CognitiveEMS: A Cognitive Assistant System for Emergency Medical Services

Sarah Preum*, Sile Shu†, Mustafa Hotaki †, Ronal Williams†, John Stankovic*, Homa Alemzadeh†
*Computer Science, University of Virginia
†Electrical and Computer Engineering, University of Virginia
{preum, ss5de, mkh3cf, rdw, jas9f, ha4d}@virginia.edu

***Abstract*—This paper presents our preliminary results on development of a *C*ognitive assistant system for *E*mergency *M*edical *S*ervices (*CognitiveEMS*) that aims to improve situational awareness and safety of first responders. *CognitiveEMS* integrates a suite of smart wearable sensors, devices, and analytics for real-time collection and analysis of in-situ data from incident scene and delivering dynamic data-driven insights to responders on the most effective response actions to take. We present the overall architecture of *CognitiveEMS* pipeline for processing information collected from the responder, which includes stages for converting speech to text, extracting medical and EMS protocol specific concepts, and modeling and execution of an EMS protocol. The performance of the pipeline is evaluated in both noise-free and noisy incident environments. The experiments are conducted using two types of publicly-available real EMS data: short radio calls and post-incident patient care reports. Three different noise profiles are considered for simulating the noisy environments: cafeteria, people talking, and emergency sirens. Noise was artificially added at 3 intensity levels of low, medium, and high to pre-recorded audio data. The results show that the i) state-of-the-art speech recognition tools such as Google Speech API are quite robust to low and medium noise intensities; ii) in the presence of high noise levels, the overall recall rate in medical concept annotation is reduced; and iii) the effect of noise often propagates to the final decision making stage and results in generating misleading feedback to responders.**

***Index Terms*—Cognitive assistant system, Medical emergency, Speech recognition, Natural language processing, EMS.**

I. Introduction

In an accident scene, emergency medical responders and firefighters initially assess and control the situation and assist victims by providing basic medical care before transferring them to hospital. In such situations, even a few minutes of delay in response time, or tiny errors in the information gathered from the accident scene can largely affect the rescue outcomes. So, first responders need to process substantial amount of information with different levels of importance and confidence and quickly prioritize available information for situation assessment and response. They also need to consider circumstances and history of the incident, communicate with the command center, other responders and the victims and then take actions based on this information and the knowledge of established emergency response protocols. Collecting, gathering, filtering, interpreting and processing such data at the incident scene or control center requires lots of human cognitive efforts.

In this paper, we present *CognitiveEMS*, a **cognitive** assistant for emergency medical services (**EMS**), that will improve first responders' situational awareness and safety in the incident scene. *CognitiveEMS* leverages responder-worn devices and smart sensors to monitor their activities and communications at the incident scene as shown in Figure 1. This data is then aggregated with static data sources, such as, emergency response protocol guidelines to generate real-time insights that can assist the first responders with making effective decisions and taking safe response actions.

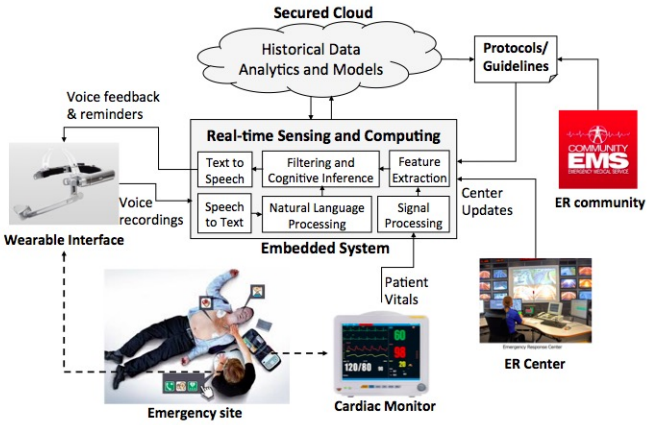


Fig. 1: The system architecture of *CognitiveEMS*

The overall architecture of the system is described in [1] and also shown in Figure 1. In this paper, we present the preliminary implementation and performance evaluation of different modules of the pipeline using real data and in presence of noise, namely, (i) speech to text conversion, (ii) context specific concept extraction, and (iii) modeling and executing protocol guidelines. The main contributions of this paper are as follows.

- Evaluating the performance of the Google Cloud Speech API, the state-of-art tool for speech recognition using EMS radio call data under three different noise profiles common in emergency situations (i.e., noise from conversation of people, cafeteria, and emergency sirens) with three different intensities (i.e., low, medium, and high). The results indicate that among the three different noise types considered, the conversation noise with high intensity degrades the performance of speech recognition

to the highest extent. However, the effects of noise with low and medium intensities are similar for all three types of noise considered.
- Adapting two state-of-the-art medical concept annotation tools, namely, MetaMap [2] and CLAMP [3] to the EMS domain and comparing their performance in extracting EMS related concepts from real EMS incident reports and radio calls.
- Evaluating the performance of the above medical concept annotation tools in processing noisy radio call data. Specifically, how the accuracy of concept annotation stage is affected by the noisy data fed from the speech recognition stage. The evaluation results indicate that these tools achieve high precision in identifying relevant concepts due to very low false positive rate. However, their recall varies from 0.25 to 1 based on the target concept list and the type and intensity of noise considered.
- Developing the first version of a rule engine that utilizes a Prolog based logic programming language to model one of the most commonly used EMS protocol guidelines, called the *Primary Survey Protocol* [4], and to automatically generate insights on the actions to be taken by the first responder.

