corresponding methods. These structures are used in the training procedure for CAE models and other dimension reduction methods. Then, using all trained models the feature vectors are obtained for each student for each model. The interaction statistics metrics are considered as a feature set for student activity. Hence, the feature vector obtained with these metrics is also included in the feature vectors list.

After the feature generation for each student is finished, the *Expert* of the course is determined using Algorithm 3.2. The *Expert* student is removed from the list of all students since that student will be omitted from the analysis and the *Activity* value for each student will be calculated based on their similarity with this student. Then, using expert similarity with dot product operation, an *Activity* value for each feature vector in the features list is obtained. Finally, all the sets of single *Activity* values for each method are exported to be used in analyses.

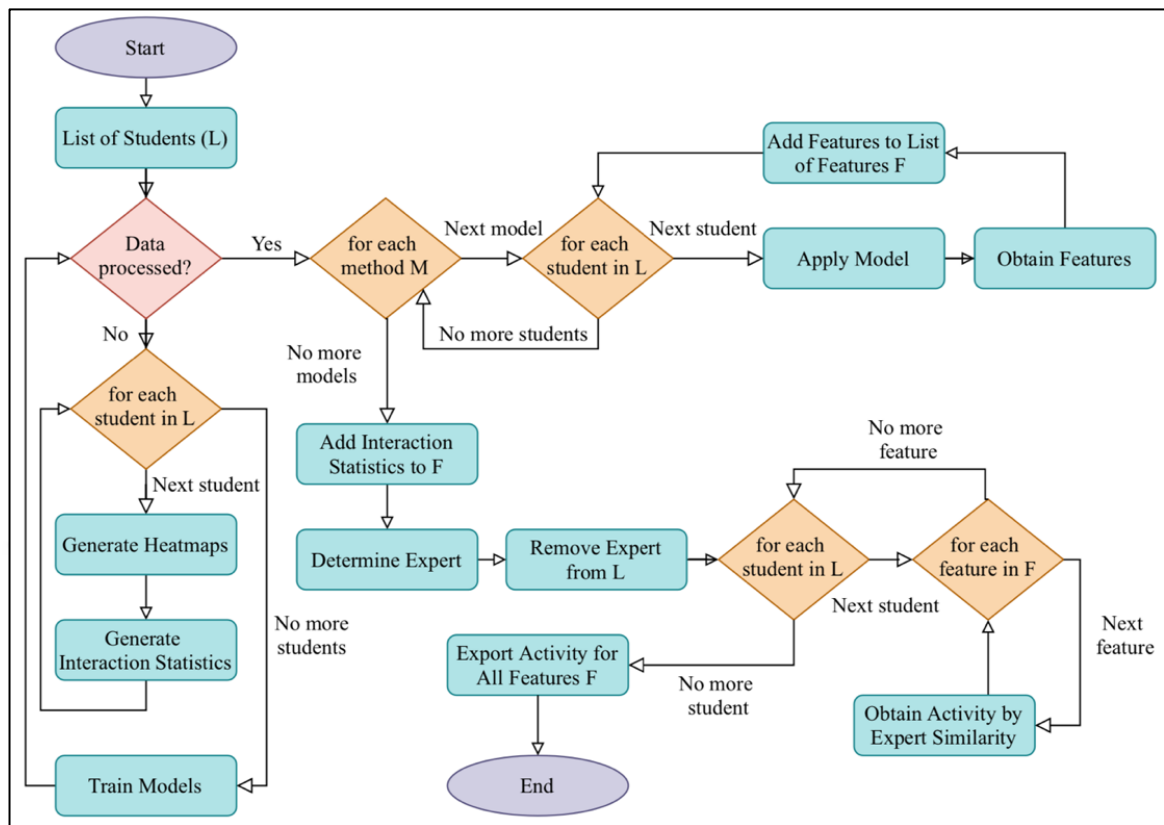


Figure 3.19. The flowchart of Activity dataset preparation procedure

3.4.1. Activity Vector Generation using Interaction Statistics

The high-level interaction statistics are obtained from the raw values stored in the Click and Stream tables. There are four high-level measurements to calculate: total spent time, time spent actively, number of mouse clicks, and mouse movement path. The measurements are calculated for each interface by dividing the table into two subtables according to the interface (StreamT, SteamE, ClickT, and ClickE). Total spent time is the measurement that represents the total amount of time spent in a lecture and this value is calculated from the latest entry in the Stream table. Then the maximum duration values obtained from each subtable represent the total spent time on each interface (MaximumDuration). Time spent actively measures the time spent in each interface. This value is calculated using the number of entries in the Stream table for each interface. The number of entries (CountRows) is multiplied by three since each log entry in the Stream table stores information collected while the student spends three seconds in the interface. The number of mouse clicks is obtained with the number of entries in the Click table for

each interface separately and mouse movement paths is the number of mouse cursor locations obtained from log files stored in the Stream table. Finally, the interaction statistics measurements are collected in a single vector in order to obtain the activity vector that represents the Activity of a student. The normalized interactions statistics vector is obtained by dividing time spent actively, number of mouse clicks, and mouse movement paths by total spent time. Algorithm 3.3 presents the activity vector generation procedure using interaction statistics.

Algorithm 3.3. Interaction statistics vector generation algorithm

```

ClickT ← Click["Timeline Tool"]
ClickE ← Click["E-learning Web Pages"]
StreamT ← Stream["Timeline Tool"]
StreamE ← Stream["E-learning Web Pages"]

Procedure Calculate( student, Click, Stream, normalized )
    active_time ← 3 × CountRows( Stream[student] )
    total_time ← MaximumDuration( Stream[student] )
    click ← CountRows( Click[student] )
    movement ← 0
    for all logfiles in Stream[student] do
        movement ← movement + CountRows( logfile )
    end for
    if normalized is True then
        activity_vector[0] ← total_time
        activity_vector[1] ← active_time
        activity_vector[2] ← click
        activity_vector[3] ← movement
    else
        activity_vector[0] ← active_time ÷ total_time
        activity_vector[1] ← click ÷ total_time
        activity_vector[2] ← movement ÷ total_time
    end if
    return activity_vector
end procedure

for all students in lecture do
    activity_t ← Calculate( student, ClickT, StreamT, False )
    activity_e ← Calculate( student, ClickE, StreamE, False )
    n_activity_t ← Calculate( student, ClickT, StreamT, True )
    n_activity_e ← Calculate( student, ClickE, StreamE, True )
    activity ← Concat( activity_t, activity_e )
    normalized_activity ← Concat( n_activity_t, n_activity_e )
end for

```

3.4.2. Activity Heatmaps Generation

The interaction statistics use time and mouse actions statistics but do not consider the coordinates of the mouse actions. The activity heatmaps are generated in order to create an interaction capturing method that emphasizes the importance of the mouse action locations. The number of click and movement measurements together with their coordinates are extracted from Click and Stream tables in order to create activity heatmaps.

The students explore the interfaces in individual computers. The screen resolutions of each computer can be different or a student could change the size of the browser. To tackle this problem, the page width and page height information for each mouse action is collected. Page width and page height information is used to convert all mouse action coordinates to a manageable scale. In order to perform this conversion, the page width is selected as 640 and the page height is 480. These values together form the standard definition video display resolution which is called 480p. Before the activity heatmaps are created, all coordinates are scaled to 480p resolution using the page width and page height values stored for each entry in Click and Stream tables.

The heatmaps are created using the values in Click and Stream tables and an empty matrix with a size of 640×480 . Firstly, the empty matrix is filled with the values obtained from the Click table. The mouse click actions are counted for each cell of the empty matrix using the scaled coordinate values. Then, the log files in the Stream table are used to obtain the scaled coordinates of the cells visited by the mouse movements where the number of visits in each location is added to the values in the matrix. Then, the values in the matrix are distributed to their neighbors using the Gaussian heat distribution algorithm. The heat distribution algorithm is presented in Algorithm 3.1. Finally, the value in each cell is scaled according to the maximum cell value to values between 0 and 1 where 1 is the maximum cell value and 0 indicates an empty cell.

The final matrix is converted to an image using equation 3.3 where the hue values of the image are obtained using the values in the scaled matrix. The value and saturation values of the image are set to 1. The resulting image is then converted to the RGB representation to obtain colored activity heatmaps of all students in a lecture. The RGB representations

are utilized in grayscale and weighted scale activity heatmaps according to equation 3.4 and 3.5 respectively. Algorithm 3.4 presents the activity heatmap generation.

Algorithm 3.4. Activity heatmap generation procedure

```

function scale( x, y, page_width, page_height )
     $x \leftarrow x \times (640 \div \text{page\_width})$ 
     $y \leftarrow y \times (480 \div \text{page\_height})$ 
end procedure

Procedure GenerateHeatmaps( Click, Stream )
    matrix[640][480]  $\leftarrow$  0
    i  $\leftarrow$  0
    for all students in lecture do
        for all Click[student] do
            scale(x,y,page_width,page_height)
            matrix[x][y]  $\leftarrow$  matrix[x][y] + 1
        end for

        for all Stream[student] do
            for all logfiles do
                scale(x,y,page_width,page_height)
                matrix[x][y]  $\leftarrow$  matrix[x][y] + 1
            end for
        end for

        matrix  $\leftarrow$  heatdistribute(matrix)
        matrix  $\leftarrow$  matrix  $\div$  maximum(matrix)
        image[640][480][3]  $\leftarrow$  0
        for i from 0 to 640
            for j from 0 to 480
                image[i][j][0]  $\leftarrow$  (1-matrix[i][j])  $\times$  240
                image[i][j][1]  $\leftarrow$  1
                image[i][j][2]  $\leftarrow$  1
            end for
        end for

        colorheatmap  $\leftarrow$  hsv2rgb(image)
        grayscaleheatmap  $\leftarrow$  rgb2gray(colorheatmap)
        weightedheatmap  $\leftarrow$  rgb2weighted(colorheatmap)

        heatmap[‘color’]  $\leftarrow$  colorheatmap
        heatmap[‘gray’]  $\leftarrow$  grayscaleheatmap
        heatmap[‘weighted’]  $\leftarrow$  weightedheatmap
        heatmaps[i]  $\leftarrow$  heatmaps
        i  $\leftarrow$  i + 1
    end for
end procedure

```

3.4.3. Activity Vector Generation using Activity Heatmaps

The activity heatmaps are utilized to obtain activity vectors with the proposed Convolutional Autoencoder method and other dimension reduction methods. Initially, the width and the height of the activity heatmap images are 640 and 480 respectively. This image size is empirically determined to be large for fast and robust convolutional autoencoder training. Consequently, before the heatmaps are utilized in the dimension reduction methods and the convolutional autoencoders, they are resized to a smaller representation where the width is 32 and the height is 24 pixels. Every cell in the RGB representation of a heatmap image has a value between 0 and 255. These values are divided by 255 to obtain values between 0 and 1. This scaling operation is performed in order to decrease the learning error values such that these values do not affect the weight update procedure of the network structure dramatically. If these values are not scaled, some weights would overcompensate while the others would undercompensate the calculated error and this would lead to undesired outcomes in the model outputs.

The proposed Convolutional Autoencoder model comprises an encoder and a decoder. The encoder composes of three encoder blocks where each block contains a convolutional layer and is accompanied by a max-pooling layer. In the first encoder block, the convolutional layer contains 16 filters and the size of each filter is 3×3 . The activation function of this layer is a rectified linear unit (ReLU) described in equation 3.8 which converts the negative numbers to 0 and keeps the positive numbers as they are [74].

$$R(x) = \max(0, x), x \in \mathbb{R} \quad (3.8)$$

The padding and stride are selected to be 1 to keep the size of the convolutional results the same as the input size. This selection ensures that the convolutional layers are tasked with extracting patterns and the max-pooling layers aim to decrease the dimensions. The window size of the max-pooling layer is 2×2 . The configurations of the second and third encoder blocks are the same except for the number of filters in the convolutional layers which are 8 and 4 respectively. Algorithm 3.5 presents the procedure to create the proposed encoder.

Algorithm 3.5. Encoder generation procedure

```

Procedure Encoder(input_layer)
    padding  $\leftarrow$  1
    stride  $\leftarrow$  1
    filter_size  $\leftarrow$  (3,3)
    pooling_size  $\leftarrow$  (2,2)

    block1  $\leftarrow$  Convolution( 16, filter_size, padding, stride, input_layer )
    block1  $\leftarrow$  Activation( 'ReLU', block1 )
    block1  $\leftarrow$  MaxPooling( pooling_size, padding, stride, block1 )

    block2  $\leftarrow$  Convolution( 8, filter_size, padding, stride, block1 )
    block2  $\leftarrow$  Activation( 'ReLU', block2 )
    block2  $\leftarrow$  MaxPooling( pooling_size, padding, stride, block2 )

    block3  $\leftarrow$  Convolution( 4, filter_size, padding, stride, block2 )
    block3  $\leftarrow$  Activation( 'ReLU', block3 )
    block3  $\leftarrow$  MaxPooling( pooling_size, padding, stride, block3 )
    encoder  $\leftarrow$  block3
    return encoder
end procedure

```

The decoder composes of three decoder blocks where each block contains a convolutional layer accompanied by an up-sampling layer. In the first decoder block, the convolutional layer contains 4 filters and the size of each filter is 3×3 . The activation function of this layer is a rectified linear unit. The padding and stride are selected to be 1 to keep the size of the convolutional results the same as the input size. This selection ensures that the convolutional layers are tasked with extracting patterns and the up-sampling layers are tasked with increasing the dimensions. The window size of the up-sampling layer is 2×2 . The configurations of the second and third decoder blocks are the same except for the number of filters in the convolutional layers which are 8 and 16 respectively. Algorithm 3.6 presents the procedure to create the proposed decoder.

