

BACHELOR THESIS

Development of Analysis Software for User Studies Based on iMotions Data

submitted on 7. August 2026
Ai-Trinh Van

First examiner: Prof. Dr. Larissa Putzar
Second examiner: Kilian Hildebrandt

**HOCHSCHULE FÜR ANGEWANDTE
WISSENSCHAFTEN HAMBURG**

Fakultät Informatik und Digitale Gesellschaft
Finkenau 35
22081 Hamburg

Abstract

Facial Emotion Recognition (FER) has become a standard component of multimodal usability research, yet platforms such as *iMotions* that excel at data acquisition provide only limited support for subsequent statistical analysis. Researchers are typically forced to write custom analysis scripts for each new experimental design, contributing to non-reproducible workflows and questionable research practices in Human-Computer Interaction.

This thesis presents a Python-based analysis tool that closes this gap by directly integrating *iMotions* CSV exports. The system lets researchers configure between-subjects and within-subjects study designs through a Tkinter interface, automatically selects an appropriate parametric or non-parametric statistical test based on a Shapiro-Wilk normality check, and renders results in an interactive dashboard covering dimensional emotions, categorical emotions, and time-course visualizations.

Zusammenfassung

Facial Emotion Recognition (FER) hat sich zu einem Standardbaustein der multimodalen Usability-Forschung entwickelt, doch Plattformen wie *iMotions*, die bei der Datenerfassung überzeugen, bieten nur begrenzte Unterstützung für anschließende statistische Analysen. Forschende sind daher häufig gezwungen, für jedes neue Versuchsdesign individuelle Analyse-Skripte zu schreiben, was zu nicht reproduzierbaren Arbeitsabläufen und fragwürdigen Forschungspraktiken im Bereich Human-Computer Interaction beiträgt.

Diese Thesis stellt ein auf Python basierendes Analysetool vor, das diese Lücke schließt, indem es direkt mit *iMotions*-CSV-Exporten arbeitet. Das System ermöglicht Forschenden, über eine Tkinter-Oberfläche sowohl Between-Subjects- als auch Within-Subjects-Studiendesigns zu konfigurieren. Es wählt automatisch einen geeigneten parametrischen oder nicht-parametrischen statistischen Test auf Basis eines Shapiro-Wilk-Normalitätstests aus und stellt die Ergebnisse in einem interaktiven Dashboard dar, das dimensionale Emotionen, kategoriale Emotionen sowie zeitliche Verläufe abdeckt.

Contents

List of Figures	IV
List of Tables	V
1 Introduction	1
1.1 Problem Statement	1
1.2 Aims and Objectives	1
1.3 Research Question	2
1.4 Thesis Structure	2
2 Fundamentals	3
2.1 Affective Computing	3
2.1.1 Discrete Emotion Models	4
2.1.2 Dimensional Emotion Models	5
2.2 User Studies in Human-Computer-Interaction	6
2.2.1 Experimental Designs	6
2.2.2 Emotion Recognition Methods	7
2.2.3 Multimodal Data Sources	8
2.3 Statistical Tests	9
2.3.1 Parametric vs. Non-Parametric Tests	10
2.3.2 Between-Subjects Comparisons	10
2.3.3 Within-Subjects Comparisons	10
2.3.4 Statistical Significance	11
2.4 Speech-To-Text	11
3 Related Work	13
3.1 iMotions Study Software	13
3.2 AP5 Virtual Assistant	14
3.2.1 Basic Chat Mode	14
3.2.2 Complex Chat Mode	14
4 Software Design	16
4.1 System Overview	16

4.2	Requirements Analysis	16
4.2.1	Functional Requirements	16
4.2.2	Non-Functional Requirements	17
4.3	Technologies	17
4.4	System Architecture	17
4.4.1	Presentation Layer	18
4.4.2	Application Layer	18
4.4.3	Data Layer	19
5	Software Implementation	20
5.1	Development Environment and Tools	20
5.2	Setup and Installation	21
5.3	Configuration Interface	21
5.4	Processing Engine	23
5.4.1	Reading iMotions exports	23
5.4.2	Temporal binning	24
5.5	Test Selection	25
5.6	Visualization Dashboard	25
5.6.1	Lazy tab construction	26
5.6.2	Dimensional and Categorical Emotions tabs	26
5.6.3	Time Course tab	27
5.6.4	Design-aware labeling	27
6	Experimental Evaluation	28
6.1	Study Objectives	28
6.2	Participants	28
6.3	Setup and Procedure	29
6.4	Data Sources	30
6.5	Data Analysis	30
7	Results	32
7.1	Software Evaluation	32
7.1.1	Processing Performance	32
7.1.2	Generated Visualizations	33
7.1.3	Supplementary Within-Subjects Case Study	36
7.2	Study Findings	37
7.2.1	Usability Survey Results	37
7.2.2	Speech-To-Text Accuracy and Duration Results	38
7.2.3	Statistical Test Results	38

8	Discussion	39
8.1	Interpretation	39
8.1.1	Software System Reflection	39
8.1.2	Speech-To-Text Study Reflection	40
8.2	Strengths of the Software	42
8.3	Limitations and Future Outlook	42
9	Conclusion	44
	Bibliography	46
	Appendix	51

List of Figures

2.1	Robert Plutchik's Wheel of Emotions [13]	4
2.2	The circumplex model by the American Psychological Association, modified by Feldman Barret and Russell [15]	5
3.1	The Virtual Assistant operating in Basic Chat Mode	15
3.2	The Virtual Assistant operating in Complex Chat Mode	15
4.1	Three-layer architecture of the study analysis system	18
5.1	The Study Configurator interface	22
6.1	The Virtual Assistant operating in Basic Chat Mode, expanded with the voice input feature	29
7.1	Total processing time (left axis) and peak memory usage (right axis) in relation to the number of synthetic respondents, ranging from 4 to 52	33
7.2	Mean engagement per participant and group comparison with automatically selected statistical test	34
7.3	Engagement over time for all participants	34
7.4	Dimensional emotion space per participant	35
7.5	Mean joy per participant and group comparison with automatically selected statistical test	35
7.6	Most dominant categorical emotion per participant	35
7.7	Mean engagement per participant and group comparison with automatically selected statistical test, extended with condition comparison for each participant	36
7.8	Most dominant categorical emotion per condition grouped by participant, exemplary for two participants	36

List of Tables

7.1	Benchmark results: timings (s) and peak memory (MB) as a function of synthetic respondent count	33
7.2	System Usability Scale (SUS) scores by respondent and group	37
7.3	Mean speech recognition satisfaction ratings by group (1–5 scale)	37
7.4	Word Error Rate (WER) and mean duration per respondent and group	38
1	Raw speech-to-text transcriptions for respondent ST1 (Group A, tiny model)	52

1 Introduction

Facial Emotion Recognition (FER) has become increasingly popular in usability studies over the last five years. This trend is evidenced by a surge in literature focusing on the transition from algorithmic benchmarks with controlled datasets to practical usability integration, applying FER technology to real-world HCI scenarios [1, 2, 3].

While FER has gained traction in usability studies, relying solely on a single data source, such as FER, often leads to ambiguity. Consequently, there is a growing consensus that integrating multimodal data significantly enhances the accuracy and robustness of FER systems [4].

1.1 Problem Statement

Industry standard user study software tools like *iMotions* have capitalized on this trend by offering integrated multimodal data acquisition [5]. However, while these tools streamline data collection, they often lack streamlined workflows for the subsequent statistical analysis of complex, synchronized datasets.

Researchers frequently resort to manual scripting for each new experimental design, leading to non-reproducible workflows. This lack of standardization is a key factor in the reproducibility crisis, where studies are often criticized for underpowered designs, undisclosed procedures, or the misuse of statistical methods. These issues are collectively termed 'questionable research practices' (QRPs), which include practices that increase false positives and are often exacerbated by the flexibility of manual scripting [6].

1.2 Aims and Objectives

To mitigate these risks, this project proposes a standardized evaluation environment that integrates directly with *iMotions* exports, automates the generation of standardized visualizations for dimensional and categorical emotion measures, and automatically selects an appropriate statistical test based on the data's distribution and its independent or paired

structure while supporting both between-subjects and within-subjects designs common in usability research, thereby reducing the likelihood of accidental QRPs.

1.3 Research Question

This thesis aims to answer the following research questions:

- What software architecture supports automated, reproducible statistical analysis across both between-subjects and within-subjects study designs, without requiring researchers to write custom analysis scripts for each new design?
- How can an experimental user study be structured to evaluate the adaptability and reproducibility gains of the proposed architecture?

Addressing this question is critical because current manual scripting workflows contribute to QRPs and hinder reproducibility in HCI. By identifying architectural patterns that support adaptability without sacrificing methodological rigor, this work seeks to bridge the gap between data acquisition and statistical analysis in user studies.

1.4 Thesis Structure

This thesis is organized into nine chapters.

Chapter 2 establishes the theoretical foundations, covering Affective Computing, user study methodologies, statistical techniques for data analysis, and speech-to-text concepts. Building on this groundwork, Chapter 3 reviews existing literature, including the *iMotions* software and the *AP5 Virtual Assistant*, the latter of which is mentioned in the Evaluation chapter. The proposed system’s architecture and requirements are described in Chapter 4, introducing the three-layer design that guides the implementation detailed in Chapter 5, which documents the development environment and software components employed.

The empirical assessment unfolds across Chapters 6 and 7. Chapter 6 outlines the methodology used to assess the system, including performance benchmarks and a within-subjects user study comparing two application prototypes. Chapter 7 then presents the outcomes, encompassing the software’s processing performance, generated visualizations, and statistical test results.

Finally, Chapter 8 interprets these findings and addresses limitations of the current implementation, while Chapter 9 summarizes contributions and outlines directions for future work.

2 Fundamentals

The development of analysis software for user studies requires a fundamental understanding of the theoretical concepts and methodological approaches.

The chapter begins by examining affective computing , which defines how human emotions are conceptualized and measured in HCI contexts. Subsequently, the methodology of user studies is discussed, establishing protocols for collecting valid empirical data. The collected data is then analyzed using statistical tests, detailing significance measures and analytical frameworks. Finally, the preprocessing of verbal feedback through speech-to-text conversion is addressed as it is part of Chapter 6.

2.1 Affective Computing

Affective computing is a field introduced by Picard [7, p. 3], who defined it as “... computing that relates to, arises from, or deliberately influences emotions”. The primary goal of this discipline is to equip machines with the ability to recognize, interpret, process, and simulate human affect [8].

However, accurately capturing human emotion presents significant challenges. Individual sensing modalities provide valuable insights into specific aspects of behavior, yet relying on a single data source often leads to ambiguous interpretations. This approach aligns with natural human perception, which inherently relies on the convergence of diverse signals to accurately interpret contexts [9].

To overcome these limitations, multimodal fusion has emerged as a critical paradigm in affective computing. By integrating synchronized data streams, such as combining visual cues with associated emotional states and physiological arousal measurements, systems can achieve a more robust and accurate estimation of the user’s state [5, 10]. This approach leverages the principle of complementarity, where one modality compensates for the weaknesses of another [9]. Multimodal integration is essential for developing analysis software capable of delivering reliable, context-aware insights in user studies.

Emotion models are crucial to affective computing as they define a criterion for measuring emotion or affect [8]. The two types of generic models are as follows:

2.1.1 Discrete Emotion Models

The discrete emotion model, also known as the categorical emotion model, classifies emotion into distinct classes [11]. Within the field of emotion recognition, research predominantly utilizes Ekman’s six basic emotions, which are typically defined as *anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise* [8, 11, 12].

Despite their widespread adoption, discrete emotion models face significant challenges due to cultural differences and the inherent complexity of human affect. Cultural contexts shape how emotions are expressed and interpreted, meaning that a single emotional label may carry vastly different connotations across societies, rendering this model imprecise [8, 11]. Furthermore, human emotional experiences are rarely singular. Individuals frequently navigate mixed or ambivalent states where multiple emotions coexist and interact [8].

These limitations motivated the development of dimensional approaches starting in the 1980s, which sought to represent emotional states as continuous coordinates rather than fixed categories, thereby capturing gradations and mixed affective states more effectively.