II. Related Work

A. Cognitive assistant systems

Cognitive assistant systems have been applied to health applications [5]. Specifically, in [5] a Google glass based assistive system is developed to perform context-aware real-time scene interpretation by identifying objects, faces, and activities for people suffering from cognitive decline. In the context of emergency situations, the form factor and interface of cognitive assistant systems and their availability and ability to respond in real-time are of particular importance. So, we consider wearable microphones instead of Google glass.

In the context of emergency medical services, existing systems try to reduce the responders' cognitive load by providing new interfaces for electronic incident scene reporting [6], [7]. ImageTrend [6] provides virtual data entry interfaces for EMS responders. But a significant part of scene reporting is still composed of narratives written in free-form text, describing the observations and actions performed by the first responders at the incident scene. In [7], authors develop a mobile entry solution to aid data collection by dynamic customization of data fields. But these systems still rely on touch screens and messaging interfaces that are hard to manipulate in the midst of an incident. Hence *CognitiveEMS* aims at automatically extracting data from the responders' speech to reduce the cognitive burden of the first responders.

B. EMS and medical decision support

Meneguzzi et al. present *ANTICO* [8], an emergency agent architecture for emergency response managers that integrates plan recognition, current and future user information needs, and workload estimation and offers dynamic data visualization on weather and traffic. They use an XML based language to specify the domain description in terms of potential user activities. They model potential workflow of a user (i.e., an emergency response manager) using Hidden Markov Model where each state represents a potential activity of the user. They evaluate the system using a simulation of a chemical attack. Although both *ANTICO* and *CognitiveEMS* are emergency response assistant, they target different user groups involved in emergency response management and thus tackles different sets of challenges. *ANTICO* focuses on providing information aggregation and visualization support through a graphical user interface to the emergency response manager who communicates with the first responders at the scene. On the other hand, *CognitiveEMS* aims at providing real time decision support through wearable interfaces directly to the first responders present in an emergency scene.

In [9], a Pressure Injury Clinical Decision Support System (PI-CDSS) with an expert knowledge base is developed to help nurses decide on the best wound products in wound healing. PI-CDSS framework applies decision-making theory, knowledge representation and process modeling to develop an expert system for wound assessment and treatment.

Shang et al. develop a framework of clinical decision support system (CDSS) for chronic diseases based on ontology and service-oriented architecture (SOA) [10]. Ontologies are used for knowledge base construction on multiple clinical practice guidelines. Further, a CDSS web service is developed to provide clinical decision support via ontology reasoning based on a knowledge base constructed from clinical practice guidelines of Type 2 diabetes mellitus and hypertension.

There are also a number of mobile applications developed for smart devices to help with the decision making process of first responders. They provide information in an organized manner and provide search options to find critical information based on keywords. *Informed's Emergency & Critical Care Guide* is one such commercial application that is developed for iOS devices [11]. It allows the user to search for medication information; calculate dosage or other necessary measurements for intervention; tabulate scores for pediatric trauma, Glasgow Coma, Apgar, and the NIH stroke scale; and access information about pediatric normal vital signs and protocols, including, resuscitation and airway management. Another mobile EMS application is TJEMS [12]. It contains regional EMS protocol guidelines and suggested interventions for first responders.

C. Noise profiling using speech APIs

There are existing works to evaluate performance of speech recognition systems in noisy environments. For example, in [13] the authors create an approach and a database to evaluate the performance of speech recognition systems in noisy environments. They consider eight different real-world noise profiles, namely, noise from suburban train, crowd of people, car, exhibition hall, restaurant, street, airport, and train station. However, the speech data used in this study consists of male and female American English speakers reading isolated digits

and sequences of up to seven digits rather than natural text containing partial or complete sentences.

There are also existing works on comparison of different off-the-shelf speech recognition APIs. Authors in [14] compare commercial speech recognition systems, Google Cloud Speech API and the Microsoft Speech API, with open-source systems such as CMU Sphinx4 [15]. Audio containing phonetically rich sentences from various sources are collected and tested with these speech recognition tools. Measurements of the word error rate show that the Google Cloud Speech API is superior. Another research [16] analyzes several automatic speech recognizers (ASRs) in terms of their suitability for use in different dialogue systems. They consider PocketSphinx [17], Apple Dictation, Google Cloud Speech API, AT&T Watson, and Otosense-Kaldi. Datasets from 6 domains of dialogue systems differing in genre, type of users, and number of users are tested. They find that speech recognizers perform differently in different domains. Overall, Cloud-based recognizers (i.e., Google and AT&T) outperform the other recognizers for four out of the six datasets. For the other two datasets, local customizable recognizers (i.e., PocketSphinx and Otosense-Kaldi) perform most accurately when used with custom language models.

III. Solution Overview

The overall architecture of *CognitiveEMS* is presented in [1] and also shown in Figure 1. In this paper we present the analytics pipeline of the system as depicted in Figure 2.