Algorithm 3.6. Decoder generation procedure

```

Procedure Decoder( input_layer )
    padding  $\leftarrow$  1
    stride  $\leftarrow$  1
    filter_size  $\leftarrow$  (3,3)
    sampling_size  $\leftarrow$  (2,2)

    block1  $\leftarrow$  Convolution( 4, filter_size, padding, stride, input_layer )
    block1  $\leftarrow$  Activation( 'ReLU', block1 )
    block1  $\leftarrow$  UpSampling( pooling_size, padding, stride, block1 )

    block2  $\leftarrow$  Convolution( 8, filter_size, padding, stride, block1 )
    block2  $\leftarrow$  Activation( 'ReLU', block2 )
    block2  $\leftarrow$  UpSampling( pooling_size, padding, stride, block2 )

    block3  $\leftarrow$  Convolution( 16, filter_size, padding, stride, block2 )
    block3  $\leftarrow$  Activation( 'ReLU', block3 )
    block3  $\leftarrow$  UpSampling( pooling_size, padding, stride, block3 )
    decoder  $\leftarrow$  block3
    return decoder
end procedure

```

The decoder produces 16 channels of information in its output. The model proposes a final convolutional layer where the number of filters depends on the number of channels the input layer has since the autoencoder aims to reproduce the input data. The final convolutional layer has 3 filters if the input heatmaps are color heatmaps since this heatmap has three information channels as RGB. This layer is utilized with a single filter if the input heatmaps are either grayscale or weighted scale heatmaps in order to match the input size. The activation function used in this convolutional layer is the sigmoid function where the input values are transformed into values between 0 and 1 [75]. The final layer utilizes sigmoid function described in equation 3.9 since in the preprocessing section the values in the input heatmaps are transformed into values between 0 and 1.

$$\sigma(x) = \frac{1}{1 + e^x} \quad (3.9)$$

The optimizer for the proposed model is selected as the adaptive moment estimation (Adam) [76]. Adam optimizer is a recent extension of stochastic gradient descent (SGD) [77]. In SGD, a network utilizes a static learning rate which is applied to all the weights in the network. On the other hand, Adam optimizer maintains individual learning rates for the

weights and adapts these individual learning rates according to the changes in the network throughout the training process. The training loss for the model is obtained using cross-entropy [78]. Cross-entropy loss C is calculated with equation 3.10 where p is the probability distribution in training output and q is the probability distribution in model prediction. Algorithm 3.7 presents the creation of a convolutional autoencoder model.

$$C = - \sum_i p(i) \cdot \log(q(i)) \quad (3.10)$$

Algorithm 3.7. Model generation procedure

```

Procedure CreateModel( input_shape )
    padding  $\leftarrow$  1
    stride  $\leftarrow$  1
    filter_size  $\leftarrow$  (3,3)
    input  $\leftarrow$  Input( input_shape )
    encoder  $\leftarrow$  Encoder( input )
    decoder  $\leftarrow$  Decoder( encoder )
    if input_shape[2] is 3 then
        output  $\leftarrow$  Convolution( 3, filter_size, padding, stride, decoder )
    else
        output  $\leftarrow$  Convolution( 1, filter_size, padding, stride, decoder )
    end if
    model  $\leftarrow$  Activation( 'sigmoid', model)
    model  $\leftarrow$  Optimizer( 'Adam', model)
    model  $\leftarrow$  Loss( 'cross-entropy', model)
    return model
end procedure

```

The proposed Convolutional Autoencoder models are trained with all the corresponding heatmaps of students in a lecture. Initially, the models are intended to be trained in 10,000 epochs but the training procedure also utilizes the early stopping criteria where if achieved the training stops earlier than intended. The training stops if the minimum loss value achieved in the training does not decrease in 200 epochs the network stops the training process. The final step in creating the activity vectors using the proposed model is to generate the set of vectors that contains the features obtained from different types of heatmaps and with different heatmap generation methods. The first method is to generate individual heatmaps from each interface and concatenate the features obtained from the individually trained models in order to produce a single activity vector. The second method is to have a single heatmap generated for each student without separating the interfaces and

considering the platform as a single interface. Training is performed for all combinations of heatmap type and generation method separately. The features are extracted from the output of the encoder of each trained model. The activity heatmaps are also used with other dimension reduction methods using different components for each method which are presented in Table 3.4. Algorithm 3.8 presents the procedure to obtain the activity vectors for each student for the proposed Convolutional Autoencoder method.

Algorithm 3.8. Activity vector generation with proposed model

```

Procedure TrainModel( heatmaps, channels )
    epochs  $\leftarrow$  10000
    patience  $\leftarrow$  200
    heatmaps  $\leftarrow$  Resize( heatmaps, height, width ) / 255
    input_shape[3]  $\leftarrow$  {32, 24, channels}
    model  $\leftarrow$  CreateModel( input_shape )
    stopping_criteria  $\leftarrow$  EarlyStopping( 'loss', 'minimum', patience )
    trained_model  $\leftarrow$  Train( model, heatmaps, epochs, stopping_criteria )
    return trained_model
end procedure

ClickT  $\leftarrow$  Click["Timeline Tool"]
ClickE  $\leftarrow$  Click["E-learning Web Pages"]
StreamT  $\leftarrow$  Stream["Timeline Tool"]
StreamE  $\leftarrow$  Stream["E-learning Web Pages"]
heatmap_total  $\leftarrow$  GenerateHeatmaps( Click, Stream )
heatmap_t  $\leftarrow$  GenerateHeatmaps( ClickT, StreamT )
heatmap_e  $\leftarrow$  GenerateHeatmaps( ClickE, StreamE )
methods  $\leftarrow$  { 'PCA', 't-SNE', 'FA', 'MDS', 'ICA', 'GRP', 'Isomap' }
for each heatmapset in heatmap_total do
    model  $\leftarrow$  TrainModel( heatmaps, channels )
    features_total  $\leftarrow$  GetEncoderOutput( heatmaps, model )
    features_total_other  $\leftarrow$  DimensonReduction( heatmaps, methods, config )
end for
for each heatmapset in heatmap_t do
    model  $\leftarrow$  TrainModel( heatmaps, channels )
    features_t  $\leftarrow$  GetEncoderOutput( heatmaps, model )
    features_t_other  $\leftarrow$  DimensonReduction( heatmaps, methods, config )
end for
for each heatmapset in heatmap_e do
    model  $\leftarrow$  TrainModel( heatmaps, channels )
    features_e  $\leftarrow$  GetEncoderOutput( heatmaps, model )
    features_e_other  $\leftarrow$  DimensonReduction( heatmaps, methods, config )
end for
features_concat  $\leftarrow$  Concat( features_t, features_e )
features_concat_other  $\leftarrow$  Concat( features_t other, features_e other )

```

3.5. COURSES

A course in the proposed e-learning environment comprises of lectures that are created by the instructor. Every lecture in each course is subject to an evaluation on its own. In other words, the learning states of the students are evaluated for each lecture individually. Consequently, each lecture comprises at least one e-learning web pages content, one timeline, and one quiz. The timeline contains at least one item that can be presented both in the timeline structure and the map. The items in the timeline consist of a date, a location, brief information about the importance of the item according to the course, and detailed information about the item. The quiz has at least two questions where the subject of each question is associated with content from one of the interfaces. Half of the questions in the quizzes are associated with the information available in the timeline tool and the other half are from the e-learning web pages interface.

3.6. SURVEY

The students fill a survey that utilizes the analysis of their their emotional states during the lectures. The survey is adapted from a widely used flow state identification technique, the Flow State Scale which is proposed by Jackson and Marsh in 1996 [79]. The survey uses a 5-point Likert scale where 5 indicates “Strongly agree”, 1 indicates “Strongly disagree” and the values 2, 3, and 4 indicate “Disagree”, “Neither disagree nor agree”, and “Agree” respectively. The questions of the survey are presented in Table 3.6.

Table 3.6. Survey questions on a 5-point Likert scale.

Questions
I struggled. But, I realized that I was able to overcome the challenge.
I didn't have to think about what to do.
During the lecture, I was aware that I was performing well.
My concentration level was high and I wasn't distracted during the lecture.
I felt like time was passing differently than usual. (Faster or slower)
I wasn't worried about my performance during the lecture.
I was sure about what I had to do.
I enjoyed the lecture.
During the lecture, I felt that I had everything under control.

4. IMPLEMENTATION

The implementation of the e-learning platform utilized in this thesis is partially supported by a grant acquired from European Union under the Erasmus+ programme for the project “Timeline Travel”. The platform is designed with feedbacks obtained in the project meetings where the user experience of each partner is discussed in detail. The data utilized in the analysis of this thesis is collected from this platform which is publically available online. The platform is deployed in a Linux Apache Server [80] and its configuration is presented in Table 4.1.

Table 4.1. Server configuration

CPU	Intel Xeon Platinum 8168 2.7GHz
RAM	1GB
Operating System	Ubuntu 18.04.3 (LTS) x64
Apache Server	Version 2.4.29
PHP	Version 7.2.24
Python	Version 3.6
MySQL	Version 5.7.32
SQLite	Version 3.20.1
Django	Version 2.1.3

The system has two software structures: e-learning platform and activity vector generation. E-learning platform contains the interactive timeline tool, classical e-learning web pages interface, and the data collection tool, Open Web Analytics. The users interact with this platform to create or attend courses. The activity vector generation contains the extraction of the interaction statistics, generation of activity heatmaps, feature extraction using Convolutional Autoencoders (CAE), dimension reduction using other methods in the literature, and *Activity* value generation through Expert similarity. The *Activity* values, together with the survey responses of students and the *Performance* values extracted from the e-learning platform are used in the validation of the proposed methods. This validation is performed on a third party software. Figure 4.1 illustrates this software structure.

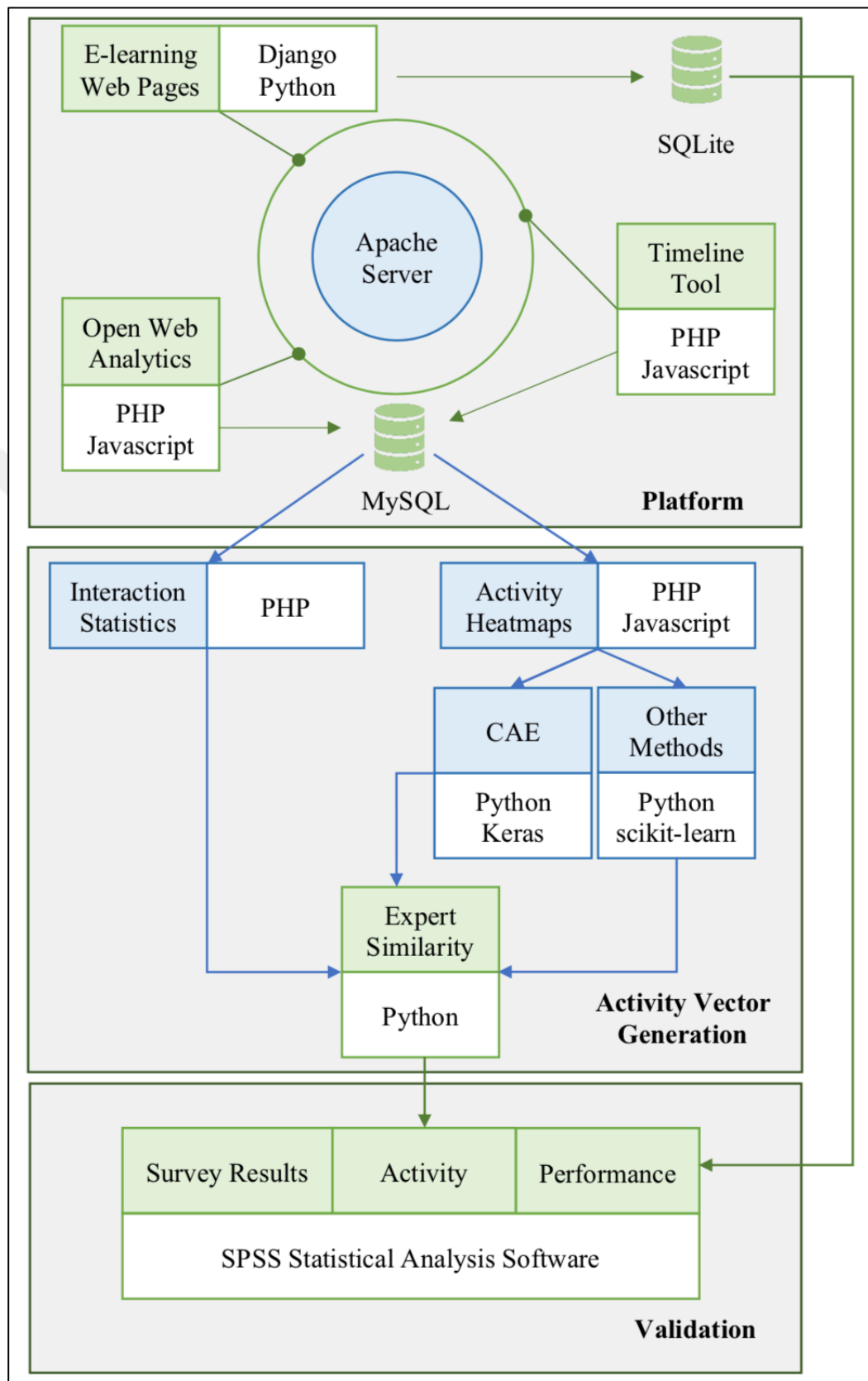


Figure 4.1. Overview of the software structure

4.1. THE E-LEARNING ENVIRONMENT

The e-learning web pages interface is developed using the Django web development framework [81]. The functionalities are handled by Javascript and the data processing is performed through Python functions. The necessary information is stored in a relational database system using SQLite [82]. The e-learning web pages interface is implemented by one of the partners of the Erasmus+ project. Figure 4.2 demonstrates an example course in the e-learning web pages interface with electronic documents, a timeline link, and a quiz.

