Real-Time Multimodal Cognitive Assistant for Emergency Medical Services

Keshara Weerasinghe
Electrical and Computer Engineering
University of Virginia
cjh9fw@virginia.edu

Saahith Janapati
Computer Science
University of Virginia
jax4zk@virginia.edu

Xueren Ge
Electrical and Computer Engineering
University of Virginia
zar8jw@virginia.edu

Sion Kim
Computer Science
University of Virginia
sk9uth@virginia.edu

Sneha Iyer
Computer Science
University of Virginia
ssi3ka@virginia.edu

John A. Stankovic
Computer Science
University of Virginia
jas9f@virginia.edu

Homa Alemzadeh
Electrical and Computer Engineering
University of Virginia
ha4d@virginia.edu

Abstract*—Emergency Medical Services (EMS) responders often operate under time-sensitive conditions, facing cognitive overload and inherent risks, requiring essential skills in critical thinking and rapid decision-making. This paper presents CognitiveEMS, an end-to-end wearable cognitive assistant system that can act as a collaborative virtual partner engaging in the *real-time* acquisition and analysis of *multimodal data* from an emergency scene and interacting with EMS responders through Augmented Reality (AR) smart glasses. CognitiveEMS processes the continuous streams of data in real-time and leverages edge computing to provide assistance in EMS protocol selection and intervention recognition. We address key technical challenges in real-time cognitive assistance by introducing three novel components: (i) a Speech Recognition model that is fine-tuned for real-world medical emergency conversations using simulated EMS audio recordings, augmented with synthetic data generated by large language models (LLMs); (ii) an EMS Protocol Prediction model that combines state-of-the-art (SOTA) tiny language models with EMS domain knowledge using graph-based attention mechanisms; (iii) an EMS Action Recognition module which leverages multimodal audio and video data and protocol predictions to infer the intervention/treatment actions taken by the responders at the incident scene. Our results show that for speech recognition we achieve superior performance compared to SOTA (WER of 0.290 vs. 0.618) on conversational data. Our protocol prediction component also significantly outperforms SOTA (top-3 accuracy of 0.800 vs. 0.200) and the action recognition achieves an accuracy of 0.727, while maintaining an end-to-end latency of 3.78s for protocol prediction on the edge and 0.31s on the server.

I. Introduction

Emergency Medical Services (EMS) responders provide life support and stabilization for patients before transporting them to a hospital. They first decide on the appropriate emergency response protocol to follow based on their assessment at the incident scene, then execute the medical interventions (treatments such as Cardiopulmonary Resuscitation (CPR) and medications) according to the protocol. They often make critical decisions under time-sensitive conditions and cognitive overload [1], [2], [3]. The right selection of the emergency response protocol is crucial but demanding, requiring the processing of information from many different sources under pressure. Even after selecting the right protocol, responders still need to be on high alert as they execute interventions and monitor the patient's response. Finally, they need to recall all details of the incident and report them in a post-incident form.

Several previous works have focused on developing assistive technologies and cognitive assistant systems to help first responders with decision making [3], [4], [5], [6] and information collection [7], [8], [9]. However, there are still several challenges and research gaps to be addressed.

Challenge of Modeling Domain Knowledge: Previous works have developed pre-defined rule-based systems [3], Behavior Trees (BTs) [4] and state machines [5] for modeling the knowledge on a limited set of EMS protocols and interventions for decision support. These systems, although interpretable, are not easily scalable because of the large amount of time and expert knowledge needed to manually encode and update their models. On the other hand, large deep learning and language models could provide generalization and scalability at the risk of losing transparency [4]. There exists a need for scalable and explainable models that capture domain knowledge while supporting a diverse range of scenarios.

Challenge of Unreliable Communication: Existing cognitive assistant systems leverage cloud-based models and services for transcribing audio from responder conversations into text (e.g., Google Speech-to-Text) [2], [3] and medical information extraction and protocol inference (e.g., MetaMap) [7], [3]. However, responders are often called to areas or situations with unreliable network connections, such as rural areas, sites of catastrophe, and natural disasters. A reliance on the cloud is a liability for a system designed to operate in such scenarios.

Challenge of Noisy and Incomplete Sensor Data: Previous works have mainly focused on developing unimodal systems based on audio data and speech recognition for assisting responders. However, audio in emergency scenes is prone to environmental noise that severely affects the quality of data and accuracy of speech recognition [3], [8], [5]. Also, audio data from responder conversations might not capture all the critical events and observations from the incident scene. Fusing other modalities such as vision and hand activity data [10]

could enable a more holistic understanding of the incident scene and more insightful feedback to responders.

Challenge of Real-time Cognitive Assistance at the Edge: Existing EMS cognitive assistants, such as EMSAssist [6], rely on one-time audio recordings as input to provide one-time protocol predictions as output. But in reality, incidents progress quickly with dynamic levels of information to be processed and decisions to be made. Therefore, reliance on recorded audio as one-time input is insufficient for supporting responders' continuous decision-making in real-time. In addition, previous works utilize unrealistic non-conversational speech data to evaluate their systems [6]. However, responders' conversations are verbalized full-phrased observations and essential information about the scene and patient status [4].

To address these challenges, we present CognitiveEMS, a *real-time multimodal* cognitive assistant system that provides *continuous end-to-end* support to responders through protocol prediction and intervention recognition at the *edge*.

Our contributions are the following:

- Real-time hands-free cognitive assistance at the edge through wireless streaming and parallel processing of multimodal data (audio and vision) from smart glasses.
- EMS-Whisper, a speech recognition model fine-tuned on realistic EMS audio recordings and synthetic EMS conversational audio, generated using a text-to-speech model and an LLM prompted with domain knowledge, for improved real-time transcription at the edge (with WER of **0.290** vs. SOTA's **0.618**).
- EMS-TinyBERT, a tiny language model integrated with EMS domain knowledge and fine-tuned with real EMS data for protocol prediction (with a significantly higher top-3 accuracy of **0.800** vs. **0.200** than SOTA).
- Real-time EMS intervention recognition at the edge using protocol knowledge and zero-shot image classification (with end-to-end accuracy of **0.727**).
- End-to-end performance and latency evaluation of the overall system both on an edge device and a server with multimodal data from simulated EMS scenarios.
- New publicly-available datasets, including human and synthetic EMS conversational audio, multimodal data from simulated EMS scenarios, and an open-source codebase, online at https://github.com/UVA-DSA/EMS-Pipeline.

II. Background and Related Work

Upon arrival at an incident scene, EMS responders interact with the patient and bystanders to collect information about the incident. Then, based on the patient's condition and status they choose an appropriate set of EMS protocols to follow. Each EMS protocol defines a specific set of steps and actions called interventions to stabilize the patient before being transported to the hospital. Various EMS agencies adhere to protocol guidelines established by regional planning agencies. In this work, we follow the protocol guidelines delineated by the Old Dominion EMS Alliance (ODEMSA)[1].

Deciding on the correct protocol is essential for the health and well-being of the patient and is arguably the most important and effortful task taken by the responders that requires experience and domain knowledge. Imagine a scenario where responders arrive at a shopping mall to find a middle-aged patient lying on the floor surrounded by concerned bystanders. Bystanders testify that the patient suddenly collapsed after complaining of chest pain. Upon quick examination, responders find the patient is unresponsive to painful stimuli and without a pulse or respiration. Based on this information, the responders choose the Cardiac Arrest Protocol [11] and conduct the proper interventions in a specific order and duration: start CPR, check rhythm, administer drugs, and support the airway (as illustrated in Figure 1). Like the Cardiac Arrest protocol, all EMS protocols are dynamic in nature. Responders need to continuously make decisions and follow different intervention paths based on how the patient's condition evolves.

Responders document the EMS incidents in the form of electronic Patient Care Reports, referred to as ePCR. ePCR includes information similar to a patient's Electronic Health Records (EHR), such as a textual narrative describing the observations made and actions taken by the responders during the incident, as well as structured data on call type, chief complaints, impressions, procedures, and medications [9], [4].