For example, Plutchik’s Wheel of Emotions, proposed in 2001, is based on Ekman’s model and expands it into eight primary emotions: *joy*, *trust*, *fear*, *surprise*, *sadness*, *disgust*, *anger*, and *anticipation*, which can be combined to form secondary emotions, such as love from joy and acceptance, and arranged by intensity levels, as shown in Figure 2.1. It differs from Ekman’s purely discrete approach by incorporating intensity gradients and combinatorial relationships, effectively bridging categorical and dimensional perspectives [13].

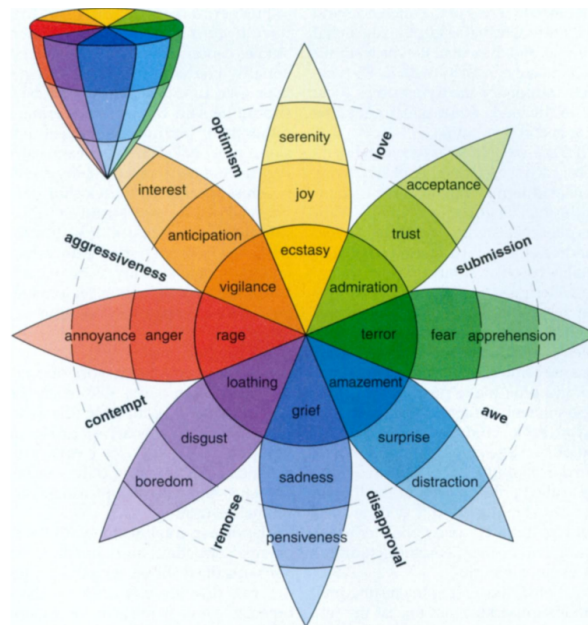


Figure 2.1: Robert Plutchik’s Wheel of Emotions [13]

2.1.2 Dimensional Emotion Models

In contrast to the discrete emotion model that assigns emotions to fixed labels, the dimensional, or continuous, emotion model serves as a complement to its downsides, depicting emotion as coordinates within a multidimensional space [8, 11]. The most widely adopted framework is Russell's Circumplex Model of Affect from 1980, which organizes emotions along two orthogonal axes forming a circular layout [14]:

- **Valence** represents the pleasure or displeasure, ranging from positive to negative
- **Arousal** indicates the activity level, which differentiates between high energy and low energy states

The two-dimensional valence-arousal plane divides into four quadrants, each encompassing emotionally related states [14]:

- **Upper-right quadrant** (high valence, high arousal): Excitement, happiness, astonishment, pleasantness
- **Lower-right quadrant** (high valence, low arousal): Calm, contentment, relaxation, serenity
- **Upper-left quadrant** (low valence, high arousal): Anger, anxiety, fear, tension, distress, annoyance
- **Lower-left quadrant** (low valence, low arousal): Sadness, depression, tiredness, gloom, boredom

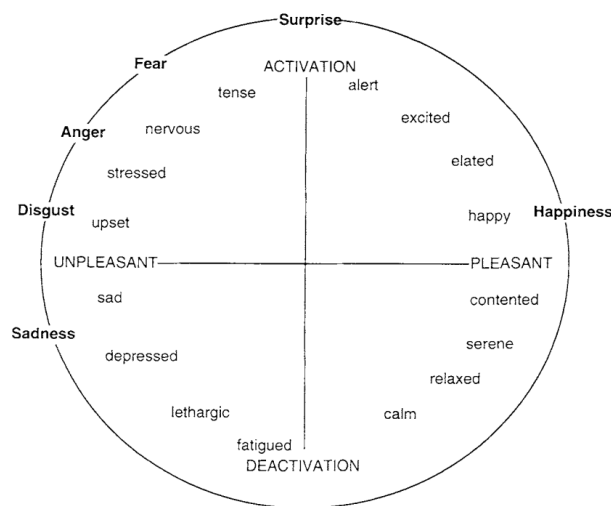


Figure 2.2: The circumplex model by the American Psychological Association, modified by Feldman Barret and Russell [15]

Figure 2.2 illustrates this two-dimensional arrangement.

This circular arrangement reflects the premise that adjacent emotions share similar affective qualities, while opposing emotions represent contrasting states. Unlike discrete models that impose hard boundaries between categories, the circumplex model allows for smooth transitions between emotional states, better reflecting the fluid nature of affective experience.

Some extensions incorporate a third dimension, **Dominance**, which portrays the feeling of control and influence over one's environment, resulting in the Valence-Arousal-Dominance space, though empirical research suggests valence and arousal account for the majority of variance in self-reported emotional states [15].

2.2 User Studies in Human-Computer-Interaction

A way to utilize the aforementioned aspects of affective computing are user studies.

User studies are a fundamental methodological approach in HCI and are used to evaluate how users interact with interactive systems under controlled or natural conditions. The primary goal of user studies is to generate empirical evidence about usability, user experience, performance, and behavioral patterns when interacting with a system. This distinguishes HCI from purely system-centered disciplines, as the effectiveness of a system is ultimately determined by how well it supports human users rather than its technical correctness alone [16, p. 2].

HCI research is fundamentally human-centered and therefore relies on systematic observation and measurement of user behavior. According to MacKenzie [16, p. 130ff], empirical user studies are essential in HCI because they allow researchers to test hypotheses about interaction design and quantify differences between interface conditions using controlled experimental methods.

2.2.1 Experimental Designs

A key aspect of comparative user studies is the selection of an appropriate experimental design. In HCI, two of the most common designs are between- and within-subjects designs.

Within-Subjects Design

In a within-subjects design, each participant experiences all conditions. Conditions are organized into blocks, with each block consisting of one or more levels corresponding to the experimental variables being tested.

To mitigate order effects, conditions are counterbalanced: one group experiences condition A followed by condition B, while the other group encounters them in reverse order.

This approach allows comparisons within the same subject, making it statistically powerful and requiring fewer participants. A disadvantage, however, is a potential learning effect or fatigue, which can influence later performance [16, p. 175f].

Between-Subjects Design

Conversely, a between-subjects design assigns participants randomly or systematically to different conditions, with each participant experiencing exactly one condition. For example, one group is assigned to condition A, while the other is assigned to condition B.

A major advantage of this design is that it avoids carryover effects such as learning and fatigue since participants are not exposed to multiple conditions. Nevertheless, individual differences introduce variability, so larger sample sizes are often required to have sufficient statistical power [16, p. 175f].

Mixed Design

A mixed design combines both within-subject and between-subject factors in a single study. MacKenzie [16, p. 177] illustrates this with an example where handedness serves as the between-subjects variable, while multiple text entry methods act as the within-subject factor.

Mixed designs combine the strengths of both within- and between-subjects designs by including at least one factor of each type. This allows researchers to study naturally occurring group differences while still benefiting from the statistical power of repeated measures for manipulated variables [16, p. 177].

2.2.2 Emotion Recognition Methods

Human emotion recognition has evolved from relying solely on subjective reporting to increasingly sophisticated multimodal approaches. The fundamental distinction lies between methods that capture conscious emotional disclosure versus those that infer affective states

through measurable signals. Both methods offer complementary strengths for understanding emotional experiences in research and applied settings.

Direct Methods

These methods rely on conscious, voluntary input from the participant regarding their emotional state. Self-report instruments remain the standard for validating subjective emotional experiences, with widely adopted tools including the Self-Assessment Manikin (SAM) [17], and various Likert-scale questionnaires, such as the System Usability Scale (SUS) [18, 19]. Structured interviews and survey-based approaches provide additional flexibility depending on research constraints [20]. The primary advantage of direct methods lies in their high validity for capturing subjective experience and straightforward interpretation, which establishes ground truth for validating objective measures [21].

However, these benefits come with notable trade-offs. Self-reports are susceptible to recall bias and interrupt natural interaction patterns, making it difficult to capture real-time emotional dynamics without disrupting the task or context being studied [22].

Indirect Methods

These methods infer emotional states from observable signals without requiring explicit user input during the task. Physiological monitoring and behavioral analysis enable continuous, passive emotion assessment. Indirect methods excel in unobtrusive deployment and enable real-time adaptation to emotional states, while also capturing non-conscious reactions that participants may not be aware of or willing to disclose [23]. Nevertheless, they exhibit lower face validity compared to self-report and must contend with substantial cultural and individual differences in emotional expression. Additionally, sensor calibration challenges and the requirement for extensive validation against ground-truth self-report measures complicate their practical deployment [22].

2.2.3 Multimodal Data Sources

Modern emotion recognition systems increasingly leverage multiple data modalities to overcome the limitations of unimodal approaches. Research demonstrates that combining subjective reports with physiological and behavioral signals tend to yield superior accuracy compared to isolated modalities [20].

Physiological Signals

Physiological measures provide objective indicators of autonomic nervous system activation associated with emotional states. Common signals include electrodermal activity (EDA), heart rate variability (HRV), EEG, electromyography (EMG), respiration patterns, and peripheral temperature [20]. These signals are advantageous because they reflect genuine autonomic responses that are difficult to voluntarily control and thus may better reflect genuine affective state, though they require careful sensor placement and calibration to ensure reliable measurements across individuals.

Self-Report Measures

Self-report instruments continue serving as essential validation benchmarks despite their practical limitations. The Self-Assessment Manikin (SAM) captures dimensional emotional ratings along valence, arousal, and dominance axes [17]. Survey-based methodologies allow assessment across varied research contexts without disrupting task flow [22].

However, researchers must account for potential recall biases and the intrusive nature of these instruments when designing experimental protocols.

Behavioral Metrics

Behavioral analysis extracts emotional cues from observable actions including facial expressions, vocal inflection, posture, and gesture patterns. Body pose estimation and eye-tracking data provide additional contextual signals for interpreting emotional engagement. Behavioral metrics excel at capturing dynamic expression but exhibit significant inter-individual variance across cultures and demographics [24].

2.3 Statistical Tests

This section describes the statistical procedures the software uses to compare experimental conditions. The overarching goal is to determine whether observed differences in facial-expression measures between groups or conditions are statistically meaningful or likely due to chance.

2.3.1 Parametric vs. Non-Parametric Tests

Statistical tests for comparing two or more samples generally fall into two families. *Parametric* tests, such as a *t*-test or analysis of variance (ANOVA), assume that the underlying data are approximately normally distributed and are more statistically powerful when that assumption holds. *Non-parametric* tests, such as the Mann-Whitney U test or the Kruskal-Wallis test, make no such distributional assumption, operating on the ranks of the data instead of their raw values, and are the safer choice when normality cannot be assumed [25].

2.3.2 Between-Subjects Comparisons

In a Between-Subjects design, each participant belongs to exactly one group, so the values being compared come from different, independent people.

Two groups

When exactly two groups are compared, Welch's *t*-test [26] is an appropriate choice when the normality assumption is met. Unlike the original Student's *t*-test, Welch's variant does not assume equal variances between the two groups, which makes it a safer general-purpose default than the classic pooled-variance *t*-test [26]. When normality does not hold, the Mann-Whitney U test [27] serves as the non-parametric alternative, providing a rank-based test for whether one group's values tend to be systematically larger than the other's.

Three or more groups

When more than two groups are compared, a one-way ANOVA [28] is applicable when the normality assumption is met, or the Kruskal-Wallis test [29] otherwise. Both are *omnibus* tests: a significant result indicates that at least one group differs from the others, but does not by itself indicate which pair of groups differs.

2.3.3 Within-Subjects Comparisons

In a Within-Subjects design, the same participant is measured under every condition, so each observation in one condition is paired with the same person's observation in the other condition.

Two conditions

For exactly two conditions, the paired-samples t -test is appropriate when the normality assumption is met. When normality does not hold, the Wilcoxon signed-rank test [30] serves as the non-parametric alternative. Similarly to the Mann-Whitney U test, rather than ranking all observations together, it ranks the absolute per-respondent differences and tests whether the signed ranks are symmetric around zero.

Three or more conditions

For more than two conditions, the Friedman test [31], which serves similarly to the Kruskal-Wallis test as a non-parametric generalization of the paired comparison to more than two related samples, provides a general-purpose default. A repeated-measures ANOVA [28] is the parametric equivalent, but it requires estimating the covariance structure between conditions and is less straightforward to apply in practice.

2.3.4 Statistical Significance

Following common convention by Fisher [32], p -values are typically rounded to two or three decimal places depending on their magnitude, with values below 0.001 reported as $p < .001$.

While p -values provide a standardized measure of statistical evidence against the null hypothesis, they should be interpreted in context and not as dichotomous decision rules alone [33].