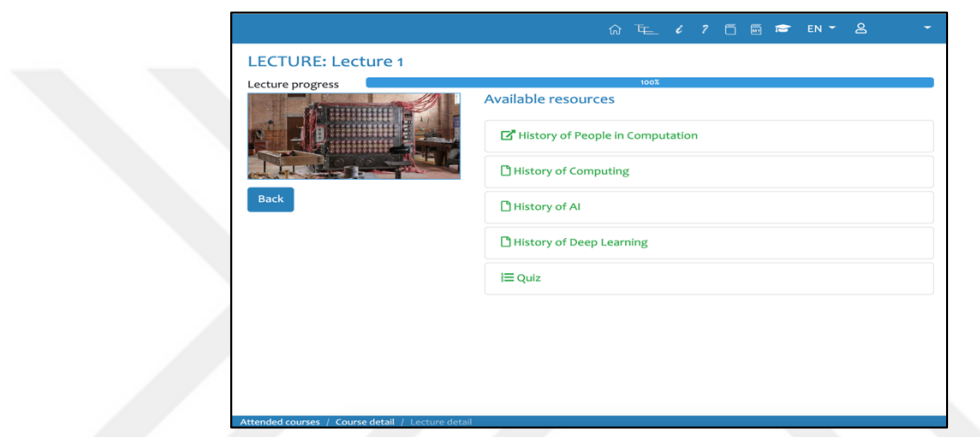


Figure 4.2. Presentation of a course in E-learning Web Pages

The timeline tool interface is developed by dividing the system into four modules: timeline, map, infobox, and editor panel. The functionalities are developed with Javascript and all the data handling operations are done by PHP. The necessary information is stored in a relational database system using MySQL [83]. Figure 4.3 illustrates timeline, map, and infobox modules in an example timeline.

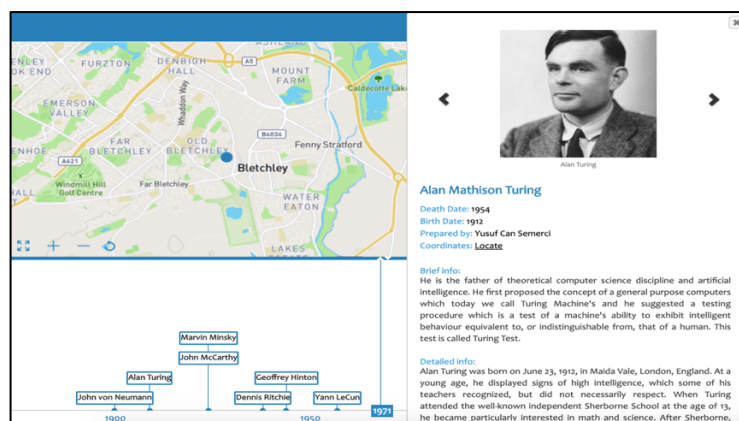


Figure 4.3. An example timeline which presents a map, a timeline, and an infobox

4.1.1. Timeline

The timeline operations are developed using a Javascript library called “timeline.js” [84] as the basis for the structure. The functionalities and the visual aspects of the library are adapted and reimplemented to the needs of the interface. The items are placed according to dates where the timeline location of each item is selected as the smallest date provided by the content creator. The timeline structure presents the items in a time range that changes according to the zoom level. The horizontal actions on the timeline move the structure to a different time range even though the zoom level is constant. The time intervals in each visible period also depend on the zoom level of the timeline. The vertical bar in the middle of the timeline shows the current date a user sees in both the timeline and the map. The visible markers on the map are the items that have dates before the current time. The changes in the timeline also affect the visibility of markers on the map where the items that have dates after the current time disappears. Hovering the mouse on items reveals an image of the item. The items in the timeline can also be clicked to open the corresponding infobox. Figure 4.4 presents an example timeline at different zoom levels with different time intervals.

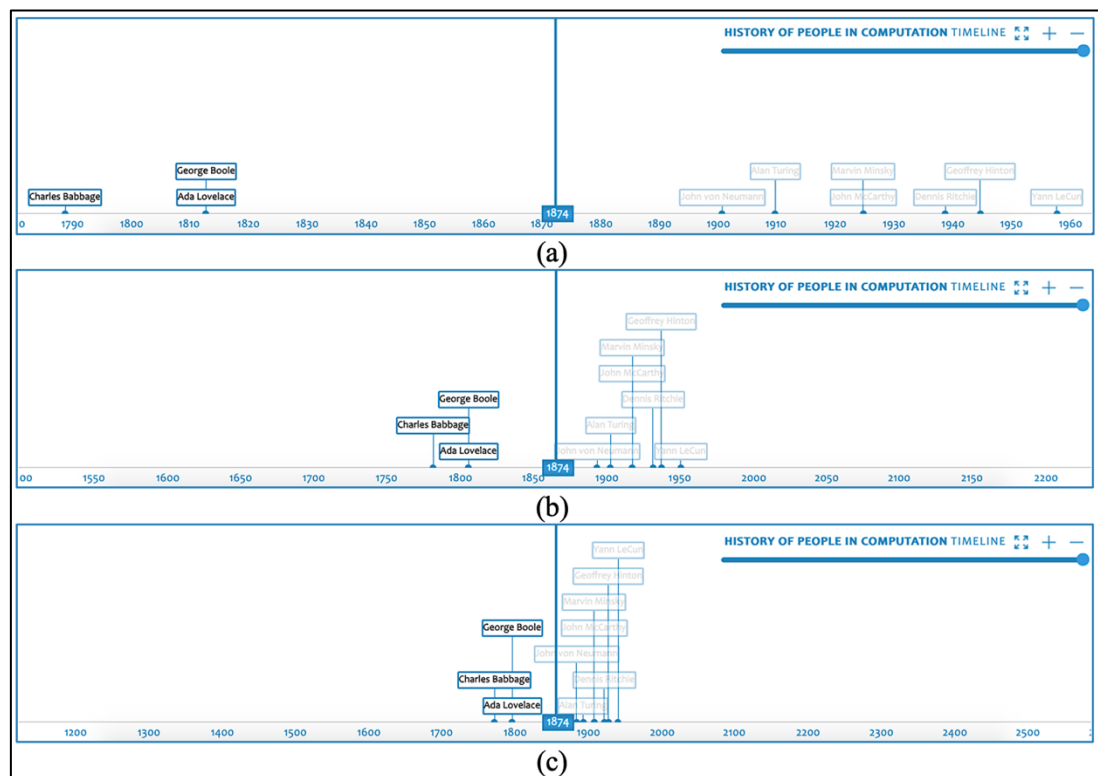


Figure 4.4. An example timeline with different intervals a) 10-year b) 50-year c) 100-year

4.1.2. Map

The map operations are developed and adapted on the structure of “leaflet.js” [85] that is a Javascript library. The map itself is provided by “Mapbox” [86]. The markers for each item are placed on the map according to the coordinates given as latitude and longitude. Each marker opens the corresponding infobox when clicked. The tool has the capability of presenting the map in three different views: Road, Satellite, and Terrain. The Road view illustrates the map with country and city names and the current roads connecting these cities. This view provides a visual representation of the relation between the markers, the surrounding cities, and the roads that connect the items. The Satellite view presents the map with current satellite images underneath the markers of the items. This view provides an understanding of how the surrounding area developed after the events the item depicts and a better view of the neighborhood the item is related. Finally, in the Terrain view, the map presents the physical features of the surrounding areas to provide an insight on the relations between the items and the physical attributes of the neighborhood if it is important for the content of a course. Figure 4.5 presents example areas illustrated according to the provided map views.



Figure 4.5. Examples of map views a) Road b) Satellite c) Terrain

4.1.3. Infobox

The infobox is a popup box that is developed on the structures of the Bootstrap front-end framework [87]. The structure provides only the box that popup when clicked on a trigger object. An image gallery, a title area, and a textual area are implemented on this structure to present detailed information regarding each item. The image gallery contains all the images provided for each item and can be navigated horizontally through arrow icons. The title area presents the full name of the item, the important dates, and a link that takes the map to the location provided for the item. The textual area contains brief information where the importance of the item is presented, detailed information where more details about the item can be given, and the references the information provided are taken from. An infobox can be opened from both the map and the timeline. Figure 4.6 presents an example infobox.

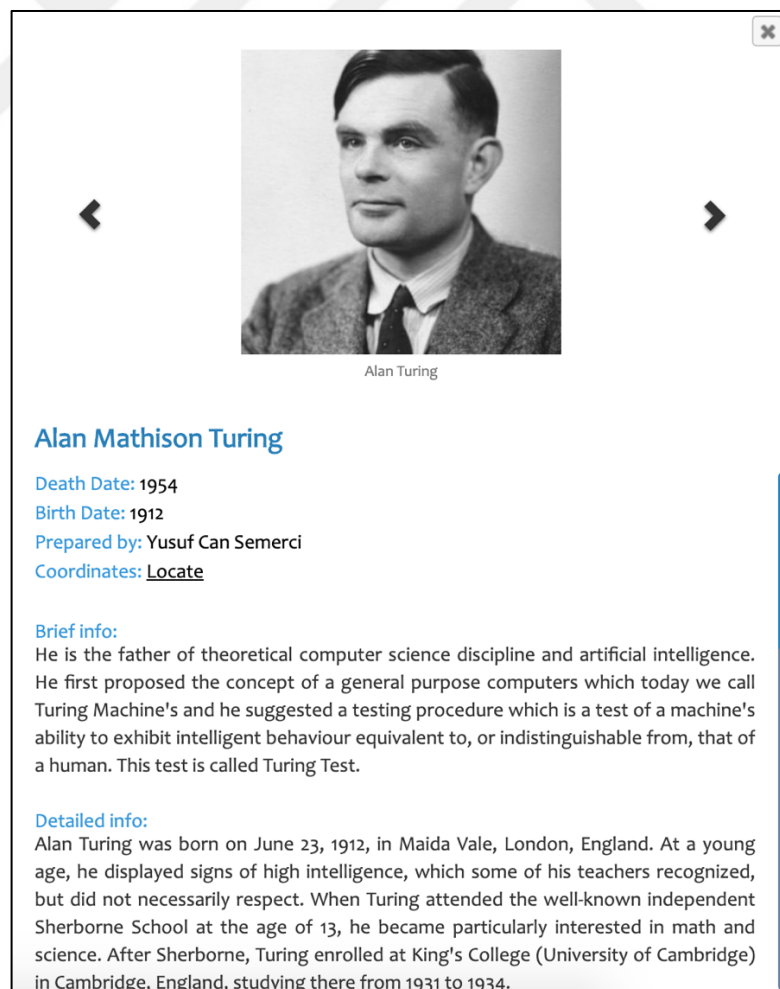


Figure 4.6. An example infobox that contains information about Alan Turing

4.1.4. Editor Panel

The content management system, in other words, editor panel is where instructors create the timelines and their content. All operations in the editor panel are performed with a combination of SQL queries and PHP scripts. The editor panel allows its users to create timelines with only a timeline name. The panel prevents the users to create content without creating a timeline since without connecting the content to a timeline that item would not be visible. When creating an item for a timeline there are several fields that need to be filled. The full name and the short name of the item, the associated dates, the corresponding location, and the textual information are these necessary fields. The location could be inserted either as latitude and longitude or as selected from a map in the content creation page. The map also allows drawing shapes on the map in order to indicate a region rather than a single location. The content creation page is presented in Figure 4.7.

Figure 4.7. The content creation page of the editor panel

The editor panel also allows inserting images for each item. One of these images can be selected as the image to be displayed when the mouse hovers on an item in the timeline structure or when an infobox is opened. Teachers can see the current content of the timeline on the timeline management page where the content could be viewed, edited, and deleted. The images of each content can be accessed through a link from this page. Figure 4.8 presents the timeline management page.

