



Hochschule für angewandte Wissenschaften Coburg
Fakultät Elektrotechnik und Informatik

Studiengang: INFORMATIK

Praxisbericht

El Ouarghi El Maizi, Salma

Unternehmen	Nara Institute of Science and Technology Social Computing Laboratory 630-0192 Nara, Ikoma-shi, Takayama-cho 8916-5 Nara, Japan
Zeitraum	01.10.2025 bis 15.02.2026

Abgabe des Berichts: 28.02.2026

Freigabe zur Vorlage des Praxisberichts an der HS Coburg:

Betreuungsperson	Prof. Eiji Aramaki	
Funktion	Laborleiter	15.02.2026
Telefon	+81743725250	
E-Mail	aramaki@is.naist.jp	Prof. Eiji Aramaki

Inhaltsverzeichnis

Abbildungsverzeichnis	II
Tabellenverzeichnis	III
Symbolverzeichnis	VI
Abkürzungsverzeichnis	VI
1 Einleitung	1
1.1 Motivation, Problemstellung und Zielsetzung	1
2 Theoretische Grundlagen	2
2.1 Medizinische Dokumentation und Triage (SOAP & ESI)	2
2.2 KI-gestützte Informationsextraktion (Whisper & LLMs)	2
2.3 Menschliche Faktoren und kognitive Belastung	2
3 Systemarchitektur und Implementierung	3
3.1 Systemarchitektur und Backend-Logik	3
3.2 Benutzeroberfläche und Lokalisierungsstrategie	6
3.2.1 Haupt-Dashboard	6
3.2.2 Transparent Monitor	7
3.2.3 Dual Mode Monitoring	8
3.2.4 Hybrid-System: Strategie der Mehrsprachigkeit	9
3.2.5 Patient Journey Dashboard und Reporting	11
3.2.6 Klinischer Berichts-Export (PDF)	12
3.2.7 Sprachmarker zur kognitiven Analyse	12
3.2.8 Qualitätsmetriken und Validierung	14
4 Technische Herausforderungen und Lösungsansätze	15
4.1 Latenzfreie Audio-Segmentierung (Ping-Pong-Recording)	15
4.2 State-Synchronisation in Multi-Window-Umgebungen	15
4.3 Resilienz durch Partial Persistence	15
4.4 Thread-Sicherheit in asynchronen Pipelines	15
4.5 API-Rate-Limits und Kosten	16
5 Evaluation und Ergebnisse	17
5.1 Evaluationsmethodik	17
5.2 Triage-Evaluation: Einzelfallanalyse und Performance-Statistik	17
5.3 Informationsextraktion: Identität, SOAP, Translation	18

Inhaltsverzeichnis

6 Fazit und Ausblick	19
6.1 Projekterfolg und Reflexion (NAIST)	19
6.2 Ethischer Rahmen und Zukunftsfähigkeit	19
Glossar	X
Literaturverzeichnis	XV
Ehrenwörtliche Erklärung	XX

Abbildungsverzeichnis

Abb. 1:	Architektur-Übersicht	3
Abb. 2:	System-Flow	4
Abb. 3:	SOAP-Detailansicht	4
Abb. 4:	Haupt-Dashboard in verschiedenen Lokalisierungsstufen	7
Abb. 5:	Transparent Monitor: Dashboard und Idle-Anzeige	8
Abb. 6:	Simultane Anzeige (Dual Mode)	9
Abb. 7:	Patient Journey Dashboard	11
Abb. 8:	Generierte klinische Berichte via PDF-Export	12
Abb. 9:	Linguistische Marker und Qualitätsanalyse	13
Abb. 10:	Transcription Accuracy Validator: BLEU- und CUDA-Evaluation	14
Abb. 11:	Evaluationsworkflow: Python-Struktur und generiertes Nurse-PDF für CALL_110.	17

Tabellenverzeichnis

Tab. 1:	Struktur der SOAP-Dokumentation	2
Tab. 2:	ESI-Triage-Stufen	2
Tab. 3:	Übersicht der Lokalisierungsstrategie (EN / JA)	9
Tab. 4:	System-Evaluations-Dashboard: REDSS vs. Pflege-Ground-Truth (n=15) . .	18
Tab. 5:	Informationsextraktion: Identitätsfelder, Klinische SOAP-Bewertung, Übersetzungsqualität	18

Codebeispielverzeichnis

1	Orchestrierung der Verarbeitungspipeline	3
2	SOAP-Extraktions-Prompt (Auszug)	4
3	Intelligente Feldextraktion	5
4	Evidence-Based Hybrid Urgency Classifier (ESI v4 + AI)	5
5	Hauptendpunkte der API	6
6	React Dashboard - State Management	6
7	Translation Cache Validation	7
8	Consolidated Localization	8
9	Statische Lokalisierung im Frontend	9
10	KI-Instruktionen zur medizinischen Lokalisierung	10
11	Validierung der Sprachreinheit	10
12	Auto-Sync Mechanismus	11
13	Implementierung des PDF-Exports	12
14	Analyse linguistischer Komplexitätsmarker	13
15	Validierungslogik und multilinguale Normalisierung	14
16	Ping-Pong-Buffer zur lückenlosen Audioerfassung	15
17	Thread-sicheres Session-Management mit expliziter Konvertierung	16
18	Evaluationsdaten CALL_110	17

Symbolverzeichnis

Symbol	Bedeutung	phys. Einheit
α	Krippendorff's Alpha - Koeffizient zur Beurteilung der Verlässlichkeit von Daten	[$-$]
κ	Cohen's Kappa - Statistisches Maß für die Inter-Rater-Reliabilität	[$-$]
$BLEU$	Bilingual Evaluation Understudy - Metrik zur Bewertung von Übersetzungen	[$-$]
F_1	F1-Score - Harmonisches Mittel aus Präzision und Trefferquote	[$-$]
G	Guiraud-Index - Maß für lexikalische Diversität	[$-$]
n	Stichprobengröße	[$-$]
P	Präzision (Precision) - Maß für die Genauigkeit der Extraktion	[$-$]
R	Trefferquote (Recall) - Maß für die Vollständigkeit der Extraktion	[$-$]
t_{ASR}	Dauer der Spracherkennung (Whisper)	[s]
t_{LLM}	Dauer der Extraktion und Klassifizierung (LLM)	[s]
t_{total}	Gesamtverarbeitungszeit der Pipeline	[s]
TTR	Type-Token-Ratio - Maß für lexikalische Variabilität	[$-$]
v	Verarbeitungsgeschwindigkeit der Audio-Pipeline	[s_{Audio}/s_{Real}]
WER	Word Error Rate - Wortfehlerrate bei der Spracherkennung	[$\%$]

Abkürzungsverzeichnis

AI	Artificial Intelligence
ALS	Advanced Life Support
API	Application Programming Interface
ASGI	Asynchronous Server Gateway Interface
ASR	Automatic Speech Recognition
BLEU	Bilingual Evaluation Understudy
BLS	Basic Life Support
CAD	Computer-Aided Dispatch
CJK	Chinese, Japanese, Korean
CORS	Cross-Origin Resource Sharing
CPU	Central Processing Unit
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
CUDA	Completion, Understanding, Detail, Accuracy
DNP	Digital Neuropsychological
DOM	Document Object Model
DP	Dynamic Programming
DSGVO	Datenschutz-Grundverordnung
DTO	Data Transfer Object
EMD	Emergency Medical Dispatcher
EMS	Emergency Medical Services
ESI	Emergency Severity Index
FastAPI	Python Web-Framework
FHIR	Fast Healthcare Interoperability Resources
GPT	Generative Pre-trained Transformer
GPU	Graphics Processing Unit
HL7	Health Level 7
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
KI	Künstliche Intelligenz

KIS	Krankenhausinformationssystem
LLM	Large Language Model
ML	Machine Learning
NAIST	Nara Institute of Science and Technology
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
OpenAI	OpenAI Inc.
ORM	Object-Relational Mapping
PDF	Portable Document Format
RAM	Random Access Memory
React	JavaScript-Bibliothek für Benutzeroberflächen
REDSS	Real-Time Emergency Decision Support System
Regex	Regular Expressions (Reguläre Ausdrücke)
REST	Representational State Transfer
SOAP	Subjective, Objective, Assessment, Plan
SQL	Structured Query Language
SQLite	Structured Query Language Lite
TTR	Type-Token Ratio
UI	User Interface
URL	Uniform Resource Locator
WER	Word Error Rate

1 Einleitung

1.1 Motivation, Problemstellung und Zielsetzung

In Rettungsleitstellen bilden Emergency Medical Dispatcher (EMDs) die kritische Schnittstelle zwischen Hilfesuchenden und der präklinischen Notfallversorgung. Unter extremem Zeitdruck müssen aus unvollständigen oder emotional geführten Gesprächen präzise medizinische Informationen extrahiert, dokumentiert und priorisiert werden.[KVS18] Hierbei sind Disponenten durch begrenzte Arbeitsgedächtniskapazität einer hohen kognitiven Belastung ausgesetzt [Mil56; Zap+25]. Moderne Technologien der Spracherkennung und Sprachverarbeitung bieten hier neue Entlastungspotentiale [Blo+19; Luo+25].

Die derzeitige manuelle Notrufbearbeitung weist strukturelle Defizite auf. Erstens führt die sequenzielle Bearbeitung von Gespräch und Dokumentation zu Verzögerungen, da relevante Informationen oft erst nach dem Gespräch finalisiert werden [Mel+25; Sco+16] oder Handbücher sequenziell konsultiert werden [CDM08, S. 45–52]. Zweitens unterliegt die medizinische Ersteinschätzung (Triage) einer hohen interindividuellen Variabilität, was zu inkonsistenten Entscheidungen führen kann [Mis+18; Sax+23]. Drittens erschweren unstrukturierte Freitextnotizen eine standardisierte klinische Übergabe und Nachvollziehbarkeit der erhobenen Daten [PLG23; Wee68]. Viertens beeinträchtigt die hohe Stressbelastung das Arbeitsgedächtnis der Disponenten erheblich und erhöht das Risiko von Informationsverlusten [KVS18; LeB09; Pin17]. Computergestützte Assistenzsysteme versprechen in diesem Kontext eine höhere diagnostische Stabilität sowie eine verbesserte prozessuale Sicherheit.

Vor diesem Hintergrund untersucht diese Arbeit die folgende zentrale Forschungsfrage: *Inwieweit kann ein KI-gestütztes Echtzeitsystem die Qualität der Notrufbearbeitung durch automatisierte SOAP-Extraktion und standardisierte Triage verbessern?*

Zur Beantwortung dieser Frage werden folgende Teilziele verfolgt:

1. **Systementwicklung:** Konzeption und Implementierung einer Echtzeit-Pipeline bestehend aus automatischer Transkription (Whisper), LLM-basierter SOAP-Extraktion und ESI-basierter Triage.
2. **Evaluation:** Quantitative Bewertung der Transkriptionsgenauigkeit sowie der Systemperformance.
3. **Klinische Validierung:** Qualitative Evaluation des Systems durch medizinisches Fachpersonal ($n = 15$).
4. **Mehrsprachigkeit:** Implementierung einer bidirektionalen Übersetzung zwischen Englisch und Japanisch.