A. EMS Cognitive Assistants

Cognitive assistants are context-aware and adaptive systems that augment the cognitive capabilities of their users [12]. Previous works have introduced unimodal EMS cognitive assistant systems that process speech from an incident scene to

[1]https://odemsa.net/

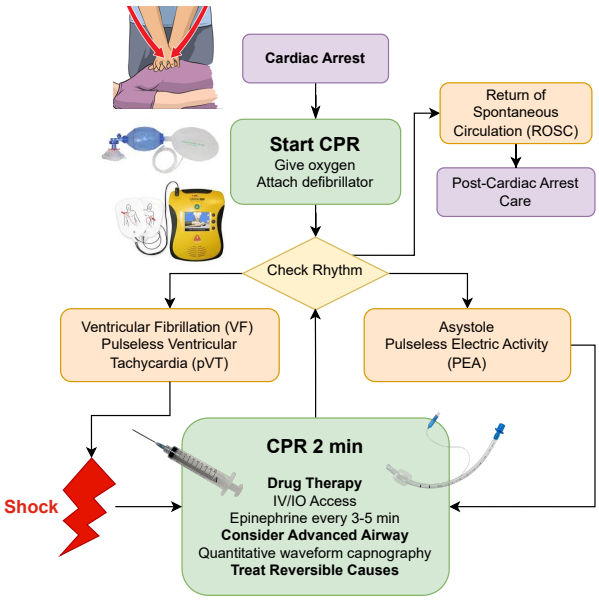

Fig. 1. Responders Continuously Make Decisions During the Cardiac Arrest Protocol (Based on ODEMSA Document [11])

provide first responders with suggestions and feedback regarding the protocols and associated interventions they perform at a scene [4], [6] or to help the responders with automated information extraction and filling of ePCR [7], [8]. These cognitive assistants are limited in their ability to accurately capture the incident context and provide meaningful feedback to responders because speech is not sufficient to obtain a holistic understanding of an emergency scenario. Crucial information is often not explicitly verbalized by responders and patients; instead, it is communicated through other modalities, such as visual signals and cues. For example, the presence or use of specific objects and devices (e.g., ECG monitors, AED devices, etc.) at the incident scene or treatment actions performed by the responders (e.g., clinical procedures such as CPR, Oxygen Administration, Epinephrine Administration) might not be verbalized by the responders. Analysis of multimodal data (e.g., speech and video) from an incident scene can enable the recognition of important objects and actions and more accurate detection of protocols and interventions, allowing timely relevant feedback for improving responders' skills and performance in critical situations. This could be helpful in training where detailed and explainable feedback on executing interventions according to the protocols is needed.

To our knowledge, CognitiveEMS is the first real-time multimodal cognitive assistant system in the EMS domain which leverages both audio and video data from the incident scene for protocol prediction and subsequent intervention recognition.

B. Automatic Speech Recognition

Modern day Automatic Speech Recognition (ASR) models leverage deep learning architectures such as the Transformer [13] and are trained using supervised and unsupervised learning techniques. Conformer [14] and Whisper [15] are both models that utilize the Transformer architecture. In the domain of transcription for medical audio, [16] has developed specialized ASR models solely trained on speech from the medical domain. This requires the curation of a large medical-focused dataset, which is difficult due to financial constraints and patient privacy restrictions. A more feasible approach found to yield improved transcription performance is to fine-tune an existing pretrained model using a (relatively) small amount of domain-specific audio [17]. In addition, synthetic audio has previously been shown to improve the performance of ASR models in specific domains. For example, SynthASR proposes using synthetically generated audio to adapt existing ASR models for specific applications [18].

Within the EMS domain, [3] utilizes cloud services (e.g., Google Cloud Speech-To-Text) for automatic speech recognition. While cloud services offer improved performance via larger models, they are a liability in a cognitive assistant system because they rely on stable network connections, which can be weak or nonexistent where responders are called. [6] proposes the EMSConformer, a fine-tuned Conformer model for offline ASR model for EMS. We find several limitations in this work. First, the mobile application developed in [6] requires responders to manually record and upload their speech on the mobile application, which may increase the cognitive overload on responders. Second, incidents progress at a dynamic and quick rate, with new information to process arising continuously from start to end. Due to the reliance on one-time recorded audio as input, EMSConformer is incapable of transcribing constantly changing information and supporting the decision-making of responders in real-time. Third, EMSConformer has been evaluated on unrealistic and non-conversational speech input. The evaluation data consists of isolated medical terminology and fragmented groups of words. However, responders' conversations and verbalized observations during real incidents will be fully-formed phrases and include words outside of medical terminology to describe signs and symptoms, such as saying, "The patient says they feel a ton of bricks on their chest" instead of saying "chest pain" [9]. Good-quality transcription of conversational audio typically heard at EMS scenes is crucial to obtain a holistic understanding of the incident scene and the status of patients. To our knowledge, EMS-Whisper is the first offline ASR model for EMS conversational data that can continuously transcribe an audio stream, allowing for effective edge deployment of an EMS cognitive assistant. In addition, we develop a novel method for generating synthetic EMS conversational audio to fine-tune pretrained ASR models.

C. Multi-Label Text Classification

Automatic medical diagnosis is the task of assigning diagnosis codes to a patient record based on free-form text. The task of assigning the most relevant subsets of labels to an instance is known as multi-label text classification (MLTC) in NLP. There is extensive prior research on MLTC in the medical domain, mainly for ICD code classification (a medical classification list designated by the World Health Organization) based on EHR data. [19], [20] utilized graph neural networks to combine domain knowledge with BERT models [21] to find the most informative n-grams in the EHR for each ICD9 code. However, most prior works developed models for the clinical domain and cannot be directly applied to the EMS protocol prediction for several reasons. First, there is a domain mismatch between EHR and EMS data (ePCR). Compared with EHR, ePCR and EMS transcripts are collected over a short time frame and contain unconfirmed diagnoses, domain-specific terminology, and abbreviations. Second, most previous works for clinical ICD code classification ignore the impact of model size and inference latency when deployed in real-world scenarios on edge devices. For protocol prediction in EMS domain, [4] proposed a weakly supervised approach to determine the relevance between the current situation at the scene and each EMS protocol by calculating the similarity between their feature vectors. [22] proposed to extract medical entities from EMS domain knowledge to build a heterogeneous graph and fuse domain knowledge with text features to compensate for EMS data scarcity problem. In [6], a deep learning based approach, called EMSMobileBERT, was proposed by fine-tuning light-weighted MobileBERT(25M) on an EMS corpus. However, the input text to EMSMobileBERT

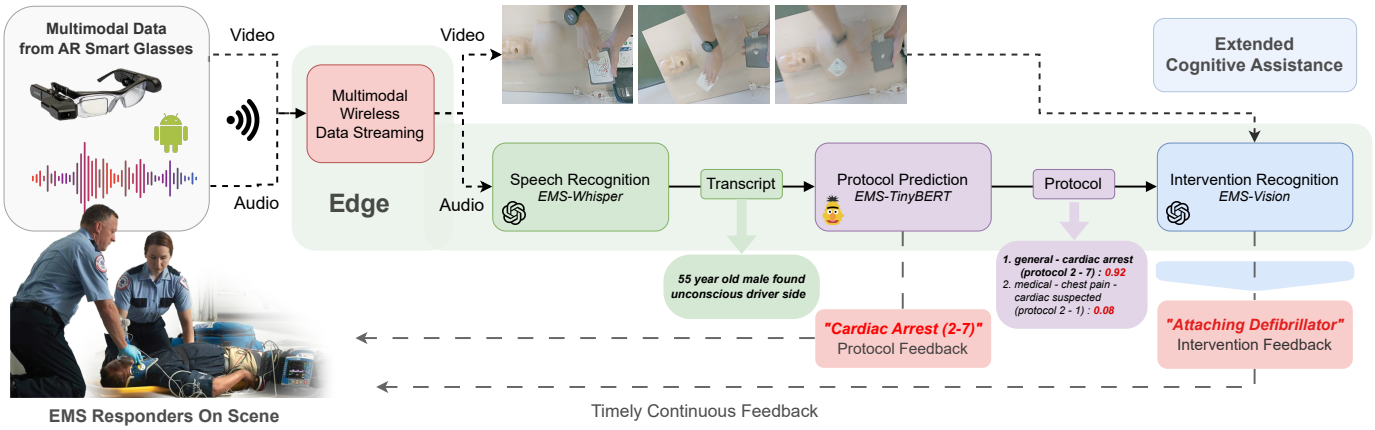

Fig. 2. Overall Architecture of the Real-Time Multimodal CognitiveEMS Pipeline

is in the form of symptom-based medical phrases which is not representative of real EMS scenarios where textual narratives from ePCR or transcribed conversational audio are expected. To our knowledge, our proposed model, EMS-TinyBERT, is the first real-time protocol prediction model that can process realistic textual transcripts (from ePCR or transcribed audio) as input, with a significantly higher top-3 accuracy than SOTA.