Select	Name	Start Period	End Period	Created by	Create Date	Approved	Images	Edit	Delete
☐	Augusta Ada Byron King	1852	1815	ycsemerci	2019-12-02 15:11:07	✓ ✗	View	Edit	Delete
☐	Charles Babbage	1871	1791	admin	2019-12-24 20:01:52	✓ ✗	View	Edit	Delete
☐	John von Neumann	1957	1903	admin	2019-12-24 20:15:32	✓ ✗	View	Edit	Delete
☐	Alan Mathison Turing	1954	1912	admin	2019-12-24 20:21:48	✓ ✗	View	Edit	Delete
☐	Dennis MacAlistair Ritchie	2011	1941	admin	2019-12-24 20:29:11	✓ ✗	View	Edit	Delete
☐	John McCarthy	2011	1927	admin	2019-12-25 07:43:12	✓ ✗	View	Edit	Delete
☐	Marvin Lee Minsky	2016	1927	admin	2019-12-25 07:48:17	✓ ✗	View	Edit	Delete
☐	Geoffrey Everest Hinton	1947	1947	admin	2019-12-25 07:59:21	✓ ✗	View	Edit	Delete
☐	Yann LeCun	1960	1960	admin	2019-12-25 08:18:00	✓ ✗	View	Edit	Delete
☐	George Boole	1864	1815	admin	2019-12-25 08:29:42	✓ ✗	View	Edit	Delete

Figure 4.8. The timeline management page of the editor panel

4.2. DATA COLLECTION

The e-learning platform is equipped with a web analytics tool called Open Web Analytics (OWA) that provides raw statistics for web-based applications. Each action taken by the students in the e-learning platform is recorded to a MySQL database using the OWA operations. The interaction data collected in the MySQL database are raw statistics.

In the Open Web Analytics database, two tables are considered in data collection procedures: Click and Stream. For each lecture, a unique session number is assigned to each student. This session id, together with the unique student user name, indicates the data collected from the interaction of a student in a single lecture. Click table stores the coordinates of mouse click locations, the screen resolution at the time of the click action, and which interface this mouse click action is performed. The mouse click locations are stored separately as x and y values of the coordinates. The screen resolution is also divided into two entities: page width and page height. Figure 4.9 presents example values of a student in the Click table.

timestamp	user_name	session_id	click_x	click_y	page_width	page_height	interface
1577374454	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1256	7	1280	874	E-learning Web Pages
1577374458	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	118	871	1280	874	E-learning Web Pages
1577374459	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	284	865	1280	874	E-learning Web Pages
1577374467	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	826	604	1280	625	E-learning Web Pages
1577374474	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	513	621	1280	625	E-learning Web Pages
1577374484	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	379	581	1280	625	E-learning Web Pages
1577374488	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	644	610	1280	616	E-learning Web Pages
1577374489	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	383	577	1280	616	E-learning Web Pages
1577374494	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	470	548	1280	616	E-learning Web Pages
1577374621	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	522	308	1280	314	E-learning Web Pages
1577374623	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	523	468	1280	864	E-learning Web Pages
1577374625	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	604	840	1280	864	E-learning Web Pages
1577374638	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	132	824	1280	864	E-learning Web Pages
1577374640	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1188	154	1280	864	Timeline Tool
1577374648	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1223	155	1280	864	Timeline Tool
1577374648	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1238	169	1280	864	E-learning Web Pages
1577374648	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1224	164	1280	864	Timeline Tool
1577374648	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1232	164	1280	864	Timeline Tool
1577374648	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1224	173	1280	864	E-learning Web Pages
1577374649	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	1241	172	1280	864	E-learning Web Pages

Figure 4.9. Example entries in Click table of Open Web Analytics

Stream table stores the mouse movements of the students every three seconds. Each entry in the table contains the unique session number, the unique user name of the student, a log file that contains the coordinates of the locations visited by the mouse movement, the amount of time passed since the start of the session until the creation of the log file, and the interface where the mouse movement is recorded. Figure 4.10 presents example values of the Stream table for a single student.

timestamp	user_name	session_id	log_file	duration	interface
1577374211	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 6.4 KiB]	55	Timeline Tool
1577374204	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.4 KiB]	49	Timeline Tool
1577374201	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 5.8 KiB]	46	Timeline Tool
1577374189	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.4 KiB]	34	Timeline Tool
1577374187	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 3.5 KiB]	31	Timeline Tool
1577374180	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.5 KiB]	25	Timeline Tool
1577374177	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 3.1 KiB]	22	Timeline Tool
1577374168	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.5 KiB]	13	Timeline Tool
1577374162	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.9 KiB]	7	Timeline Tool
1577374122	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.1 KiB]	240	E-learning Web Pages
1577374016	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 1.8 KiB]	132	E-learning Web Pages
1577373906	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.2 KiB]	24	E-learning Web Pages
1577373897	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.4 KiB]	15	E-learning Web Pages
1577373880	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.9 KiB]	331	E-learning Web Pages
1577373868	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.3 KiB]	319	E-learning Web Pages
1577373627	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 1.7 KiB]	47	E-learning Web Pages
1577373609	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 3.1 KiB]	30	E-learning Web Pages
1577373585	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 1.8 KiB]	6	E-learning Web Pages
1577373582	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 3.3 KiB]	3	E-learning Web Pages
1577373576	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 4.1 KiB]	3	E-learning Web Pages
1577373575	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.3 KiB]	25	E-learning Web Pages
1577373548	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 1.8 KiB]	340	E-learning Web Pages
1577373544	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 3 KiB]	333	E-learning Web Pages
1577373504	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.6 KiB]	294	E-learning Web Pages
1577373498	Subject 01	8s0izp9ymgaoui65igg0bcmmqv69eifl	[BLOB - 2.3 KiB]	288	E-learning Web Pages

Figure 4.10. Example entries in Stream table of Open Web Analytics

The log file in the Session table contains the x and y coordinates of the mouse cursor and the page resolution when the content of the log file is collected as page width and page height. These mouse cursor coordinates are the locations visited by the user on the corresponding interface in three seconds. It should be noted that these log files do not collect mouse movements if the student leaves the interface of the e-learning platform. If the students are performing mouse operations but they are not in either of the e-learning platform interfaces, the data collection tool continues to count the seconds that passed since the beginning of the session without collecting mouse actions. This is the reason why the duration values do not increase in monotonic three second periods. Figure 4.11 presents the content of an example log file.

```

1 timestamp:1577374171, cursor_x:628, cursor_y:819, page_width:1280, page_height:864
2 timestamp:1577374171, cursor_x:626, cursor_y:809, page_width:1280, page_height:864
3 timestamp:1577374172, cursor_x:626, cursor_y:806, page_width:1280, page_height:864
4 timestamp:1577374172, cursor_x:614, cursor_y:774, page_width:1280, page_height:864
5 timestamp:1577374176, cursor_x:613, cursor_y:762, page_width:1280, page_height:864
6 timestamp:1577374176, cursor_x:608, cursor_y:742, page_width:1280, page_height:864
7 timestamp:1577374177, cursor_x:583, cursor_y:715, page_width:1280, page_height:864
8 timestamp:1577374177, cursor_x:570, cursor_y:693, page_width:1280, page_height:864
9 timestamp:1577374177, cursor_x:568, cursor_y:687, page_width:1280, page_height:864
10 timestamp:1577374178, cursor_x:567, cursor_y:684, page_width:1280, page_height:864
11 timestamp:1577374178, cursor_x:591, cursor_y:628, page_width:1280, page_height:864
12 timestamp:1577374178, cursor_x:609, cursor_y:605, page_width:1280, page_height:864
13 timestamp:1577374178, cursor_x:638, cursor_y:540, page_width:1280, page_height:864
14 timestamp:1577374178, cursor_x:641, cursor_y:236, page_width:1280, page_height:864
15 timestamp:1577374178, cursor_x:634, cursor_y:210, page_width:1280, page_height:864
16 timestamp:1577374179, cursor_x:640, cursor_y:265, page_width:1280, page_height:864
17 timestamp:1577374179, cursor_x:636, cursor_y:290, page_width:1280, page_height:864
18 timestamp:1577374179, cursor_x:618, cursor_y:321, page_width:1280, page_height:864
19 timestamp:1577374180, cursor_x:618, cursor_y:322, page_width:1280, page_height:864
20 timestamp:1577374180, cursor_x:622, cursor_y:324, page_width:1280, page_height:864

```

Figure 4.11. The content of an example log file

4.3. DATASET PREPARATION

The data collected in the Click and Stream tables are converted into information that describes the Activity of a student. This thesis proposes an interaction statistics based approach and a heatmap based approach. The high-level interaction metrics are calculated using PHP where the data in the Click and Stream tables is utilized through SQL queries. Similarly, the required data for activity heatmaps are gathered from the Click and Stream tables using PHP through SQL queries. The PHP scripts generate the heatmap matrices where each cell holds the mouse action densities at that location. The heat and color distribution procedures are applied on these matrix using Javascript. Figure 4.12 presents example heatmaps generated from a student's interactions.

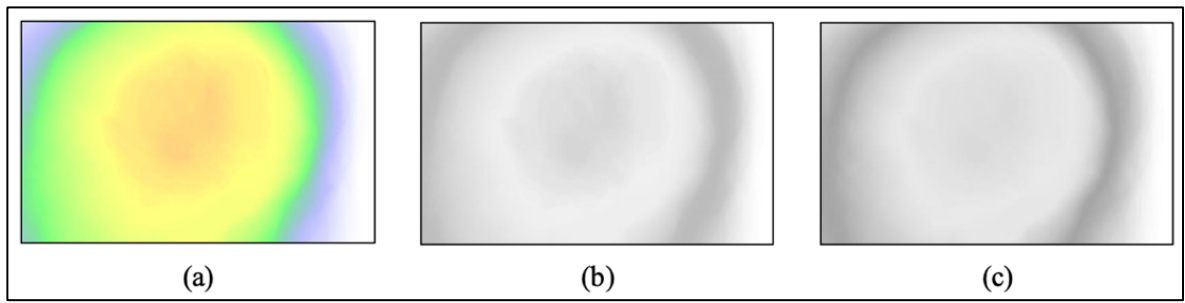


Figure 4.12. Example activity heatmaps generated from a student's interactions

a) Color b) Grayscale c) Weighted scale

The proposed Convolutional Autoencoder models are implemented using the Tensorflow [88] based Python library “Keras” [89]. Other dimension reduction methods are not implemented but are utilized from the Python library “scikit-learn” [90]. The single activity values of each student are obtained through Expert similarity and they are generated for all methods using Python. Figure 4.13 presents a reproduction of an example heatmap in different stages of the training.

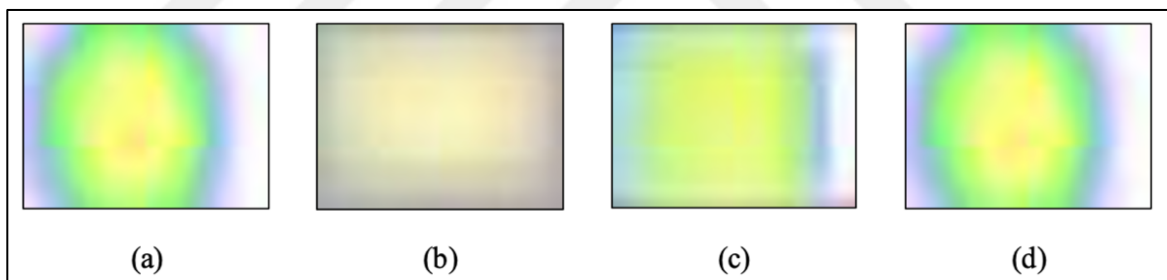


Figure 4.13. Example heatmaps generated by the proposed convolutional autoencoder model in different stages a) Original b) First Epoch c) Tenth Epoch d) Final Prediction

4.4. COURSES

In order to eliminate student profile biases in the experiments, two courses were designed for two different departments. A course about the pioneers of computing and the history of advancements in the field, named “History of People in Computation” was prepared for the students in the Department of Computer Engineering. This course was intended for the student profiles in Natural Sciences. Another course about the political figures and their effects on the events that occurred in the Middle East region was created for the students in the Department of Political Science and International Relations named as “Regional

Studies”. The content of “History of People in Computation” course was prepared in English while “Regional Studies” course was in Turkish.

Each course had only one lecture and the experiment for each course was conducted in a controlled environment. The performances of the students were measured by a quiz at the end of each lecture and for the analysis of their flow states, the students filled a survey after the quiz. The total duration of each lecture which consists of the content exploration duration and the quiz and survey completion durations was two hours for each lecture. The difficulty level for each course was considered to be average according to the instructors of the courses.

The course materials were divided between the interfaces of the timeline tool and the e-learning web pages in order to ensure that the students interact with both interfaces. The content of every interface is unique to itself and there is no duplicate information in these interfaces. Each course consists of three small chapters in the form of electronic documents, a timeline with 8-10 related persons who are important for the course content, and a quiz with 10 multiple-choice questions. Half of the questions in the quizzes are associated with the information available in timeline tool and the other half is from e-learning web pages interface. The important persons were placed in the timeline according to their birth dates and information about their life, work, and importance for the course were presented in an infobox available as a popup window. Table 4.2 presents the organization of each course.