2 Theoretische Grundlagen

2.1 Medizinische Dokumentation und Triage (SOAP & ESI)

Die medizinische Dokumentation erfolgt nach der SOAP-Methode (Tabelle 1) [PLG23; Wee68], während die Triage über den Emergency Severity Index (ESI) die Dringlichkeit priorisiert [Gil+20] (Tabelle 2). Da manuelle Triage Mistriage-Raten von bis zu 32 % aufweist [Mis+18; Sax+23; Tan+07], zielt Automatisierung auf eine höhere Reliabilität ab, wie computergestützte Systeme wie ANKUTRIAGE bereits demonstrieren [Koc+22].

Tab. 1: Struktur der SOAP-Dokumentation

Sektion	Beschreibung und Inhalte
Subj. (S)	Angaben des Patienten, Symptome, Schmerzbeschreibung (Narrativ).
Obj. (O)	Messbare Fakten: Name, Alter, Vitaldaten, Bewusstsein, Adresse.
Assess. (A)	Medizinische Arbeitshypothese und klinische Einschätzung.
Plan (P)	Therapeutische Maßnahmen und entsandtes Rettungsmittel.

Tab. 2: ESI-Triage-Stufen

ESI	Kriterien / Dringlichkeit	Res.
1 (CRIT.)	Lebensgefahr, erfordert sofortige lebenserhaltende Maßnahmen.	Sofort
2 (HIGH)	Hochdringlich; instabiler Status oder hohes Risiko.	Hoch
3 (MED.)	Stabil, jedoch hoher Ressourcenbedarf (Diagnostik).	≥ 2
4 (LOW)	Stabil, maximal eine medizinische Ressource benötigt.	1
5 (NONE)	Stabil, kein Ressourcenbedarf (z. B. Beratung).	0

Systemadaption: REDSS konsolidiert ESI 4 und 5 zu “LOW”, da beide Stufen gemäß ESI-Protokoll keinen unmittelbaren Ressourcenbedarf aufweisen [Gil+20].

2.2 KI-gestützte Informationsextraktion (Whisper & LLMs)

OpenAI Whisper dient als robustes Spracherkennungsmodell zur Transkription in akustisch schwierigen Notfallsituationen [Luo+25; Rad+23; Zec+25]. Das System nutzt das “base”-Modell für geringe Latenz. GPT-4o übernimmt die Extraktion strukturierter Informationen mit einer Genauigkeit von über 91 % [Gup+25; Per+24] und nutzt schrittweise Schlussfolgerungen (Chain-of-Thought) zur besseren Kontextualisierung [Ada+24; Wei+22].

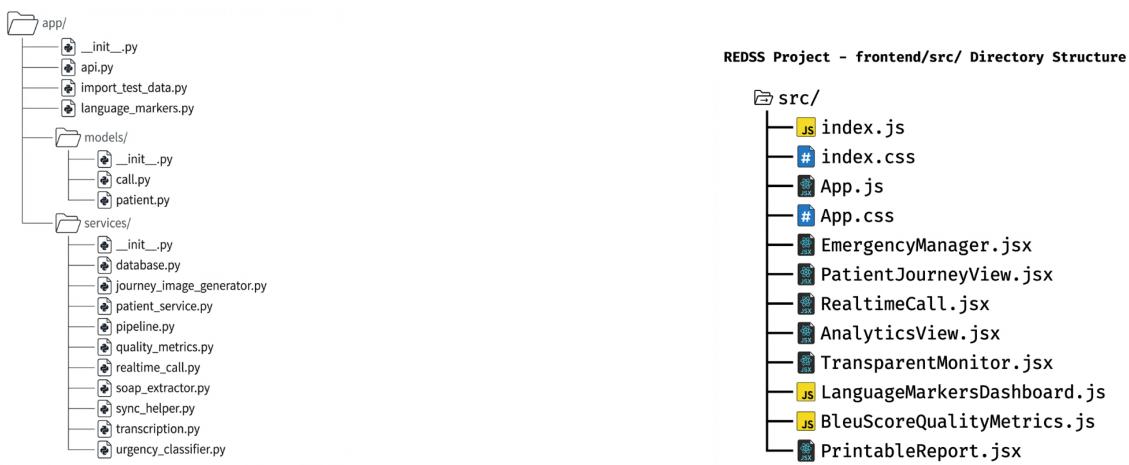
2.3 Menschliche Faktoren und kognitive Belastung

Akuter Stress reduziert die Kapazität des Arbeitsgedächtnisses [Mil56] erheblich, was parallele Aufgaben wie Dokumentation und Assessment erschwert [LeB09; Pin17]. Dies erhöht die Fehlerquote und das Burnout-Risiko [KVS18]. Computergestützte Tools entlasten das Personal, indem sie Dokumentationslast übernehmen und einen stärkeren Fokus auf die Kommunikation ermöglichen [Mel+25; Sco+16].

3 Systemarchitektur und Implementierung

3.1 Systemarchitektur und Backend-Logik

Das REDSS-System nutzt eine **Drei-Schichten-Architektur** [BCK03]. Das React-Frontend kommuniziert via REST und WebSockets mit dem FastAPI-Backend, während die Daten in SQLite gespeichert werden (Abb. 1). Die Pipeline koordiniert alle Schritte von der Aufnahme bis zur Speicherung (Abb. 2).



1.a Backend

1.b Frontend

Abb. 1: Architektur-Übersicht

Verarbeitungspipeline: Die asynchrone Pipeline orchestriert die zentralen Verarbeitungsschritte von der Audio-Aufnahme bis zur strukturierten Datenausgabe (Listing 1).

```

1  async def process_call(self, audio_path: str) -> dict:
2      call_id = f"CALL_{datetime.now().strftime('%Y%m%d_%H%M%S')}"
3      try:
4          # Schritt 1: Audio transkribieren
5          logger.info(f"Starte Transkription für {call_id}")
6          transcription = await transcription_service.transcribe(audio_path)
7          transcript = transcription['text']
8          # Schritt 2: SOAP-Notizen extrahieren
9          logger.info(f"Extrahiere SOAP-Notizen für {call_id}")
10         soap = await soap_extractor.extract(transcript)
11         # Schritt 3: Dringlichkeit klassifizieren
12         urgency = await urgency_classifier.classify(transcript, soap)
13         # Schritt 4: In Datenbank speichern (atomar)
14         with get_db() as db:
15             call = save_call(db, call_id, audio_path, transcript)
16             update_soap(db, call_id, soap)
17             update_urgency(db, call_id, urgency)
18             db.commit()
19         return self._build_response(call_id, transcript, soap, urgency)
20     except Exception as e:
21         logger.error(f"Fehler bei Verarbeitung: {str(e)}")
22         raise ProcessingError(f"Pipeline-Fehler: {str(e)}" from e)

```

Code 1: Orchestrierung der Verarbeitungspipeline

3 Systemarchitektur und Implementierung

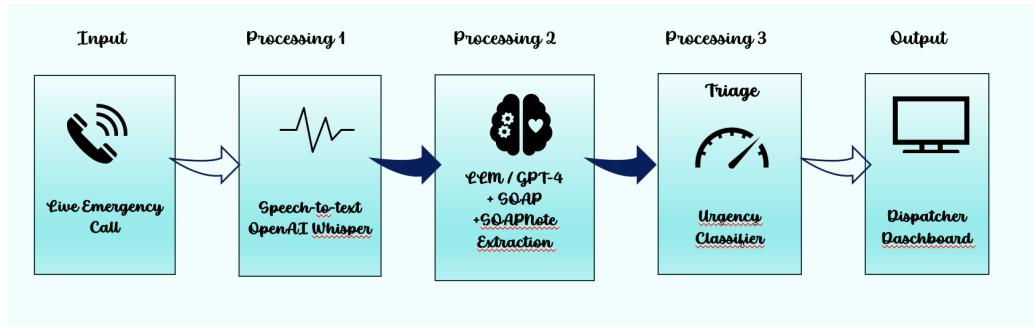


Abb. 2: System-Flow

Die **Transkription** nutzt das Whisper “base”-Modell für einen optimalen Trade-off zwischen Latenz und Genauigkeit [Rad+23; Zec+25]. Die **SOAP-Extraktion** erfolgt durch Instruktionsoptimierung (*Prompt Engineering*) mit einem *Chain-of-Thought*-Ansatz [Wei+22]:

```

1 You are a medical AI assistant. TASK: Extract structured SOAP notes.
2 GUIDELINES: Identify symptoms, vitals, patient identifiers, and clinical reasoning.
3 OUTPUT FORMAT (STRICT):
4 <S>- Subjective observations and patient complaints</S>
5 <O>- Name: [extracted name]
6   - Age: [number]
7   - Address: [full address]
8   - Symptoms: [list]
9 </O>
10 <A>- Clinical assessment and suspected condition</A>
11 <P>- Emergency response plan and instructions</P>

```

Code 2: SOAP-Extraktions-Prompt (Auszug)

The screenshot shows the SOAP Clinical Analysis interface with the following sections:

- SUBJECTIVE:** The caller is a teenager who is worried about his friend, Jake, who drank a lot at a party. They are both 18 years old, and it is Jake's birthday. The caller reports that Jake is conscious but out of it, lying on the couch and mumbling. He has thrown up about 30 minutes ago and is currently on his side with a trash can nearby. The caller is concerned about Jake's condition and is unsure if he will get in trouble since there are no adults present.
- OBJECTIVE:** Name: Jake Morrison
Age: 18
Address: 8923 University Boulevard
Phone: [Not provided]
Blood: [Not provided]
- ASSESSMENT:** The patient, Jake, is showing signs of potential alcohol poisoning due to excessive alcohol consumption (8 or 9 drinks over three hours) and recent vomiting. He is conscious but disoriented and has a slow but steady breathing pattern.
- PLAN:** Paramedics are being dispatched to the location to assess Jake's condition. The caller is advised to keep Jake on his side and stay with him until help arrives.
- System Performance (BLEU):** EXTRACTION QUALITY: System Validated | BLEU OK
- Clinical Transcription:** Are there adults at this party?
[Hesitant] No, his parents are out of town. Are we gonna get in trouble?
I'm more concerned about Jake right now. Has he taken any drugs? Billie... weird

Abb. 3: SOAP-Detailansicht

3 Systemarchitektur und Implementierung

Der LLM-basierte Extraktionsansatz folgt etablierten Methoden zur strukturierten klinischen Informationsextraktion [Ada+24; Gup+25].