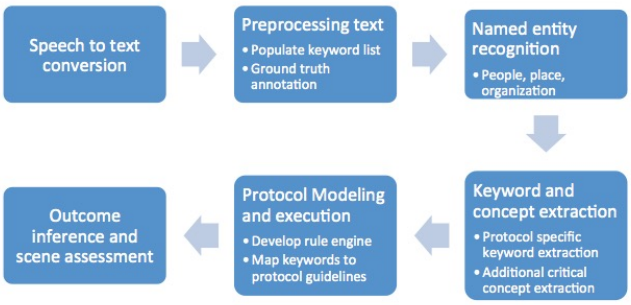


Fig. 2: Basic analytics pipeline of *CognitiveEMS*

A. Analytics Pipeline

As shown in Figure 2, the basic analytics pipeline of our proposed system, *CognitiveEMS*, consists of (i) speech to text conversion, (ii) pre-processing text data originated from different sources (e.g., output from speech recognition stage, incident report, electronic health record data, etc.), (iii) named entity recognition to identify people, place, organization, and temporal expressions, (iv) heterogeneous concept extraction in the context of EMS including, EMS protocol specific concepts and additional critical concepts, (v) protocol modeling and execution, and (vi) inferring the outcome and assessing the scene to track the status of the scene after the interventions are performed by the first responders. Overall, goal of the pipeline is to extract useful information from real time data collected from different sources (both on-scene and outside) and provide feedback to the first responders in order to increase their situational awareness, decision making capacity and safety. Results on comparing different off-the-shelf speech recognition tools and named entity recognition (NER) tools are presented in [1]. There four off-the-shelf state-of-art speech recognition tools namely, Google Cloud Speech API, Microsoft speech API, PocketSphinx, and IBM BlueMix API, are compared in both noisy and noise-free environment. The performance of these tools is measured in terms of word error rate and computation time using transcripts that are not specific to EMS scenes. The evaluation demonstrates that the Google Speech API outperforms the other APIs in terms of both performance metrics for both noisy and noise-free data. We also compare three state-of-art NER tools in extracting named entities (i.e., people, place, organization) and temporal expressions from EMS specific textual data in [1]. The tools are Stanford coreNLP, Apache openNLP, and Illinois tagger. The results indicate that the Stanford coreNLP NER tool outperforms the other tools in terms of both precision and recall for all of the four types of entities considered.

In this paper, we further evaluate the Google Speech API using real EMS radio call data and in presence of varying levels of noise and noise types. We also adapt two medical concept extraction tools to EMS domain and compare their performance on EMS concept extraction on both noisy and noise-free EMS data. Finally, a rule engine is developed to model the primary survey protocol [4]. The protocol modeling is performed using both noisy and noise-free data. Thus we extend the work in [1] in this paper by (i) performing noise profiling under three different noise types and three noise levels, (ii) implementing additional two modules of *CognitiveEMS*, and (iii) evaluating their performances under nine different noise profiles.

B. Speech-to-text conversion

Speech data is collected using the wearable devices worn by the first responders. This data can include incident scene description, patient's status description, radio calls, conversation between responders and others present on the scene, etc. This data is converted to text to extract critical information. As mentioned above, previous research shows that out of available commercial and open-source speech recognition technologies, the Google Cloud Speech API provides the best results [1], [14]. Hence, in this paper, the audio data is converted to speech data using the Google Cloud Speech API.

C. Concept extraction

After speech to text conversion, medical and EMS relevant concepts are extracted from the converted text as well as other text data, e.g., patient's electronic health record (EHR), EMS report, etc. Concepts are collected based on an existing ontology of EMS concepts and the knowledge base of EMS protocols. In this paper, concepts are extracted based on a preliminary ontology and two EMS protocols, primary and secondary surveys [4] using two state-of-the-art medical NLP

tools, namely, MetaMap [2] and CLAMP [3]. This section presents a brief description of these tools.

1) MetaMap: MetaMap is a highly configurable tool for mapping biomedical text to the concepts in the unified medical language system (UMLS) Metathesaurus [2]. MetaMap uses a knowledge-intensive approach based on natural-language processing and computational-linguistic techniques. The input text to MetaMap undergoes lexical, syntactic, and semantic analysis consecutively. The lexical analysis consists of tokenization, sentence identification, and acronym or abbreviation identification. The syntactic analysis includes part-of-speech tagging, lexicon matching from input to SPECIALIST lexicon, and shallow parsing to identify phrases and their lexical heads. Finally, MetaMap performs semantic analysis, including, but not limited to, variant generation of identified phrases and words, candidate identification, mapping and word sense disambiguation. MetaMap classifies concepts into semantic types that can be further utilized to extract information of specific types, e.g., medication or disease name. There are 133 semantic types categorized in 14 semantic groups. An alternate way to filter concepts is using concept unique identifiers (CUI) of known or target concepts. In this paper, we used CUIs to filter concepts.

2) CLAMP: Clinical Language Annotation, Modeling, and Processing (CLAMP) is another customizable natural language processing pipeline for processing clinical text data [3]. It is trained on electronic health record (EHR) data and provides an interface to extract three semantic classes of clinical terms from text, namely, *problem*, *test*, and *treatment*. The *problem* class includes patient history, injury, symptoms, and diagnoses. The *test* class encapsulates concepts indicating diagnostic medical procedure (e.g., 12 lead EKG) and vitals (e.g., blood pressure). The treatment class covers concepts related to treatment, such as, medication name, medical procedures. CLAMP also identifies some temporal concepts, such as, duration and interval. CLAMP formulates the problem of concept extraction as a classification problem, e.g., classifying phrases or words in a text into one of the three classes mentioned above. It offers a variety of classification approaches for concept extraction: (i) machine learning based classifier including support vector machines and conditional random fields, (ii) rule based classifier and (iii) a combination of both (i) and (ii).