Table 4.2. Course details

Department	Course Name	Interface	Syllabus
Computer Engineering	History of People in Computation	E-learning Web Pages	History of Computing History of Artificial Intelligence History of Deep Learning
		Timeline Tool	History of People in Computation
Political Science and International Relations	Regional Studies	E-learning Web Pages	Iran-Iraq Wars Arab-Israeli Wars American-Arab Wars
		Timeline Tool	Political Figures in Middle East

The “History of People in Computation” course contains a timeline entitled “History of People in Computation” which contains 10 figures from the computation history who were pioneers of computing, artificial intelligence, and deep learning, and three electronic documents entitled “History of Computing”, “History of Artificial Intelligence” and “History of Deep Learning”. The History of People in Computation timeline is presented in Figure 14.

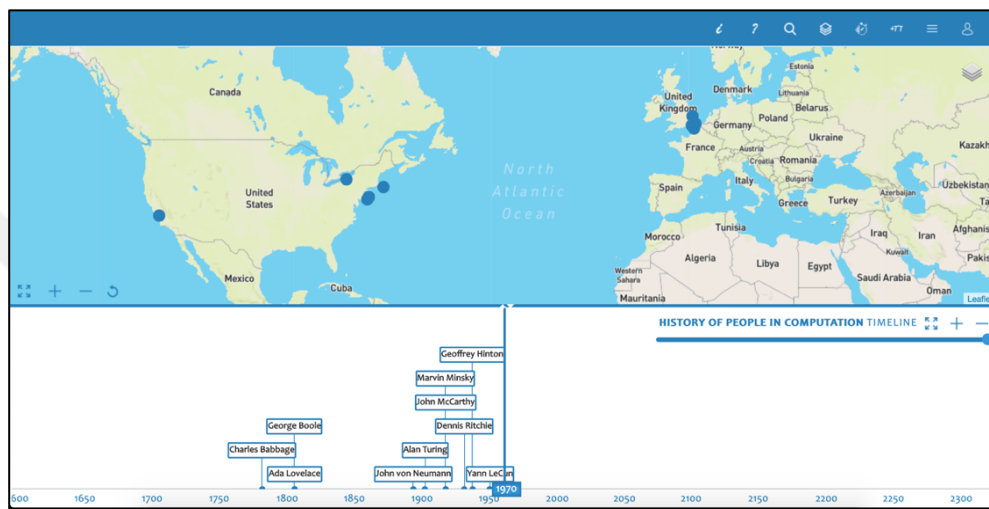


Figure 4.14. History of People in Computation timeline

“Regional Studies” course consists of a timeline called “Political Figures in the Middle East” which is prepared with 8 political figures in the history of the Middle East and three small chapters in e-learning web pages interface: “Iran-Iraq Wars”, “Arab-Israeli Wars”, and “American-Arab Wars”. The timeline of the course is presented in Figure 4.15.

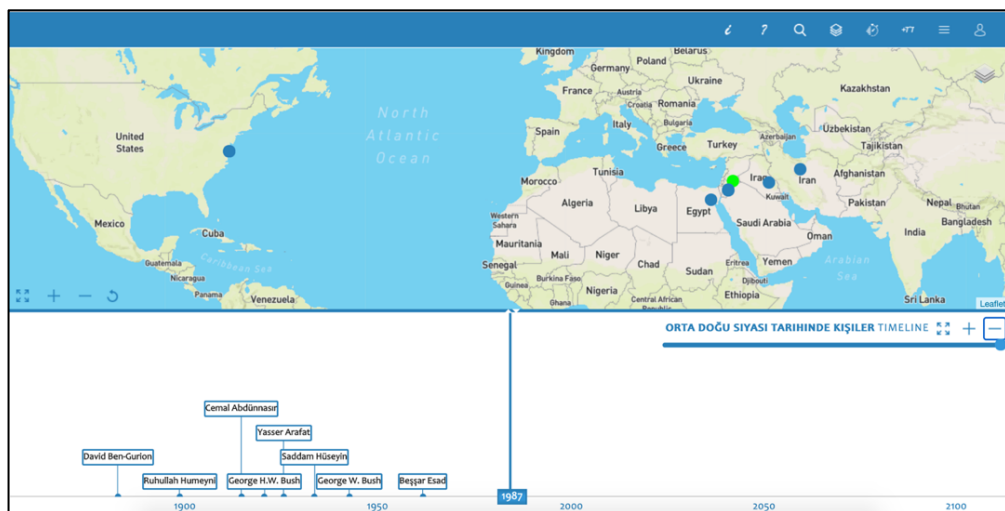


Figure 4.15. Political figures in the Middle East timeline

The quiz for each course has 10 questions prepared in the language of the course content. Five questions were asked from the content of the timeline of the course and the other five were prepared based on the content of the electronic document in the e-learning web pages. Each question has four choices and the exam is closed book. The quizzes are chosen as the technique of formative assessment because a quiz can have questions from a wide range of topics in a single lecture, it can be completed by the students in a short period of time, and it can be graded objectively. The quiz questions and their answers are presented in Appendix A and Appendix B.



5. RESULTS

The interaction and examination data are collected from students who participated in the courses described in the previous chapter. The *Activity* measurement is calculated to reflect the interaction patterns of the students generated through their explorations in the environment. The *Performance* measurement is calculated to associate a value to the acquired skill of each student. Hence, as described in previous chapters, *Activity* is proposed to represent the challenge difficulty, while *Performance* indicates the skill of a student. In order to verify these hypotheses, a statistical analysis should be conducted using the survey responses of the students as ground truth.

This statistical analysis aims to verify the validity of the proposed concepts by examining the existence of correlations between the concepts and the survey responses. If there is a correlation between the *Activity*-Survey Responses and *Performance*-Survey Responses pairs, these measurements can be used to identify the interaction and the acquired skills respectively. Although the existence of these correlations indicate that the measurements can be used to identify these behaviors, if they correlate among them, they can not be used in the flow theory framework. The *Activity-Performance* correlation would present a linear relation between these two concepts, hence a high Activity would imply High Performance and vice versa. This relation is not desired since high Activity would not exist with high Performance in all cases and the related emotions would not be induced. In summary, in order to validate the reliability of these measurements this analysis has three goals:

- Investigate the existence of a correlation between the survey responses and *Activity* values generated to represent the student interactions.
- Investigate the existence of a correlation between the survey responses and *Performance* values generated to represent the acquired skill level.
- Confirm that a correlation between *Activity* and *Performance* values does not exist.

This analysis is performed on IBM's statistical analysis software which is called SPSS [91]. Before the correlation analysis is performed, the Shapiro-Wilk test for normality is performed with each value set to examine their distributions. The evaluations presented in Table 5.3 and Table 5.4 are obtained using one of the two correlation analysis methods; Pearson correlation and Spearman's rank correlation. The correlation analysis method

selection is based on the evaluation of normality tests. If both value sets are normally distributed Pearson correlation analysis is selected, otherwise, the analysis evaluations are obtained using Spearman's rank correlation.

5.1. PARTICIPANTS

The participants who attended the first course were students from the Department of Computer Engineering. Initially, 52 students, ranging from 19 to 21 years old were present in the first course but one of the students was omitted from the analysis because he did not complete the requirements of the course. Among the remaining 51 students, 11 of them were female.

In the second course, the participants were students from the Department of Political Science and International Relations. There were 35 students initially, ranging from 19 to 21 years old and 27 of them were female students. 3 female students were omitted from the analysis of this course as a result of an absence of the required quiz or the survey responses.

The students were in their bachelor's degree studies and they participated in the courses voluntarily. Before starting the experiments, students were asked for permission to store their activity and performance data anonymously and all students voluntarily agreed to give permission. The procedure for the experiments and the participation of the students were also approved by Yeditepe University Ethical Committee. Table 5.1 presents the details of the participants.

Table 5.1. Participants in the e-learning platform courses

Deparment	Number	Age	Male/Female
Computer Engineering	52	19-21	41/11
Political Science and International Relations	35	19-21	8/27

5.2. CONVOLUTIONAL AUTOENCODER MODEL SELECTION

The proposed convolutional autoencoder model is selected as the best model in terms of activity heatmap reproduction. The model selection analysis is performed with heatmaps generated from the interaction patterns of the students who participated in the “History of People in Computation” course. Each model presented in Table 3.3 is trained individually using three different activity heatmaps: color heatmaps generated from the Timeline Tool interface, color heatmaps generated from the E-learning Web Pages interface, and total color heatmaps generated from the e-learning environment considering the environment as a single interface. The models are trained with 10.000 epochs and training is stopped when an early stopping criterion is met. The early stopping activates when the minimum error value is reached and even though the training continues, the error obtained between the reproduced heatmap and the input heatmap does not decrease in 200 epochs. On average the models reach the minimum error at 4.000 epochs. Table 5.2 presents the error values obtained from each activity heatmap for each model.

The model which presents the minimum error value is selected as the proposed model. This model has three convolution layers and three maximum pooling layers in the encoder and three convolutional layers and three up-sampling layers in the decoder. The convolutional layers in the encoder has filters in 3×3 size and the number of filters are 16, 8, and 4 respectively. The convolutional layers in the decoder has filters in 3×3 size and the number of filters are 4, 8, and 16 respectively.

Table 5.2. The loss values obtained for different types of heatmaps in CAE model selection

Model*	Loss (Cross-entropy)		
	Timeline Tool Heatmap	E-learning Web Pages Heatmap	Total Heatmap
L1-E4D4-F3-MP	0.35629	0.31505	0.32266
L1-E4D4-F3-AP	0.36601	0.31524	0.31653
L1-E4D4-F5-MP	0.36242	0.31608	0.31555
L1-E4D4-F5-AP	0.35764	0.31555	0.32302
L1-E8D8-F3-MP	0.35029	0.31123	0.30920
L1-E8D8-F3-AP	0.35076	0.31088	0.31069
L1-E8D8-F5-MP	0.35000	0.31190	0.31094
L1-E8D8-F5-AP	0.34960	0.31105	0.30994
L1-E16D16-F3-MP	0.34807	0.30982	0.30591
L1-E16D16-F3-AP	0.34653	0.30967	0.30540
L1-E16D16-F5-MP	0.34711	0.30893	0.30590
L1-E16D16-F5-AP	0.34767	0.30999	0.30506
L2-E4D4-F3-MP	0.36179	0.35416	0.32885
L2-E4D4-F3-AP	0.36248	0.41225	0.31713
L2-E4D4-F5-MP	0.36823	0.31746	0.32119
L2-E4D4-F5-AP	0.35962	0.31652	0.31197
L2-E8D8-F3-MP	0.35301	0.31231	0.31142
L2-E8D8-F3-AP	0.35097	0.31221	0.31024
L2-E8D8-F5-MP	0.35031	0.31193	0.31118
L2-E8D8-F5-AP	0.35240	0.31271	0.31037
L2-E16D16-F3-MP	0.34780	0.31040	0.30399
L2-E16D16-F3-AP	0.34915	0.30946	0.30682
L2-E16D16-F5-MP	0.34678	0.31000	0.30675
L2-E16D16-F5-AP	0.34713	0.30954	0.30543
L2-E84D48-F3-MP	0.35464	0.32343	0.31591
L2-E84D48-F3-AP	0.35712	0.31254	0.30978
L2-E84D48-F5-MP	0.35363	0.31269	0.31497
L2-E84D48-F5-AP	0.35311	0.31322	0.31120
L2-E168D816-F3-MP	0.35041	0.31180	0.30814
L2-E168D816-F3-AP	0.35350	0.31169	0.30885
L2-E168D816-F5-MP	0.34788	0.31045	0.30654
L2-E168D816-F5-AP	0.34764	0.31142	0.30818
L3-E4D4-F3-MP	0.38266	0.31600	0.32035
L3-E4D4-F3-AP	0.36889	0.33113	0.32603
L3-E4D4-F5-MP	0.37358	0.31906	0.31598
L3-E4D4-F5-AP	0.36162	0.37438	0.32780
L3-E8D8-F3-MP	0.35450	0.31208	0.30732
L3-E8D8-F3-AP	0.35482	0.31247	0.31218
L3-E8D8-F5-MP	0.35079	0.31206	0.30852
L3-E8D8-F5-AP	0.35169	0.31353	0.30962
L3-E16D16-F3-MP	0.34747	0.30994	0.30807
L3-E16D16-F3-AP	0.34832	0.31055	0.30863
L3-E16D16-F5-MP	0.34791	0.30988	0.30709
L3-E16D16-F5-AP	0.34804	0.31052	0.30866
L3-E1684D4816-F3-MP**	0.34405	0.30776	0.30263
L3-E1684D4816-F3-AP	0.35282	0.31907	0.32344
L3-E1684D4816-F5-MP	0.35300	0.31252	0.31064
L3-E1684D4816-F5-AP	0.35282	0.31382	0.31799

* The models are named according to the parameters. L denotes the number of layers, E and D denote the number of filters in the encoder and decoder respectively, F denotes the filter size, and MP and AP denote Maximum and Average pooling respectively.