Zur robusten Extraktion der Patientendaten werden zusätzliche Regex-Muster für die Feldextraktion eingesetzt, um die Strukturierung durch das LLM abzusichern und potenzielle Formatierungsfehler abzufangen:

```
1 def parseField(text, fieldName):
2     # Mehrsprachige Label-Unterstützung
3     translations = {
4         'Name': ['Name', 'Nom', 'Nombre'],
5         'Age': ['Age', 'Alter', 'Edad'],
6         'Address': ['Address', 'Adresse', 'Location']
7     }
8     keys = translations.get(fieldName, [fieldName])
9
10    for key in keys:
11        patterns = [
12            f"^{key}\\s*[:]\\s*([^\n\r\\.]+)", # Standard
13            f"\\*\\s*{key}\\s*[:]\\s*([^\n\\.]+)", # Bullet
14            f"^{key}\\s*[:]\\s*([^\n\\.]+)" # Start-of-line
15        ]
16        for pattern in patterns:
17            match = re.search(pattern, text, re.IGNORECASE)
18            if match and is_valid_value(match.group(1)):
19                return match.group(1).strip()
20
21    return 'N/A'
```

Code 3: Intelligente Feldextraktion

Die **Dringlichkeitsklassifizierung** folgt einem hybriden Ansatz aus Keyword-Matching und kontextueller KI-Analyse nach ESI-Stufen [Gil+20]. Diese Strategie kombiniert deterministische ESI-Vorgaben mit probabilistischer KI-Kontextanalyse:

1. **ESI Level 1/2 (Critical/High-Risk):** Trust ESI (Evidence-based, immer eskalieren).
2. **ESI Level 4/5 (Low/Minimal):** Trust ESI, außer die KI erkennt höhere Dringlichkeit (Safety-first).
3. **ESI Level 3 (Medium):** Nutzung der KI-Verfeinerung (Context-aware).
4. **Safety-first:** Immer auf die höhere Dringlichkeit eskalieren, wenn eine der Methoden dies vorschlägt.
5. **Spezialfall:** Die KI erkennt Testanrufe oder Unsinn und stuft diese angemessen herab.

```
1 class HybridUrgencyClassifier:
2     # ESI Level 1: Immediate Lifesaving Criteria
3     CRITERIA = {
4         "en": ["intubated", "cardiac arrest", "unconscious", "apneic"],
5         "ja": ["挿管", "心停止", "意識不明", "無呼吸"]
6     }
7     def classify(self, transcript, soap, lang="en"):
8         # 1. Rule-based ESI Check (Speed & Safety)
9         if any(term in transcript for term in self.CRITERIA.get(lang, [])):
10             return {"level": "CRITICAL", "esi": 1}
11         # 2. AI Contextual Analysis (Instruktionsoptimierung)
12         prompt = f"""
13             Role: Emergency MD. Task: ESI Triage.
14             Input: {transcript} | Context: {json.dumps(soap)}
```

3 Systemarchitektur und Implementierung

```
15     Format: {{ "level": "EST_LEVEL", "logic": "Rationale in {lang}" }}
16     """
17     return self.call_ai(prompt)
```

Code 4: Evidence-Based Hybrid Urgency Classifier (ESI v4 + AI)

Die Client-Kommunikation wird über folgende REST-Endpunkte abgewickelt:

```
1 # Audio-Upload und Verarbeitung
2 POST /api/upload
3 # Text-basierte Eingabe
4 POST /api/process-text
5 # Notrufliste abrufen (mit Übersetzung)
6 GET /api/calls?lang=en|ja
7 # Einzelnen Notruf abrufen
8 GET /api/calls/{call_id}?lang=en|ja
9 # WebSocket fuer Live-Anrufe
10 WS /ws/realtime-call
11 # Patientenverwaltung
12 GET/POST/PUT /api/patients
```

Code 5: Hauptendpunkte der API

3.2 Benutzeroberfläche und Lokalisierungsstrategie

Das Benutzerinterface ist für die simultane Nutzung in Englisch und Japanisch optimiert. Das React-basierte Frontend umfasst Dashboards für Einsatzmanagement, Live-Anrufe und Behandlungsverlauf. Das zentrale Dashboard verwaltet den globalen Zustand:

```
1 const EmergencyManager = () => {
2     const [calls, setCalls] = useState([]);
3     const [language, setLanguage] = useState('en');
4     const [urgencyFilter, setUrgencyFilter] = useState('ALL');
5     // Polling fuer Echtzeit-Updates
6     useEffect(() => {
7         const fetchCalls = async () => {
8             const response = await fetch('/api/calls?lang=${language}');
9             const data = await response.json();
10            setCalls(data);
11        };
12        fetchCalls();
13        const interval = setInterval(fetchCalls, 5000); // Alle 5s
14        return () => clearInterval(interval);
15    }, [language]);
16    return (
17        <div className="dashboard">
18            <Header language={language} setLanguage={setLanguage} />
19            <StatsCards calls={calls} />
20            <CallList calls={calls} urgencyFilter={urgencyFilter} />
21        </div>
22    );
23};
```

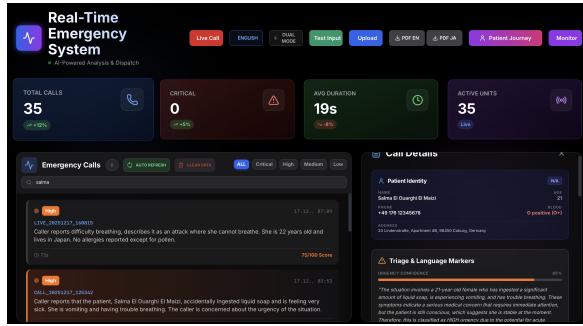
Code 6: React Dashboard - State Management

Das Polling-Intervall von 5 Sekunden stellt einen Kompromiss zwischen Echtzeit-Aktualisierung und Server-Last dar, da kürzere Intervalle die API-Rate-Limits unnötig belasten würden [BG21].

3.2.1 Haupt-Dashboard

Das zentrale Dashboard für das Einsatzmanagement bietet eine Übersicht über alle Notrufe und erlaubt das Filtern nach Dringlichkeit sowie das Umschalten der Systemsprache (Abb. 4).

3 Systemarchitektur und Implementierung



4.a Englische Ansicht



4.b Japanische Ansicht

Abb. 4: Haupt-Dashboard in verschiedenen Lokalisierungsstufen

Dynamische Lokalisierungs-Pipeline (Backend) Die Backend-Intelligenz (`app/api.py`) operiert zweistufig: Ein **Cache-Check** prüft zunächst auf vorhandene Übersetzungen, um Latenz und Kosten zu minimieren. Bei einem Cache-Miss führt die **KI-Translation** (GPT-4o) eine medizinisch präzise Lokalisierung durch, die klinische Fachtermini und kulturelle Besonderheiten (z. B. japanische Adressformate) wahrt.

3.2.2 Transparent Monitor

Dieses Sekundärdisplay ermöglicht eine Echtzeit-Synchronisation [WHA23] und zeigt dem Personal die wichtigsten Daten auf einen Blick, ohne die Interaktion am Hauptgerät zu stören. Die Lokalisierung erfolgt konsolidiert in einem einzelnen Request, was die Antwortzeit von ca. 20s auf unter 1s (Cache-Hit) reduziert (Abb. 5). Das Mehrsprachigkeitssystem (EN/JA) unterstützt bidirektionale Übersetzung und nutzt Caching zur Vermeidung redundanter API-Aufrufe:

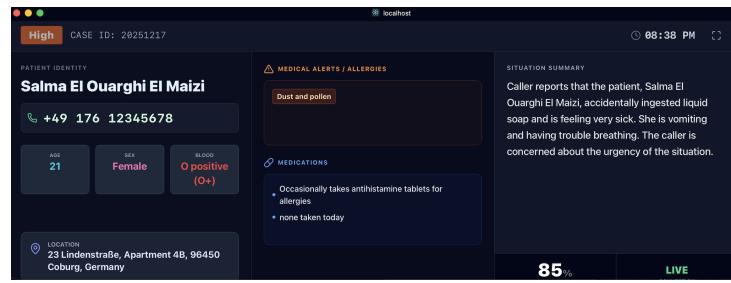
```

1 CACHE_VERSION = "v2"
2
3 def is_valid_cache(cached_data, target_lang):
4     # Version-Check
5     if cached_data.get("_version") != CACHE_VERSION:
6         return False
7     # Sprachreinheits-Check
8     if target_lang == "ja":
9         # Pruefe auf englische Labels
10        english_indicators = ["Name:", "Age:", "not provided"]
11        if any(ind in cached_data["soap_objective"] for ind in english_indicators):
12            return False
13    elif target_lang == "en":
14        # Pruefe auf japanische Zeichen
15        japanese_pattern = r'[\u3040-\u309F\u30A0-\u30FF\u4E00-\u9FFF]'
16        if re.search(japanese_pattern, cached_data["soap_objective"]):
17            return False
18    return True

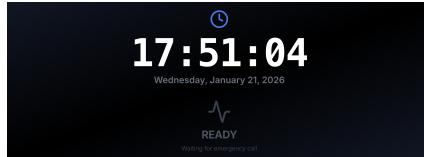
```

Code 7: Translation Cache Validation

3 Systemarchitektur und Implementierung



5.a Transparent Dashboard



5.b Uhrzeit (EN)



5.c Uhrzeit (JA)

Abb. 5: Transparent Monitor: Dashboard und Idle-Anzeige

```
1 def localize_call_data(transcript, soap_notes, metadata, target_lang):
2     """
3         EINE API-Anfrage statt 3+ separater Calls
4         Uebersetzt: Transcript, SOAP (alle 4 Sektionen), Metadata
5     """
6     prompt = f"""
7         Translate and localize the following emergency call data to {target_lang}.
8         IMPORTANT:
9             - Translate field labels (Name, Age)
10            - Preserve medical accuracy
11            - Maintain SOAP structure
12        TRANSCRIPT: {transcript}
13        SOAP NOTES: {json.dumps(soap_notes)}
14        METADATA: {json.dumps(metadata)}
15        OUTPUT: Return JSON with all translated fields
16    """
17
18    response = openai.ChatCompletion.create(
19        model="gpt-4-mini",
20        messages=[{"role": "user", "content": prompt}]
21    )
22    return parse_localized_response(response)
```

Code 8: Consolidated Localization

3.2.3 Dual Mode Monitoring

Ein **Dual Mode**-Button ermöglicht die simultane Anzeige beider Sprachen zur Qualitätskontrolle. Dies unterstreicht das **Human-in-the-Loop**-Designkonzept, da der Disponent jederzeit die KI-Übersetzung verifizieren und korrigierend eingreifen kann. Das gesamte Interface ist lückenlos lokalisiert, wie in Abbildung 6 dargestellt.