D. Human Action Recognition

Human Action Recognition (HAR) has been extensively studied, with applications in numerous domains such as video surveillance, entertainment, sports, [23] and medicine [24], [25], [26]. Previous work has explored various sources of data, [27] including RGB images, depth information, and wearable sensors. However, within the EMS domain, there is limited work on responder action and intervention recognition. [28] proposes a method that utilizes a spatial-temporal fusion convolution neural network (CNN) to recognize actions taken place during emergency incidents by responders. However, like most previous works in other domains, this method relies on a significant volume of annotated EMS data [29], which is notably scarce and costly to obtain. Recently, zero-shot learning methods have overcome traditional supervised learning challenges, including the need for extensive annotations [29] and the inability to generalize to different classes [30]. For instance, CLIP [31] excels in zero-shot image classification by associating images with relevant natural language captions. It is trained by contrastive learning based on a large dataset of image-text pairs from the Internet.

This paper presents our preliminary work on real-time recognition of EMS interventions using the knowledge of protocol guidelines and zero-short image classification. The real-time recognition of responders' actions enables monitoring of the critical aspects of interventions (e.g., CPR rate and depth [10]) and validating them for adherence to established protocol guidelines during training or actual incidents to enhance responders' skills and quality of care. It can also assist in the automated logging of EMS data and ePCR filing [8].

III. Real-time Cognitive Assistant Pipeline

This section presents the overall architecture of the CognitiveEMS pipeline as illustrated in Figure 2. Our primary objective is the creation of a system designed to provide uninterrupted, real-time assistance to EMS responders. As depicted in Figure 3, we develop the CognitiveEMS as a real-time, multi-threaded architecture, aiming for a Service Level Objective (SLO) of a 4-second response interval for delivering protocol prediction feedback. This design criterion is pivotal for offering prompt support to responders amidst critical situations, with the system delivering continuous feedback at 4-second intervals based on the most recent data inputs. The establishment of this SLO was informed by consultations with our Emergency Medical Technician (EMT) collaborators [32] and by considering the response times for critical medical emergencies and current hardware constraints. Future work will focus on overcoming current limitations to achieve better response times and further improve the effectiveness and timeliness of feedback to responders.

Response time is measured from the start of speech recognition to protocol prediction completion and feedback generation. We first implement an Android application to stream the multimodal data from the AR smart glasses to the cognitive assistant pipeline for further processing. The first stage of our pipeline is the speech recognition thread that continuously ingests an audio stream as input, transcribing 4 seconds of buffered audio at a time. These transcriptions are subsequently passed to the next stage for protocol prediction.

Then, the protocol prediction thread produces a ranked list of candidate protocols based on confidence scores. The protocol with the highest confidence is sent to a priority queue for immediate delivery to the responder's AR smart glasses. Next, the protocol output is combined with real-time video from the AR smart glasses for intervention recognition. Only protocols with high confidence are processed, reducing unnecessary video frame processing and increasing system efficiency. Recognized interventions are then sent to the feedback queue. Each stage of the pipeline will be described next by highlighting, where appropriate, what was necessary to

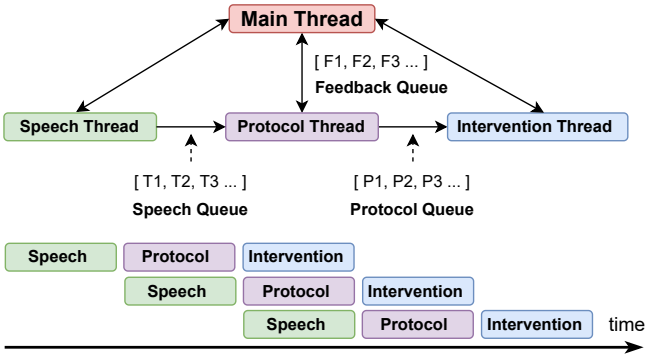

Fig. 3. Parallel Processing at the Edge to Reduce Latency and Provide Timely Continuous Feedback

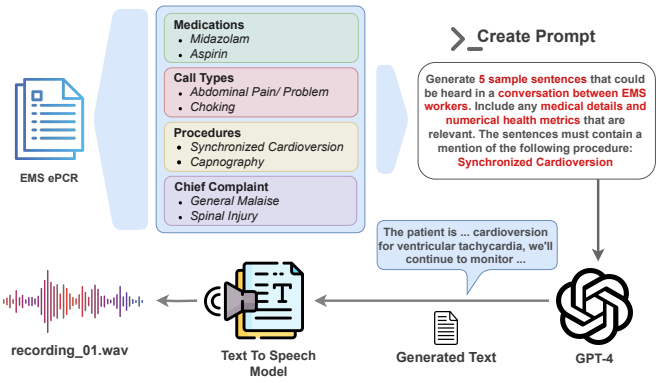

Fig. 4. Generating Synthetic EMS Audio to Address Data Scarcity

make the solution real-time and deployable on edge.

A. Multimodal Wireless Data Streaming

Our cognitive assistant system functions by processing continuous streams of multimodal data in real-time. In emergency situations, EMS responders will wear AR smart glasses (e.g., Vuzix M4000), which continuously transmit audio and video data to the central device of the cognitive assistant pipeline. Once the protocol is successfully predicted, the cognitive assistant relays feedback to the AR display on smart glasses.

We have created an Android application for the AR smart glasses, responsible for three key tasks: continuous streaming of audio from the microphone to capture responders' speech, video capture from the built-in camera, and displaying feedback from the cognitive assistant on the AR display over wireless networks (Wi-Fi). Our primary goal was to optimize low-latency transmission without affecting performance. To achieve this, we designed and implemented the app using multithreading (see Figure 3). For video and audio streams, we use the User Datagram Protocol (UDP) [33] for its lightweight, minimal packet overhead transmission. For feedback, we use the Transmission Control Protocol (TCP) for its reliable delivery, crucial for just-in-time feedback to responders. We have designed an intuitive user interface that ensures minimal interference with the responder's focus during emergencies by delivering feedback in a non-disruptive manner.

B. EMS-Whisper for Speech Recognition

To develop a speech recognition model for the EMS domain, we fine-tune the English-only tiny (39M parameters) and base (74M parameters) sizes of the Whisper family of models. We choose to build off the Whisper architecture because it has been found to be robust to noisy environments, diverse ranges of accents, and has strong out-of-the-box performance on numerous datasets [15]. To make the two models more accurate for the EMS domain, we curate a dataset of conversational EMS speech for domain adaptation.

Dataset: Our dataset to fine-tune the two Whisper models combines speech data from four different sources:

1. Snippets from Simulated Emergency Scenarios from EMS Responders: We consulted with several local EMS agencies and asked responders at these agencies to simulate treating patients with a variety of common patient conditions such as heart attacks, seizures, and breathing difficulties. We then recorded the simulations of enacted scenarios and manually chunked them into shorter audio segments.

2. Read-aloud Snippets From EMSContExt Dataset: We read aloud short segments of reports from the publicly available EMSContExt dataset [7]. These texts contained vitals, medications, and symptoms commonly encountered in EMS scenes that are present in conversational EMS dialogue.

3. EMSAssist Training Dataset: We incorporated the publicly released EMSAssist training set, which contains numerous medical terms and phrases [6]. The EMSAssist training set does not have the issue of long, incoherent sequences of medical keywords that we observe in the test set.

4. Synthetic Data Generated using GPT-4 and ElevenLabs Audio Generator: To train our speech recognition models on a diverse range of EMS terminology, we synthetically generate EMS conversational text using the GPT-4 Large Language Model [34] and synthetic audio of this text with the ElevenLabs Synthetic Voice Generator API.