D. Protocol Modeling and Execution

1) EMS Protocols: Primary and secondary surveys are two of the most commonly used protocols for initial patient assessment in emergency situations. The primary survey is an initial, rapid assessment of the patient to identify and treat those conditions that present an immediate threat to life [4]. Once the primary survey has been finished, first responders can proceed to secondary survey. Generally, the overall purpose of secondary survey is to examine the problems that do not threaten the patients life immediately but might become severe, even life-threatening, if they are not treated properly. The secondary survey contains two phases: information-gathering phase and examination phase. In the information-gathering phase, rescuers try to determine the nature of a patients issue by asking questions and observing the environments where the patient is found. In the examination phase, physical assessments are performed to determine the patients vital signs and detect injuries or signs of illness. The decision process flowcharts based on the primary and secondary survey protocols are shown in Figure 3 and Figure 4, respectively.

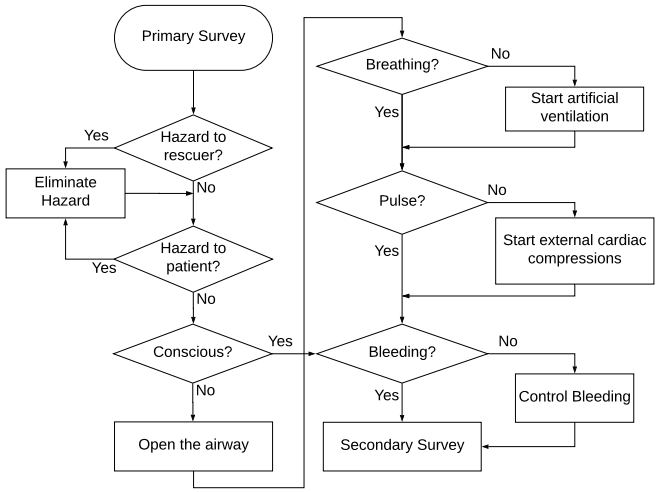


Fig. 3: EMS Protocol 1: Primary Survey [4]

2) Rule Engine: The rule engine is a software system that executes one or more rules in real-time. A rule system enables the policies and operational decisions to be defined, tested and executed. Such rule engines can be implemented by logic programming languages (e.g., Prolog), which are sets of sentences in logical form to express facts and rules about the target problem domain. In this work, we translate the EMS protocol guidelines into rules in logic programming language and then generate a custom rule engine to realize a real-time executable model for cognitive interface and decision making. In this paper, we applied pydatalog [18], a logic programming language library performing Datalog, implemented in Python, to develop the first version of a rule engine that only models the primary survey of the EMS protocol guidelines. The concepts required for the primary survey can be extracted and their negation conditions can be detected by concept annotation tools such as MetaMap. These extracted concepts can then be directly used as the inputs to the executable model of primary survey. The default input considers that there is no hazardous situation and the patient does not have any of the symptoms that are observed according to the primary survey protocol. For every concept extracted by MetaMap, we first identify whether it is required as an input to the primary survey. Then we apply the detected negation condition of the concept as an input to the rule engine. Finally, a treatment suggestion is generated based on the relevant concepts fed to the rule engine.

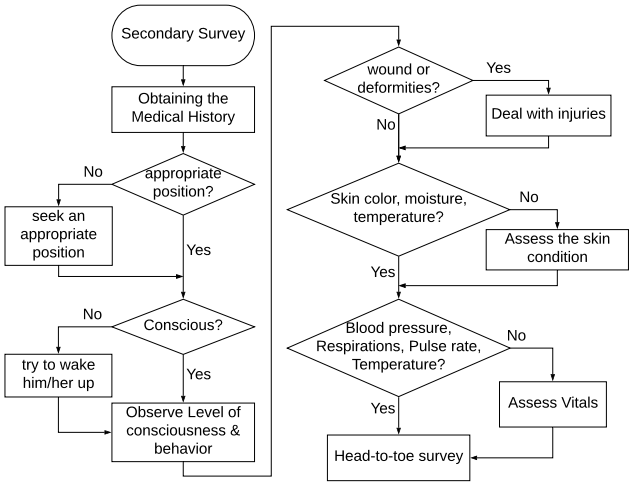


Fig. 4: EMS Protocol 2: Secondary Survey [4]

IV. Evaluation

In this section we evaluate multiple stages of our pipeline, namely, speech to text conversion under noisy environment, context-aware concept extraction, and modeling and execution of the EMS protocols.

A. Speech to text conversion under noise

In this experiment, we assess the performance of the Google Cloud Speech API under different noise profiles that are likely to appear in an emergency situation.

Dataset: The experiment is conducted using 4 short radio calls. A radio call contains audio data where a paramedic reports on the condition of a patient in an emergency incident. The radio calls considered in this experiment cover the following EMS situations: shortness of breath, multiple injuries with bleeding, myocardial infarction, and motor vehicle accident. The audio files are short, containing about 162 words and spanning about 1.5 minutes on average.

Noise profile: For this experiment, three types of real-world noise profiles are chosen: cafeteria, people talking, and emergency sirens. For each noise profile, three noise levels are artificially added: low, medium, and high noise.