3 Systemarchitektur und Implementierung

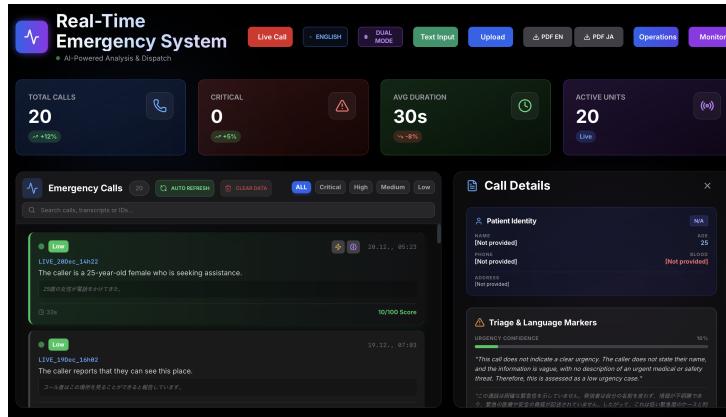


Abb. 6: Simultane Anzeige (Dual Mode)

3.2.4 Hybrid-System: Strategie der Mehrsprachigkeit

Die Implementierung der Zweisprachigkeit folgt einem hybriden Ansatz, der präzise Kontrolle über das Benutzerinterface mit der Flexibilität von KI-gestützten Inhalten kombiniert. Dabei wird strikt zwischen statischen (Hard-Coded) und dynamischen (AI-Driven) Inhalten unterschieden.

Implementierungs-Matrix Die folgende Tabelle fasst die Verteilung der Verantwortlichkeiten im System zusammen:

Tab. 3: Übersicht der Lokalisierungsstrategie (EN / JA)

Feature	Typ	Komponente / Ort
Buttons & Menüs	Hard-Coded	EmergencyManager.jsx (TRANSLATIONS)
Tabellen-Header	Hard-Coded	Frontend UI Layer
Notruf-Transkript	AI-Driven	soap_extractor.py (OpenAI)
SOAP-Notizen	AI-Driven	OpenAI GPT-4o-mini
Urgency Reasoning	AI-Driven	OpenAI GPT-4o-mini
Validierung	Logik	app/api.py (Mixing Check)

Durch diese hybride Architektur wird sichergestellt, dass das UI stets professionell und stabil bleibt, während die komplexen medizinischen Daten dynamisch und skalierbar übersetzt werden können.

Ebene 1: UI-Übersetzung (Statisch) Um Navigationsfehler und "KI-Halluzinationen" im Menü auszuschließen, werden alle festen UI-Texte über ein Dictionary-System gesteuert. Beim Umschalten der Sprache wird der gesamte UI-Zustand unverzüglich aktualisiert.

```

1 // frontend/src/EmergencyManager.jsx
2 const TRANSLATIONS = {
3   en: {
4     title: "Real-Time Emergency System",
5     liveCall: "Live Call",

```

3 Systemarchitektur und Implementierung

```
6     patientIdentity: "Patient Identity",
7     urgency: "Urgency Level",
8     critical: "Critical"
9 },
10 ja: {
11     title: "救急意思決定支援システム",
12     liveCall: "ライブ通話",
13     patientIdentity: "患者基本情報",
14     urgency: "緊急度レベル",
15     critical: "最優先(重症)"
16 }
17 };
```

Code 9: Statische Lokalisierung im Frontend

Ebene 2: KI-Instruktionen und Instruktionsoptimierung (AI-Driven) Das Herzstück der medizinischen Lokalisierung ist die dynamische Generierung von Prompts. Hierbei wird der KI nicht nur der Text übergeben, sondern auch die exakte medizinische Struktur der Zielkultur (z. B. japanische Adressformate) beigebracht.

```
1 # app/services/soap_extractor.py
2 def localize_call_data(self, transcript, soap_notes, metadata, target_language):
3     is_japanese = target_language in ["ja", "jp", "japanese"]
4     lang_name = "JAPANESE" if is_japanese else "ENGLISH"
5     # Zielsprachen-spezifische Labels definieren
6     if is_japanese:
7         labels = "氏名:, 年齢:, 住所:, 電話:, 血液型:"
8         missing = "[不明]"
9     else:
10        labels = "Name:, Age:, Address:, Phone:, Blood:"
11        missing = "[Not provided]"
12     # Instruktionsoptimierung fuer medizinische Korrektheit
13     prompt = f"""
14 You are a medical localization expert.
15 Localize the following emergency call data into {lang_name}.
16 EXTREMELY IMPORTANT: In the Objective section, you MUST use these exact labels: {labels}.
17 If information is missing, use "{missing}".
18 Maintain the clinical tone in the Reasoning section.
19 """
20     return self.call_openai_api(prompt, transcript, soap_notes)
```

Code 10: KI-Instruktionen zur medizinischen Lokalisierung

Ebene 3: Vermischungsschutz und Qualitätssicherung Um die Reinheit der Zielsprache zu gewährleisten, wurde eine automatisierte Qualitätskontrolle implementiert. Dieser Algorithmus erkennt, ob die KI fälschlicherweise Labels in der Quellsprache hinterlassen hat ("Japanglisch") und erzwingt in solchen Fällen eine Neuübersetzung.

```
1 # app/api.py
2 def validate_localization(lang, obj_text):
3     # Enhanced detection: Check for language mixing
4     if lang == "ja":
5         # Japanisch sollte keine englischen Labels enthalten
6         has_english_labels = any(label in obj_text for label in ["Name:", "Age:", "Address:])
7         if has_english_labels:
8             print(f"Warnung: Englischer Inhalt im japanischen Cache. Re-translating...")
9             return False # Erzwingt Neu-Uebersetzung
10    elif lang == "en":
11        # Englisch sollte keine japanischen Schriftzeichen enthalten
12        import re
13        has_japanese_chars = bool(re.search(r'[\u3040-\u309F\u30A0-\u30FF\u4E00-\u9FFF]', obj_text))
14        if has_japanese_chars:
15            print(f"Warnung: Japanischer Inhalt im englischen Cache. Re-translating...")
```

3 Systemarchitektur und Implementierung

```
16     return False  
17     return True
```

Code 11: Validierung der Sprachreinheit

3.2.5 Patient Journey Dashboard und Reporting

Auto-Synchronisation Der Patient Journey Tracking-Prozess ermöglicht die automatische Synchronisation neuer Notrufe mit bestehenden Datensätzen:

```
1 def sync_emergency_call_to_patient_journey(call_id):  
2     call = db.query(EmergencyCall).filter_by(id=call_id).first()  
3     patient = db.query(Patient).filter_by(name=call.patient_name).first()  
4     if not patient:  
5         patient = Patient(  
6             name=call.patient_name,  
7             primary_condition=call.disease,  
8             medical_history=call.soap_subjective,  
9             journey_events=json.dumps([{"segment": 1, "date": call.created_at.date(),  
10                "description": f"Initial: {call.disease}", "status": "completed"}])  
11         )  
12         db.add(patient)  
13     else:  
14         events = json.loads(patient.journey_events or "[]")  
15         events.append({"segment": len(events) + 1, "date": call.created_at.date(),  
16             "description": f"Follow-up: {call.disease}", "status": "completed"})  
17         patient.journey_events = json.dumps(events)  
18     db.commit()
```

Code 12: Auto-Sync Mechanismus

Das Tracking des Patientenverlaufs ermöglicht eine kontinuierliche Überwachung von der ersten Notrufaufnahme bis zur Behandlung.

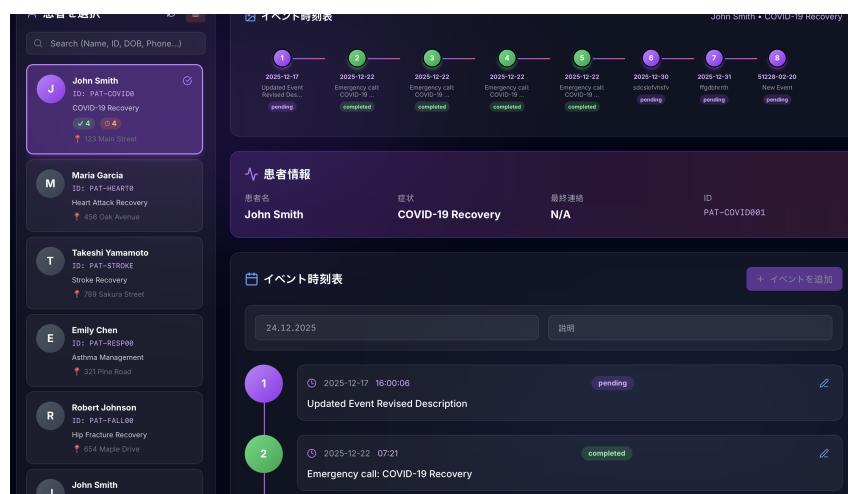


Abb. 7: Patient Journey Dashboard

3 Systemarchitektur und Implementierung

3.2.6 Klinischer Berichts-Export (PDF)

Die generierten Berichte dienen als Brücke zwischen der Notrufaufnahme und der klinischen Weiterbehandlung. Der Fokus liegt dabei auf der kompakten Übergabe: Der Export enthält ausschließlich die S- und O-Daten (Subjective und Objective), um die initiale Anamnese prägnant abzubilden.

```

1 const exportToPDF = async () => {
2   const { default: jsPDF } = await import('jspdf');
3   const doc = new jsPDF('p', 'mm', 'a4');
4   // Header & Patienten-Metadaten
5   doc.setFontSize(22);
6   doc.text('Patient Journey Report', 20, 20);
7   doc.text(`Name: ${selectedPatient.name}`, 20, 45);
8   // Iteration über Journey-Events
9   journeyEvents.forEach((event, i) => {
10     doc.text(`${event.date}: ${event.description}`, 20, 70 + i*10);
11   });
12   doc.save(`Patient_Journey_${selectedPatient.name}.pdf`);
13 };

```

Code 13: Implementierung des PDF-Exports

8.a EN-Bericht

8.b JA-Bericht

Abb. 8: Generierte klinische Berichte via PDF-Export

3.2.7 Sprachmarker zur kognitiven Analyse

Der LanguageMarkerAnalyzer wurde zu einer wissenschaftlich fundierten Version weiterentwickelt, die Transkripte mittels strukturierter Schwellenwerte (MarkerThresholds) auf kognitive Beeinträchtigungen untersucht. Die Analyse erfolgt modular über *Speech Fluency*, kognitive, linguistische und semantische Marker. Ein Kernbestandteil ist die Analyse der lexikalischen Diversität mittels Type-Token-Ratio (TTR) und dem Guiraud-Index. Die TTR ist ein Maß für die lexikalische Variabilität ($TTR = \text{Types}/\text{Tokens} \in [0, 1]$) [CM10], während der Guiraud-Index ($G = \text{Types}/\sqrt{\text{Tokens}}$) herangezogen wird, um eine verzerrungsfreie Analyse unabhängig