We first generate natural language prompts for GPT-4 utilizing information extracted from over 35,000 anonymized emergency Patient Care (ePCR) reports from a local urban EMS agency [9]. This extracted information contains 703 unique medications, call types, procedures, and symptoms. For each unique value, we create a prompt to ask GPT-4 to generate conversational EMS text that includes that value. We then feed this generated text to the ElevenLabs synthetic audio API and generated human-like audio with voices corresponding to a diverse range of genders, accents, and speaking rates. This synthetic audio contains rare, specialized terms that do not occur frequently, but are vital for a holistic understanding of emergency scenarios. The synthetic data generation pipeline is depicted in Figure 4.

Table I describes our train/validation/test splits and the quantity of data we obtain from each source. We publicly release the audio data from sources 2 and 4.

Fine-tuning: We used the curated conversational EMS dataset to fine-tune the tiny and base versions of Whisper for the

TABLE I
EMS AUDIO DATASET CURATED FROM DIFFERENT SOURCES

Split	Source (Duration in Minutes) 1	2	3	4	Total
Training set	20.8	0.0	80.4	41.2	142.4
Validation set	0.0	53.0	0.0	17.8	70.8
Test set	21.1	0.0	0.0	0.0	21.1
Total	41.9	53.0	80.4	59.0	234.3

EMS domain. We perform fine-tuning of Whisper with HuggingFace's Transformer's library and Trainer API [35]. We performed hyperparameter tuning using our validation set. Our final tiny model was trained for 1 epoch with a dropout rate of 0.1. Our final base model was trained for 2 epochs with a dropout of 0. Both models were trained using a batch size of 8 and a learning rate of 1×10^{-5} with the Adam optimizer.

We additionally train variants of the tiny and base models without synthetic data to assess whether the addition of the synthetic data improves performance. We designate Whisper fine-tuned on the full dataset with the suffix **fine-tuned** and models fine-tuned without synthetic data with the suffix **wo-synthetic**. Using our full training set, we also fine-tune the Conformer model pretrained on 1,000 hours of the LibriSpeech dataset to compare our performance with SOTA [14], [36].

Edge Deployment: To reduce computational strain on the edge device, we convert the fine-tuned Whisper models to the GGML format [37] with whisper.cpp [38], an open-source library that provides a zero-dependency high-performance C++ inference API for the Whisper models. We further optimize inference of the Encoder portion of the Whisper model using NVIDIA's cuBLAS library, which allows for partial utilization of the GPU of our edge device during inference time.

Streaming Setup: We adapt whisper.cpp's open-souce stream code to ingest an audio stream from an audio device. We process the audio stream at a sampling rate of 16,000Hz. A buffer accumulates chunks of 1024 audio samples until four seconds worth of samples (16000 x 4 = 48000) are collected. The buffer is then fed to the Whisper model for inference. This accumulation and transcription of audio data proceeds continuously until the cognitive assistant is terminated. We utilize multiprocessing to handle the audio buffering task to maximize efficiency and parallelism and ensure low latency.

C. EMS-TinyBERT for Protocol Selection

We propose the EMS-TinyBERT which can be deployed on edge devices for accurate real-time protocol selection. Our model consists of three parts as shown in Figure 5: a text encoder to encode EMS transcripts (from ePCR or input audio) into text features; a graph neural network to fuse domain knowledge with text features; a group-wise training strategy to deal with scarcity of training samples in rare classes.

Text Encoder: We adopt TinyClinicalBERT [39], a BERT model pretrained on medical corpus, as the text encoder. After removing the punctuations and stopwords in the EMS transcripts t, we use TinyClinicalBERT to encode the transcript

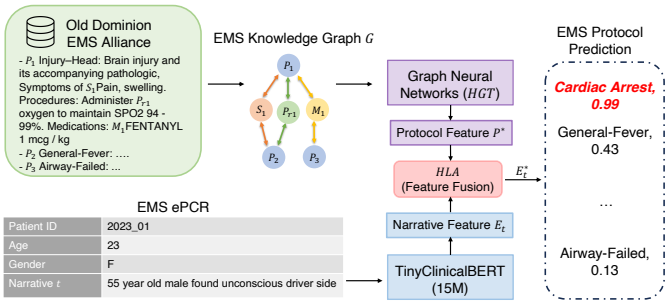

Fig. 5. EMS-TinyBERT: A Tiny Model that Fuses EMS Domain Knowledge (Old Dominion EMS Alliance) to Improve Protocol Prediction Performance

and take the last hidden states of TinyClinicalBERT as text features $\mathbf{E}_t \in \mathbb{R}^{\text{seq}\times\delta}$, where seq is the length of text and δ is the dimension of hidden states in TinyClinicalBERT.

Domain Knowledge Fusion: To utilize external medical domain knowledge for EMS protocol prediction, we follow the methods in [22] to construct a heterogeneous graph G and utilize a feature fusion module to integrate domain knowledge into text features for classification.

ODEMSA is utilized as medical guidelines in our works, which contain detailed descriptions of different medical conditions, diverse relations between medical entities like signs and symptoms of specific conditions, and the interventions that medical professionals should perform to provide consistent patient care. Specifically, we manually extract the symptoms, medications, and procedures for every EMS protocol and then construct a graph $G = (N, E)$, with N as the set of nodes (protocols, medications, symptoms, procedures) and E as the set of edges. We use GatorTron [40] to generate initial node embeddings based on the descriptions or names of the nodes in the protocol guidelines. Then we adopt a 1 layer heterogeneous graph transformer (HGT) [41] to capture the relations between different protocol nodes in graph G. The output of HGT are updated node embeddings ($\mathbf{P}^\star = \text{HGT}(G)$), from which we only use the updated protocol features for feature fusion.

Then a heterogeneous label-wise attention [22] (HLA) module takes both protocol features $\mathbf{P}^\star$ and text feature E_t as input and fuses the features by having the protocols assign different weights for each token in the text representation ($\mathbf{E}_t^\star = \text{HLA}(\mathbf{P}^\star, \mathbf{E}_t)$). Finally, the fused features $\mathbf{E}_t^\star$ are flattened and fed into a linear layer for classification.

Group-wise Training: In the EMS domain, there are often different protocols for the treatment of the same condition for patients with different pre-existing conditions (e.g., pediatric vs. adult patients). While the signs and symptoms for such protocols are similar to those of adults, the guidelines on specific interventions that are appropriate for children are different. Therefore, we group the highly similar protocols(e.g.: *Medical-seizure (adult protocol 3-12)* and *Medical-seizure (pediatric protocol 9-12)*) during the model training phase for coarse-grained classification. During inference, we used predefined rules (e.g.: age $\leq$ 18 indicates a pediatric case) to do fine-grained classification. We use the sigmoid function to normalize the model's final output to be between 0 and 1.

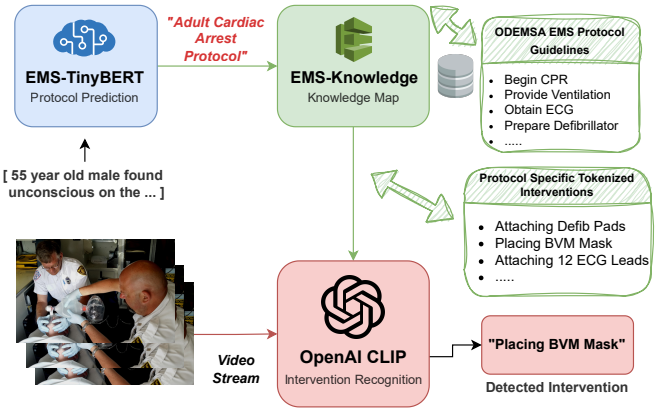

Fig. 6. Integrating EMS Protocol Knowledge to Improve Intervention Recognition Module Performance

A binary cross entropy loss is used to measure the distance between prediction and ground truth. The final output from the model is the EMS protocol predictions and their corresponding confidence scores (see Figure 5).

Edge Deployment: We implement two strategies to shorten the inference time on the edge device. First, we used a lightweight 15M parameter pretrained model, TinyClinicalBERT, as our text encoder. Secondly, to reduce the time of constructing an EMS knowledge graph, we store all the initial node embeddings in the memory so that the model does not need to repetitively generate initial embeddings during inference. Moreover, during the system startup, we initialize the protocol prediction model with a warm-up inference to allocate GPU resources efficiently to minimize runtime overheads.