Noise is added to the clean audio files in LabVIEW [19]. Wave files for both the clean recording and noise tracks are imported and added sample by sample. In this approach, the sampling rate has to be the same for both tracks. To create a noisy recording in a controlled manner, the noise track is given a pre-gain to make its magnitude roughly equal to the magnitude of the clean recording. Then, the tracks are given a weight and are added to create the output. To keep the clean recording's magnitude constant, the weight for it is chosen to always be 1. The weights for the noise tracks are 0.1, 0.4, and 1 for low, medium, and high noise levels, respectively. These values are not linearly-spaced but do cover a relatively wide range of signal-to-noise ratios. This method allows adding noise in a controlled manner, as it keeps the noise and clean audio waveforms constant and only varies the noise type and the noise level.

Performance metric: The performance is measured using two metrics: word error rate (WER) and accuracy.

- WER is a commonly used metric for determining the performance of a speech recognition system. It works on the word level rather than the phoneme level, counting the percentage of errors in the words in the recognized text against the reference text. WER is calculated using the following formula:

$$WER = \frac{(I + D + S)}{N}$$

  Here, I, D, S, and N indicate the number of inserted words, the number of deleted words, the number of substituted words, and the total number of words in the reference, respectively.
- Accuracy is calculated as:

$$Accuracy = \frac{(N - D - S)}{N}$$

  Accuracy is a less preferred metric than the WER in most situations, since it does not take into account the number of incorrectly inserted words.

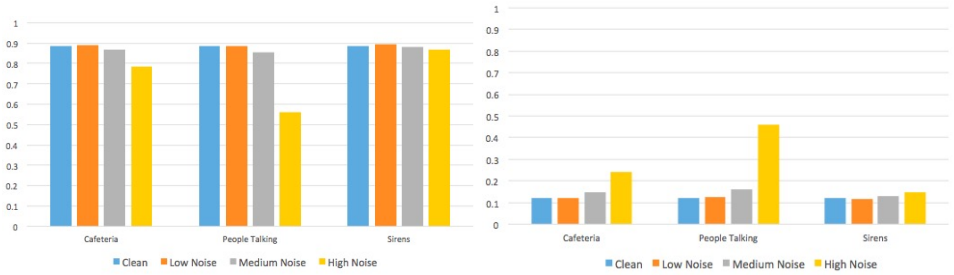
1) Effect of Noise on WER: Figure 6 depicts how WER varies across different noise profiles and noise levels. As expected, WER increases as more noise is added to the data. It can be seen that people talking has the most adverse effect on the recognition results. On average, the increase in WER from medium to high noise is larger than the increase in WER from low to medium noise. This can be partially explained by the fact that the noise levels are not increased linearly. However, the fact that WER increases more sharply for people talking at high noise than the other noise profiles suggests that noise that contains speech affects WER more adversely at louder levels.

2) Effect of Noise on Accuracy: The effect of noise on accuracy is consistent with its effect on WER as shown in Figure 5. Here the average WER and accuracy is shown across all radio calls for different noise profiles. In this case, when high levels of noise are added, the accuracy sharply declines. As expected, the effects are again most adverse for the case of people talking and least adverse for emergency sirens.

3) Effect of adding phrase hint: Speech recognition with the Google Cloud Speech API can be tailored to a specific application by providing it with a list of word hints. In the earlier settings, the API used its default model to convert speech to text. In this experiment, the API is provided with a list of common medical terms. This made a slight improvement of about one percent in both accuracy and WER for one of the datasets. It is likely that with a more comprehensive list of EMS relevant concepts, the performance improvement is even higher.

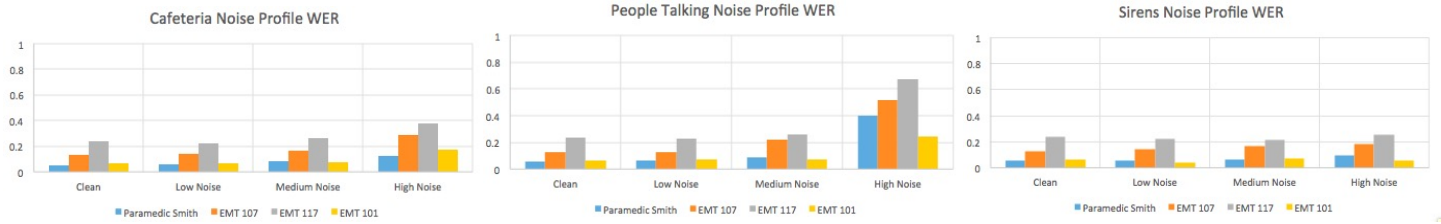
B. EMS concept extraction

One of the fundamental stages of the pipeline is extracting medical and EMS relevant concepts from the text data. The



(a) Average accuracy under different noise profiles (b) Average WER under different noise profiles

Fig. 5: Effect of noise on speech to text conversion: the X axis represents the variation of the noise levels and the noise types and the Y axis represents accuracy (Figure a) and WER (Figure b). On average, when noise level is increased the accuracy decreases and the word error rate (WER) increases. The change in performance is more drastic for the people talking noise followed by the cafeteria noise.



(a) Cafeteria noise profile (b) People talking noise profile (c) Siren noise profile

Fig. 6: For all noise profiles, WER increases with the increase of noise level. The decline in performance is most significant for the people talking noise. The amount of change in WER varies across different radio call data files.

text data can originate from heterogeneous sources, such as patient care reports, patient's history, electronic health records, radio calls, etc. Also, the text data can either appear in raw text format or as output from the speech recognition stage.