3 Systemarchitektur und Implementierung

von der Textlänge zu ermöglichen. Das System dient hierbei der klinischen Unterstützung durch passive Sprachanalyse, um in Notfällen subtile neurologische Veränderungen (z. B. Anzeichen von Demenz) objektiv einzuschätzen [Roa+11].

```
1 def _analyze_linguistic_markers(self, transcript: str, clean_words: List[str], sentences: List[str]) -> Dict:
2     """Analyse linguistischer Komplexitätmarker"""
3     total_words = len(clean_words)
4     unique_words = len(set(clean_words))
5     # 1. Type-Token Ratio (TTR) - Basis-Diversität
6     ttr = unique_words / total_words if total_words > 0 else 0
7     # 2. Guiraud's Index - Längenbereinigte lexikalische Diversität
8     # Diese Metrik ist robuster gegenüber unterschiedlichen Textlängen als TTR.
9     guiraud = unique_words / math.sqrt(total_words) if total_words > 0 else 0
10    return {
11        "ttr_score": self._create_marker_result(
12            marker_name="TTR (Type-Token Ratio)",
13            value=round(ttr, 3),
14            unit="ratio",
15            judgment=self._judge_ttr(ttr),
16            normal_range="0.50-0.70",
17            risk_level=self._assess_risk(ttr, self.thresholds.ttr_low,
18                                         self.thresholds.ttr_very_low, reverse=True),
19            description="Lexikalische Diversität: Eindeutige Wörter / Gesamtzahl Wörter"
20        ),
21        "guiraud_index": self._create_marker_result(
22            marker_name="Guiraud's Index",
23            value=round(guiraud, 2),
24            unit="index",
25            judgment=self._judge_guiraud(guiraud),
26            normal_range="7.0-12.0",
27            risk_level=self._assess_risk(guiraud, self.thresholds.guiraud_low,
28                                         self.thresholds.guiraud_very_low, reverse=True),
29            description="Längenbereinigte lexikalische Diversität: Eindeutige Wörter / sqrt(Gesamtwörter)"
30        )
31    }
```

Code 14: Analyse linguistischer Komplexitätmarker

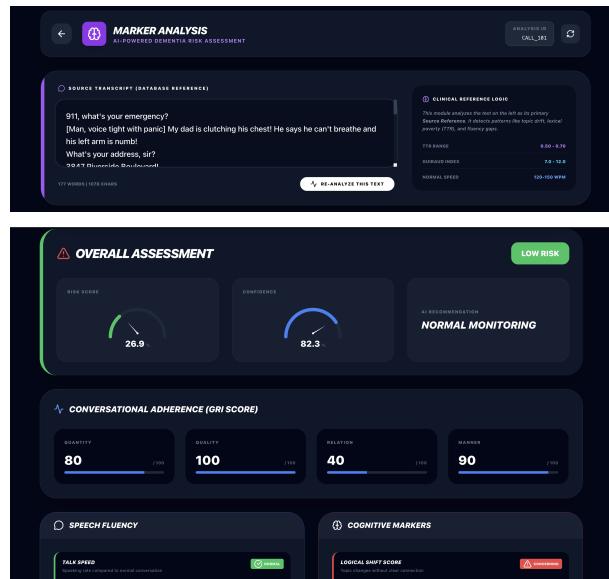


Abb. 9: Linguistische Marker und Qualitätsanalyse

3.2.8 Qualitätsmetriken und Validierung

Die Evaluation von Spracherkennungssystemen im medizinischen Kontext erfordert eine Differenzierung zwischen mechanischer Erkennungsrate (WER) und semantischer Korrektheit (BLEU-Score sowie das klinisch orientierte Bewertungssystem CUDA) [Gup+25; Pap+02]:

$$WER = \frac{S + D + I}{N} \quad BLEU = BP \cdot \exp \left(\sum_{n=1}^N w_n \ln p_n \right) \quad (3.1)$$

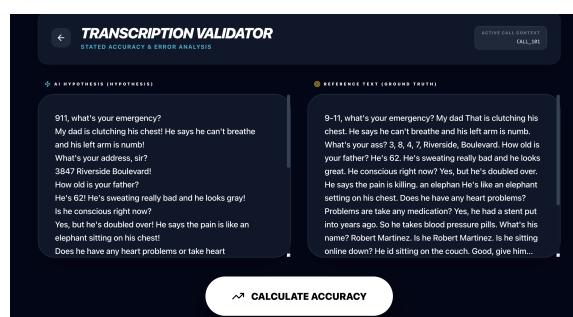
Ergänzend wird das **CUDA-Modell** zur Bewertung der medizinischen Integrität herangezogen. Dieses klinisch orientierte Bewertungssystem für SOAP-Notizen aggregiert vier Dimensionen: **Completion** (Vollständigkeit der Referenzinformationen), **Understanding** (Jaccard-Ähnlichkeit der Wortüberschneidung), **Detail** (Verhältnis der Informationsdichte zur Vermeidung von Informationsverlust) und **Accuracy** (Präzision zur Vermeidung von Halluzinationen). Zusammen mit dem BLEU-Score liefert CUDA eine datenbasierte Einschätzung darüber, wie präzise und nützlich die automatisch extrahierten Informationen für das medizinische Personal sind.

```

1 def clean_text(self, text):
2     # 1. Noise Reduction (Text-Normalisierung)
3     text = re.sub(r'\[.*?\]', '', text.lower())
4     # 2. Schutz von Kanji/Kana Unicode-Bereichen
5     text = re.sub(r'[^w\s\u3040-\u309F\u30A0-\u30FF\u4E00-\u9FFF]', ' ', text)
6     return text.strip()
7 def validate_transcription(self, ref, hyp):
8     # Mechanisch (WER): Edit-Distanz via DP-Matrix
9     wer = calculate_levenshtein(ref, hyp) / len(ref.split())
10    # Semantisch (CUDA): Klinische Datenintegrität [Gup+25]
11    cuda = (completion + understanding + detail + accuracy) / 4
12    return {"wer_acc": max(0, (1-wer)*100), "cuda": cuda}

```

Code 15: Validierungslogik und multilinguale Normalisierung



10.a Textvergleich (Ref vs. Hyp)



10.b BLEU-Score und CUDA-Metriken

Abb. 10: Transcription Accuracy Validator: BLEU- und CUDA-Evaluation

Dieser **Development-Loop** ermöglicht es dem Entwickler, die KI-Ergebnisse im *Validator* (Abb. 10) anhand farblich hervorgehobener Diskrepanzen zu prüfen. Um die Zuverlässigkeit des Systems zu testen und zu verifizieren, wie korrekt oder fehlerhaft die Transkription tatsächlich ist, wurde dieses Dashboard implementiert.

4 Technische Herausforderungen und Lösungsansätze

4.1 Latenzfreie Audio-Segmentierung (Ping-Pong-Recording)

Eine kritische Herausforderung bei Echtzeit-Transkriptionen ist der Datenverlust während der Chunk-Generierung. Klassische MediaRecorder-Implementierungen führen zu Lücken im Audiostream, wenn ein Segment abgeschlossen und das nächste gestartet wird.

Die Lösung erfolgte durch ein **Ping-Pong-Buffering-Verfahren** auf Client-Seite. Hierbei werden zwei parallele Instanzen des MediaRecorder zeitversetzt gesteuert, um eine lückenlose Erfassung zu garantieren:

```
1 const switchRecorders = () => {
2     const nextIndex = (activeIndex + 1) % 2;
3     const current = recorders[activeIndex];
4     const next = recorders[nextIndex];
5     next.start(); // Starte neue Aufnahme bevor die alte stoppt
6     if (current.state === 'recording') {
7         current.stop(); // Schliesst den Chunk ab und sendet ihn
8     }
9     activeIndex = nextIndex;
10};
```

Code 16: Ping-Pong-Buffer zur lückenlosen Audioerfassung

4.2 State-Synchronisation in Multi-Window-Umgebungen

Die Anforderung, ein Dashboard und einen isolierten Monitor (*Transparent Monitor*) simultan zu aktualisieren, ohne auf eine persistente Datenbank-Abfrage (*Polling*) angewiesen zu sein, wurde über die **BroadcastChannel-API** gelöst. Dies ermöglicht eine ereignisgesteuerte Kommunikation zwischen Browser-Kontexten mit minimaler Latenz.

4.3 Resilienz durch Partial Persistence

Um Systemausfälle (z. B. durch API-Timeouts) abzufangen, wurde ein Pipeline-Muster integriert, das **Partial Results** sichert. Selbst wenn die KI-Analyse fehlschlägt, wird das Transkript und die Audio-Dateipersistiert, um die medizinische Dokumentationspflicht zu erfüllen.

4.4 Thread-Sicherheit in asynchronen Pipelines

Eine zentrale Herausforderung war das Management von SQLAlchemy-Sessions zur Vermeidung von `DetachedInstanceError`-Exceptions im asynchronen Kontext von FastAPI. Diese treten auf, wenn nach dem Schließen der Session auf Objektattribute zugegriffen wird [Fow03].

Die Lösung erfolgte durch die Konvertierung in Python-Dictionaries innerhalb der aktiven Session (Data Transfer Object Pattern):

```
1 def get_call_data(call_id: str) -> dict:
2     """Sichere Datenextraktion mit Session-Management"""
3     with get_db() as db:
4         call = db.query(EmergencyCall).filter_by(id=call_id).first()
5         # Konvertiere zu Dict INNERHALB der aktiven Session
6         call_dict = {
7             'id': call.id,
8             'transcript': call.transcript,
9             'soap_subjective': call.soap_subjective,
10            'soap_objective': call.soap_objective,
11            'urgency_level': call.urgency_level,
12            'created_at': call.created_at.isoformat()
13        }
14     # Session ist geschlossen, aber Dict ist sicher verwendbar
15     return call_dict
```

Code 17: Thread-sicheres Session-Management mit expliziter Konvertierung

Diese Entkopplung der Datenbankschicht von der asynchronen Pipeline gewährleistet Thread-Sicherheit und Datenkonsistenz am Client [BG21; Fow03].

4.5 API-Rate-Limits und Kosten

Aufgrund der Instabilität von Google Gemini 1.5 Flash (Verbindungsfehler) erfolgte der Wechsel auf OpenAI GPT-4o-mini. Die API-Kosten wurden durch Caching (Listing 7), Prompt-Minimierung und konsolidierte Requests (Listing 8) optimiert.