** Proposed Model

5.3. PAIRWISE CORRELATION ANALYSIS OF COURSE 1: HISTORY OF PEOPLE IN COMPUTATION

Table 5.3 presents the evaluation of pairwise correlation analysis for the variables of the first course which is the “History of People in Computation”. The analysis is performed with data collected from 50 students instead of the actual number of students who participated in the lecture which is 52. Out of these two participants one of them was selected to be the expert based on the proposed Expert selection criteria and excluded from the analysis. The other student is omitted from the pairwise analysis since a complete survey responses could not be obtained from the survey form the student filled after the lecture ended. In quizzes, the maximum grade is 10 and the minimum is 4 ($\mu=6.92$, $SD=1.39$).

Table 5.3. Evaluation of the analysis for the History of People in Computing course

Variables	Survey Responses	Performance
Survey Responses ^a	1	
Performance ^b	$r=.269$ $p=.059^{**}$	1
Activity (from Color Heatmaps, Convolutional Autoencoders, and Concatenation Process) ^a	$r=-.272$ $p=.056^{**}$	$p=.102^{**}$
Activity (from Color Heatmaps and t-Distributed Stochastic Neighbor Embedding) ^b	$r=-.258$ $p=.094^{*}$	$p=.740^{**}$
Activity (from Interaction Statistics) ^b	$p=.354^{**}$	$p=.212^{**}$
Activity (from other Dimension Reduction Methods) ^c	$p>.102$	$p>.102^{**}$
^a According to the Shapiro-Wilk normality test, these variables present normal distributions. ^b These variables deviate from a normal distribution ^c This field presents the correlation analysis evaluation for all other methods described in this thesis which are not present in the table. They are combined because they do not have significant correlations. [*] Pearson correlation analysis. ^{**} Spearman's rank correlation analysis.		

When Table 5.3 is observed according to the goals established before the analysis is performed, the analysis confirms the hypotheses proposed in this thesis. There is a marginally significant ($p < 0.1$) correlation between the *Performance* measure and the survey responses. According to this finding, the *Performance* measure could be utilized in flow state diagrams which leads to learning state identification.

The most challenging hypotheses to be investigated is the existence of a method that can extract interaction patterns to obtain *Activity* values. This thesis proposes several techniques to extract interaction patterns from heatmaps and statistical information. According to the pairwise correlation analysis, the interaction statistics do not correlate with the survey responses for both Interaction Statistics Only and Normalized Interaction Statistics methods. This indicates that the low-level statistics are not able to represent the interactions of the students as intended. Furthermore, the *Activity* values collected using grayscale and weighted scale heatmaps, whether they are concatenated or generated as a total heatmap, do not present correlations that would be interpreted as being able to represent student interactions.

Among all the *Activity* value sets generated using the proposed methods, only one value set has a marginally significant correlation with the survey responses. This *Activity* measurement is generated from colored activity heatmaps using the proposed Convolutional Autoencoder model with the concatenation process. The only other method that correlates with the survey responses for this course is t-Distributed Stochastic Neighbor Embedding method. According to these findings, t-SNE and the proposed CAE methods could be utilized in flow state diagrams to represent *Activity*.

The last investigation performed on these variables is the confirmation that a correlation between the *Activity* and the *Performance* does not exist. Both t-SNE and the proposed CAE method do not have a significant correlation with the *Performance* measure which confirms the validity of these methods in the flow state framework. If a distinction to be made between these two methods for Activity, the r-value could be investigated. The proposed method has a bigger r-value than the t-SNE method. Hence, *Activity* measurement generated from colored activity heatmaps using the proposed Convolutional Autoencoder model with the concatenation process has a stronger relation with the survey responses and can be considered as superior to t-SNE in the interaction pattern extraction.

5.4. PAIRWISE CORRELATION ANALYSIS OF COURSE 2: REGIONAL STUDIES

Table 5.4 presents the evaluation of pairwise correlation analysis for the variables of the second course which is called “Regional Studies”. The analysis is performed with data collected from 29 students instead of the actual number of students who participated in the lecture which is 32. Out of these three participants one of them was selected to be the expert based on the proposed Expert selection criteria and excluded from the analysis. The other two students are omitted from the pairwise analysis since a complete survey responses could not be obtained from the survey form the students filled after the lecture ended. In quizzes, the maximum grade is 10 and the minimum is 4 ($\mu=7.03$, $SD=1,62$).

Table 5.4. Evaluation of the analysis for the Regional Studies course

Variables	Survey Responses	Performance
Survey Responses ^b	1	
Performance ^a	$r=.314$ $p=.097^{**}$	1
Activity (from Color Heatmaps, Convolutional Autoencoders, and Concatenation Process) ^a	$r=-.389$ $p=.037^{**}$	$p=.723^*$
Activity (from Color Heatmaps and Factor Analysis) ^b	$r=.334$ $p=.076^{**}$	$p=.172^{**}$
Activity (from Color Heatmaps and Isometric Feature Mapping) ^a	$r=-.385$ $p=.039^{**}$	$p=.513^*$
Activity (from Color Heatmaps and Gaussian Random Projection) ^a	$r=.345$ $p=.067^{**}$	$p=.207^*$
Activity (from Interaction Data Only) ^a	$p=.924^{**}$	$p=.824^*$
Activity (from other Dimension Reduction Methods) ^c	$p>.106^{**}$	$p>.106$
^a According to the Shapiro-Wilk normality test, these variables present normal distributions. ^b These variables deviate from a normal distribution ^c This field presents the correlation analysis evaluation for all other methods described in this thesis which are not present in the table. They are combined because they do not have significant correlations. [*] Pearson correlation analysis. ^{**} Spearman's rank correlation analysis.		

When Table 5.4 is observed, the analysis of this course also confirms the hypotheses proposed in this thesis. There is a marginally significant ($p < 0.1$) correlation between the *Performance* measure and the survey responses. According to this finding, the *Performance* measure could be utilized in flow state diagrams.

Among all the *Activity* value sets generated using the proposed methods, only one value set has a significant correlation ($p < 0.05$) with the survey responses. This *Activity* measurement is generated from colored activity heatmaps using the proposed Convolutional Autoencoder model with the concatenation process. The other methods that have correlations with the survey responses for this course are Factor Analysis, Isometric Feature Mapping, and the Gaussian Random Projection methods. These methods could be utilized in flow state diagrams to represent *Activity*.

The last investigation performed on these variables is the confirmation that a correlation between the *Activity* and the *Performance* does not exist. The *Performance* measure does not have a significant correlation with any of the valid *Activity* methods which confirms the validity of these methods in the flow state framework. When the r-values of the valid *Activity* generation methods are examined, the proposed method presents a bigger r-value than all the other methods. Hence, *Activity* measurement generated from the proposed Convolutional Autoencoder concatenation approach has a stronger relation with the survey responses and can be considered as superior to the other methods in the interaction pattern extraction.

5.5. FLOW STATE DIAGRAMS OF THE TWO COURSES

The flow state diagram is a graphical representation of the emotional states individuals experience during an activity. According to the evaluation of the correlation analysis, the proposed measurements can be used to illustrate the emotional states of the students in the experimental courses. In this graphical representation, the *Activity* measure represents the y-axis in the diagram since *Activity* is utilized instead of the challenge difficulty in the flow state diagram of Csikszentmihalyi. Similarly, the *Performance* measure represents the acquired skills of the students and can be used to represent the x-axis in the diagram instead of the skill level in the flow state diagram. Figure 5.1 presents the flow diagrams for the experimental courses. The positions of the borders that separate the *Flow* region

from *Anxiety* and *Boredom* regions are obtained from the study of Lynch and Ghergulescu [92]. These border positions were defined by Lynch and Ghergulescu after an analysis including 7614 students and 25945 lessons.

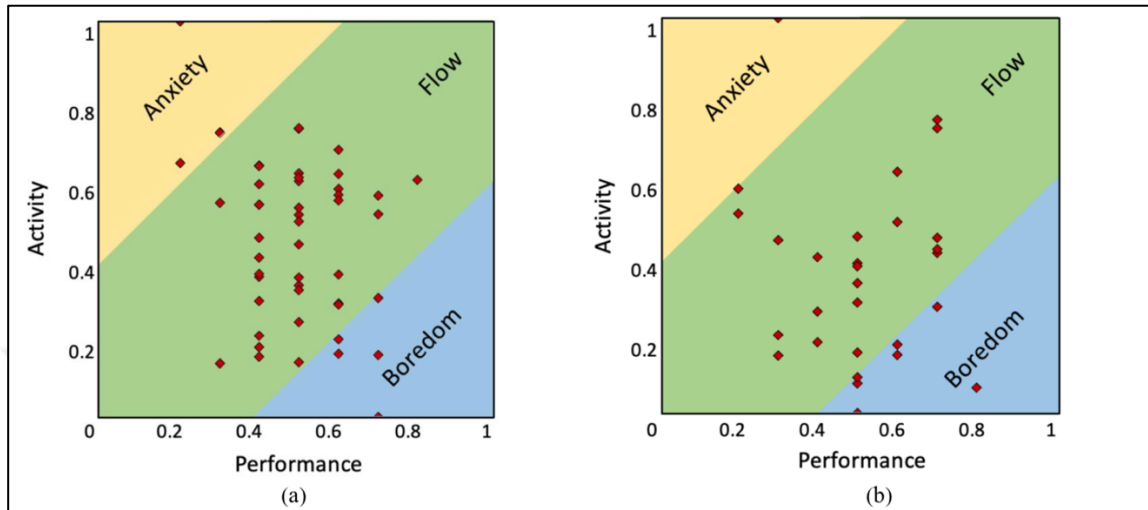


Figure 5.1. Flow state diagrams for a) Course 1: History of People in Computation
b) Course 2: Regional Studies

Although the flow states and their relations with *Activity* and *Performance* measures are verified with the correlation analysis, the relation between the flow state diagrams and the student survey responses are also examined. Table 5.5 presents the student survey mean values for each experimental course.

The student responses showed that in both courses most of them are in a positive sentimental state. The main factor that contributed to this outcome is related to the average difficulty of the course. Based on their answers, we can expect that most of them should be placed on the flow channel. The flow diagrams show that in both courses the majority of the students are in flow. This finding presumes that the content of the course was generally good for most students. We can suppose that making the course content harder would move some of the students to the Anxiety region while making it much easier would make students be uninterested and displace them towards the Boredom region.

Table 5.5. Average student survey responses in 5-point Likert scale

Questions	Course 1	Course 2
I struggled. But, I realized that I was able to overcome the challenge.	3.26	3.07
I didn't have to think about what to do.	4.08	4.52
During the lecture, I was aware that I was performing well.	4.16	4.00
My concentration level was high and I wasn't distracted during the lecture.	3.82	3.41
I felt like time was passing differently than usual. (Faster or slower)	3.46	3.55
I wasn't worried about my performance during the lecture.	3.94	4.03
I was sure about what I had to do.	4.31	4.34
I enjoyed the lecture.	4.56	4.62
During the lecture, I felt that I had everything under control.	4.62	4.38
Mean	4.02	3.99
SD	0.46	0.54

5.6. DISCUSSION

This thesis proposes a new approach for measuring the learning states of students in an e-learning environment. The experiments need to support the propositions of the thesis in several aspects. In summary, these aspects are as follows:

- The proposed approach needs to propose a valid method that would be able to extract meaningful representations from behavioral patterns generated through the interactions of the students in order to be used in learning state evaluation.
- The proposed method should present consistent outcomes in correlation analysis for different course materials with different student profiles so that the approach can be considered generalizable.
- The approach should propose a way to collect behavioral pattern data from students in order to have the ability to be transferred to other e-learning environments.

Two methods are proposed for capturing student interactions in an e-learning environment: interaction statistics and activity heatmaps. The correlation analyses of both experimental courses show that there are no correlations between the survey responses and the methods that use only interaction statistics. On the other hand, there are methods that present correlations with the survey responses and are based on activity heatmaps. According to the evaluation of the experimental courses, the activity heatmaps could be considered as superior to the interaction statistics in capturing behavioral patterns. The difference between these two methods could be explained according to the way they represent the interaction. The interaction statistics uses the total mouse clicks and the number of screen locations visited by the mouse movements and does not consider the actual locations of these actions. However, the activity heatmaps utilize both the number of actions and their actual positions. The superior performance of the activity heatmaps could be a result of their ability to represent the interaction statistics in a spatial domain. Consequently, the activity heatmaps can be used to capture behavioral patterns in an e-learning environment.

The experiments include three different activity heatmap types: color heatmaps, grayscale heatmaps, and weighted scale heatmaps. According to the correlation analysis performed on the data collected in both experiments, only the methods that are based on the color heatmaps present correlations with survey responses. This means that the weighted scale and grayscale heatmaps do not present correlations and do not have the ability to represent the behavioral patterns of the students. Consequently, these findings suggest that the conversion between the color heatmaps to other scales might cause a loss of information and the color channels provide critical information for interactions. The loss of information occurs with the reduction of channels in the images since the color heatmaps have three channels and grayscale and weighted scales heatmaps have only one channel of information.