2.4 Speech-To-Text

Speech-to-text (STT) technology converts spoken language into written text, enabling hands-free and eyes-free interaction with digital systems. It has become increasingly prevalent in everyday applications, ranging from dictation software and virtual assistants to real-time captioning services.

For individuals with motor impairments who may struggle with traditional keyboard input or individuals with writing difficulties, STT offers an alternative input method that can significantly reduce barriers to text entry [34, 35]. Beyond accessibility in the traditional sense, STT also benefits users in situational contexts where hands are occupied and can serve as a faster input alternative for users who type slowly regardless of ability.

Research on voice assistants shows that performance for users with disabilities is closely linked to cognitive and linguistic abilities, underscoring the need for adaptive interfaces [36]. Given these implications, evaluating the performance and usability of STT systems remains an important concern in HCI research.

3 Related Work

3.1 iMotions Study Software

iMotions is a commercial multimodal biometric research platform designed for human behavior research, integrating data streams from various biosensors into a single synchronized workflow. The software is widely used in academic and industrial settings for studying cognitive processes, emotional responses, and behavioral patterns through the combination of physiological and behavioral data sources [5].

iMotions includes an integrated statistics module that enables users to compute descriptive metrics, paired and independent t-tests, ANOVAs, correlation matrices, and basic regression on biometric streams collected during a single study session [37, 38]. However, the platform does not provide the full suite of advanced modeling, multilevel analyses, or meta-analytic procedures available in dedicated statistical packages such as SPSS, R, or SAS.

Researchers wishing to compare results across multiple independent studies must manually export raw data, typically as per-participant CSV files, and analyze these files externally [39]. Each export follows a study-specific schema, and there is no native cross-study harmonization or batch-export feature. This limitation makes large-scale comparative analyses more time-consuming and dependent on external statistical software, particularly for projects involving longitudinal or multi-site research designs.

The system developed in this thesis directly addresses this identified gap. While *iMotions* efficiently handles data acquisition and intra-study analysis, it lacks automated workflows for cross-study evaluation and standardized visualization exports. The proposed Python-based tool complements *iMotions* by enabling analyses across multiple independent studies and providing reproducible, publication-ready visualizations. This modular approach allows researchers to leverage *iMotions*' strengths in data collection while extending analytical capabilities beyond its native limitations.

3.2 AP5 Virtual Assistant

The *AP5 Virtual Assistant* is a not yet publicly released virtual assistant that was developed as part of the fifth work package within the *IntelMOD* project to support functional cost splitting. It serves as a digital tool for the systematic recording of modernization measures. In the context of this thesis, the assistant is used to evaluate and test the developed software. Furthermore, it will serve as a basis for subsequent empirical studies planned within the *IntelMOD* project.

There are two versions of the assistant:

3.2.1 Basic Chat Mode

In Basic Chat Mode, the AP5 Virtual Assistant answers simple questions to support users in filling out the form application developed by a project partner. The assistant is based on a Retrieval-Augmented Generation (RAG) system, which draws on an external knowledge database to provide accurate and context-specific answers. To ensure transparency and traceability, all responses are accompanied by a reference to the corresponding sources. Figure 3.1 shows the assistant operating in this mode.

3.2.2 Complex Chat Mode

Complex Chat Mode offers the entire functional cost splitting process as an alternative to the form application. It combines a language model with a fixed script, allowing the assistant to guide users through the process conversationally while collecting the required data in the background and controlling the corresponding API calls. Figure 3.2 illustrates this interaction mode.

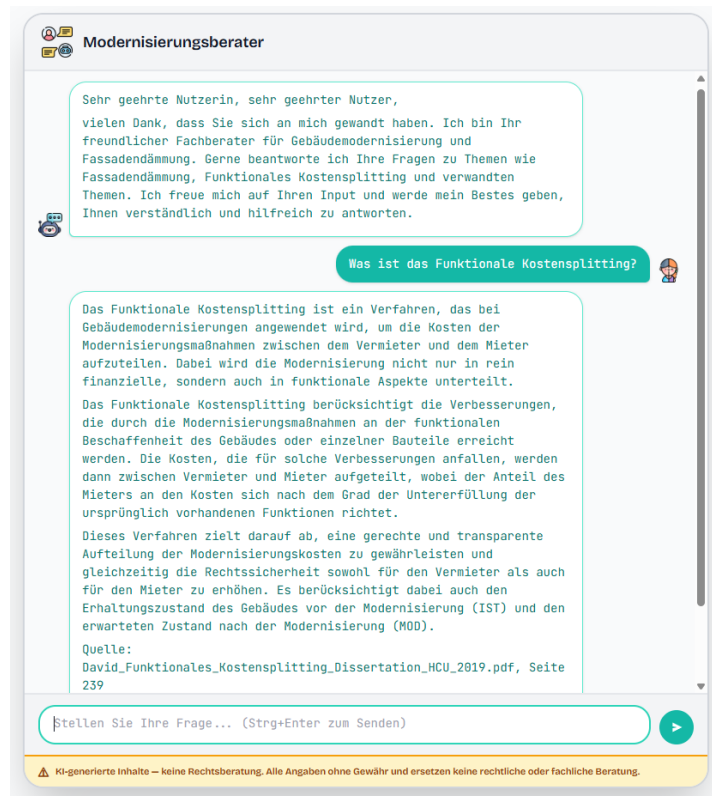


Figure 3.1: The Virtual Assistant operating in Basic Chat Mode



Figure 3.2: The Virtual Assistant operating in Complex Chat Mode

4 Software Design

This chapter outlines the architectural and technical foundations of the study analysis system, detailing its requirements, chosen technologies, and overall system design.

4.1 System Overview

The study analysis software is a Python-based tool designed to support researchers in evaluating and visualizing experimental studies that were conducted in iMotions. Users can define study parameters, such as the study design, through a flexible configuration interface, initiate automated statistical analyses, and export generated visualizations for reporting purposes.

The system targets researchers with varying levels of programming experience, providing both automated workflows and customizable analysis pipelines.

4.2 Requirements Analysis

A thorough understanding of both functional and non-functional requirements is essential for designing a system that meets the needs of researchers conducting study evaluations. The following analysis identifies the core capabilities the system must deliver as well as the quality attributes it must uphold.

Functional requirements define the specific features and operations the system must support, while non-functional requirements address broader concerns such as performance, and modularity. Together, these requirements form the basis for subsequent design and implementation decisions.

4.2.1 Functional Requirements

The system must support flexible configuration of study design parameters, allowing researchers to adapt experimental settings without code modifications.

It should perform automated study evaluations and compute relevant statistical measures from collected data.

Additionally, the system must enable export functionality for visualizations in common formats, facilitating integration into publications or presentations.

4.2.2 Non-Functional Requirements

The architecture should prioritize modularity and extensibility, ensuring that new features or analysis methods can be added with minimal impact on existing components.

Performance and reliability are critical—the system must process datasets efficiently while maintaining consistent results across multiple runs.

4.3 Technologies

The implementation utilizes Python as the primary programming language due to its established dominance in data analysis and scientific computing. According to the official Python Developers Survey 2023 conducted by the Python Software Foundation and JetBrains, “Python is regularly recommended as a good programming language for beginners due to its readable syntax, wide applicability (from data science to web development), and great community.” [40].

The graphical user interface is implemented with Tkinter, Python’s standard GUI toolkit. Since Tkinter ships with the standard library, it introduces no additional installation overhead for researchers running the tool, which supports the goal of accessibility for users with limited programming experience. According to comparisons of Python GUI frameworks, Tkinter is considered the best choice for beginners and small-scale tools because it requires no extra installation, has a tiny footprint, and runs on Windows, macOS and Linux with a stable API [41]. For research tools where rapid deployment and minimal dependency requirements are prioritized over advanced UI customization, Tkinter provides a pragmatic trade-off that aligns with the project’s accessibility goals.

4.4 System Architecture

The system’s module structure follows the classic three-layer application architecture: a presentation layer, an application layer, and a data layer [42]. Figure 4.1 presents an overview of this architecture.

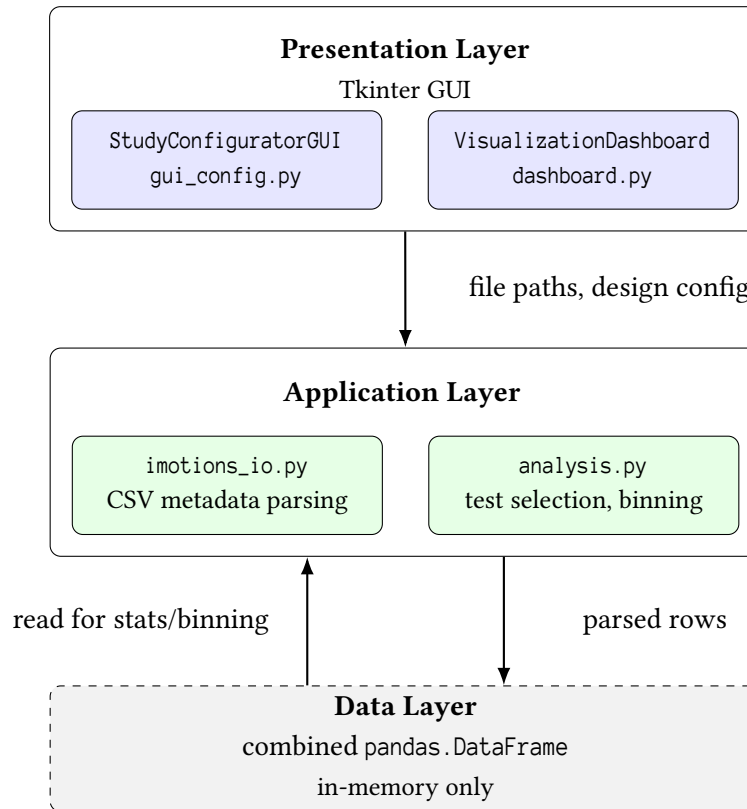


Figure 4.1: Three-layer architecture of the study analysis system

4.4.1 Presentation Layer

Both GUI-facing classes belong here: `StudyConfiguratorGUI` (`gui_config.py`) collects the study design, group/condition setup, and CSV file selection, while `VisualizationDashboard` (`dashboard.py`) renders the resulting tabbed interface. Neither performs analysis itself, their role is to collect information from the researcher and present information back, handing everything else to the Application Layer [42].

4.4.2 Application Layer

Also called the business logic layer, this is where the system’s processing happens, standing between the Presentation and Data Layers [42]. It is split into two focused modules:

- `imotions_io.py` reads the metadata header at the top of every iMotions export – locating where the actual measurement rows begin, and extracting the respondent’s name and the study name – producing the parsed rows that make up the Data Layer,

- `analysis.py` reads from the Data Layer while the dashboard is rendering: it selects and runs the appropriate statistical test for a given number of groups or conditions, and bins a respondent's recording into smoothed time windows for the Time Course view.

Neither module touches Tkinter, Matplotlib, or any other GUI state, which is what allows both to be tested directly and independently of the interface built on top of them.

4.4.3 Data Layer

Normally this layer stores, manages, and retrieves an application's data via a database [42]. In this case it only consists of the single combined `pandas.DataFrame` assembled by the Application Layer from all loaded CSV files. The system has no persistence requirement between sessions, so an in-memory `DataFrame` fulfills the Data Layer's role without the overhead of an actual database.

5 Software Implementation

This chapter describes how the system presented in the previous chapter was implemented. It covers the tools used to build and version it, how a researcher sets it up on their own machine, and the internal logic behind the configuration interface and the visualization engine.

5.1 Development Environment and Tools

The project was developed and version-controlled in a private repository on *GitHub*, which provided a central remote for the codebase and a history of incremental changes throughout development.

Visual Studio Code was used as the primary editor with **Python 3.8** being the runtime version chosen for development.

Dependencies are managed with *pip* and isolated in a Python *venv* (virtual environment). Using a virtual environment keeps the project's package versions separate from the system-wide Python installation, which avoids version conflicts with other projects and makes the setup reproducible on a different machine: anyone can recreate the exact same environment from a `requirements.txt` file rather than relying on whatever happens to be globally installed.

The project relies on a set of packages, each responsible for one stage of the pipeline:

- **pandas** for loading, cleaning and aggregating the *iMotions* CVS exports,
- **NumPy** for numerical array operations,
- **SciPy** for chosen statistical tests described in [Section 2.3](#),
- **Matplotlib** and **Seaborn** for rendering the dashboard's charts,
- **Tkinter** for the GUI itself.