In this experiment, we compare two state-of-the-art medical concept extraction tools in extracting context specific terms from EMS related text. The two contexts that are considered are: EMS protocol specific concepts and an ontology of medical concepts that do not appear on the prior list. We also consider the effect of noise on the accuracy of concept extraction. In addition, we consider how the performances of these tools vary across different data sources. The datasets and settings used in this experiment are described below.

Data: We use two different datasets in this experiment, including: (i) text converted by the Google Speech API from the EMS radio call recorded by the first responder, and (ii) post-incident patient care reports written by the responder(s). We use the post-incident reports as a surrogate of the overall incident description and workflow of the first responders, as any other data containing these information aren't available for the real scenarios considered in this paper. These two types of data vary in terms of source, format, and content as described below.

- The radio calls are made by the first responders during service time while transferring the patient to hospital to give a concise summary of the incident. On the other hand, the patient care reports are recorded by the responders post incident and after returning from service.
- The radio call data is comparatively much shorter than the patient care report data. The radio call data contains the summary of the work flow of the responder(s) in a chronological order. It usually contains the symptoms of the patients, relevant vitals for potential diagnosis, and sudden change of status of the patient. The patient care report includes more comprehensive details on the incident, including chief complaint, other complaints, patient's demographic information, history (i.e., past disease, diagnosis, medication, allergy), all recorded vitals, procedures performed on scene and their outcomes, progression of vitals or status over time, etc. Also, it contains context specific abbreviations, e.g., *level of consciousness and awareness of people, place, time* may expressed as "CAO times 3" or "CAO X 3".
- The radio call data is usually collected in free-form audio format and need to be converted to text for further processing. Based on the API used to converting the audio, the generated text may not include any punctuation or sentence identifiers. As this is derived from speech data, the text is often grammatically inaccurate and consists of partial sentences or only phrases. On the other hand, the patient care reports are usually written in semi-structured format or punctuated grammatically accurate text, often consisting of complete sentences. Hence, unlike the patient care reports, the radio call data cannot be easily analyzed by dependency parsing or extracting relationships between different concepts.

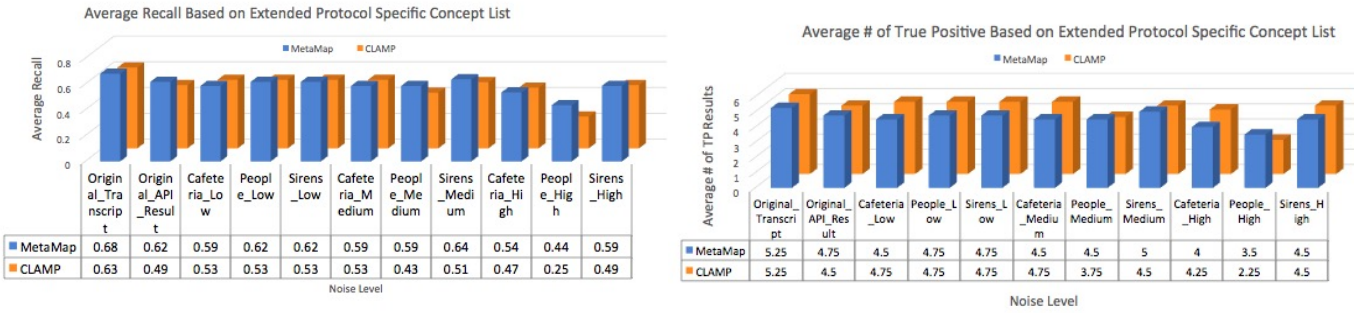


Fig. 7: Comparing the performances of MetaMap and CLAMP under noise in terms of average recall and number of true positives for the extended protocol specific concept list. Overall, MetaMap performs better than CLAMP in case of both noisy and noise-free data. The effects of low and medium levels of noise are similar for all three types of noise. The effects of high level of noise is significant, specially, in case of the noise of people talking.

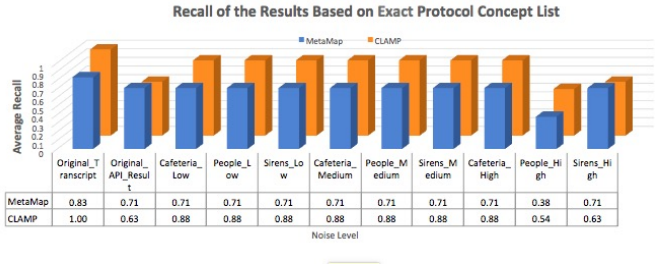


Fig. 8: Comparing the performances of MetaMap and CLAMP in terms of average recall for the exact protocol specific concept list. CLAMP outperforms MetaMap for most of the noise profiles. For both tools the recall drops when the outputs of speech recognition stage are fed to the tools. For noisy input data, the performances of each of the tools are adversely affected only for the high level of the noise of people talking.

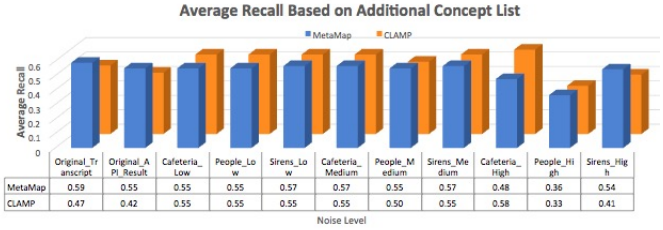


Fig. 9: Comparing the performances of MetaMap and CLAMP in terms of average recall for the additional critical EMS concept list. MetaMap outperforms CLAMP for both noisy and noise-free data.