Although the activity heatmaps can represent interactions using both statistics and location information, the ability to extract meaning from these heatmaps could differ between methods. According to the correlation analysis performed with the activity heatmaps obtained in the “History of People in Computation” course, there are two methods that can extract *Activity* measures: Convolutional Autoencoders and t-Distributed Stochastic Neighbor Embedding and for the “Regional Studies” course there are four methods: Convolutional Autoencoders, Factor Analysis, Isometric Feature Mapping, and the

Gaussian Random Projection. These methods present a way to extract meaningful representations of *Activity* in their respective courses. Although all these *Activity* values could be used in the learning state evaluation in accordance with the student behaviors, there must be a method that presents superior correlations with the survey responses. The proposed Convolutional Autoencoder method is superior to all other methods in both courses in terms of r-values which represents the strength of the correlation between the Activity and the Survey Responses. The higher r-value indicates that the proposed method has a stronger relation with the learning state of a student than all other methods.

The main factor behind the superiority of the proposed Convolutional Autoencoder model is its ability to extract local patterns in the images. The activity heatmaps are images that are obtained to represent the density of information in a spatial domain where the locations and the corresponding density values are obtained from the behavioral patterns of students. The methods that can extract local patterns are expected to present superior correlations since the information in heatmaps highly relies on the critical content presented in the e-learning environment. The analysis verifies that the superior correlation is obtained with Convolutional Autoencoders which takes advantage of the pattern extraction ability of convolution operation. Although it could be argued that this method requires a much bigger amount of data compared to the amount of data the experiments provide, the correlation analysis present a satisfactory correlation with the use of Convolutional Autoencoders in the proposed approach.

The purpose of designing the experiments in different departments is to examine the performance of the methods to extract meaningful representations of student interactions from different student profiles. The experimental courses are conducted with two different student profiles who are bachelor students from two different disciplines. According to the correlation analysis performed on the data obtained from these courses, the only method that present correlations in both courses is the *Activity* measurement generated from colored activity heatmaps using the proposed Convolutional Autoencoder model with the concatenation process. This means that the proposed method is generalizable while the other methods are not.

Concerning the proposed approach's ability to be transferred and applied to other e-learning environments, the approach is designed according to this requirement without needing any external devices. The related works utilize external devices such as EEG

headsets and video cameras. Most of the web-based learning environments are operated using a mouse and these environments require mouse activity to interact with their content. Consequently, this thesis proposes an approach that relies only on the behavioral patterns created by mouse activity. The analysis verifies that the use of mouse interaction patterns can provide a satisfactory measurement for learning state evaluation.

In summary, this thesis proposes a learning state identification approach that provides objective measurements for e-learning systems which are based on information that can be transferred to other domains, present similar correlations with different student profiles, and can be extracted with state of the art methods.



6. LIMITATION

The e-learning platform developed in this thesis ensures mouse activity through an interactive learning tool that contains a map and a timeline. Since the behavioral patterns extracted from the mouse activity are crucial for the proposed method, the e-learning environments that might use the method should also provide sufficient mouse activity. The proposed method also uses mouse activity collected from the classical e-learning web pages. According to the correlation analysis, the most important criteria for the proposed method is to provide mouse activity patterns from two different interfaces that can produce behavioral patterns. Most e-learning environments require sufficient mouse interactions and they contain several tools that can be separated from the classical web pages in order to provide a second interface for mouse activity.

7. CONCLUSION

An emerging challenge arising from the rapid increase in the number of e-learning systems that provide education in distance is the design of a system that can interpret the behaviors of its users in order to identify their learning states. These systems, when they are capable of identifying the emotional states, can provide valuable information to the instructors of online courses so that they can alter the content or the exams of their courses accordingly.

The previous studies in learning identification lack objective, transferable, and generalizable state of the art methods. Consequently, a learning state identification method that provides objective measurements which can capture behavioral patterns of students, are based on information that can be transferred to other e-learning systems, and present similar performances with different student profiles does not exist.

To tackle this problem, this thesis proposes a learning state identification approach where the learning states could be obtained using objective measurements defined to reflect the constructs of flow theory. These objective measures, *Activity* and *Performance*, represent the behavioral patterns generated through interactions and acquired skills of students respectively. The activity heatmaps are utilized to capture behavioral patterns which are then used to extract meaningful representations with Convolutional Autoencoders. Single *Activity* values are obtained by selecting an Expert among the students of a course and by finding the interaction similarities using dot product operation with the Expert's interaction patterns. The *Performance* values of students in a course are calculated using their grades in exams designed to reflect only the information that can be obtained from the materials of the course. When these measures are combined to illustrate a flow state diagram, they present the students in one of the three emotional states: flow, anxiety, and boredom where flow represents the state in which the students are learning. This diagram present instructors, an understanding of the students and the course materials.

The performance of the proposed approach is analyzed using behavioral and performance data collected from the students in two different courses which contain materials from two different disciplines. There are two goals of having two courses for two different student profiles which are to eliminate student profile biases in the experiments and to examine the existence of a method that can provide a generalizable performance in terms of

correlations. The correlation analyses are performed for each course using the Performance values of the students and for comparison purposes Activity values generated using the proposed method and several other methods. According to the correlation analysis, the proposed method presents a superior performance in terms of p-values and r-values. It should also be noted that the only method that presents satisfactory correlation with survey responses for both courses is the proposed method which generates Activity values using activity heatmaps and Convolutional Autoencoders. In summary, The proposed approach provides an objective method to identify learning states since it depends only on mouse interactions and exam grades. It is also generalizable since it is not specific to a particular discipline or profile. Finally, there is no indication that the method is not transferable and applicable to other e-learning systems that can provide a sufficient amount of mouse interactions.

This thesis proposes to fill a critical gap in the studies towards intelligent tutoring systems. In order to achieve full functionality in an intelligent tutor, the system should identify the current emotional states and the needs of its students to such an extent that the content alteration or individual feedback could be fully personalized. The emotional state identification and personalized feedback generation could be achieved using the proposed approach in future studies. Furthermore, the effects of personalized feedback on learning can be examined with a semester-long course where consecutive feedback is provided to the students after each lecture. A further study is also needed for verifying the transferability of the proposed approach in a different e-learning system.

In an era where the world is witnessing a rapid increase in online education, distance learning, and interactive e-learning environments, multidisciplinary studies in education and computer science become significantly important in order to provide objective, automated, and valid learning evaluations. This thesis aims to contribute towards this direction by proposing an approach that provides valuable insight into a course and its students by taking advantage of educational and computational theories.

REFERENCES

1. Siemens G, Gašević D. Special issue on learning and knowledge analytics. *Educational Technology & Society*. 2012;15(3):1-63.
2. Van Barneveld A, Arnold KE, Campbell JP. Analytics in higher education: Establishing a common language. *EDUCAUSE Learning Initiative*. 2012;1(1):I-II.
3. Siemens G. Learning analytics: The emergence of a discipline. *American Behavioral Scientist*. 2013;57(10):1380-400.
4. Rich E. User modeling via stereotypes. *Cognitive Science*. 1979;3(4):329-54.
5. Fischer G. User modeling in human–computer interaction. *User Modeling and User-Adapted Interaction*. 2001;11(1-2):65-86.
6. Anderson JR, Corbett AT, Koedinger KR, Pelletier R. Cognitive tutors: Lessons learned. *The Journal of the Learning Sciences*. 1995;4(2):167-207.
7. Desmarais MC, Baker RS. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Modeling and User-Adapted Interaction*. 2012;22(1-2):9-38.
8. Burns HL, Capps CG, editors. *Foundations of intelligent tutoring systems*. Hove: Psychology Press; 2013.
9. Anderson T, editor. *The theory and practice of online learning*. Edmonton: Athabasca University Press; 2008.
10. Andrews R, Haythornthwaite C, editors. *The Sage handbook of e-learning research*. California: Sage; 2007.
11. Self JA. Bypassing the intractable problem of student modelling. *Intelligent Tutoring Systems: At the Crossroads of Artificial Intelligence and Education*. 1990;41:1-26.
12. Csikszentmihalyi M. *Beyond boredom and anxiety*. San Francisco: Jossey-Bass; 1975.

13. dos Santos WO, Bittencourt II, Isotani S, Dermeval D, Marques LB, Silveira IF. Flow Theory to Promote Learning in Educational Systems: Is it Really Relevant?. *Brazilian Journal of Computers in Education*. 2018;26(02):29.
14. Liu SH, Liao HL, Pratt JA. Impact of media richness and flow on e-learning technology acceptance. *Computers & Education*. 2009;52(3):599-607.
15. Katuk N, Ryu H. Does a longer usage mean flow experience? An evaluation of learning experience with curriculum sequencing systems (CSS). In *2011 Sixth IEEE International Symposium on Electronic Design, Test and Application*. 2011:IEEE.
16. Van Schaik P, Martin S, Vallance M. Measuring flow experience in an immersive virtual environment for collaborative learning. *Journal of Computer Assisted Learning*. 2012;28(4):350-65.
17. Hou HT. Integrating cluster and sequential analysis to explore learners' flow and behavioral patterns in a simulation game with situated-learning context for science courses: A video-based process exploration. *Computers in Human Behavior*. 2015;48:424-35.
18. San Pedro MO, Baker RS, Gowda SM, Heffernan NT. Towards an understanding of affect and knowledge from student interaction with an intelligent tutoring system. In *International Conference on Artificial Intelligence in Education*. 2013:Springer, Berlin, Heidelberg.
19. Kiili K. Content creation challenges and flow experience in educational games: The IT-Emperor case. *The Internet and Higher Education*. 2005;8(3):183-98.
20. Faiola A, Newlon C, Pfaff M, Smyslova O. Correlating the effects of flow and telepresence in virtual worlds: Enhancing our understanding of user behavior in game-based learning. *Computers in Human Behavior*. 2013;29(3):1113-21.
21. D'Mello S, Graesser A. Dynamics of affective states during complex learning. *Learning and Instruction*. 2012;22(2):145-57.

22. Erhel S, Jamet E. Improving instructions in educational computer games: Exploring the relations between goal specificity, flow experience and learning outcomes. *Computers in Human Behavior*. 2019;91:106-14.
23. Özhan ŞÇ, Kocadere SA. The effects of flow, emotional engagement, and motivation on success in a gamified online learning environment. *Journal of Educational Computing Research*. 2020;57(8):2006-31.
24. Ha Y, Im H. The Role of an Interactive Visual Learning Tool and Its Personalizability in Online Learning: Flow Experience. *Online Learning*. 2020;24(1):205-26.
25. Bitrián P, Buil I, Catalán S. Flow and business simulation games: A typology of students. *The International Journal of Management Education*. 2020;18(1):100365.
26. Rodríguez-Ardura I, Meseguer-Artola A. Flow experiences in personalised e-learning environments and the role of gender and academic performance. *Interactive Learning Environments*. 2019:1-24.
27. Verma V, Rheem H, Amresh A, Craig SD, Bansal A. Predicting Real-Time Affective States by Modeling Facial Emotions Captured During Educational Video Game Play. *In International Conference on Games and Learning Alliance*. 2020;447-52:Springer.
28. Standen PJ, Brown DJ, Taheri M, Galvez Trigo MJ, Boulton H, Burton A, Hallowell MJ, Lathe JG, Shopland N, Blanco Gonzalez MA, Kwiatkowska GM. An evaluation of an adaptive learning system based on multimodal affect recognition for learners with intellectual disabilities. *British Journal of Educational Technology*. 2020;51(5):1748-65.
29. Wu SF, Lu YL, Lien CJ. Detecting Students' Flow States and Their Construct Through Electroencephalogram: Reflective Flow Experiences, Balance of Challenge and Skill, and Sense of Control. *Journal of Educational Computing Research*. 2020;58(8):1515-40.

30. Georgiou T, Demiris Y. Adaptive user modelling in car racing games using behavioural and physiological data. *User Modeling and User-Adapted Interaction*. 2017;27(2):267-311.
31. Pearson K. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*. 1901;2(11):559-72.
32. Spearman C. General intelligence objectively determined and measured. *American Journal of Psychology*. 1904;15:107-97.
33. Thurstone LL. The vectors of mind. *Psychological Review*. 1934;41(1):1.
34. Torgerson WS. *Theory and methods of scaling*. New York: John Wiley and Sons, Inc. 1958:460
35. Hérault J, Ans B. Réseau de neurones à synapses modifiables: Décodage de messages sensoriels composites par apprentissage non supervisé et permanent. *Comptes rendus des séances de l'Académie des sciences. Série 3, Sciences de la vie*. 1984;299(13):525-8.
36. Comon P. Independent component analysis, a new concept?. *Signal Processing*. 1994;36(3):287-314.
37. Johnson WB, Lindenstrauss J. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*. 1984;26(189-206):1.
38. Bingham E, Mannila H. Random projection in dimensionality reduction: applications to image and text data. *In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2001:245-50.
39. Tenenbaum JB, De Silva V, Langford JC. A global geometric framework for nonlinear dimensionality reduction. *Science*. 2000;290(5500):2319-23.
40. Maaten LV, Hinton G. Visualizing data using t-SNE. *Journal of Machine Learning Research*. 2008;9(Nov):2579-605.