5.2 Setup and Installation

The system has no server-side or database component, so setup only involves preparing a local Python environment. The following steps outline the installation procedure:

1. **Copy the project files:**

Transfer the project directory from the provided USB storage device to your local machine.

2. **Verify the Python environment:**

Ensure Python 3.8 or higher is installed:

```
1 python --version
```

3. **Create and activate a virtual environment:**

```
1 python -m venv venv
2 # Windows
3 venv\Scripts\activate
4 # macOS / Linux
5 source venv/bin/activate
```

4. **Install dependencies via pip:**

```
1 pip install -r requirements.txt
```

5. **Launch the application:**

```
1 python gui_config.py
```

This opens the Study Configurator window described in the following section.

5.3 Configuration Interface

The `StudyConfiguratorGUI` class (`gui_config.py`) is the entry point of the application and the only screen the researcher interacts with before analysis begins. It is deliberately kept free of any analysis logic, so that the configuration workflow stays simple regardless of how the processing engine or the visualization dashboard evolve.

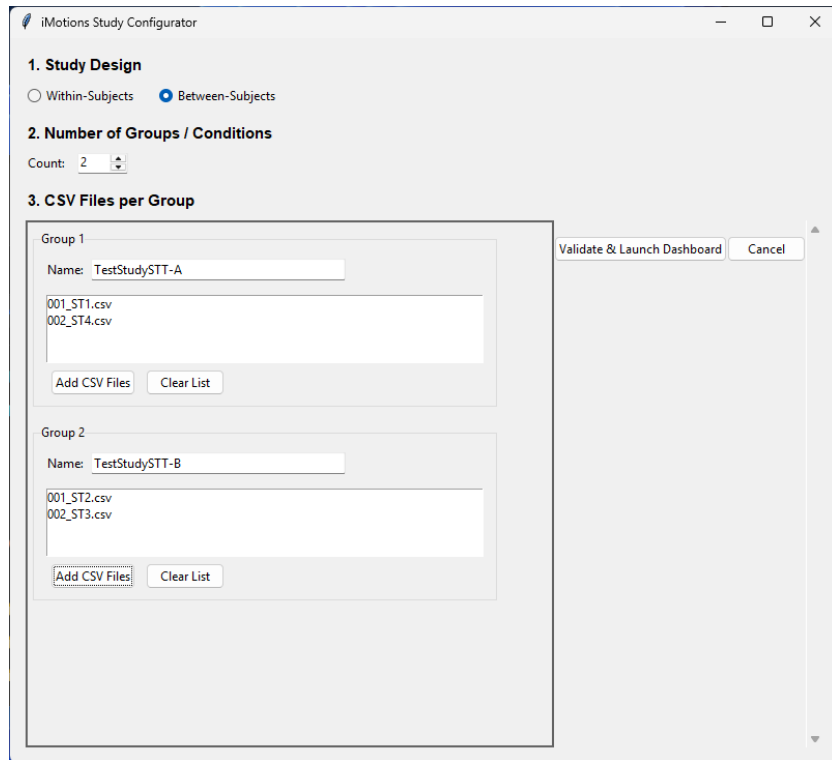


Figure 5.1: The Study Configurator interface

From this window, the researcher can:

- choose the **study design**, either within-subjects or between-subjects, via radio buttons;
- set the **number of groups or conditions** using a spin box, which dynamically rebuilds the form below it;
- for each group or condition, **add one or more iMotions CSV exports** through a file dialog, or clear the current selection;
- optionally **rename** a group or condition, overriding its default label;
- **validate and launch** the dashboard once every group has at least one file assigned.

On the backend, changing the design type or the group count triggers `rebuild_group_fields()`, which destroys and redraws one labeled frame (with a listbox, an editable name field, and Add/Clear buttons) per group, so the form always matches the current configuration.

A small usability feature is built into file selection: as soon as the first CSV is added to a group, its display name is automatically filled in from the `#Study` name entry in that file's *iMotions* metadata header. This removes a manual, error-prone naming step while still letting the researcher type a different name if they prefer one:

Codeblock 5.1: Auto-filling a group's name from the CSV's study-name metadata (gui_config.py)

```
1 if added_any:
2     name_var = self.group_name_vars[group_idx]
3     current = name_var.get().strip()
4     if not current or current == self.group_labels[group_idx]:
5         first_file = self.group_files[group_idx][0]
6         study_name = find_study_name(first_file)
7         if study_name:
8             name_var.set(study_name)
```

When the researcher clicks *Validate & Launch Dashboard*, `validate_and_launch()` first checks that every group has at least one file, then hands each file to the processing engine described in Section 5.4. The resulting combined dataset, together with the design type and the group labels, is packaged into a plain dictionary and passed directly to `VisualizationDashboard`, which is opened in a new Toplevel window.

5.4 Processing Engine

Once the configuration interface hands off a list of files, the processing engine is responsible for turning raw iMotions exports into the single, clean, normalized `DataFrame` that both the statistical tests and the dashboard operate on. This logic lives in two modules, `imotions_io.py` and `analysis.py`, which are deliberately kept free of any Tkinter or Matplotlib dependency so that they can be unit-tested in isolation from the GUI.

5.4.1 Reading iMotions exports

Each iMotions export begins with a proprietary metadata header (study name, respondent information, sensor configuration) followed by a `#DATA` marker that precedes the actual measurement rows. `imotions_io.py` locates this marker to skip the header, and separately scans the same metadata block for the respondent's name and the study name:

Codeblock 5.2: Locating the data section and extracting the respondent and study name from an iMotions CSV header (`imotions_io.py`)

```
1 def find_data_start(filepath):
2     with open(filepath, "r", encoding="utf-8-sig") as f:
3         for line_num, line in enumerate(f):
```

```

4         if line.startswith("#DATA"):
5             return line_num + 1
6     return 0
7
8 def find_respondent_name(filepath):
9     with open(filepath, "r", encoding="utf-8-sig") as f:
10         for line in f:
11             if line.startswith("#Respondent_Name"):
12                 parts = line.split(",")
13                 if len(parts) > 1 and parts[1].strip():
14                     return parts[1].strip()
15             if line.startswith("#DATA"):
16                 break
17     return Path(filepath).stem

```

Each loaded file is tagged with a Group column (from the configuration, see Section 5.3) and a Respondent column (from the metadata header), and all files are concatenated with pandas into a single combined dataset.

5.4.2 Temporal binning

Per-frame facial-expression data is inherently noisy, so the Time Course view in the dashboard (Section 5.6) does not plot raw frames directly. Instead, the processing engine splits each respondent's recording into a fixed number of equal-width time windows and averages the selected channels within each window:

Codeblock 5.3: Binning a respondent's recording into smoothed time windows (analysis.py)

```

1 def binned_time_series(df, value_cols, n_bins=120):
2     durations = df.groupby(["Group", "Respondent"])["Timestamp"].apply(lambda s:
3         s.max() - s.min())
4     max_duration = durations.max()
5     bin_width = max_duration / n_bins
6
7     parts = []
8     for (group, respondent), sub in df.groupby(["Group", "Respondent"]):
9         sub = sub.sort_values("Timestamp")
10        t0 = sub["Timestamp"].min()
11        t_max = sub["Timestamp"].max()
12
13        n_local_bins = max(1, int(-(t_max - t0) // bin_width))

```

```

13     edges = [t0 + i * bin_width for i in range(n_local_bins + 1)]
14     bins = pd.cut(sub["Timestamp"], bins=edges)
15
16     agg = sub.groupby(bins, observed=True)[value_cols].mean()
17     bin_centers = agg.index.map(lambda iv: (iv.left + iv.right) / 2)
18     agg = agg.reset_index(drop=True)
19     agg["Time(s)"] = (pd.Series(bin_centers).astype(float) - t0) / 1000.0
20     agg["Group"] = group
21     agg["Respondent"] = respondent
22     parts.append(agg.dropna(subset=value_cols, how="all"))
23     return pd.concat(parts, ignore_index=True)

```

For the “most dominant categorical emotion” view in the dashboard, the same binned values are compared row-wise to determine which categorical channel has the highest average value in each time window.

5.5 Test Selection

To decide which statistical test type mentioned in Section 2.3 applies, the software first tests each group’s values for normality using the Shapiro-Wilk test [43], one of the most widely used and, for small sample sizes in particular, one of the most powerful tests for departure from normality [25].

If the Shapiro-Wilk test rejects normality for any group, or if a group has too few observations to test reliably, the software falls back to the corresponding non-parametric test rather than risk violating the parametric test’s assumptions. This conservative default is particularly relevant because per-respondent facial-expression aggregates are typically based on small sample sizes, where normality is both hard to verify and easy to violate.

5.6 Visualization Dashboard

The VisualizationDashboard class (dashboard.py) receives the combined, normalized dataset from the processing engine and renders it as a tabbed interface with three tabs: Dimensional Emotions, Categorical Emotions, and Time Course. The same three tabs are used for both study designs; what changes between them is which statistical test is run (Section 5.5) and, for within-subjects, one additional chart described below.

5.6.1 Lazy tab construction

All three tabs are built through a `ttk.Notebook`, but their content is not created immediately when the dashboard opens. A Matplotlib canvas created inside a still-hidden notebook page can be given a size of 0x0 pixels and render blank even after the tab is later selected, so each tab's charts are instead built the first time that tab actually becomes visible:

Codeblock 5.4: Building each tab's content lazily, only once it is actually selected
(`dashboard.py`)

```
1 tab_builders = {
2     str(dim_tab): lambda: self._build_emotion_tab(dim_tab, available_dim,
3         "dimensional"),
4     str(cat_tab): lambda: self._build_emotion_tab(cat_tab, available_cat,
5         "categorical"),
6     str(time_tab): lambda: self._build_timeline_tab(time_tab, available_dim,
7         available_cat),
8 }
9 built_tabs = set()
10
11 def on_tab_changed(_event=None):
12     selected = notebook.select()
13     if selected and selected not in built_tabs:
14         built_tabs.add(selected)
15         tab_builders[selected]()
16
17 notebook.bind("<<NotebookTabChanged>>", on_tab_changed)
18 on_tab_changed()
```

5.6.2 Dimensional and Categorical Emotions tabs

Each of these tabs lets the researcher pick an emotion channel from a dropdown and shows two linked panels: a bar chart of the mean value per test person, colored by group or condition respectively, and a box plot comparing those means between groups or conditions, annotated with the result of the automatically selected statistical test. The dimensional tab additionally shows a fixed Valence x Engagement scatter map, giving a view of where each test person falls on both dimensions at once.

For a within-subjects design, a third panel is added: a slope chart connecting each respondent's own value across conditions, colored depending on whether that person's value increased, decreased, or stayed the same. This directly shows whether individual participants moved in

a consistent direction. Only respondents with a value in every condition can be included in this chart and in the paired test itself, and the panel title reports how many respondents that applied to, so incomplete pairings are never silently hidden.

5.6.3 Time Course tab

This tab plots the binned time series produced by the processing engine in Section 5.4. Two modes are available: a single chosen emotion channel plotted as one line per respondent over time, or, for categorical emotions, a “most dominant emotion” view that shows, per respondent, which categorical channel had the highest value in each time window.

5.6.4 Design-aware labeling

Rather than duplicating the dashboard for each study design, a single `group_label` attribute is threaded through all axis labels, legends, and chart titles, and an `is_within` flag determines which statistical test function from Section 5.5 is called. This keeps the two designs visually and behaviorally consistent while avoiding a second, parallel implementation of the same charts.

6 Experimental Evaluation

This chapter presents an experimental study designed to evaluate the analysis system developed as part of this thesis. The study serves as a proof-of-concept, demonstrating that the software can successfully process and visualize performance data across varying model configurations. Specifically, the experiment investigates the trade-offs between transcription accuracy and system responsiveness for two German Whisper model variants, embedded within a realistic voice assistant interaction.

6.1 Study Objectives

The primary objective of this experimental study was to evaluate the performance characteristics of different German Whisper model sizes, specifically comparing *whisper-tiny-german-1224* [44] against *whisper-large-v3-turbo-german* [45] by primeline using *faster-whisper* [46], within the context of a voice assistant. The core research question focused on identifying the optimal balance between transcription accuracy, and responsiveness.