In this experiment, four real radio calls and five patient care reports from publicly-available sources are used. The radio call data are the same as the ones described in Section IV-A. We use noise free and noisy versions of the radio calls to measure the performances of CLAMP and MetaMap in extracting the relevant concepts from text. Then we compare the performances of CLAMP and MetaMap in extracting concepts from raw, unaltered text.

Ground truth annotation: The text data are manually annotated by two annotators to identify the following three types of concepts:

- **Protocol specific exact concepts**: Each text data file is annotated to identify whether it contains the exact concepts from the two EMS protocols: primary survey and secondary survey (described in Section III-D1). In total, we identified 19 concepts for these protocols: *conscious, breath, pulse, bleed, skin color, skin temperature, skin moisture, blood pressure, temperature, heart disease, circulation problem, stroke, COPD, asthma, diabetes, last visit, medication, wound, position*.
- **Protocol specific extended concepts**: Each text data is annotated to detect whether it contains any of the concepts from an extended list of protocol specific concepts. The original list of concepts is extended in two ways: (i) by adding the ”preferred name” of the most relevant concept unique identifiers (CUIs) for each concept in the original list (extracted manually from the UMLS Metathesaurus API [20]), and (ii) by manually identifying

phrases or words in the dataset of EMS radio calls and patient care reports that are semantically similar to the original protocol specific concepts. For example, using CUIs, we are able to add terms such as *blood*, *bleeding* for the original protocol specific concept *bleed*. We also add medical specific abbreviations, e.g., *BP* for *blood pressure* and *oriented times 3* for *consciousness*. Number of concepts in the extended lists for the radio call and patient care report data are 248 and 233, respectively.

- **Additional critical EMS concepts**: Finally, each text data file is also annotated for the additional EMS concepts that are not covered in the above two lists but may be critical for decision making or for executing other EMS protocols. This set includes the concepts indicating (i) physiological vital signs, (ii) chief complaints, (iii) clinical or emergency procedures, (iv) accident type, (v) pains, (vi) medications, and (vii) symptoms. Empirically, this list is found to be specific to our dataset. Although this list is curated manually in this work, in future we plan to further expand and populate it using word embedding models that are trained on larger domain-specific corpora. Number of additional concepts added in the extended lists for the radio call and patient care report data are 27 and 49, respectively.

Performance Metrics:

1) Comparing performance of MetaMap and CLAMP in noisy data: In this experiment, the text data from the four radio calls are used to measure the performances of MetaMap and CLAMP in medical and EMS concept extraction under noise. Specifically, for each of the 4 radio calls, the input consists of (i) the original transcript of the radio call, (ii) the output from the Google cloud API without any noise (clean data), and (iii) the nine noisy text outputs from Google cloud API (i.e., three noise profiles with three different noise levels). Thus, in this experiment we evaluate the performance of MetaMap and CLAMP in concept extraction on 44 (4 times 11) different text scripts.

The results of this experiment are presented in Figures 8, 7, and 9 for exact, extended protocol specific concepts, and for additional critical concepts, respectively. The number of false positives for both MetaMap and CLAMP are found to be low or zero in these cases, even when the data is very noisy. This is because both MetaMap and CLAMP are trained on domain specific concepts and they classify terms as medical concept with high confidence. Hence, precision is measured to be 1 for almost all cases. So, we consider the recall and number of true positives.

For exact protocol specific concept list (referring to Figure 8), CLAMP is found to perform better than MetaMap in terms of both true positive rate and recall. There is difference in performance from the original transcript to the noise-free output from the Google API. The effect of noise is minimal. Specifically, there is no difference in the effect of low and medium noise. The recall drops only for high level of people talking noise.

concept Type	Tool Name	Avg. number of true positives	Avg. number of false negatives	Avg. Recall
Exact	CLAMP	13.8	12.6	**0.524**
	MetaMap	0.8	0.2	0.043
Extended	CLAMP	16.2	13.6	0.547
	MetaMap	12.6	7	**0.696**
Additional	CLAMP	25.2	4.6	**0.88**
	MetaMap	13.2	10.8	0.527

TABLE I: Comparing the performances of CLAMP and MetaMap on patient care report data: CLAMP on average extracts a larger number of true positives in for all three concept lists. The precision is 1 in all the cases as the tools result in no false positives. So, we consider three other measures. The performances of these tools vary based on the concept list, e.g., while MetaMap performs better for the extended concept list, CLAMP outperforms MetaMap for the other two concept lists.

For extended protocol specific concept list (referring to Figure 7), MetaMap outperforms CLAMP in all cases by at least 7% in terms of recall. Like the previous case, there is no difference in performance from the original transcript to the noise-free output from Google API. In this case, the effect of noise is more prominent than the previous case. This is because, the exact concept list consists of only 19 concepts whereas the extended concept list consists of 248 concepts. For example, for the people talking noise profile, the performance under the low level of noise is about 30% better than the performance under the high level of noise. For cafeteria noise, the performance under the low level of noise is about 8% better than the performance under the high level of noise. The level of siren noise does not affect the performance of concept extraction.

Finally, as shown in Figure 9, for additional critical EMS concept list MetaMap outperforms CLAMP in most of the cases. Interestingly, in this case the recall is higher for the noisy output than the original scripts for CLAMP under low and medium noise. This is because, as the Google API output is not punctuated, so it does not have any sentence structure. We found that although some concepts are not identified in the original punctuated file by CLAMP, they are found in the punctuation-free text generated by the Google API. Similar to the two above-mentioned cases, the effect of people talking noise is higher than the effects of cafeteria and sirens noise.