Streaming Setup: EMS-TinyBERT is capable of receiving and processing text input in chunks during the inference phase, enabling real-time prediction based on streaming textual transcripts from audio data. These textual chunks are progressively accumulated over the course of the EMS responder interactions to achieve a holistic understanding of the incident context for more accurate prediction. We utilize this ability to support end-to-end evaluations of our system.

D. EMS-Vision for Intervention Recognition

Our intervention recognition approach (Figure 6) consists of three subcomponents. The protocol predictions from EMS-TinyBERT serve as a contextual basis, and a knowledge agent established upon protocol guidelines provides a subset of interventions associated with the protocol to the vision module. The vision module utilizes a zero-shot image classification model, CLIP proposed by [31], which takes in video frames and candidate labels for interventions. To fully exploit the potential of the CLIP model, identifying a pertinent and concise subset of interventions in natural language is critical, considering the comprehensiveness of EMS protocols that encompass extensive interventions.

For our preliminary implementation, we consider two protocols *Medical - chest pain - cardiac suspected (protocol 2 - 1)* and *Medical - respiratory distress/asthma/copd/croup/reactive airway (respiratory distress)* defined in ODEMSA. For every 4 seconds of multimodal audio/video data, the output with the highest confidence from the protocol prediction module is fed to a knowledge agent as depicted in 6. The knowledge agent will provide the associated interventions for the protocol as input to the CLIP model. The CLIP model will then generate a predicted intervention output for every frame in the window. The windows where the protocol prediction confidence is low will not have any intervention predictions. The knowledge agent will provide interventions described in natural language to provide CLIP with an image-text pair that encompasses a stronger semantic relationship while minimizing the similarity with other non-matching pairs.

Edge Deployment: The initial configuration of the CLIP model demands computational resources and dependencies that are not available on our edge device, the NVIDIA Jetson Nano. Hence, we utilize a lightweight implementation of CLIP written in C/C++ to support CPU-Only inference designed for resource-constrained devices [42]. Moreover, we apply 8-bit integer quantization [43] which effectively transforms the trained model to represent weights and activations of layers in the neural net with 8-bit integers over high-precision values such as 32-bit floating point numbers. This significantly reduces the model size which essentially leads to lower memory requirements and faster inference times.

IV. Evaluation

We perform a comprehensive evaluation of the performance and latency of the standalone modules and the end-to-end pipeline compared to EMSAssist [6] models, including EMSConformer for speech recognition and EMSMobileBERT for protocol prediction, using the following datasets and metrics.

A. Evaluation Datasets

Electronic Patient Care Report (ePCR) Dataset is a collection of 4,417 anonymized pre-hospital ePCRs obtained from a local urban ambulance agency. Each ePCR contains first responders' textual descriptions of the patient's situation, interventions performed, patient's medical history, and EMS protocols used.

EMS Audio Dataset is comprised of two subsets. The first subset is used for standalone evaluation of speech recognition models to assess the effect of fine-tuning on model performance. This subset consists of short pre-chunked segments of audio, as is standard in general speech recognition datasets. The second subset is used to evaluate the speech recognition and protocol selection models in the context of our end-to-end pipeline. This subset contains longer post-incident narratives, along with transcripts and protocol labels for each narrative.

EMS Video Dataset is a collection of 40 videos (equating to approximately 17000 images) of two distinct intervention actions: 'Defibrillation' using AED and 'Supporting Airway' using a Bag Valve Mask (BVM). Defibrillation is relevant for cardiac-related emergencies, and BVM is relevant to respiratory distress emergencies. This dataset is used for the standalone evaluation of the intervention recognition module.

EMS Multimodal Dataset consists of eight video and audio scenarios simulating a realistic EMS incident from start to end. This dataset is used for an integrated evaluation of speech recognition and intervention recognition. To collect this dataset, we first developed eight realistic EMS incident scenarios for cardiac-related and respiratory distress emergencies by referencing several example EMS training scenarios from Alabama Public Health [44] and the EMS Online [45] websites, combined with a team member's background knowledge from their time as an EMT. We then acted out the scenarios in a simulated setting with a dummy patient and recorded synchronized audio and video using the CognitiveEMS pipeline. Due to insufficient incident details in the conversational part of this dataset, the protocol prediction module could not function as intended on the small chunks of transcribed audio, resulting in low performance. So we decided to not use this dataset for the end-to-end evaluation of the protocol prediction module.

B. Evaluation Metrics

Speech Recognition: We use the standard metrics of Word Error Rate (WER) and Character Error Rate (CER) to assess the performance of our speech recognition module. When calculating metrics for the Whisper models, we first normalize the predictions using the model's tokenizer, which standardizes punctuation and special characters. When calculating metrics for SOTA's EMSConformer, we first preprocess the ground truth transcripts remove all punctuation except for periods, and convert numbers to their phonetic representations because the EMSConformer has a limited vocabulary.

We also report latency measurements (in milliseconds) for the different speech recognition models when they are run as part of the end-to-end pipeline. These measurements correspond to the time it takes for the model to transcribe one chunk of audio, which we have chosen to be 4 seconds long.

EMS Protocol Prediction: We used multiple metrics for the evaluation of the protocol selection module to have a fair and complete comparison to SOTA. Threshold-based metrics like Micro F1(miF) and Macro F1(maF) are used by setting the threshold as 0.5. Ranking based metrics like top-k accuracy($Acc@K$) which do not require a specific threshold are also reported. Since the average number of labels per instance in the ePCR dataset is 1.2, we set $K = 1$ for top-k accuracy($Acc@K$), but also report $Acc@3$. We describe these metrics and their drawbacks below:

- miF is heavily influenced by frequent protocols and thus can be used to evaluate the overall performance.
- maF weighs the F1 score achieved on each protocol equally and is used to evaluate the performance for the rare protocols.
- $Acc@K$ computes the number of times where the correct label is among the top k predicted labels, which are ranked by confidence scores.

Intervention Recognition: We use the following metrics for the evaluation of intervention recognition module. These metrics are used to assess the impact of protocol knowledge input on the intervention recognition and the end-to-end real-time performance.

- *Accuracy* refers to the ratio of video frames classified as the correct intervention to all video frames associated with the intervention. Each video frame is labeled by a human with the intervention taking place at that moment, and this is used to identify whether the intervention recognition outputs agree with the ground truth.
- *Latency* refers to the duration between the retrieval of a video frame by the vision module and the generation of recognized intervention.

End-to-End Evaluation: We evaluate the performance and latencies of each module in the pipeline when operated in an end-to-end setting. We use the same metrics defined for each module with the addition of latency measurement. Moreover, we report two overall latencies for protocol feedback generation and intervention recognition in an end-to-end execution.

C. Experimental Platforms

We implement our cognitive assistant system on two platforms for performance comparison and end-to-end evaluation.

Edge Device: We utilize a Jetson Nano, an Edge AI apparatus manufactured by NVIDIA. The Jetson Nano provides an ARM A57 Quad Core processor with a clock speed of 1.43Ghz, RAM of 4GB, and a 128-Core Maxwell GPU running on Ubuntu 18.04 with CUDA 10.2. Our evaluation only involves the execution of the inference phase of the models, with no involvement of the edge device in the training process.

Server: We utilize a server configured with a 13th Generation Intel(R) Core(TM) i9-13900KF processor, operating at a clock speed of 3.0 GHz and featuring a total of 32 CPU cores. This server is complemented by 32 gigabytes of system RAM and accommodates an NVIDIA RTX 4080 GPU, which possesses 9728 CUDA cores with 16 gigabytes of dedicated GPU memory running on Ubuntu 20.04.

D. EMS-Whisper Evaluation

We perform two evaluations of our ASR models. First, we evaluate the models' standalone performance with short pre-chunked segments of audio, as is common in standard evaluations of ASR systems. We perform this evaluation on the server using the test split of our EMS Audio dataset.

We also assess the performance of the speech recognition models when integrated into the end-to-end pipeline and processing a stream of audio in parallel with the other modules. For this second evaluation, we use the full-length conversational post-incident EMS reports from subset (2) of the aforementioned EMS Audio Dataset and the recordings from the EMS Multimodal dataset. We also calculate the WER and CER of the combined transcription output of the entire report (which is the finalized transcripts of each chunk concatenated together). We also calculate the average latency to process every 4-second chunk of audio. We do not report results for Whisper base results on the Jetson edge device because the model exhausted the computational resources of

the device when run in parallel with other modules of the pipeline. We also only report metrics for the EMSConformer on the server because we encountered compatibility issues when implementing the model on the NVIDIA Jetson.