To investigate these performance trade-offs, three hypotheses were formulated.

H1: The large model achieves a better accuracy in transcribing.

H2: On average, the tiny model transcribes the voice inputs faster than the large model.

H3: Participants show significantly more engagement when using the large model.

6.2 Participants

Given the exploratory nature of this pilot study, the participant pool was intentionally limited to four individuals. These participants were recruited to interact with the *AP5 Virtual Assistant* in a controlled environment. Due to the small sample size, the demographic details regarding age and occupation were not the primary focus. Instead, the emphasis was placed on their ability to provide consistent voice input under the experimental conditions. The study utilized a between-subjects design, where participants were assigned to one of two

groups to ensure no single user experienced both model variations simultaneously, thereby preventing fatigue from skewing the results.

6.3 Setup and Procedure

The study was conducted using the *AP5 Virtual Assistant* as mentioned in Section 3.2, operating in its Basic Chat Mode. A voice input feature was added to the chat interface, as illustrated in Figure 6.1.

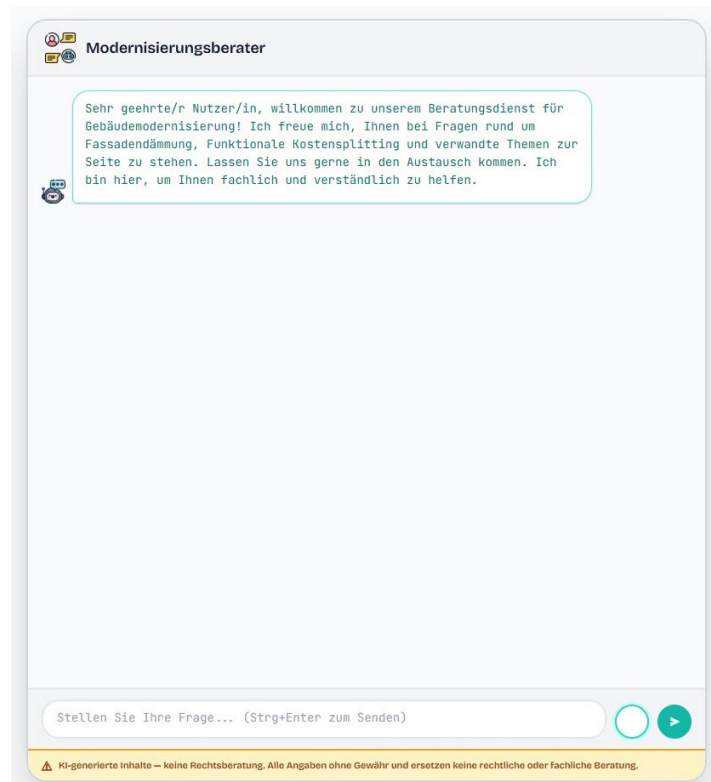


Figure 6.1: The Virtual Assistant operating in Basic Chat Mode, expanded with the voice input feature

The setup involved two distinct groups:

- **Group A** interacted with the assistant powered by the tiny speech-to-text model.
- **Group B** interacted with the assistant powered by the large speech-to-text model.

Participants were provided with predefined questions based on a golden set created by project partners (see Appendix 9), which covers the most likely asked questions about functional cost splitting. Their task was to ask these questions via voice input to the assistant by pressing

a designated button, as seen next to the send button in Figure 6.1, to activate the audio recording, and pressing it a second time to stop it and initiate the transcription. The whole study was conducted in German language, each session lasting approximately 20 minutes.

6.4 Data Sources

This study collects multiple data streams to evaluate the performance and user experience of the speech-based interaction system. Each source was selected to capture complementary dimensions of system quality and user perception.

- **Demographic Questionnaire:** Participants completed a brief demographic survey collecting age, gender, language background, and prior experience with chatbots. This enables analysis of whether usability and performance vary across user subgroups.
- **iMotions:** Video recordings were captured using iMotions software with integrated AFFDEX [47] facial expression analysis. This provides objective measures of emotional responses during task completion [48].
- **Transcribed Questions:** All spoken user queries were transcribed and stored together with associated metadata, including timestamps, recognized text, and response latency. This allows both the accuracy of the speech recognition and the temporal characteristics of the interaction to be analyzed in detail.
- **Usability Survey:** Participants completed a German translation [49] of the System Usability Scale (SUS) [19] with a focus on the speech-to-text input component. The survey was supplemented with three more questions accessing satisfaction of the recognition accuracy and response speed, and trust in the system.

The Speech User Interface Service Quality (SUISQ) [50] questionnaire was also considered as a potential evaluation tool. However, it was not adopted for this study, as it largely overlaps with constructs already captured by the SUS and is primarily designed for voice assistants with auditory feedback. Furthermore, this study evaluates only the speech-to-text output itself and does not investigate the quality of the chatbot’s generated response.

6.5 Data Analysis

The collected data were processed and analyzed using the proposed software, which automated the parsing, evaluation, and visualization of each data stream.

Facial expression data exported from iMotions were automatically parsed by the software to generate comparative visualizations and statistical tests, allowing emotional engagement to be compared across the two model groups.

Transcription accuracy was evaluated using a dedicated script based on the JiWER [51] library, which computed the Word Error Rate (WER) [52] between the recognized text and a manually verified ground truth for each transcribed question. In addition, the response speed metadata was compared across groups to assess differences in response latency between the tiny and large model configurations.

Finally, a separate script processed the results of the usability survey, computing SUS scores as well as the responses to the three supplementary items on recognition accuracy, response speed, and trust in the system, to provide a quantitative measure of perceived user experience.

7 Results

This chapter presents the results obtained from evaluating the implemented software system and from the empirical speech-to-text study conducted with it.

7.1 Software Evaluation

This section evaluates the software independently of any specific study’s findings, focusing on how it performs and what it produces. Section 7.1.1 reports processing time and memory usage across varying respondent counts, while Section 7.1.2 presents the tables and charts the software generates for a between-subjects study.

7.1.1 Processing Performance

To evaluate the scalability of the proposed software, a synthetic benchmarking approach was used. All four base recordings exported from *iMotions* were loaded and used as templates to generate synthetic datasets of varying size: the four base recordings were cycled through and duplicated, with independent Gaussian jitter ($\sigma = 2.0$) added to each dimensional and categorical emotion channel per duplicate, in order to avoid identical, degenerate copies while preserving the overall structure and volume of a real recording. Each duplicate inherited the real group label of its source recording, as determined by `find_study_name`, ensuring the synthetic datasets preserved the same group structure as the actual study. Benchmarks were run for $n \in \{4, 8, 20, 52\}$ synthetic respondents.

Table 7.1 summarizes the measured stage timings and peak memory usage for each dataset size. Total processing time ranged from 0.78 s at $n = 4$ respondents to 4.90 s at $n = 52$ respondents. Peak memory usage ranged from 39.4 MB at $n = 4$ to 497.9 MB at $n = 52$, corresponding to approximately 9.6–9.9 MB per respondent across all tested sizes. Among the individual pipeline stages, the dominant-emotion timeline rendering accounted for 29.8 % of total processing time at $n = 4$ and 57.3 % at $n = 52$.

Figure 7.1 shows total processing time and peak memory usage in relation to respondent count.

Table 7.1: Benchmark results: timings (s) and peak memory (MB) as a function of synthetic respondent count

n	Build	Comparison	Dim. Map	Timeline	Dominant	Total	Peak Mem.
4	0.028	0.313	0.083	0.127	0.233	0.784	39.4
8	0.053	0.315	0.083	0.183	0.389	1.024	77.5
20	0.130	0.467	0.099	0.367	1.018	2.082	192.2
52	0.339	0.721	0.132	0.902	2.806	4.900	497.9

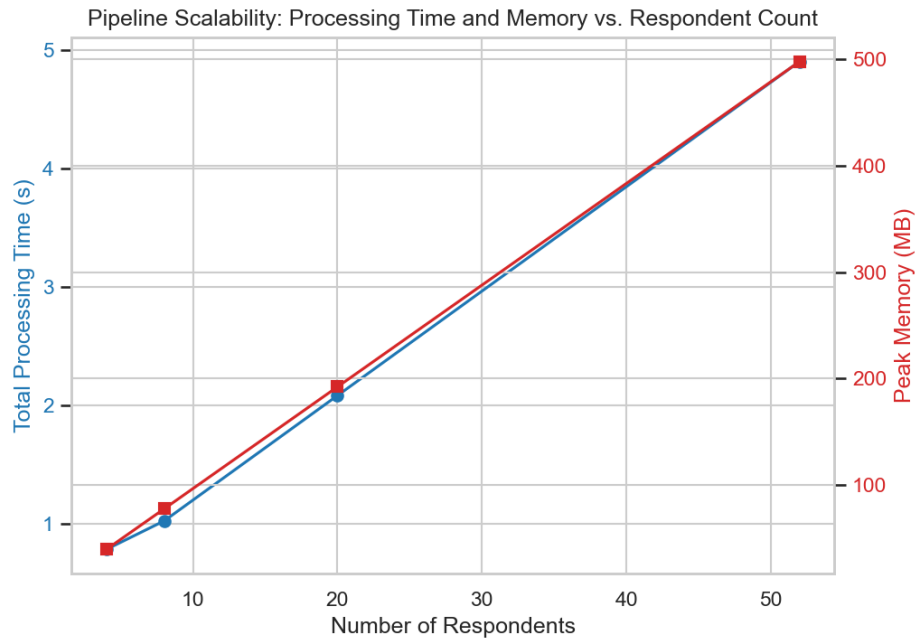


Figure 7.1: Total processing time (left axis) and peak memory usage (right axis) in relation to the number of synthetic respondents, ranging from 4 to 52

7.1.2 Generated Visualizations

This section shows the tables and charts the software generates for a between-subjects study, using the speech-to-text evaluation dataset as an example.

Figure 7.2 presents mean FER-derived Engagement scores per respondent and group, alongside the between-group comparison and the automatically selected statistical test.

Beyond aggregate comparisons, the software also visualizes how a given emotion channel develops over the course of the recording for each respondent, as shown for Engagement in Figure 7.3.

In addition to single-emotion views, the dashboard maps each respondent's mean Valence

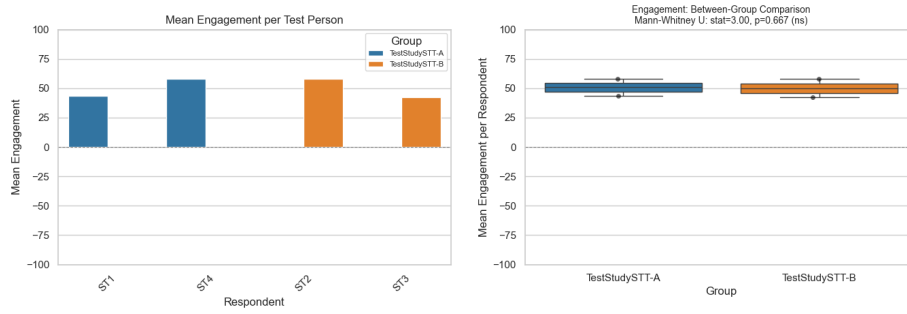


Figure 7.2: Mean engagement per participant and group comparison with automatically selected statistical test

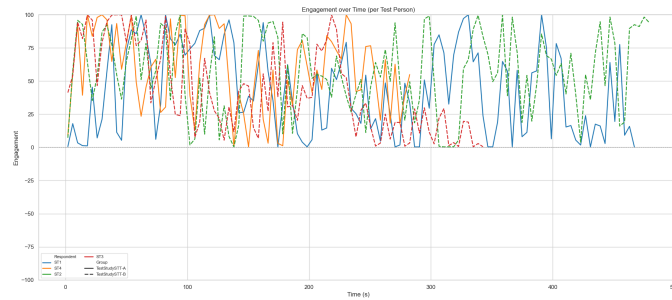


Figure 7.3: Engagement over time for all participants

against their mean Engagement, positioning every participant within a two-dimensional emotional space, as shown in Figure 7.4.

The same set of visualizations is available for categorical emotions. As an example, Figure 7.5 shows the per-participant comparison and group test for Joy, following the same layout used for Engagement above.

Finally, rather than tracking a single emotion, the software can identify which categorical emotion was most dominant at each point in time for every respondent, as shown in Figure 7.6.