5 Evaluation und Ergebnisse

5.1 Evaluationsmethodik

Die Evaluation basiert auf 15 synthetischen Notrufszenarien, die in einem dreiphasigen Prozess geprüft wurden: (1) Initialisierung und REDSS-Verarbeitung der Transkripte, (2) Konsolidierung der Ergebnisse in validierbare PDF-Berichte und (3) fachliche Validierung durch diplomiertes Pflegepersonal unter Verwendung einer Likert-Skala (1–5). Diese Experten-Korrekturen bilden die klinische Ground Truth (`expected_urGENCY`) für alle Vergleiche. Abbildung 11 illustriert diesen Workflow: Die technische Datenstruktur (links) wird dem klinischen Assessment im PDF-Format (rechts) gegenübergestellt.

```

1 EVALUATED_CALLS = [
2     "call_id": "CALL_110",
3     "text": "***911, what's your emergency?
4 [Elderly woman, calm] Hello dear, I'm calling because I've had a
5         nosebleed for about 45 minutes and it won't stop.
6 What's your address, ma'am?
7 3421 Sycamore Boulevard, apartment 12C.
8 How old are you?
9 I'm 81. My name is Helen Reed.
10 Are you on any blood thinners?
11
12     "expected_urGENCY": "medium",
13     "expected_type": "medical",
14     "expected_location": "3421 Sycamore Boulevard, apartment 12C",
15     "expected_agent": "Helen Reed",
16     "expected_soap": [
17         "subjective": "81-year-old female with unilateral epistaxis (right naris) for 45 minutes, on warfarin for mechanical heart valve, reports lightheadedness, history of hypertension, lives alone",
18         "objective": "Patient is pale and conversational, applying pressure first site (pinching and forward lean), prolonged bleeding despite measures, anticoagulated patient, hypertensive history, mild orthostatic symptoms reported",
19         "assessment": "Prolonged epistaxis in anticoagulated elderly patient, possible posterior bleed, mild hypovolemia",
20         "plan": "BLS transport for ENT evaluation, likely nasal packing needed, check INR/PT levels, blood pressure monitoring, assess for posterior vs anterior source, possible cauterization or packing, ensure family support, monitor for continued bleeding"
21     ]
1]

```

Code 18: Evaluationsdaten
CALL_110

Case 10 of 15: CALL_110		Urgency: HIGH
Patient: Helen Reed		
AI TRIAGE CLASSIFICATION		
AI Predicted Urgency: HIGH		
Triage Logic: The patient is an 81-year-old woman with a prolonged nosebleed lasting 45 minutes, which is common given her age, warfarin and her age. The combination of anticoagulation therapy and the duration of bleeding increases her risk of significant complications. Although she is not in immediate distress, the lightheadedness she reports, along with her medical history, indicates a potential for deterioration. Therefore, she requires urgent evaluation and intervention, placing her at ESI Level 2. (AI-enhanced from ESI Level 3)		
Nurse Assessment:		
<input checked="" type="checkbox"/> ACCURATE	<input type="checkbox"/> SHOULD BE HIGHER	<input checked="" type="checkbox"/> SHOULD BE LOWER
Comments: medium		
CLINICAL SUMMARY (S + O)		
The caller is an 81-year-old woman named Helen Reed who has been experiencing a nosebleed for about 45 minutes that won't stop. She reports that she has had occasional nosebleeds in the past, which usually resolve in 10 to 15 minutes. She feels a little lightheaded but attributes it to her age. She takes warfarin and has an artificial heart valve. The nosebleed started while she was reading her book, and she has been pinching her nose and leaning forward as instructed. Name: Helen Reed Age: 81 Address: 3421 Sycamore Boulevard, apartment 12C Phone: [Not provided] Blood: [Not provided]		
<input checked="" type="checkbox"/> ACCURATE	<input type="checkbox"/> INCOMPLETE	<input type="checkbox"/> INCORRECT
Comments: _____		
For Context:		
Assessment: The patient is experiencing a prolonged nosebleed, likely exacerbated by the use of blood thinners (warfarin) and her age. She is lightheaded but does not report severe dizziness or weakness.		
Plan: Paramedics will be dispatched to check on the patient due to the duration of the nosebleed and her use of blood thinners. She is advised to continue pinching her nose until help arrives.		

Nurse-PDF Output

Abb. 11: Evaluationsworkflow: Python-Struktur und generiertes Nurse-PDF für CALL_110.

5.2 Triage-Evaluation: Einzelfallanalyse und Performance-Statistik

In Tabelle 4 werden die detaillierten Einzelergebnisse der 15 Notrufszenarien gegenübergestellt. Die Tabelle vergleicht die klinische Einschätzung des Fachpersonals (Expected GT) direkt mit der automatisierten Klassifizierung des REDSS-Systems und analysiert die Abweichungen hinsichtlich ihrer Sicherheit (Status & Analyse). Insgesamt erzielt das System solide Ergebnisse, wobei insbesondere die Rate der Unter-Triage (2/15 Fälle) kritisch beleuchtet wird, um die Patientensicherheit zu gewährleisten. Die zugrunde liegenden Verarbeitungszeiten wurden auf einem MacBook Air M2 gemessen, wobei die Pipeline eine durchgehende Echtzeitfähigkeit (6–20 s) aufweist.

5 Evaluation und Ergebnisse

Tab. 4: System-Evaluations-Dashboard: REDSS vs. Pflege-Ground-Truth (n=15)

Call ID	Szenario	Expected (GT)	System	Status & Analyse
CALL_101	Herzinfarkt (Brustschmerz)	CRITICAL	HIGH	! Unter-Triage (-1)
CALL_102	Kettensägenverletzung	CRITICAL	CRITICAL	✓ Match
CALL_103	Pädiatr. Anfall (>5 min)	CRITICAL	CRITICAL	✓ Match
CALL_104	Asthma (Schwer)	HIGH	HIGH	✓ Match
CALL_105	Kopfverletz. + Warfarin	HIGH	HIGH	✓ Match
CALL_106	Geburt (Crowning)	CRITICAL	CRITICAL	✓ Match
CALL_107	Hypoglykämie (Diabetes)	CRITICAL	HIGH	! Unter-Triage (-1)
CALL_108	Armfraktur	LOW	MEDIUM	↑ Sicherer (+1)
CALL_109	Gastroenteritis	LOW	HIGH	↑ Sicherer (+2)
CALL_110	Nasenbluten (Anticoag.)	LOW	HIGH	↑ Sicherer (+2)
CALL_111	COPD-Exazerbation	MEDIUM	HIGH	↑ Sicherer (+1) / Name: N/A
CALL_112	Niereninfektion	MEDIUM	HIGH	↑ Sicherer (+1)
CALL_113	Welfare Check	LOW	HIGH	↑ Sicherer (+2)
CALL_114	Perle in Nase (Kind)	MEDIUM	MEDIUM	✓ Match
CALL_115	Alkoholvergiftung	HIGH	HIGH	✓ Match

Aggregierte Performance-Metriken (pro ESI-Level):				
Level	Genaugigkeit	GT-Count	Korrekt	Over / Under
CRITICAL (ESI-1)	60 %	5	3	0 / 2
HIGH (ESI-2)	100 %	3	3	- / 0
MEDIUM (ESI-3)	33 %	3	1	2→H / 0
LOW (ESI-4/5)	0 %	4	0	4→M/H / 0
Gesamt (4 Levels)	46,7 %	15	7	6 / 2
±1 Level	80,0 %	—	12	—

Die Triage folgt einem *Safety-First*-Prinzip (40 % Over- vs. 13,3 % Under-Triage). Trotz Fehlklassifizierungen bei CALL_101/107 (HIGH statt CRITICAL) sichern konservative Hochstufungen kritische Indikationen ab [Tan+07].

5.3 Informationsextraktion: Identität, SOAP, Translation

Tabelle 5 vereint Feldextraktion (Multi-Pattern vs. Regex), klinische SOAP-Bewertung (Likert 1–5) und Mehrsprachigkeit (BLEU EN→JA):

Tab. 5: Informationsextraktion: Identitätsfelder, Klinische SOAP-Bewertung, Übersetzungsqualität

Kategorie	Feld/Kriterium/Metrik	Regex	Multi-P.	Δ	Ø	σ	Bewertung
Identität	Alter	42 %	89 %	+47 PP	—	—	Sehr gut
	Name	68 %	93 %	+25 PP	—	—	Sehr gut (1 Fall n.e.)
	Adresse	55 %	100 %	+45 PP	—	—	Perfekt
	Telefon	71 %	94 %	+23 PP	—	—	Sehr gut
SOAP	Vollständigkeit (S)	—	—	—	3,6	0,9	Befriedigend
	Genauigkeit (S)	—	—	—	4,1	0,7	Gut
	Vollständigkeit (O)	—	—	—	3,4	1,0	Befriedigend
	Genauigkeit (O)	—	—	—	3,9	0,8	Gut
	Klinische Relevanz	—	—	—	4,0	0,8	Gut
	Gesamt	—	—	—	3,8	0,8	Gut
Translation	BLEU (Objective)	—	—	—	0,81	—	Sehr gut
	BLEU (Subjective)	—	—	—	0,73	—	Gut
	BLEU (Assess./Plan)	—	—	—	0,69	—	Befriedigend
	Cache-Hit-Rate (>90 %)	—	—	—	0,83s	—	Echtzeit

Key Findings: Multi-Pattern-Matching verbessert die Feldextraktion signifikant (Adresse: 100 %). Die klinische Relevanz (4,0/5) und Übersetzungsqualität (BLEU 0,81) bestätigen die Praxistauglichkeit.

6 Fazit und Ausblick

Das im Rahmen dieser Arbeit entwickelte REDSS-System stellt einen signifikanten Fortschritt in der computergestützten Notfallmedizin dar. Es konnte demonstriert werden, dass die gezielte Synergie aus Spracherkennung (Whisper), Large Language Models (GPT-4o) und klinisch etablierten Triage-Standards (ESI) nicht nur technisch realisierbar ist, sondern eine messbare prozessuale Entlastung für das medizinische Fachpersonal generiert. Das gesamte Projekt einschließlich Quellcode, Demonstrationsvideos und Validierungsdaten ist öffentlich auf GitHub verfügbar (<https://github.com/salmaelouar/REDSS-Emergency-AI>).

6.1 Projekterfolg und Reflexion (NAIST)

Sämtliche zu Beginn definierten Forschungsfragen wurden erfolgreich adressiert: Die automatisierte SOAP-Extraktion wurde implementiert und bewies eine hohe klinische Relevanz. Bei der ESI-Triage zeigte das System eine hohe Treffsicherheit innerhalb der medizinisch vertretbaren Toleranzbereiche, was es als wertvolle Zweitmeinung ("Conservative Safety Net") qualifizierte. Zudem konnte die durchgehende Echtzeitfähigkeit der Verarbeitungspipeline erfolgreich nachgewiesen werden.

Die Zeit am **Nara Institute of Science and Technology (NAIST)** in Japan ermöglichte in einer internationalen Spitzensforschungsumgebung tiefe Einblicke in die Full-Stack-Entwicklung und das LLM-Engineering. Besonders die Optimierung von Prompt-Ketten stellte eine lohnende Herausforderung dar, während der interkulturelle Austausch die Relevanz global einsetzbarer Medizinal-Software (EN/JA) unterstrich.