41. Kullback S, Leibler RA. On information and sufficiency. *The Annals of Mathematical Statistics*, 1951;22(1):79-86.
42. Mitchell TM. *Machine learning*. New York: McGraw Hill;1997.
43. LeCun Y, Touresky D, Hinton G, Sejnowski T. A theoretical framework for back-propagation. *In Proceedings of the 1988 Connectionist Models Summer School*. 1988;1:21-8.
44. Kramer MA. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*. 1991;37(2):233-43.
45. Masci J, Meier U, Cireşan D, Schmidhuber J. Stacked convolutional auto-encoders for hierarchical feature extraction. *In International Conference on Artificial Neural Networks*. 2011;52-9:Springer.
46. Shapiro SS, Wilk MB. An analysis of variance test for normality (complete samples). *Biometrika*. 1965;52(3/4):591-611.
47. Pearson K. VII. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*. 1895;58(347-352):240-2.
48. Spearman C. The proof and measurement of association between two things. *The American Journal of Psychology*, 1904;15(1):72-101.
49. Hong JC, Liu MC. A study on thinking strategy between experts and novices of computer games. *Computers in Human Behavior*. 2003;19(2):245-58.
50. Nakamura J, Csikszentmihalyi M. The concept of flow. *In Flow and the Foundations of Positive Psychology*. 2014;239-63:Springer.
51. Seifert T, Hedderson C. Intrinsic motivation and flow in skateboarding: An ethnographic study. *Journal of Happiness Studies*. 2010;11(3):277-92.
52. Csikszentmihalyi M, Larson R, Prescott S. The ecology of adolescent activity and experience. *In Applications of Flow in Human Development and Education*. 2014;241-54:Springer.

53. Jackson SA, Martin AJ, Eklund RC. Long and short measures of flow: The construct validity of the FSS-2, DFS-2, and new brief counterparts. *Journal of Sport and Exercise Psychology*. 2008;30(5):561-87.
54. Magyaródi T, Nagy H, Soltész P, Mózes T, Oláh A. Psychometric properties of a newly established flow state questionnaire. *The Journal of Happiness & Well-Being*. 2013;1(2):85-96.
55. Bradley MM, Lang PJ. Measuring emotion: Behavior, feeling, and physiology. *Cognitive Neuroscience of Emotion*. 2000;25:49-59.
56. Baumeister RF, Vohs KD, Nathan DeWall C, Zhang L. How emotion shapes behavior: Feedback, anticipation, and reflection, rather than direct causation. *Personality and Social Psychology Review*. 2007;11(2):167-203.
57. Mauss IB, Levenson RW, McCarter L, Wilhelm FH, Gross JJ. The tie that binds? Coherence among emotion experience, behavior, and physiology. *Emotion*. 2005;5(2):175-90.
58. Hinbarji Z, Albatal R, Gurrin C. Dynamic user authentication based on mouse movements curves. *In International Conference on Multimedia Modeling*. 2015;111-22:Springer.
59. Khan IA, Brinkman WP, Fine N, Hierons RM. Measuring personality from keyboard and mouse use. *In Proceedings of the 15th European Conference on Cognitive Ergonomics: The Ergonomics of Cool Interaction*. 2008;1-8:Association for Computing Machinery.
60. Azcarraga J, Suarez MT. Recognizing student emotions using brainwaves and mouse behavior data. *International Journal of Distance Education Technologies (IJDET)*. 2013;11(2):1-5.
61. Korkofingas C, Macri J. Does time spent online have an influence on student performance? Evidence for a large business studies class. *Journal of University Teaching & Learning Practice*. 2013;10(2):2.

62. Calafiore P, Damianov DS. The effect of time spent online on student achievement in online economics and finance courses. *The Journal of Economic Education*. 2011;42(3):209-23.
63. Allen MJ. *Assessing academic programs in higher education*. New Jersey: John Wiley & Sons; 2003.
64. Madaus GF, Airasian PW. Placement, Formative, Diagnostic, and Summative Evaluation of Classroom Learning. *Paper presented at the meeting of the American Educational Research Association*. 1970.
65. Buchanan T. The efficacy of a World-Wide Web mediated formative assessment. *Journal of Computer Assisted Learning*. 2000;16(3):193-200.
66. Justham D, Timmons S. An evaluation of using a web-based statistics test to teach statistics to post-registration nursing students. *Nurse Education Today*. 2005;25(2):156-63.
67. Brewer CA. Near real-time assessment of student learning and understanding in biology courses. *BioScience*. 2004;54(11):1034-9.
68. Open Web Analytics: Open Web Analytics; [cited 2018 11 June]. Available from: <http://www.openwebanalytics.com>
69. Rec IT. BT 601: Studio encoding parameters of digital television for standard 4: 3 and wide-screen 16: 9 aspect ratios. *ITU-R Rec. BT*. 1995;656.
70. Nishio M, Nagashima C, Hirabayashi S, Ohnishi A, Sasaki K, Sagawa T, Hamada M, Yamashita T. Convolutional auto-encoder for image denoising of ultra-low-dose CT. *Heliyon*. 2017;3(8):e00393.
71. LeCun Y, Cortes C, Burges CJ : The MNIST database of handwritten digits; [cited 2020 21 December]. Available from <http://yann.lecun.com/exdb/mnist>
72. Turchenko V, Chalmers E, Luczak A. A deep convolutional auto-encoder with pooling-unpooling layers in caffe. *International Journal of Computing*. 2019;18(1):8-31.

73. Seyfioğlu MS, Özbayoğlu AM, Gürbüz SZ. Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities. *IEEE Transactions on Aerospace and Electronic Systems*. 2018;54(4):1709-23.
74. Hahnloser RH, Sarpeshkar R, Mahowald MA, Douglas RJ, Seung HS. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*. 2000;405(6789):947-51.
75. Han J, Moraga C. The influence of the sigmoid function parameters on the speed of backpropagation learning. *In International Workshop on Artificial Neural Networks*. 1995;195-201:Springer.
76. Kingma DP, Ba J. Adam. A method for stochastic optimization. *In Proceedings of 3rd International Conference on Learning Representations*. 2015.
77. Robbins H, Monro S. A stochastic approximation method. *The Annals of Mathematical Statistics*. 1951:400-7.
78. Murphy KP. *Machine learning: a probabilistic perspective*. Cambridge: MIT Press; 2012.
79. Jackson SA, Marsh HW. Development and validation of a scale to measure optimal experience: The Flow State Scale. *Journal of Sport and Exercise Psychology*. 1996;18(1):17-35.
80. Apache HTTP Server: Apache Software Foundation; [cited 2017 1 September]. Available from: <https://github.com/apache/httpd>
81. Django: Django Software Foundation; [cited 2018 1 September]. Available from: <https://github.com/django/django>
82. SQLite: Hipp DR; [cited 2018 1 September]. Available from: <https://sqlite.org>
83. MySQL: Oracle Corporation; [cited 2017 1 September]. Available from: <https://github.com/mysql/mysql-server>
84. TimelineJS: Knightlab; [cited 2017 30 September]. Available from: <https://github.com/NUKnightLab/TimelineJS>

85. Leaflet.JS: Leaflet; [cited 2017 18 October]. Available from: <https://github.com/Leaflet/Leaflet>
86. Mapbox: Mapbox; [cited 2017 18 October]. Available from: <https://www.mapbox.com>
87. Bootstrap: Otto M, Thornton J; [cited 2017 1 September]. Available from: <http://getbootstrap.com>
88. Tensorflow: Google Brain Team; [cited 2018 1 September]. Available from: <https://github.com/tensorflow/tensorflow>
89. Keras: Chollet F, others; [cited 2018 1 September]. Available from: <https://github.com/fchollet/keras>
90. Scikit-learn: Cournapeau D; [cited 2018 1 September]. Available from: <https://github.com/scikit-learn/scikit-learn>
91. IBM SPSS Statistics for Windows, Version 22.0: Armonk, NY, IBM Corp; [cited 2019 14 October]. Available from: <https://www.ibm.com>
92. Lynch T, Ghergulescu I. Large scale evaluation of learning flow. *In 2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*. 2017;62-4: IEEE.

APPENDIX A: HISTORY OF PEOPLE IN COMPUTATION QUIZ

Q1) Who is the father of Deep Learning?

- a) **Geoffrey Hinton** b) Alan Turing c) George Boole d) John von Neumann

Q2) What was the biggest obstacle for the development of Artificial Intelligence in the 20th century?

- a) The lack of appropriate algorithms
b) The lack of political will
c) The lack of intelligent people to set the right specifications and requirements
d) **The lack of computer power and memory**

Q3) What did Dennis Ritchie create?

- a) Convolutional Networks b) Linux and C++ c) **Unix and C** d) Windows

Q4) When did the first theoretical work on computation started?

- a) 12th century b) 14th century c) 20th century d) **17th century**

Q5) What did Yann LeCunn first create?

- a) **Convolutional Neural Networks** b) Artificial Neural Network
c) Unix and C d) Windows

Q6) What type of algebra modern computers use today?

- a) Convolutional b) **Boolean** c) Computational d) Vectoral

Q7) Which is the importance of Marvin Minsky's creation?

- a) The first use of back propagation computing machine
b) The first convolutional neural network
c) **The first artificial network that can learn**
d) The first general purpose computing machine

Q8) Deep Learning and Machine Learning algorithms are based on what?

- a) **Only on Neural Networks**
b) On Neural Networks and Oriented Graphs
c) On Neural Networks and Back Propagation Networks
d) On Neural Networks and Directed Neuronic Networks

Q9) What is the purpose of Turing Test?

- a) To measure a program's intelligence level
b) To find stupid programs
c) To measure if there is an artificial intelligence or not
d) **To test if a program's intelligence is distinguishable from a human or not**

Q10) The fundamental elements of a computer were defined by who?

- a) John von Neumann in the 20th century b) **Charles Babbage in the 19th century**
c) William Shakespeare in the 16th century d) Archimedes in the 3rd century BC

APPENDIX B: REGIONAL STUDIES QUIZ

Q1) When has the Syrian Civil War ended?

- a) In 2015 b) In 2016 c) In 2017 **d) It continues at the beginning of the year of 2020**

Q2) What was Yasser Arafat a leader of?

- a) The Egyptian independent move b) The Syrian opposition
c) The Palestinians d) The nationalistic Iranian movement

Q3) When was Ben Gurion the Israeli minister?

- a) The first Israeli-Arab war **b) The first and second Israeli-Arab war**
c) The third Israeli-Arab war d) All the Israeli-Arab wars

Q4) What was the reason for the intervention in Afghanistan by the United States?

- a) The control of the opium in Afghanistan
b) The attack of 9/11
c) The suppression of press freedom in Afghanistan
d) The 465 American hostages held by Mustafa Abdel Hacı Hacaturian

Q5) What was the event that provoked The Gulf War?

- a) The Invasion of Iran by Ayatollah Khomeini
b) The Invasion of Kuwait by Saddam Hussein
c) The Invasion of Kuwait by Ayatollah Khomeini
d) The Invasion of Iran by Saddam Hussein

Q6) What was the reason of the Iran-Iraq war?

- a) The desire of Saddam Hussein to be the leader of the Arab war**
b) The desire of Ayatollah Khomeini to be the leader of the Arab war
c) The desire of Saddam Hussein to have access to the gas pipes coming from China
d) The desire of Ayatollah Khomeini to have access to the water resources of Iraq.

Q7) What was the reason for the Suez Crisis?

- a) The nationalization of the Suez Canal by Nasser**
b) The nationalization of the Suez Canal by Yasser Arafat
c) The nationalization of Gaza Canal by Yasser Arafat
d) The nationalization of Gaza Canal by Naser

Q8) In the Six Days War, the Arab States fought against which country?

- a) Israel** b) USA c) Egypt d) Jordan

Q9) The Syrian civil war started because of which event?

- a) A football game **b) Massive protests asking for political reforms**
c) High unemployment d) Beautiful Helene

Q10) Just before the first Arab-Israeli War, Palestine was controlled by who?

- a) The Americans **b) The British** c) The French d) The Mongolians

APPENDIX C: SURVEY IN TURKISH

Öğrenci Anketi

- 1 - Kesinlikle Katılmıyorum
 2 - Katılmıyorum
 3 - Ne Katılıyorum Ne de Katılmıyorum
 4 - Katılıyorum
 5 - Kesinlikle Katılıyorum

1	Zorlandım. Fakat karşılaştığım zorluğu aşabildiğimi farkettim.	1	2	3	4	5
2	Ne yapmam gerektiği konusunda düşünmek zorunda kalmadan platform içerisinde kendimi yönlendirebildim.	1	2	3	4	5
3	Ders sırasında iyi bir performans gösterdiğimin farkındaydım.	1	2	3	4	5
4	Konsantrasyonum tamdı. Ders boyunca dikkatim dağılmadı.	1	2	3	4	5
5	Zaman normalden farklı geçiyormuş gibi hissettim. (Daha hızlı ya da daha yavaş)	1	2	3	4	5
6	Ders sırasında performansım konusunda endişeli değildim.	1	2	3	4	5
7	Ne yapmam gerektiği konusunda kendimden emindim.	1	2	3	4	5
8	Dersten keyif aldım.	1	2	3	4	5
9	Ders sırasında kontrolün bende olduğunu hissettim.	1	2	3	4	5

Ad Soyad	
Doğum Tarihi	
Tarih	
İmza	