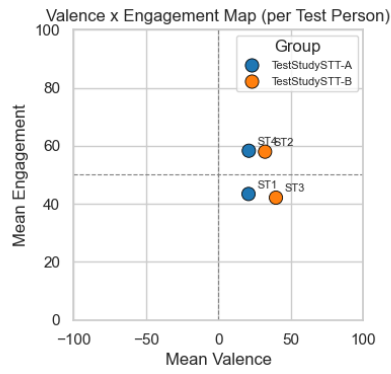


Figure 7.4: Dimensional emotion space per participant

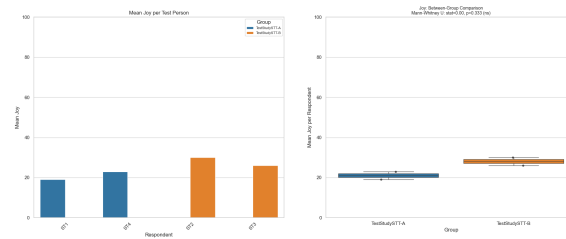


Figure 7.5: Mean joy per participant and group comparison with automatically selected statistical test

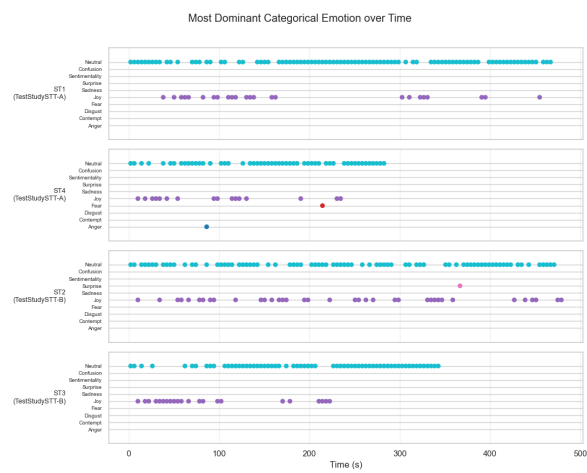


Figure 7.6: Most dominant categorical emotion per participant

7.1.3 Supplementary Within-Subjects Case Study

To demonstrate that the software also supports within-subjects designs, this section presents results from a supplementary study conducted by Kilian Hildebrandt, comparing participant acceptance of the AP5 Virtual Assistant’s Basic Chat Mode and Complex Chat Mode (Section 3.2). With a total of 16 participants, each participant used both modes, making this a within-subjects comparison.

Figure 7.7 shows mean engagement per participant, the between-condition comparison with automatically selected statistical test, and the paired per-participant comparison across both modes.

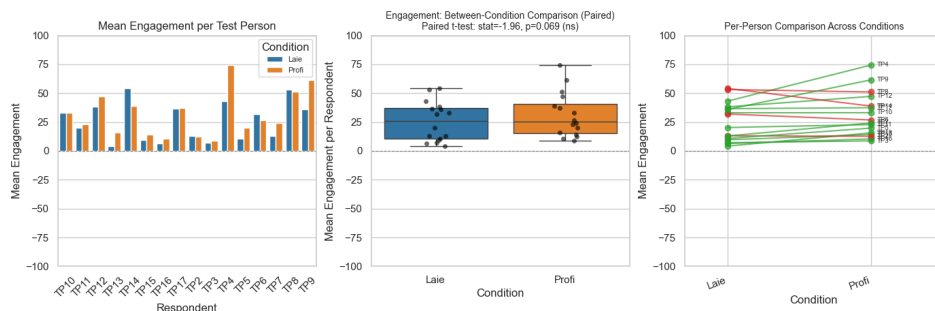


Figure 7.7: Mean engagement per participant and group comparison with automatically selected statistical test, extended with condition comparison for each participant

Figure 7.8 shows the most dominant categorical emotion over time for two example participants, illustrating how the software groups a single respondent’s conditions together for direct comparison in within-subjects designs.



Figure 7.8: Most dominant categorical emotion per condition grouped by participant, exemplary for two participants

7.2 Study Findings

This section reports the empirical results of the speech-to-text study, covering the usability survey, transcription accuracy and duration, and the statistical tests conducted for each hypothesis.

7.2.1 Usability Survey Results

Table 7.2 presents the SUS scores obtained from four respondents, two in each group. In Group A, respondent ST1 scored 70.0 and respondent ST4 scored 62.5, giving a group mean of 66.25 and a standard deviation of 5.30. In Group B, respondent ST2 scored 66.7 and respondent ST3 scored 85.0, giving a group mean of 75.83 and a standard deviation of 12.96.

Table 7.2: System Usability Scale (SUS) scores by respondent and group

Group	Respondent	SUS Score
A	ST1	70.0
A	ST4	62.5
B	ST2	66.7
B	ST3	85.0
A	<i>Mean (SD)</i>	66.25 (5.30)
B	<i>Mean (SD)</i>	75.83 (12.96)

Table 7.3 presents mean ratings for three items related to speech recognition, rated on a 1–5 scale, separately for Group A and Group B. For “Satisfied with recognition accuracy,” Group A had a mean rating of 1.50 and Group B had a mean rating of 4.50. For “Satisfied with recognition speed,” Group A had a mean rating of 4.00 and Group B had a mean rating of 4.50. For “Trust in the speech recognition,” Group A had a mean rating of 1.50 and Group B had a mean rating of 3.50.

Table 7.3: Mean speech recognition satisfaction ratings by group (1–5 scale)

Item	Group A	Group B
Satisfied with recognition accuracy	1.50	4.50
Satisfied with recognition speed	4.00	4.50
Trust in the speech recognition	1.50	3.50

7.2.2 Speech-To-Text Accuracy and Duration Results

Table 7.4 presents the Word Error Rate (WER) and mean segment duration. In Group A, respondent ST1 had a WER of 1.516 and a mean duration of 6.89 seconds, while respondent ST4 had a WER of 0.170 and a mean duration of 4.99 seconds, giving a group mean WER of 0.843 with a standard deviation of 0.952 and a group mean duration of 5.94 seconds and a standard deviation of 1.35. In Group B, respondent ST2 had a WER of 0.101 and a mean duration of 5.55 seconds, while respondent ST3 had a WER of 0.064 and a mean duration of 5.51 seconds, giving a group mean WER of 0.082 with a standard deviation of 0.026 and a group mean duration of 5.53 seconds and a standard deviation of 0.02.

Table 7.4: Word Error Rate (WER) and mean duration per respondent and group

Group	Respondent	WER	Mean Duration (s)
A	ST1	1.516	6.89
A	ST4	0.170	4.99
B	ST2	0.101	5.55
B	ST3	0.064	5.51
A	<i>Mean (SD)</i>	0.843 (0.952)	5.94 (1.35)
B	<i>Mean (SD)</i>	0.082 (0.026)	5.53 (0.02)

7.2.3 Statistical Test Results

For each emotion, the software automatically selected and ran the appropriate statistical test based on the distribution of the underlying data. A Mann-Whitney U test found no statistically significant difference in engagement between groups in Figure 7.2, with $U = 3.000$, $p = 0.667$.

In addition, a Mann-Whitney U test conducted for Section 7.2.2 found no statistically significant difference in WER between groups based on Table 7.4, with $U = 4.000$, $p = 0.333$.

8 Discussion

This chapter reflects on the results presented in the preceding chapters, placing them in the context of the original research questions and requirements. It begins with an interpretation of both the software evaluation outcomes and the empirical STT study findings. Following this, the inherent strengths of the implemented system are discussed, highlighting how specific architectural and functional choices contributed to meeting the stated goals. The chapter concludes with an honest assessment of limitations encountered during development and evaluation, alongside concrete directions for future work.

8.1 Interpretation

The following subsections examine two distinct aspects of the results. First, interpretation focuses on the software evaluation itself: how well the system achieved its functional and non-functional objectives and what the performance benchmarks reveal about its practical viability. Second, reflection addresses the speech-to-text study, distinguishing between statistical non-significance and methodological constraints such as sample size.

8.1.1 Software System Reflection

One key finding from the results is that the system always produces the exact same output when given the same input files. The same CSV files, under the same settings, always lead to the same values, the same statistical test, and the same charts, for example the charts seen in Section 7.1.2. This might seem obvious, but it's exactly what manual scripting struggles with: re-running an analysis by hand often introduces small, undocumented differences like a slightly different filtering step or a test chosen by gut feeling rather than a clear rule. By making every step deterministic and governed by explicit, checkable logic, the system directly addresses the reproducibility problem raised in the introduction.

A second point is how fast this reproducible result is delivered. The performance benchmark in Section 7.1.1 showed that loading four files takes under 1 second, and even a 52-respondent dataset is fully processed in under 5 seconds. In contrast, writing or adapting a manual

analysis script typically takes minutes to hours. So the system doesn't just make analysis reproducible – it makes it fast enough that there's little reason to fall back on quicker but less careful manual approaches.

The benchmark also points to where the system's scalability limits lie. Peak memory usage grew essentially linearly with the number of respondents, at a near-constant ~9.6–9.9 MB per additional respondent, a predictable cost that poses no practical concern for studies of the size typically conducted in this domain. Processing time, however, was increasingly dominated by a single stage: the dominant-emotion timeline chart, which renders one subplot per respondent, grew from 29.8 % of total runtime at four respondents to 57.3 % at 52. This is a straightforward consequence of that stage's per-respondent rendering strategy rather than a property of the underlying data processing, and it suggests that if the system were ever applied to substantially larger participant pools, this specific visualization would be the first target for optimization.

Looking back at the original requirements in Section 4.2, all functional ones are met: flexible study design configuration supports both between-subjects and within-subjects designs, and automated evaluation is provided through the dashboard's charts, automatic test selection and chart exports as PNGs. The non-functional requirement of modularity went beyond what was expected: the original three-layer architecture was later split into independent modules, so that processing and statistics logic could be tested separately from the GUI. Performance and reliability are supported by the benchmark results, though reliability at larger scales hasn't been verified with a real study yet.

The clearest sign that the architecture is extensible is how within-subjects support was added. Instead of rebuilding the dashboard, it only required a small branch driven by a `design_type` setting: swapping one statistical test for another and adding one chart. The existing visualization tabs, normalization logic, and file-loading workflow required only minimal changes to the chart visualizations themselves, such as adding a paired per-participant comparison panel. This is exactly the kind of easy adaptation the architecture was designed for, and the fact that it came at such a low cost validates the original design choices. The supplementary within-subjects case study presented in Section 7.1.3 confirms that this support works in practice, not just in principle, producing the same paired per-participant comparisons and grouped visualizations as the between-subjects results discussed above.

8.1.2 Speech-To-Text Study Reflection

The STT study did not yield a statistically significant result for **H3** about the larger model engaging the users more, which is unsurprising given the small number of participants: the automatic test selection logic in Section 5.5 is deliberately conservative with small samples, and with so few respondents per condition, most tests have limited power to detect

anything but a large effect. This should not be read as evidence that the applications being compared performed equivalently, only that the study, as conducted, was underpowered to distinguish between them. This directly connects to the recommendation in Section 8.3 for more participants in future studies: a larger sample would reduce variance and enable the system to reliably detect moderate effect sizes that were simply invisible at this scale.

The absence of a significant difference does not make the collected data uninformative, however. The usability results in Section 7.2.1 and the speech-to-text accuracy results in Section 7.2.2 both provide descriptive evidence about the applications independent of whether the emotion-based comparison reached significance.

In particular, task completion time was practically indistinguishable between the two applications, while the smaller model was noticeably more error-prone: Group A’s mean WER (0.843, SD = 0.952) was roughly ten times higher than Group B’s (0.082, SD = 0.026). This gap is driven substantially by a single outlier segment from respondent ST1 (WER = 1.516), where the smaller model entered a repetition loop and produced a faulty transcription, repeating the word “untuk” over 200 times (see Appendix 9). Even excluding this outlier, however, ST4’s WER of 0.170 still exceeds both Group B respondents’ individually (0.101 and 0.064), indicating that the smaller model’s higher error rate is not solely an artifact of the ST1 outlier. These descriptive results are broadly consistent with **H1**: the large model’s substantially lower WER supports the expectation of better transcription accuracy, though a Mann-Whitney U test found this difference not to be statistically significant ($U = 4.000$, $p = 0.333$). Notably, $U = 4.000$ represents complete separation between the two groups, both Group A respondents had higher WER than both Group B respondents, indicating the observed pattern is directionally consistent despite not reaching significance: a limitation attributable to the very small sample size ($n = 2$ per group) rather than the absence of a real effect. In contrast, **H2** was not supported: task completion time was practically indistinguishable between the two applications, with no indication that the smaller model transcribed noticeably faster. This suggests that, where resource constraints allow, the larger model should be preferred: it offers a meaningful accuracy advantage at essentially no cost in speed.