6.2 Ethischer Rahmen und Zukunftsfähigkeit

Das System folgt strikt dem **Human-in-the-Loop**-Prinzip, wodurch die Letztentscheidungsgewalt und algorithmische Transparenz als Voraussetzung für klinisches Vertrauen stets beim Menschen verbleiben. Konkret empfehlen sich für die Zukunftsfähigkeit folgende Schritte: (1) Fine-Tuning auf einen annotierten Datensatz von 10.000+ realen Notrufen unter Einsatz von Audio-Augmentation (z. B. Störgeräusch-Injektion), (2) Integration eines Konfidenz-Scores für die Triage-Entscheidung, (3) Implementierung der deutschen Lokalisierung sowie anschließendes A/B-Testing mit Rettungsleitstellen in Deutschland und Japan zur Validierung der Mehrsprachigkeit unter Realbedingungen.

Glossar

Algorithmus

Eine präzise definierte Handlungsvorschrift zur Lösung eines Problems oder einer Klasse von Problemen in endlich vielen Schritten.

Asynchrone Verarbeitung

Eine Form der Programmabarbeitung, bei der Operationen unabhängig vom Hauptprogrammfluss ausgeführt werden, sodass das System nicht auf den Abschluss zeitintensiver Aufgaben (z. B. KI-Inferenz) warten muss.

BLEU-Score

Bilingual Evaluation Understudy. Eine Metrik zur automatischen Bewertung der Qualität von maschinellen Übersetzungen, basierend auf der n-Gramm-Präzision im Vergleich zu einer Referenz.

BroadcastChannel API

Eine Web-Schnittstelle, die es verschiedenen Browsing-Kontexten (Fenstern, Tabs) derselben Herkunft ermöglicht, in Echtzeit miteinander zu kommunizieren.

Chain-of-Thought

Eine Prompting-Technik, die das Modell dazu anleitet, Zwischenschritte der Argumentation zu generieren, bevor die endgültige Antwort ausgegeben wird, um die Qualität bei komplexen Aufgaben (z. B. Triage-Begründung) zu erhöhen.

Cohens Kappa

Ein statistisches Maß (κ) zur Bewertung der Übereinstimmung zwischen zwei Ratern (z. B. Mensch und KI), das die zufällige Übereinstimmung korrigiert.

CUDA (Metrik)

Ein Qualitätsmodell (Completion, Understanding, Directness, Accuracy) zur Bewertung der medizinischen Genauigkeit und Vollständigkeit von KI-generierten Inhalten.

Data Mapper Pattern

Ein Architekturmuster, das die Geschäftslogik strikt von der Datenbankstruktur trennt, indem ein Mapper-Objekt den Datentransfer zwischen Objekten und Tabellen übernimmt.

Dispatch

Der Prozess der Entsendung und Koordination von Einsatzmitteln und Personal in einem Notfallsystem.

Emergency Medical Dispatcher

Speziell geschultes Personal in Rettungsleitstellen, das für die Notrufannahme, die medizinische Ersteinschätzung und die Koordination von Rettungsmitteln verantwortlich ist.

Endpoint

Ein durch eine URL identifizierter Zugriffspunkt einer API, über den bestimmte Daten oder Funktionen abgefragt werden können.

FastAPI

Ein modernes, leistungsstarkes Web-Framework für die Erstellung von APIs mit Python, das auf Standard-Python-Typ-Hints basiert und asynchrone Programmierung unterstützt.

Framework

Ein Programmiergerüst, das eine vordefinierte Struktur und Werkzeuge bereitstellt, um die Entwicklung von Anwendungen zu beschleunigen und zu standardisieren.

Halluzination

Ein Phänomen bei generativen Sprachmodellen, bei dem das System Texte generiert, die zwar grammatisch korrekt klingen, aber faktisch falsch sind oder nicht auf den Eingabedaten basieren.

Idle Mode

Der Ruhezustand des Transparent Monitors, wenn kein aktiver Notruf vorliegt. In diesem Modus werden das aktuelle Datum und die Uhrzeit lokalisiert (z. B. auf Japanisch oder Englisch) angezeigt.

Inferenz

Der operative Einsatz eines fertig trainierten KI-Modells auf neue Daten, um Vorhersagen, Klassifizierungen oder Extraktionen vorzunehmen.

Inter-Rater-Reliabilität

Ein statistisches Maß für die Übereinstimmung zwischen verschiedenen Beobachtern oder Bewertern (z. B. Mensch vs. KI) bei der Nutzung desselben Bewertungssystems.

JSON

JavaScript Object Notation. Ein kompaktes Datenformat für den Datenaustausch zwischen Anwendungen, das sowohl für Menschen als auch für Maschinen leicht lesbar ist.

Kontextfenster

Die maximale Anzahl von Token, die ein LLM gleichzeitig verarbeiten kann, bevor Informationen am Anfang der Konversation verloren gehen.

Latenz

Die zeitliche Verzögerung zwischen einer Eingabe (z. B. Ende eines Notrufs) und der Verfügbarkeit des Ergebnisses (z. B. fertige SOAP-Note).

Likert-Skala

Eine bipolare Skala (oft 5-stufig), die oft in Fragebögen und klinischen Studien zur Messung von Einstellungen oder Zufriedenheit eingesetzt wird.

n-Gramm

Eine zusammenhängende Folge von n Elementen (z. B. Wörtern) aus einer Textsequenz, die für statistische Textanalysen und Metriken wie den BLEU-Score verwendet wird.

Notfall

Ein plötzliches Ereignis, das eine unmittelbare Gefahr für Leben oder Gesundheit darstellt und ein schnelles Eingreifen von Rettungskräften erfordert.

One-Shot/Few-Shot Learning

Eine Technik im Prompt Engineering, bei der dem Modell keine oder nur sehr wenige Beispiele für die gewünschte Aufgabe im Kontextfenster zur Verfügung gestellt werden.

Pipeline

Eine Abfolge von Verarbeitungsschritten (z. B. Transkription, Analyse, Klassifizierung), bei der die Ausgabe eines Schritts als Eingabe für den nächsten dient.

Precision

Präzision. In der KI-Evaluation das Verhältnis der korrekt positiven Vorhersagen zur Gesamtzahl der positiven Vorhersagen (Vermeidung von Fehlalarmen).

Prompt

Eine textuelle Eingabeaufforderung an ein KI-Modell, die vorgibt, welche Aufgabe das Modell ausführen soll.

Prompt Engineering

Der Prozess der gezielten Gestaltung und Optimierung von Eingabeaufforderungen (Prompts), um die Ausgabequalität von Large Language Models für spezifische Aufgaben zu erhöhen.

Pydantic

Eine Bibliothek zur Datenvielfältigung und Einstellungsverwaltung in Python, die auf Python-Typ-Annotationen basiert und in FastAPI zur Validierung von API-Anfragen verwendet wird.

React

Eine JavaScript-Bibliothek zur Erstellung von Benutzeroberflächen, die auf der komponentenbasierten Architektur basiert und ein effizientes Rendering durch ein virtuelles DOM ermöglicht.

React Hook

Spezielle Funktionen in React (z. B. useState, useEffect), mit denen Zustände und Lifecycle-Methoden in funktionalen Komponenten genutzt werden können.

Recall

Trefferquote oder Sensitivität. Das Verhältnis der korrekt positiven Vorhersagen zur Gesamtzahl der tatsächlich positiven Fälle (Vermeidung von verpassten Fällen).

Regex

Eine Abfolge von Zeichen, die ein Suchmuster bilden. Sie werden in der Informatik verwendet, um Texte nach bestimmten Mustern (z. B. Telefonnummern, Postleitzahlen) zu durchsuchen oder zu validieren.

REST

Representational State Transfer. Ein Architekturstil für zustandslose Schnittstellen zwischen Systemen, der primär über HTTP-Methoden (GET, POST, etc.) kommuniziert.

Robustheit

Die Fähigkeit eines Systems oder Algorithmus (z. B. Whisper), auch unter schwierigen Bedingungen wie Hintergrundlärm oder schlechter Audioqualität stabile Ergebnisse zu liefern.

SQLAlchemy

Ein leistungsstarker Object-Relational Mapper (ORM) für Python, der eine flexible Abstraktionsschicht zwischen der Anwendung und der SQL-Datenbank bereitstellt.

SQLite

Ein leichtgewichtiges, dateibasiertes relationales Datenbankmanagementsystem, das keine separate Server-Infrastruktur benötigt.

Token

Die kleinste Informationseinheit (z. B. Wortteile oder Zeichen), in die ein Text von einem Sprachmodell zerlegt wird, um ihn mathematisch zu verarbeiten.

Transformer

Eine neuronale Netzwerkarchitektur, die auf dem Attention-Mechanismus basiert und die Grundlage für moderne Sprachmodelle wie GPT und Whisper bildet.

Transkription

Der Prozess des Umwandelns von gesprochener Sprache aus einer Audiodatei in geschriebenen Text.

Triage

Ein systematisches Verfahren zur Ersteinschätzung der medizinischen Behandlungsdringlichkeit von Patienten, um in Notfallsituationen eine effiziente Ressourcenallokation zu gewährleisten.

TTR

Type-Token-Ratio. Ein Maß für die lexikalische Variation eines Textes, berechnet als Verhältnis der unterschiedlichen Wörter (Types) zur Gesamtzahl der Wörter (Tokens).

Urgency Classifier

Ein Software-Modul zur automatisierten Einstufung der medizinischen Dringlichkeit eines Patienten basierend auf klinischen Kriterien und kontextueller Analyse.

Virtual DOM

Eine Speicherkopie des realen DOMs in React, die es ermöglicht, UI-Änderungen effizient zu berechnen und nur notwendige Aktualisierungen an der tatsächlichen Browser-Ansicht vorzunehmen.

WebSocket

Ein Kommunikationsprotokoll, das eine dauerhafte, bidirektionale Verbindung zwischen Client und Server ermöglicht und für die Echtzeitübertragung (z. B. Live-Transkription) genutzt wird.

Whisper

Ein von OpenAI entwickeltes, Transformer-basiertes Modell zur automatischen Spracherkennung (ASR), das für seine Robustheit gegenüber Hintergrundgeräuschen optimiert ist.

Zero-Shot Learning

Die Fähigkeit eines Modells, eine Aufgabe ohne vorherige spezifische Beispiele im Prompt zu lösen, basierend auf dem während des Trainings erworbenen Allgemeinwissen.

Übertriage

Ein vorsorgliches Einstufen eines Patienten in eine höhere Dringlichkeitskategorie als medizinisch objektiv notwendig, um das Risiko einer gefährlichen Unterversorgung (Untertriage) zu minimieren.