E. EMS-TinyBERT Evaluation

We use the scikit-multilearn [46] to split the ePCR dataset by the ratio of 70:30 for training and testing and further perform 3-fold cross-validation. The evaluation results are based on the average performance of the three test sets to alleviate the bias introduced by data splitting. There are in total 43 EMS protocol labels in our dataset.

We used GatorTron [40] to generate the initial embeddings for all the nodes in the knowledge graph. The Heterogeneous Graph Transformer(HGT) [41] was adopted as the GNN model for domain knowledge fusion. We set the number of layers in HGT as 1, and the hidden dimension in HGT is set as 256. We use Adam optimizer for training with batch sizes ranging from 4 to 32 and learning rates ranging from 1e-6 to 1e-3 for hyper-parameter tuning. To avoid over-fitting, we use regularization with the weight decay of 1e-5, a dropout rate of 0.3, and early stopping if the validation loss keeps increasing more than three times. We implemented the EMS-TinyBERT based on the Huggingface [35] and the PyTorch [47]. The model was trained with one NVIDIA GPU RTX3090.

F. EMS-Vision Evaluation

Using the intervention videos dataset, we evaluate the performance on the server with and without protocol knowledge to assess the impact of integrating the knowledge when using zero short models. Moreover, we only evaluate intervention recognition using our fine-tuned speech models to avoid redundant results. Then, we employ the end-to-end Scenarios dataset featuring simulated interactions between EMS responders and a patient encountering cardiac-related and respiratory distress emergency scenarios, which were collected to assess the performance of our intervention recognition system under more realistic conditions. This evaluation is part of the end-to-end system evaluation explained in the next section.

G. End-to-End System

We deploy the cognitive assistant on two platforms, as previously described. The edge-deployed version, however, features fundamental differences in the implementation of certain components, such as speech recognition and action recognition, aimed at optimizing computational resource utilization while ensuring acceptable response times. For evaluation purposes, we simulate audio streaming from AR smart glasses by creating a separate process that reads an audio file and writes it to a virtual microphone interface through PulseAudio [48], a networked low-latency sound server for Linux and POSIX systems. In the case of video streaming, we load the video data and transmit frames at their original rate to mimic real-time video feed, accomplished using OpenCV to ensure the intervention recognition module's proper functioning [49].

We utilized ten recordings acquired from an EMS agency in the Audio Dataset to conduct an end-to-end assessment of the Speech Recognition module, followed by the Protocol Prediction module. However, since we lack access to the videos associated with these recordings, we are unable to evaluate the Intervention Recognition module. As an alternative, we use the Multimodal Dataset, which consists of 8 self-collected scenarios, to evaluate the end-to-end performance from speech recognition to intervention recognition.

Furthermore, for a more comparative analysis of our cognitive assistant pipeline, we implement SOTA [6] speech recognition module (EMSConformer) and protocol prediction module (EMSMobileBERT) on our server. However, due to certain incompatibilities in software dependencies, we could not implement and evaluate their models on our edge device. In order to ensure a fair comparison, we trained and fine-tuned their models on our own datasets. Also, for performance comparison at the edge, we use the best results of their model performance on our server to our results at the edge. This is assuming that their model has the same or better performance on the server than the edge as reported in [6]. Performance metrics and latencies for each stage/module in our pipeline are reported in Tables V and VI. Additionally, we provide an overall latency measurement for protocol prediction feedback and intervention recognition.

TABLE II
STANDARD ASR EVALUATION ON EMS AUDIO DATASET

Model	WER	CER
tiny	0.156	0.093
tiny-fine-tuned-wo-synthetic	0.158	0.094
tiny-fine-tuned (39M)	**0.152**	**0.091**
base	0.133	0.079
base-fine-tuned-wo-synthetic	0.131	0.079
base-fine-tuned (74M)	**0.122**	**0.077**
EMSConformer-tflite (10M)	0.436	0.250
EMSConformer-base (10M)	0.355	0.203

V. RESULTS

A. EMS Speech Recognition

Standard Speech Recognition Evaluation: We first discuss the results of the traditional evaluation of the speech recognition models on the test split of our curated EMS Speech dataset. As shown in Tabel II, we observe that **Whisper models fine-tuned with both human and synthetic data achieve the best performance**. Our tiny-fine-tuned model achieves a WER of 0.152 while the original tiny model achieves a WER of 0.156. Our base-fine-tuned model achieves a WER of 0.122 while the original base achieves a WER of 0.133. Both of our fine-tuned Whisper models achieve substantially lower error rates compared to SOTA's fine-tuned base and lite models (with WER of 0.355 and 0.436, respectively). These results demonstrate that fine-tuning the pretrained Whisper models using human and synthetically-generated audio can improve transcription accuracy for conversational EMS speech. We also find that the **tiny-fine-tuned model offers the best tradeoff** between model size and WER for edge deployment.

TABLE III
PROTOCOL SELECTION COMPARISON TO STATE-OF-THE-ART ON EPCR DATASET, EMS-TINYBERT (15.9M) INCLUDES BOTH KNOWLEDGE FUSION AND GROUP-WISE TRAINING ON EPCR DATASET.

Model (Size)	miF	maF	Acc@1	Acc@3
TinyClinicalBERT (15M)	0.673	0.171	0.679	0.807
EMSMobileBERT (25M)	0.216	0.034	0.226	0.439
TinyClinicalBERT-Group (15M)	0.709	0.277	0.696	0.836
TinyClinicalBERT-Knowledge Fusion (15.9M)	0.743	0.395	0.744	0.898
EMS-TinyBERT (15.9M)	**0.756**	**0.499**	**0.765**	**0.904**

TABLE IV
PERFORMANCE OF INTERVENTION RECOGNITION WITH AND WITHOUT PRIOR KNOWLEDGE OF THE PROTOCOL ON EMS VIDEO DATASET

With Protocol Knowledge		Without Protocol Knowledge	
Intervention	Accuracy	Intervention	Accuracy
Attaching Defibrillator	**0.79**	Attaching Defibrillator	0.51
Inserting IV to arm	0.11	Attaching nebulizer	0.16
Inserting IV to leg	0.07	Defibrillator	0.09
Defibrillator	0.02	Inserting IV to arm	0.07
Placing Oxygen mask on face	**0.64**	Placing Oxygen mask on face	0.22
Attaching nebulizer	0.21	Inserting IV to arm	0.19
Inserting airway adjunct	0.07	Attaching two Defib pads on chest	0.15
Administering albuterol	0.03	Inserting IV to leg	0.15

B. EMS Protocol Prediction

We compare our protocol prediction model, EMS-TinyBERT, with SOTA models including EMSMobileBERT, TinyClinicalBERT in terms of performance and model size. Results show that EMS-TinyBERT achieves a good trade-off between memory usage and performance. Specifically, **it outperforms SOTA methods by more than 10% in performance on ePCR dataset, while only introducing 6%** overhead in number of parameters (15.9M vs. 15M).

As shown in Table III, EMS-TinyBERT has the best performance over all the baselines. For frequently used EMS protocols, EMS-TinyBERT has the highest miF 0.756, which outperforms EMSMobileBERT (0.216) and TinyClinicalBERT (0.673), showing that EMS-TinyBERT can provide correct EMS protocols for first responders when encountering common conditions. For rarely used protocols, EMS-TinyBERT has the best maF score (0.499) over all other models, indicating its good ability to provide EMS protocol recommendations for rare conditions. Using ranking based evaluation metrics, EMS-TinyBERT achieves the best performance among all models. Acc@1 in EMS-TinyBERT indicates that 76.5% top1 recommendation is correct. Furthermore, Acc@3 of EMS-TinyBERT is 90.4%, indicating there is 90.4% chance that ground truth EMS protocols are in the top-3 predictions of our model. The comparison also shows that EMSMobileBERT does not perform well on real-world ePCR data and can not be applied to EMS conversational scenarios.