The usability results point in a similar direction. Group B’s mean SUS score (75.83) exceeds the commonly cited industry-average benchmark of approximately 68 [53], while Group A’s mean (66.25) falls just below it, suggesting Group B’s application was perceived as more usable overall. However, Group B’s larger standard deviation of 12.96, compared to 5.30 for Group A, shows this was driven substantially by a single respondent’s very high score, so the apparent advantage should be treated cautiously given only two respondents per group. This pattern is consistent with the supplementary speech-recognition ratings: Group B also rated recognition accuracy and trust in the system considerably higher than Group A, while both groups rated recognition speed similarly. Taken together, this suggests that perceived speech recognition quality may be a key driver of the overall usability difference between the two

applications, though this interpretation would need confirmation with a larger sample before being treated as more than a hypothesis for future work.

8.2 Strengths of the Software

Several aspects of the implemented system stand out as genuine strengths rather than incidental features:

- **Automatic, principled test selection.** Instead of always using the same test regardless of the data, the software checks whether each group's or condition's values are normally distributed and then selects an appropriate parametric or non-parametric test: independent-samples tests for between-subjects designs, paired/repeated-measures tests for within-subjects designs (Section 5.5). This directly addresses one of the questionable research practices identified in Chapter 1: picking tests by guesswork rather than by a documented rule.
- **Both major study designs from a shared codebase.** between-subjects and within-subjects studies are both supported without duplicating the visualization or processing logic.
- **A dedicated per-person view for paired data:** The within-subjects slope chart (Chapter 5) shows whether each individual person's score went up or down between conditions: a level of detail that group averages alone can't capture.
- **Modularity that enables unit testing.** By separating the CSV-parsing and statistics logic from the GUI, the core analysis can be tested independently without having to run the interface at all. This solves a problem that is otherwise very hard to address, since analysis and interface logic are usually tightly tangled together in comparable research tools.
- **Measured performance:** Rather than just claiming the tool is "fast enough", Section 7.1.1 provides concrete timing and memory measurements across different synthetic respondent counts, making the performance claims verifiable rather than assumed.

8.3 Limitations and Future Outlook

The current implementation has several limitations that constrain the strength of its conclusions and point toward concrete future work.

- **No integrated survey analysis:** Instruments such as the SUS questionnaire used in the Speech-to-Text study were administered and scored outside the software. Integrating survey collection directly into the configuration interface would let researchers analyze self-report and FER data side by side within the same tool and same reproducible workflow, rather than combining results from two separate systems by hand – reintroducing, at that step, exactly the kind of manual, undocumented process the software is meant to reduce elsewhere.
- **Sample size in the evaluation study:** As discussed above, the speech-to-text study's small participant count limited its statistical power. Future evaluations of both the applications under test and the software itself would benefit from a larger and more diverse participant pool, both to better power the statistical comparisons and to exercise the software's between-subjects and within-subjects paths under more realistic, larger-scale conditions.

9 Conclusion

This thesis set out to answer the research question posed in Chapter 1: what software architecture supports automated, reproducible statistical analysis across both between-subjects and within-subjects study designs without requiring researchers to write custom analysis scripts for each new design, and how can an experimental user study be structured to evaluate the adaptability and reproducibility gains of such an architecture.

This yields a two-part answer. First, a three-layer architecture that isolates data parsing and statistical decision-making from the presentation layer supports automated, reproducible analysis across both designs without duplicated implementation: by parsing *iMotions* exports through a fixed pipeline and selecting statistical tests through an explicit, checkable decision rule rather than manual judgment, the software removes a key source of the questionable research practices that motivated this project. Second, evaluating such an architecture's adaptability and reproducibility gains benefits from a two-tier evaluation structure: a controlled performance benchmark to establish reproducibility and scalability claims quantitatively, paired with at least one applied study per supported design, a between-subjects pilot study and a supplementary within-subjects case study, to confirm that the architecture generalizes in practice rather than only in principle.

The three-layer architecture proved well suited to this goal. Separating the CSV parsing and statistics logic from the presentation layer not only kept the core logic unit-testable in isolation, but also made the system genuinely extensible: within-subjects support was added with only a small, targeted branch in the existing codebase rather than a parallel implementation. The supplementary within-subjects case study in Section 7.1.3 confirmed that this extension works in practice, not merely in principle.

The performance benchmarks showed that the system processes datasets of realistic size with up to 52 synthetic respondents in under five seconds with predictable, near-linear memory growth, making it fast enough to remove any practical incentive to fall back on manual scripting.

The empirical speech-to-text study served as a proof-of-concept for the software rather than a definitive comparison of the two Whisper models. With only two participants per group, none of the three hypotheses reached statistical significance, and this must be read as a limitation of the study's power rather than evidence of equivalence between conditions.

Descriptively, however, the results consistently pointed in one direction: the larger model achieved a substantially lower Word Error Rate, complete group separation in the Mann-Whitney U test, and higher usability and trust ratings, all without a corresponding cost in response time. This pattern is consistent with **H1** and inconsistent with **H2**, while **H3** remains unresolved.

Two limitations constrain the strength of these conclusions and point toward future work. First, the software does not yet integrate self-report instruments such as the SUS questionnaire, which had to be scored outside the tool, reintroducing a manual step at precisely the point the system was designed to eliminate. Second, the evaluation study's small sample size limited statistical power and should be addressed in future evaluations with a larger participant pool to strengthen the empirical comparisons.

In summary, this thesis contributes a modular, reproducible analysis pipeline for *iMotions*-based usability studies, demonstrates its practical viability through both performance benchmarking and a real empirical study, and shows that its architecture generalizes across the two study designs most common in HCI research without duplicated implementation effort.

Bibliography

- [1] M. Kaur and M. Kumar, “Facial emotion recognition: A comprehensive review,” *Expert Systems*, vol. 41, no. 10, e13670, 2024. DOI: <https://doi.org/10.1111/exsy.13670>.
- [2] M. Sajjad et al., “A comprehensive survey on deep facial expression recognition: Challenges, applications, and future guidelines,” *Alexandria Engineering Journal*, vol. 68, pp. 817–840, 2023, ISSN: 1110-0168. DOI: <https://doi.org/10.1016/j.aej.2023.01.017>.
- [3] R. Carvalho et al., “Applications and Reliability of Automatic Facial Emotion Recognition Software: A Five-Year Review (2020-2024),” in *Proceedings of the 11th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-Exclusion*, ser. DSAI ’24, New York, NY, USA: Association for Computing Machinery, 2025, pp. 368–376, ISBN: 9798400707292. DOI: [10.1145/3696593.3696604](https://doi.org/10.1145/3696593.3696604).
- [4] A.-L. Cîrneanu, D. Popescu, and D. Iordache, “New Trends in Emotion Recognition Using Image Analysis by Neural Networks, A Systematic Review,” *Sensors*, vol. 23, no. 16, 2023, ISSN: 1424-8220. DOI: [10.3390/s23167092](https://doi.org/10.3390/s23167092).
- [5] M. Pedersen, *Multimodality in iMotions Software: A Complete Guide to Synchronized Biosensor Research Across iMotions Lab, Online, Education, and Media Analytics*, Mar. 2026. Accessed: Apr. 1, 2026. [Online]. Available: <https://imotions.com/blog/learning/product-guides/multimodality-in-imotions-software/>.
- [6] M. Grunde-McLaughlin et al., “RDoFlow: Automatically assessing under-specified statistical analyses in HCI,” in *Proceedings of the 31st International Conference on Intelligent User Interfaces*, ser. IUI ’26, New York, NY, USA: Association for Computing Machinery, 2026, pp. 683–698, ISBN: 9798400719844. DOI: [10.1145/3742413.3789160](https://doi.org/10.1145/3742413.3789160).
- [7] R. W. Picard, *Affective Computing*. The MIT Press, 2000.
- [8] Y. Wang et al., “A systematic review on affective computing: Emotion models, databases, and recent advances,” *Information Fusion*, vol. 83-84, pp. 19–52, 2022, ISSN: 1566-2535. DOI: <https://doi.org/10.1016/j.inffus.2022.03.009>.
- [9] S. Poria, E. Cambria, R. Bajpai, and A. Hussain, “A review of affective computing: From unimodal analysis to multimodal fusion,” *Information Fusion*, vol. 37, pp. 98–125, 2017, ISSN: 1566-2535. DOI: <https://doi.org/10.1016/j.inffus.2017.02.003>.
- [10] S. Oviatt, “Ten myths of multimodal interaction,” *Commun. ACM*, vol. 42, no. 11, pp. 74–81, Nov. 1999, ISSN: 0001-0782. DOI: [10.1145/319382.319398](https://doi.org/10.1145/319382.319398).

- [11] S. PS and G. Mahalakshmi, "Emotion models: A review," *International Journal of Control Theory and Applications*, vol. 10, no. 8, pp. 651–657, 2017.
- [12] P. Ekman, "An argument for basic emotions," *Cognition & emotion*, pp. 169–200, 1992.
- [13] R. Plutchik, "The nature of emotions: Human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice," *American Scientist*, vol. 89, no. 4, pp. 344–350, 2001, ISSN: 00030996. [Online]. Available: <http://www.jstor.org/stable/27857503>.
- [14] J. A. Russell, "A circumplex model of affect," *Journal of Personality and Social Psychology*, vol. 39, no. 6, pp. 1161–1178, 1980, ISSN: 1939-1315(Electronic),0022-3514(Print). DOI: [10.1037/h0077714](https://doi.org/10.1037/h0077714).
- [15] J. A. Russell and L. F. Barrett, "Core affect, prototypical emotional episodes, and other things called emotion: Dissecting the elephant," *Journal of Personality and Social Psychology*, vol. 76, no. 5, pp. 805–819, 1999, ISSN: 1939-1315. DOI: [10.1037/0022-3514.76.5.805](https://doi.org/10.1037/0022-3514.76.5.805).
- [16] I. S. MacKenzie, *Human-Computer Interaction: An Empirical Research Perspective*. Morgan Kaufmann, 2013, ISBN: 978-0-12-405865-1.
- [17] M. M. Bradley and P. J. Lang, "Measuring emotion: The self-assessment manikin and the semantic differential," *Journal of Behavior Therapy and Experimental Psychiatry*, vol. 25, no. 1, pp. 49–59, 1994, ISSN: 0005-7916. DOI: [https://doi.org/10.1016/0005-7916\(94\)90063-9](https://doi.org/10.1016/0005-7916(94)90063-9).
- [18] S. Koelstra et al., "DEAP: A Database for Emotion Analysis ;Using Physiological Signals," *IEEE Transactions on Affective Computing*, vol. 3, pp. 18–31, Dec. 2011. DOI: [10.1109/T-AFFC.2011.15](https://doi.org/10.1109/T-AFFC.2011.15).
- [19] J. Brooke, "SUS: A quick and dirty usability scale," *Usability Eval. Ind.*, vol. 189, Nov. 1995.
- [20] A. Aruna Gladys and V. Vetrivel, "Survey on multimodal approaches to emotion recognition," *Neurocomputing*, vol. 556, p. 126 693, 2023, ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2023.126693>.
- [21] M. Ramaswamy and S. Palaniswamy, "Multimodal emotion recognition: A comprehensive review, trends, and challenges," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 14, Oct. 2024. DOI: [10.1002/widm.1563](https://doi.org/10.1002/widm.1563).
- [22] S. Kalateh, L. A. Estrada-Jimenez, S. Nikghadam-Hojjati, and J. Barata, "A Systematic Review on Multimodal Emotion Recognition: Building Blocks, Current State, Applications, and Challenges," *IEEE Access*, vol. 12, pp. 103 976–104 019, 2024.
- [23] S. Ding, L. Ma, and H. Li, "Multimodal physiological signal emotion recognition based on multi-head cross attention with representation learning," *Front Psychiatry*, vol. 16, p. 1 713 559, 2025. DOI: [10.3389/fpsyg.2025.1713559](https://doi.org/10.3389/fpsyg.2025.1713559).