Literaturverzeichnis

- [Ada+24] H. Adam, Y. Ming, A. J. Butte, and L. A. Nathanson. “Clinical Information Extraction with Large Language Models: A Case Study on Organ Procurement”. In: *AMIA Annual Symposium Proceedings*. Vol. 2024. American Medical Informatics Association, 2024, pp. 115–123.
- [BCK03] Len Bass, Paul Clements, and Rick Kazman. *Software Architecture in Practice*. 2nd. Boston, MA: Addison-Wesley Professional, 2003.
- [BG21] Srinivas Bangalore and Sai S. Gullapalli. “Performance Analysis of Asynchronous Server Gateway Interface (ASGI) in Python Web Frameworks”. In: *International Journal of Advanced Computer Science and Applications* 12.8 (2021), pp. 298–304.
- [Blo+19] Stig Nikolaj Blomberg, Fredrik Folke, Annette Kjær Ersbøll, Helle Collatz Christensen, Christian Torp-Pedersen, Michael R. Sayre, Jim R. Countiss, and Freddy K. Lippert. “Real-time machine learning helps dispatchers identify out-of-hospital cardiac arrest”. In: *PLOS ONE* 14.1 (2019), e0210385. DOI: [10.1371/journal.pone.0210385](https://doi.org/10.1371/journal.pone.0210385).
- [CDM08] J. J. Clawson, K. B. Dernocoeur, and C. Murray. *Principles of Emergency Medical Dispatch*. 4th. Priority Press, 2008.
- [CM10] Michael A. Covington and Joe D. McFall. “Cutting the TTR: Type-token ratio, language impairment, and language acquisition”. In: *Journal of Biomedical Informatics* 43.3 (2010), pp. 471–476.
- [Fow03] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley Professional, 2003.
- [Gil+20] Nancy Gilboy, Paula Tanabe, Debbie Travers, and Anna Marie Rosenau. *Emergency Severity Index (ESI): A Triage Tool for Emergency Department Care, Version 4, Implementation Handbook, 2020 Edition*. Tech. rep. Publication No. 12-0014. Rockville, MD: Agency for Healthcare Research and Quality (AHRQ), 2020.
- [Gup+25] R. Gupta, A. Gupta, R. Singh, A. Prasad, and P. Bansal. “Large language models for data extraction from unstructured and semi-structured electronic health records: a multiple model performance evaluation”. In: *BMJ Health & Care Informatics* 32.1 (2025), e101139. DOI: [10.1136/bmjhci-2024-101139](https://doi.org/10.1136/bmjhci-2024-101139).
- [Koc+22] Ayca Koca, Onur Polat, Ahmet Burak Oguz, and Mesut Sevindik. “Reliability and Validity of a New Computer-Based Triage Decision Support Tool: ANKUTRIAGE”. In: *Disaster Medicine and Public Health Preparedness* 16.6 (2022), pp. 2441–2445. DOI: [10.1017/dmp.2022.101](https://doi.org/10.1017/dmp.2022.101).

- [KVS18] K. E. Klimley, V. B. Van Hasselt, and A. M. Stripling. “Posttraumatic Stress Disorder in Police, Firefighters, and Emergency Dispatchers”. In: *Aggression and Violent Behavior* 43 (2018), pp. 33–44. DOI: [10.1016/j.avb.2018.08.005](https://doi.org/10.1016/j.avb.2018.08.005).
- [LeB09] V. R. LeBlanc. “The effects of acute stress on performance: implications for health professions education”. In: *Academic Medicine* 84.10 Suppl (2009), S25–S33. DOI: [10.1097/ACM.0b013e3181b37b8f](https://doi.org/10.1097/ACM.0b013e3181b37b8f).
- [Luo+25] Xiao Luo, Le Zhou, Kathleen M. Adelgais, and Zhan Zhang. “Assessing the Effectiveness of Automatic Speech Recognition Technology in Emergency Medicine Settings: A Comparative Study of Four AI-powered Engines”. In: *Journal of Healthcare Informatics Research* 9 (2025), pp. 494–512. DOI: [10.1007/s41666-025-00171-8](https://doi.org/10.1007/s41666-025-00171-8).
- [Mel+25] S. S. Meloy, E. Woltman, A. Martinez, and K. Duane. “It’s time to talk to emergency medical dispatchers: survey study on performance feedback and patient outcome follow-up to EMDs”. In: *BMC Emergency Medicine* 25.1 (2025), p. 13. DOI: [10.1186/s12873-025-01332-7](https://doi.org/10.1186/s12873-025-01332-7).
- [Mil56] George A. Miller. “The magical number seven, plus or minus two: Some limits on our capacity for processing information”. In: *Psychological Review* 63.2 (1956), pp. 81–97.
- [Mis+18] Binoy Mistry, Sarah Stewart De Ramirez, Gabor Kelen, Paulo S. K. Schmitz, Kamna S. Balhara, Scott Levin, Diego Martinez, Kevin Psoter, Xavier Anton, and Jeremiah S. Hinson. “Accuracy and Reliability of Emergency Department Triage Using the Emergency Severity Index: An International Multicenter Assessment”. In: *Annals of Emergency Medicine* 71.5 (2018), 581–587.e3.
- [Pap+02] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. “BLEU: a Method for Automatic Evaluation of Machine Translation”. In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.
- [Per+24] E. A. Perez-Alday, T. D. Li-Pershing, A. Bender, C. Hamilton, J. A. Thomas, K. Johnson, T. Lee, E. R. Gonzales, K. Taylor, and L. G. Tereshchenko. “Leveraging Large Language Models for Accurate Retrieval of Patient Information From Medical Reports: Systematic Evaluation Study”. In: *Journal of Medical Internet Research* 26 (2024), e59803. DOI: [10.2196/59803](https://doi.org/10.2196/59803).
- [Pin17] Jesse M. Pines. “What Cognitive Psychology Tells Us About Emergency Department Physician Decision-making and How to Improve It”. In: *Academic Emergency Medicine* 24 (2017), pp. 1–5.

- [PLG23] Vivek Podder, Valerie Lew, and Sassan Ghassemzadeh. “SOAP Notes”. In: *StatPearls [Internet]*. PMID: 29489268. Treasure Island (FL): StatPearls Publishing, 2023. URL: <https://www.ncbi.nlm.nih.gov/books/NBK482263/>.
- [Rad+23] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. “Robust speech recognition via large-scale weak supervision”. In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. PMLR, 2023, pp. 28492–28518.
- [Roa+11] Brian Roark, Margaret Mitchell, John-Paul Hosom, Kristy Hollingshead, and Jeffrey Kaye. “Spoken Language Derived Measures for Detecting Mild Cognitive Impairment”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.7 (2011), pp. 2081–2090. DOI: [10.1109/TASL.2011.2112351](https://doi.org/10.1109/TASL.2011.2112351).
- [Sax+23] David R. Sax, E. Margaret Warton, Dustin G. Mark, David R. Vinson, and Mamata V. Kene. “Evaluation of Version 4 of the Emergency Severity Index in US Emergency Departments for the Rate of Mistriage”. In: *JAMA Network Open* 6.3 (2023), e232404.
- [Sco+16] Greg Scott, Christopher Olola, Corike Toxopeus, Jeff J. Clawson, Adam Johnson, Brad Schultz, Kim Miller, Nicole Richmond, David Robinson, Matt Zavadsky, Leslie Burnette, Tracy Barron, and Brian Patterson. “Characterization of Call Prioritization Time in a Medical Priority Dispatch System”. In: *Annals of Emergency Dispatch & Response* 4.1 (2016), pp. 27–33. URL: <https://www.aedrjournal.org/characterization-of-call-prioritization-time-in-a-medical-priority-dispatch-system>.
- [Tan+07] Paula Tanabe, Richard Gimbel, Paul R. Yarnold, Demetrios N. Kyriacou, and James G. Adams. “Reliability and validity of scores on the Emergency Severity Index version 3”. In: *Academic Emergency Medicine* 14.3 (2007), pp. 213–218.
- [Wee68] Lawrence L. Weed. “Medical records that guide and teach”. In: *New England Journal of Medicine* 278.11 (1968), pp. 593–600. DOI: [10.1056/NEJM196803142781105](https://doi.org/10.1056/NEJM196803142781105).
- [Wei+22] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. “Chain of Thought Prompting Elicits Reasoning in Large Language Models”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 24824–24837.
- [WHA23] WHATWG. *HTML Living Standard: Web messaging*. Web Hypertext Application Technology Working Group. 2023. URL: <https://html.spec.whatwg.org/multipage/web-messaging.html>.

- [Zap+25] Jasmine S. Zaphir, Karen A. Murphy, Alex J. MacQuarrie, and Matthew J. Stainer. “Understanding the Role of Cognitive Load in Paramedical Contexts: A Systematic Review”. In: *Prehospital Emergency Care* 29.2 (2025). DOI: [10.1080/10903127.2024.2370491](https://doi.org/10.1080/10903127.2024.2370491).
- [Zec+25] J. R. Zech, D. Kim, A. Somoano, I. Landi, E. S. Krauss, J. Kim, J. Gichoya, S. Soffer, and E. Klang. “Transformer-Based Open-Source Whisper Software Versus Leading Commercial Speech Recognition Software for Radiology Transcription: Comparison Study”. In: *American Journal of Roentgenology* 225.1 (2025), e2532903.

Persönliche Angaben / Personal details

El Ouarghi El Maizi, Salma

Familienname, Vorname / Surnames, given names

28.12.2000

Geburtsdatum / Date of birth

INFORMATIK

Studiengang / Course of study

03895439

Matrikelnummer / Student registration number

Eigenständigkeitserklärung

Declaration

Hiermit versichere ich, dass ich diese Arbeit selbstständig verfasst und noch nicht anderweitig für Prüfungszwecke vorgelegt habe. Ich habe keine anderen als die angegebenen Quellen oder Hilfsmittel benutzt. Die Arbeit wurde weder in Gänze noch in Teilen von einer Künstlichen Intelligenz (KI) erstellt, es sei denn, die zur Erstellung genutzte KI wurde von der zuständigen Prüfungskommission oder der bzw. dem zuständigen Prüfenden ausdrücklich zugelassen. Wörtliche oder sinngemäße Zitate habe ich als solche gekennzeichnet.

Es ist mir bekannt, dass im Rahmen der Beurteilung meiner Arbeit Plagiatserkennungssoftware zum Einsatz kommen kann.

Es ist mir bewusst, dass Verstöße gegen Prüfungsvorschriften zur Bewertung meiner Arbeit mit „nicht ausreichend“ und in schweren Fällen auch zum Verlust sämtlicher Wiederholungsversuche führen können.

I hereby certify that I have written this thesis independently and have not submitted it elsewhere for examination purposes. I have not used any sources or aids other than those indicated. The work has not been created in whole or in part by an artificial intelligence (AI), unless the AI used to create the work has been expressly approved by the responsible examination board or examiner. I have marked verbatim quotations or quotations in the spirit of the text as such.

I am aware that plagiarism detection software may be used in the assessment of my work.

I am aware that violations of examination regulations can lead to my work being graded as "unsatisfactory" and, in serious cases, to the loss of all repeat attempts.

Unterschrift Studierende/Studierender / Signature student

Japan, den 28.02.2026

Ort, Datum / Place, date