We also validate the necessity of each module in the design of EMS-TinyBERT model. Table III shows that both Group-wise training and knowledge fusion help to improve the model's ability in protocol prediction. It is notable that the knowledge fusion module will introduce an extra 0.9M parameters to the raw model because the graph neural network is used to incorporate EMS domain knowledge. However, compared with baseline TinyClinical(15M), it is acceptable to have a 6% increase in the number of parameters to achieve more than 10% performance improvement. Besides, EMS-TinyBERT(15.9M) has better performance and less overhead than EMSMobileBERT(25M) on edge devices.

C. EMS Intervention Recognition

To assess the effect of using multimodal data, we compare the results of intervention recognition obtained with and without the knowledge of protocol. Table IV shows two examples of recognizing *Attaching Defibrillator* in *Cardiac Arrest* protocol and *Placing Oxugen Mask* in *Respiratory Distress* protocol. When incorporating the protocol knowledge (from ground truth labels), the CLIP model is provided with a subset of natural language tokens describing the interventions that are relevant to the target protocol. But without such knowledge, the CLIP model will consider the full set of possible EMS interventions. We see that by **incorporating the protocol knowledge, the accuracy of the intervention recognition improves significantly (28% (0.79 vs. 0.51)-42% (0.64 vs. 0.22))**. As indicated in Table IV, without the protocol knowledge, the CLIP model faces difficulties in identifying the most suitable intervention that describes the image, resulting in a higher number of false positives as there are too many candidate interventions to consider.

D. End-to-End System

We evaluate the end-to-end performance and latency of our cognitive assistant pipeline, including a comparative analysis of our proposed speech and protocol prediction models and the previous SOTA. In Tables V and VI, we provide performance metrics and latencies for each stage of the pipeline.

Speech Recognition Performance and Latency: In the end-to-end pipeline, we observe that our **tiny-fine-tuned model achieves the lowest WER of 0.290 on the Jetson edge device** and also outperforms SOTA's fine-tuned Conformer model running on the server (which had a WER of 0.591). Our **base-fine-tuned model achieves the lowest WER on the server** across all models with a WER of 0.225 and achieves a substantially lower WER rate than SOTA's base model (0.591).

We also observe that our **fine-tuned-tiny model achieves a 3.65 second latency on our edge device** when processing a 4-second chunk of audio, which is comparable to the latency of SOTA's EMS-Conformer (3.60 seconds).

Our speech results demonstrate that our fine-tuned Whisper models outperform SOTA and the original Whisper models on conversational EMS audio in a real-time streaming setup. We also demonstrate that our tiny-fine-tuned model runs effectively on our edge device in conjunction with other modules of our cognitive assistant pipeline. V.

Protocol Prediction Performance and Latency: Our results demonstrate a significant improvement in protocol prediction performance compared to the SOTA (EMSMobileBERT) both on the server and the edge device. On the server, **the best miF of our EMS-TinyBERT is 0.650 compared with 0.071 on EMSAssist, the best Acc@1 of EMS-TinyBERT is 0.650**

TABLE V
AVERAGE END-TO-END PERFORMANCE OF COGNITIVEEMS PIPELINE ON EDGE DEVICES VS. SERVERS COMPARED TO SOTA

Speech Model	Protocol Model	EMS Audio Dataset										EMS Multimodal Dataset	
		Speech Recognition				Protocol Prediction						Intervention Recognition	
		Server		Edge		Server			Edge			Server	Edge
		WER	CER	WER	CER	miF	Acc@1	Acc@3	miF	Acc@1	Acc@3	Accuracy	Accuracy
tiny.en	EMS-TinyBERT	0.276	0.176	0.301	0.201	0.550	0.550	0.700	**0.650**	**0.650**	**0.800**	-	-
tiny-wo-synthetic	EMS-TinyBERT	0.271	0.175	0.290	0.195	0.600	0.600	**0.800**	0.600	0.600	0.700	-	-
tiny-fine-tuned	EMS-TinyBERT	**0.266**	**0.162**	**0.290**	**0.192**	**0.600**	**0.600**	0.700	0.600	0.600	0.750	**0.743**	**0.727**
base.en	EMS-TinyBERT	0.281	0.186	-	-	**0.650**	**0.650**	0.800	-	-	-	-	-
base-wo-synthetic	EMS-TinyBERT	0.261	0.175	-	-	0.500	0.500	0.800	-	-	-	-	-
base-fine-tuned	EMS-TinyBERT	**0.225**	**0.152**	-	-	0.600	0.600	**0.850**	-	-	-	**0.775**	-
EMSConformer-tflite*	EMSMobileBERT	0.618	0.440	-	-	0.071	0.100	0.200	-	-	-	-	-
EMSConformer-tf-base*	EMSMobileBERT	0.591	0.429	-	-	0.020	0.056	0.111	-	-	-	-	-

* Based on our implementation of EMSConformer and EMSMobileBERT [6] models using our data and server. Not evaluated on our edge device due to incompatibilities.

TABLE VI
AVERAGE END-TO-END LATENCY OF COGNITIVEEMS PIPELINE ON EDGE DEVICES VS. SERVERS COMPARED TO SOTA

Speech Model	Protocol Model	Average Latency (s)									
		Speech Recognition		Protocol Prediction		Intervention Recognition		Protocol Feedback		Intervention Feedback	
		Server	Edge	Server	Edge	Server	Edge	Server	Edge	Server	Edge
tiny-fine-tuned	EMS-TinyBERT	0.16	**3.65**	0.01	**0.14**	0.02	4.46	0.18	**3.78**	0.19	8.24
base-fine-tuned	EMS-TinyBERT	0.30	-	**0.01**	0.18	0.02	-	0.31	-	0.32	-
EMSConformer-tf-lite	EMSMobileBERT	**0.10**	3.60*	0.01	0.60*	-	-	**0.11**	4.20*	-	-
EMSConformer-base	EMSMobileBERT	0.19	-	0.02	-	-	-	0.21	-	-	-

* As reported in [6] on an Essential Phone PH-1 and not evaluated on our edge device due to incompatibilities.

compared with 0.100 on EMSAssist, and the best Acc@3 **of EMS-TinyBERT is 0.850 compared with 0.200 on EMSAssit. The latency on the server of our EMS-TinyBERT is 0.01s compared with 0.02s on EMSAssist**. Compared with EMSAssist, our EMS-TinyBERT takes less inference time and achieves better protocol prediction performance. There are several reasons for the superiority of our protocol prediction model. First, compared with EMSAssist, EMS-TinyBERT has fewer parameters (15M) than EMSMobileBERT (25M), which shortens the inference time on both the server and the edge device. However, EMS-TinyBERT achieves better EMS protocol prediction because of utilizing a medical pretrained model TinyClinicalBERT as the backbone and incorporating EMS domain knowledge for decision making.

Intervention Recognition Performance and Latency: Notably, intervention recognition is able to **achieve comparable accuracy on both edge and server, 0.727 vs. 0.743, respectively**. This difference in accuracy can be attributed to the changes we applied to the base model to accommodate the resource constraints on the edge device. Due to the limited resources available, we observe a significantly higher latency on the edge device (4.46s) compared to the server (0.02s). This is primarily due to the inability to use the base CLIP model that utilizes GPU resources to accelerate the inference. Instead, the entire inference process on the edge device is performed on the CPU, which has low compute capacity and remains a limitation for this edge device. Moreover, on the server, the base fine-tuned speech model has been shown to indirectly improve intervention recognition accuracy **0.775** vs. **0.743** through more accurate transcriptions, which are utilized by the downstream protocol prediction model to output more accurate protocol predictions.

End-to-End Latency: Our results show that the **end-to-end latency for protocol feedback outperforms SOTA with a considerably lower value of 3.78s vs. 4.20s on the edge**, while meeting the pre-defined deadline of **4s** for the real-time system. While speech recognition latencies are comparable to SOTA (**3.65s** vs. 3.60s), the latency for protocol prediction is significantly reduced (**0.14s** vs. 0.60s), thereby contributing to an overall reduction in latency.