- [24] G. Udaheureka, K. Djouani, and A. M. Kurien, "Multimodal Emotion Recognition Using Visual, Vocal and Physiological Signals: A Review," *Applied Sciences*, vol. 14, no. 17, 2024, ISSN: 2076-3417. DOI: [10.3390/app14178071](https://doi.org/10.3390/app14178071).
- [25] A. G. Ghasemi and S. Z. Zahediasl, "Normality Tests for Statistical Analysis: A Guide for Non-Statisticians," *International Journal of Endocrinology and Metabolism*, vol. 10, no. 2, pp. 486–489, 2012, ISSN: 1726-9148. DOI: [10.5812/ijem.3505](https://doi.org/10.5812/ijem.3505).
- [26] B. L. Welch, "The generalization of 'student's' problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1-2, pp. 28–35, Jan. 1947, ISSN: 0006-3444. DOI: [10.1093/biomet/34.1-2.28](https://doi.org/10.1093/biomet/34.1-2.28).
- [27] H. B. Mann and D. R. Whitney, "On a Test of Whether One of Two Random Variables Is Stochastically Larger than the Other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947. DOI: [10.1214/aoms/1177730491](https://doi.org/10.1214/aoms/1177730491).
- [28] L. Stahle and S. Wold, "Analysis of variance (ANOVA)," *Chemometrics and Intelligent Laboratory Systems*, vol. 6, no. 4, pp. 259–272, 1989, ISSN: 0169-7439. DOI: [https://doi.org/10.1016/0169-7439\(89\)80095-4](https://doi.org/10.1016/0169-7439(89)80095-4).
- [29] W. H. Kruskal and W. A. Wallis, "Use of Ranks in One-Criterion Variance Analysis," *Journal of the American Statistical Association*, vol. 47, no. 260, pp. 583–621, 1952. DOI: [10.1080/01621459.1952.10483441](https://doi.org/10.1080/01621459.1952.10483441).
- [30] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945, ISSN: 00994987. Accessed: Jul. 25, 2026. [Online]. Available: <http://www.jstor.org/stable/3001968>.
- [31] M. Friedman, "The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937. DOI: [10.1080/01621459.1937.10503522](https://doi.org/10.1080/01621459.1937.10503522).
- [32] R. A. Fisher, "Statistical Methods for Research Workers," in *Breakthroughs in Statistics: Methodology and Distribution*, S. Kotz and N. L. Johnson, Eds., New York, NY: Springer, 1992, pp. 66–70, ISBN: 978-1-4612-4380-9. DOI: [10.1007/978-1-4612-4380-9_6](https://doi.org/10.1007/978-1-4612-4380-9_6).
- [33] R. L. Wasserstein and N. A. Lazar, "The ASA Statement on P-Values: Context, Process, and Purpose," *The American Statistician*, vol. 69, no. sup1, pp. 131–134, 2016. DOI: [10.1080/00031305.2016.1154108](https://doi.org/10.1080/00031305.2016.1154108).
- [34] O. Polacek, A. J. Sporka, and P. Slavik, "Text input for motor-impaired people," *Universal Access in the Information Society*, vol. 16, no. 1, pp. 51–72, Mar. 2017, ISSN: 1615-5297. DOI: [10.1007/s10209-015-0433-0](https://doi.org/10.1007/s10209-015-0433-0).
- [35] G. Almgren Bäck et al., "Speech-to-text intervention to support text production among students with writing difficulties: A single-case study in nordic countries," *Disability and rehabilitation. Assistive technology*, vol. 19, pp. 1–20, May 2024. DOI: [10.1080/17483107.2024.2351488](https://doi.org/10.1080/17483107.2024.2351488).

- [36] F. Masina et al., “Investigating the Accessibility of Voice Assistants With Impaired Users: Mixed Methods Study,” *Journal of Medical Internet Research*, vol. 22, no. 9, e18431, Sep. 2020, ISSN: 1439-4456. DOI: [10.2196/18431](https://doi.org/10.2196/18431).
- [37] iMotions, *Analysis - iMotions*, Aug. 2024. Accessed: Jul. 17, 2026. [Online]. Available: <https://imotions.com/products/imotions-lab/features/analysis/>.
- [38] B. Murray, *Statistical Analysis [Which Test To Use] - iMotions Blog*, Section: Consumer Insights, Sep. 2019. Accessed: Jul. 17, 2026. [Online]. Available: <https://imotions.com/blog/learning/best-practice/statistical-analysis/>.
- [39] iMotions, *Data Export - iMotions*, May 2024. Accessed: Jul. 17, 2026. [Online]. Available: <https://imotions.com/products/imotions-online/data-export/>.
- [40] P. S. Foundation and JetBrains, *Python Developers Survey 2023 Results*. Accessed: Jul. 19, 2026. [Online]. Available: <https://lp.jetbrains.com/python-developers-survey-2023/>.
- [41] C. Wan, *Choosing the Best Python GUI Library: Comparing Tkinter, PyQt, and wxPython*, Jan. 2025. Accessed: Jul. 19, 2026. [Online]. Available: <https://charleswan111.medium.com/choosing-the-best-python-gui-library-comparing-tkinter-pyqt-and-wxpython-1c835746586a>.
- [42] M. Chiamonte, “The Benefits of a Three-Layered Application Architecture,” vFunction, Accessed: Jul. 15, 2026. [Online]. Available: <https://vfunction.com/blog/the-benefits-of-a-three-layered-application-architecture/>.
- [43] S. S. Shapiro and M. B. Wilk, “An analysis of variance test for normality (complete samples),” *Biometrika*, vol. 52, no. 3-4, pp. 591–611, Dec. 1965, ISSN: 0006-3444. DOI: [10.1093/biomet/52.3-4.591](https://doi.org/10.1093/biomet/52.3-4.591).
- [44] primeLine AI Services, *whisper-tiny-german-1224 (Revision 1e00ddc)*, 2024. DOI: [10.57967/hf/3826](https://doi.org/10.57967/hf/3826).
- [45] primeLine AI Services, *whisper-large-v3-turbo-german (Revision 9e7012d)*, 2024. DOI: [10.57967/hf/3827](https://doi.org/10.57967/hf/3827).
- [46] SYSTRAN, *SYSTRAN/faster-whisper: Faster Whisper transcription with CTranslate2*. Accessed: May 14, 2026. [Online]. Available: <https://github.com/SYSTRAN/faster-whisper>.
- [47] M. Bishay, K. Preston, M. Straßus, G. Page, J. Turcot, and M. Mavadati, *AFFDEX 2.0: A Real-Time Facial Expression Analysis Toolkit*, 2022. arXiv: [2202.12059](https://arxiv.org/abs/2202.12059) [cs.CV].
- [48] iMotions, *Facial Expression Analysis - Emotion Detection Software*, May 2026. Accessed: Jul. 20, 2026. [Online]. Available: <https://imotions.com/products/imotions-lab/modules/fea-facial-expression-analysis/>.
- [49] B. Rummel, “System Usability Scale (Translated into German),” Apr. 2013.

- [50] J. Lewis and M. Hardzinski, “Investigating the psychometric properties of the Speech User Interface Service Quality questionnaire,” *International Journal of Speech Technology*, vol. 18, Sep. 2015. DOI: [10.1007/s10772-015-9289-1](https://doi.org/10.1007/s10772-015-9289-1).
- [51] *Jitsi/jiwer*, Jul. 2026. Accessed: Jul. 15, 2026. [Online]. Available: <https://github.com/jitsi/jiwer>.
- [52] A. Morris, V. Maier, and P. Green, “From WER and RIL to MER and WIL: improved evaluation measures for connected speech recognition,” Oct. 2004. DOI: [10.21437/Interspeech.2004-668](https://doi.org/10.21437/Interspeech.2004-668).
- [53] A. Bangor, P. Kortum, and J. Miller, “The System Usability Scale (SUS): An Empirical Evaluation,” *International Journal of Human-Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008. DOI: [10.1080/10447310802205776](https://doi.org/10.1080/10447310802205776).

Appendix

Questions based on the Golden Set

1. Was ist das Funktionale Kostensplitting?
2. Warum wurde das Funktionale Kostensplitting entwickelt?
3. Was bedeutet „nicht-mieterhöhungsrelevant“?
4. Was ist das Zielbaumverfahren nach Aurhammer?
5. Warum ist eine exakte Ermittlung der Erhaltungskosten oft nicht möglich?
6. Welche gesetzlichen Grundlagen gelten bei der Anwendung des Funktionalen Kostensplittings?
7. Wie lassen sich die Qualitätseigenschaften eines Bauteils strukturiert beschreiben?
8. Was sind die wichtigsten Schritte bei der Anwendung des Funktionalen Kostensplittings?
9. Welche bautechnischen Prozesse führen zu Funktionsminderungen, die als Erhaltungsbedarf gelten?
10. Ist ein mangelnder Schallschutz als ein Bauschaden zu verstehen?
11. Warum spielt der Zustand (z. B. Schäden) meines alten Bauteils bei der Berechnung eine Rolle?
12. Wie funktioniert die funktionale Bewertung eines Bauteils?
13. Warum sind manche Funktionen wichtiger für die Berechnung als andere?
14. Was bedeutet es, wenn das neue Bauteil sowohl alte als auch neue Funktionen übernimmt?
15. Warum werden beim Funktionalen Kostensplitting überhaupt die Funktionen eines Bauteils betrachtet und nicht einfach die Kosten?

16. Was ist das Kopplungsprinzip und wie unterscheidet es sich vom Funktionalen Kostensplitting?
17. Was ist der Unterschied zwischen Modernisierung und Erhaltungsmaßnahme?
18. Wie wird im Funktionalen Kostensplitting ermittelt, welcher Teil der Modernisierungskosten nicht für eine Mieterhöhung angesetzt werden darf?
19. Was sind Erhaltungskosten?
20. Wie hilft das Funktionale Kostensplitting bei der Berechnung von Mieterhöhungen?

Speech-to-Text Transcriptions — Respondent ST1

Table 1 lists the raw speech-to-text output produced by the tiny Whisper model for respondent ST1 (Group A), alongside the corresponding golden-set question number and recording duration. Entry 2 corresponds to the repetition-loop outlier discussed in Section 8.1.2, in which the model produced a faulty, repeated transcription of a single non-German token instead of recognizing the intended question.

Table 1: Raw speech-to-text transcriptions for respondent ST1 (Group A, tiny model)

No.	Recognized Text	Duration (s)
1	Was ist das funktionale Kostungsblätting?	8.16

[illegible]

No.	Recognized Text	Duration (s)
9	Welche bautechnischen Prozesse führen zu Funktionsminderungen die als Erhaltungsbedarf gelten?	7.20
10	Ist ein mangelnder Schallschutz als ein Bauschaden zu verstehen?	6.06
11	Warum spielt der Zustand zum Beispiel Schäden meines alten Bauteils bei der Berechnung eine Rolle?	7.80
12	byfunktioniert die funktionale Bewertung eines Bauteils!	5.46
13	Barum sind manche Funktion wichtiger für die Berechnung als andere.	5.76
14	Was bedeutet es wenn das neue Bauteil sowohl alte als auch neue Funktionen übernimmt?	7.98
15	Warum werden beim Funktionen einen Kostensplitting überhaupt die Funktion eines Bauteils betrachtet und nicht einfach die Kosten?	10.20
16	Was ist das Kopplungsprinzip und wie unterscheidet es sich vom Funktionar-Kostenisling?	8.58
17	Wie wird im funktionalen Costance-Blitting ermittelt welcher Teil der Modernisierungskosten nicht für eine Mieterhöhung angesetzt werden darf?	11.46
18	Was ist der Unterschied zwischen Modernisierung und Erhaltungsmassnahme?	6.00
19	was sind Erhaltungskosten.	2.94
20	Wie hilft das funktionale Kostenblössing bei der Berechnung von Mieterhöhung nennen?	7.68

*This entry corresponds to the WER outlier of 1.516 reported in Table 7.4 and discussed in Section 8.1.2.

Affidavit

I hereby declare that the submitted Bachelor thesis titled

Development of Analysis Software for User Studies Based on iMotions Data

is my own work and that I have only used the sources and materials indicated. All verbatim passages taken from literature or other sources, including websites, have been clearly identified as quotations with appropriate source citations.

Hamburg, 07. August 2026