VI. CONCLUSION

This paper presents a novel real-time cognitive assistant deployed on edge that utilizes multimodal data to continuously support EMS responders during emergencies. With a fine-tuned speech recognition model for EMS conversations and a protocol prediction model built using a tiny language model incorporating EMS domain knowledge and the novel addition of intervention recognition using multimodal data, our cognitive assistant demonstrates significant performance improvement compared to SOTA. Despite using real-world data in our evaluations, the real-world deployment of the proposed system in EMS training or actual incidents requires further improvement, refinements, and evaluation in consultation with the first responders. Future work will focus on improving the real-time accuracy and performance and validating the practical utility and user-friendliness of our cognitive assistant system through real-life incident simulations and feedback integration in collaboration with our partner EMS agencies.

VII. Acknowledgement

This work was supported in part by the award 70NANB21H029 from the U.S. Department of Commerce, National Institute of Standards and Technology (NIST).

References

[1] J. Sweller, "Cognitive load theory," in *Psychology of learning and motivation*. Elsevier, 2011, vol. 55, pp. 37–76.

[2] S. M. Preum, S. Shu *et al.*, "Towards a cognitive assistant system for emergency response," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018, pp. 347–348.

[3] S. Preum, S. Shu *et al.*, "Cognitiveems: A cognitive assistant system for emergency medical services," *ACM SIGBED Review*, vol. 16, no. 2, pp. 51–60, 2019.

[4] S. Shu, S. Preum *et al.*, "A behavior tree cognitive assistant system for emergency medical services," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6188–6195.

[5] M. A. Rahman, L. Jia *et al.*, "emsreact: A real-time interactive cognitive assistant for cardiac arrest training in emergency medical services," in *2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*, 2023, pp. 120–128.

[6] L. Jin, T. Liu *et al.*, "Emsassist: An end-to-end mobile voice assistant at the edge for emergency medical services," in *Proceedings of the 21st Annual International Conference on Mobile Systems, Applications and Services*, 2023, pp. 275–288.

[7] S. M. Preum, S. Shu *et al.*, "Emscontext: Ems protocol-driven concept extraction for cognitive assistance in emergency response," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 13 350–13 355, Apr. 2020. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7048

[8] M. A. Rahman, S. M. Preum *et al.*, "Grace: generating summary reports automatically for cognitive assistance in emergency response," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 08, 2020, pp. 13 356–13 362.

[9] S. Kim, W. Guo *et al.*, "Information extraction from patient care reports for intelligent emergency medical services," in *2021 IEEE/ACM Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*. IEEE, 2021, pp. 58–69.

[10] M. A. Rahman, K. Weerasinghe *et al.*, "Senseems-towards a hand activity recognition and monitoring system for emergency medical services," in *The 22nd International Conference on Information Processing in Sensor Networks*, 2023, pp. 310–311.

[11] O. Protocols, "2.7 general – cardiac arrest," https://www.odemsaprotocols.com/Protocols/Section02/2.7%20ALS%20Adult%20Cardiac%20Arrest.pdf.

[12] S. M. Preum, S. Munir *et al.*, "A review of cognitive assistants for healthcare: Trends, prospects, and future directions," *ACM Computing Surveys (CSUR)*, vol. 53, no. 6, pp. 1–37, 2021.

[13] A. Vaswani, N. Shazeer *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[14] A. Gulati, J. Qin *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," 2020.

[15] A. Radford, J. W. Kim *et al.*, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518.

[16] C.-C. Chiu, A. Tripathi *et al.*, "Speech recognition for medical conversations," 2018. [Online]. Available: https://arxiv.org/pdf/1711.07274.pdf

[17] J. Luo, J. Wang *et al.*, "Cross-language transfer learning and domain adaptation for end-to-end automatic speech recognition," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2021, pp. 1–6.

[18] A. Fazel, W. Yang *et al.*, "Synthasr: Unlocking synthetic data for speech recognition," *arXiv preprint arXiv:2106.07803*, 2021.

[19] A. Rios and R. Kavuluru, "Few-shot and zero-shot multi-label learning for structured label spaces," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, vol. 2018. NIH Public Access, 2018, p. 3132.

[20] J. Mullenbach, S. Wiegreffe *et al.*, "Explainable prediction of medical codes from clinical text," *arXiv preprint arXiv:1802.05695*, 2018.

[21] J. Devlin, M.-W. Chang *et al.*, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[22] X. Ge, R. D. Williams *et al.*, "Dkec: Domain knowledge enhanced multi-label classification for electronic health records," *arXiv preprint arXiv:2310.07059*, 2023.

[23] Y. Kong and Y. Fu, "Human action recognition and prediction: A survey," *International Journal of Computer Vision*, vol. 130, no. 5, pp. 1366–1401, 2022.

[24] L. Schrader, A. Vargas Toro *et al.*, "Advanced sensing and human activity recognition in early intervention and rehabilitation of elderly people," *Journal of Population Ageing*, vol. 13, pp. 139–165, 2020.

[25] E. Kańtoch, "Human activity recognition for physical rehabilitation using wearable sensors fusion and artificial neural networks," in *2017 Computing in Cardiology (CinC)*. IEEE, 2017, pp. 1–4.

[26] D. Mukherjee, R. Mondal *et al.*, "Ensemconvnet: a deep learning approach for human activity recognition using smartphone sensors for healthcare applications," *Multimedia Tools and Applications*, vol. 79, pp. 31 663–31 690, 2020.

[27] P. Pareek and A. Thakkar, "A survey on video-based human action recognition: recent updates, datasets, challenges, and applications," *Artificial Intelligence Review*, vol. 54, pp. 2259–2322, 2021.

[28] Y. Zhang, Q. Guo *et al.*, "Human action recognition for dynamic scenes of emergency rescue based on spatial-temporal fusion network," *Electronics*, vol. 12, no. 3, p. 538, 2023.

[29] W. Wang, V. W. Zheng *et al.*, "A survey of zero-shot learning: Settings, methods, and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–37, 2019.

[30] Y. Xian, B. Schiele, and Z. Akata, "Zero-shot learning-the good, the bad and the ugly," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4582–4591.

[31] A. Radford, J. W. Kim *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[32] M. A. Rahman, L. Jia *et al.*, "emsreact: A real-time interactive cognitive assistant for cardiac arrest training in emergency medical services," in *2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT)*. IEEE, 2023, pp. 120–128.

[33] J. Postel, "User datagram protocol," Tech. Rep., 1980.

[34] R. OpenAI, "Gpt-4 technical report," *arXiv*, pp. 2303–08 774, 2023.

[35] T. Wolf, L. Debut *et al.*, "Huggingface's transformers: State-of-the-art natural language processing," *arXiv preprint arXiv:1910.03771*, 2019.

[36] V. Panayotov, G. Chen *et al.*, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[37] G. Gerganov, "ggml," https://github.com/ggerganov/ggml, 2023.

[38] ——, "whisper.cpp," https://github.com/ggerganov/whisper.cpp, 2023.

[39] O. Rohanian, M. Nouriborji *et al.*, "Lightweight transformers for clinical natural language processing," *arXiv preprint arXiv:2302.04725*, 2023.

[40] X. Yang, A. Chen *et al.*, "Gatortron: A large clinical language model to unlock patient information from unstructured electronic health records," *arXiv preprint arXiv:2203.03540*, 2022.

[41] Z. Hu, Y. Dong *et al.*, "Heterogeneous graph transformer," in *Proceedings of the web conference 2020*, 2020, pp. 2704–2710.

[42] M. Sariogoz, "clip.cpp," https://github.com/monatis/clip.cpp, 2023.

[43] M. Nagel, M. Fournarakis *et al.*, "A white paper on neural network quantization," *arXiv preprint arXiv:2106.08295*, 2021.

[44] A. P. H. O. of Emergency Medical Services, "Alabama paramedic protocol scenarios," https://adph.org/ems/assets/ALParamedicProtocolScenario021113.pdf.

[45] E. Online, "Cbt 434 cardiac 5," https://www.emsonline.net/cbtinstructor/assets/cbt434cardiacscenarios1-5.pdf.

[46] P. Szymański and T. Kajdanowicz, "A scikit-based python environment for performing multi-label classification," *arXiv preprint arXiv:1702.01460*, 2017.

[47] A. Paszke, S. Gross *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[48] *pulseaudio(1) - Linux man page*, https://linux.die.net/man/1/pulseaudio, Linux Die.

[49] G. Bradski, "The opencv library." *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, vol. 25, no. 11, pp. 120–123, 2000.