Overall, the performances of the concept extraction tools are similar for the original transcripts and the noise-free output form the Google API for different concept lists. Among different types of noise considered in this experiment, the people talking noise affects the performance of concept extraction the most. The effects of siren noise is negligible. Among different levels of noise, the high levels of noise affect concept extraction the most. The effects of low and medium levels of noises are similar for different noise profiles. The performances of MetaMap and CLAMP vary based on the concept list and textual data type. This demands further exploration by adding domain adaptive, robust concept extraction models that are

trained on larger heterogeneous EMS datasets to improve the overall concept extraction accuracy.

2) Comparing performance of MetaMap and CLAMP in raw text data: In this experiment, we compare the performance of MetaMap and CLAMP using 5 public patient care reports. The results are presented in Table I. In this experiment, both MetaMap and CLAMP result in zero false positives and thus the precision is 1 in all cases. So, we compare them in terms of recall and the number of true positives and false negatives. CLAMP outperforms MetaMap in terms of recall for extracting concepts from both exact protocol specific concept list and additional critical concept list. In case of extended protocol specific concept list, MetaMap performs better. This is because: (i) for exact protocol specific concept list and additional critical concept list, MetaMap is sometimes not triggered by the key phrases present in the text. This is evident by the lower number of true positives for MetaMap in case of considering these two concept lists; (ii) On the other hand, the extended protocol specific concept list contains much larger number of concepts than the other two lists. These concepts are identified by MetaMap more frequently as MetaMap identifies 133 semantic types of concepts, while CLAMP identifies only three types of concepts. This is demonstrated by the lower number of false negatives for MetaMap in case of considering the extended protocol specific concept list.

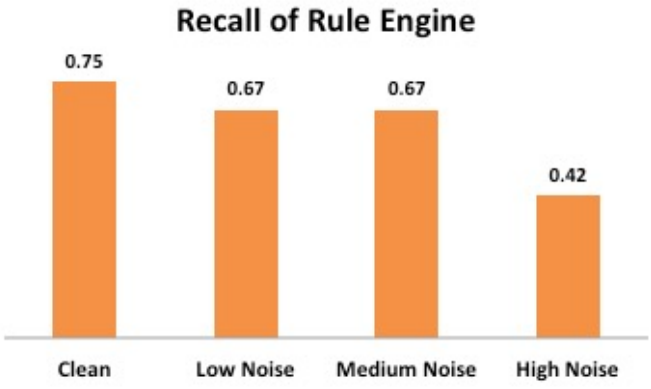


Fig. 10: Recall rate of the primary survey protocol rule engine applied on both noise-free and noisy outputs of the radio call data from the speech recognition stage. On average, higher levels of noise cause more decline in the recall rate and accuracy of detecting correct status and thus can lead to erroneous action suggestions.

C. EMS Protocol Modeling

In this experiment, we apply the Primary survey protocol rule engine (described in Section III-D2) on the noisy output from the concept annotation stage to generate a final action suggestion for the first responders based on the concepts extracted from the radio call data. As a patient's condition changes over time, in this experiment the suggestions are generated based on the final stable condition of the patient as reported in the radio call. The results are presented in Figure 10 in terms of the recall of the correct actions suggested by the rule engine based on the primary survey protocol versus the level of noise present in the data. It is evident that higher levels of noise affect the correctness of the rule engine more adversely. Similar to the concept extraction phase, the recall of this stage is the same for both low and medium noise levels but drops sharply for high level of noise. This is because with the noisy data, the accuracy of MetaMap for concept extraction declines. Specifically, three cases of error are identified empirically. Firstly, concepts required by protocols are failed to be extracted, e.g., the concept bleed is not detected in some noisy files, thus the action suggested by the rule engine is to perform secondary survey instead of control bleeding (referring to Figure 3). Secondly, extra concepts are mistakenly identified by MetaMap. Finally, negation is not detected correctly by MetaMap. These errors get propagated from the concept annotation (NLP) stage to the protocol execution stage, resulting in incorrect identification of patient's condition and generating incorrect or misleading suggestions to the responders. The recall rate of correct outputs are, respectively, 0.75, 0.67, 0.67, and 0.42, for noise free, low-noise, medium-noise, and high-noise data, respectively.

V. Discussion and Future work

This section describes some of the major challenges identified through the experiments conducted in this paper and the potential future work to address those challenges.

Despite being the state-of-the art speech recognition technology, the Google API has a number of limitations as follows. (i) The textual output of the API does not contain any punctuation marks. While humans can infer sense from conversations using pauses and verbal ques and distinguish sentences and flow of meaning from each other without explicitly using punctuation marks, the Google API fails to identify the implicit punctuations. Although, this does not impact calculating WER and accuracy, it makes it harder to perform natural language processing on the returned transcript. Because, the traditional NLP tools require proper punctuation marks to capture sentence structures [21]. (ii) The Google Speech API is also inconsistent in interpreting numbers. For example, it interpreted what was clearly dictated as 52, five-two, year-old male as 5052 year-old male. Errors like this affect both the accuracy and WER. These erroneous speech recognition results can make it hard to derive meaningful data out of the returned transcripts, might lead to error in decision making, and result in safety violations. In future, we would like to explore techniques to resolve these issues and enhance the resilience of different stages of the pipeline to errors. Also, this paper used the asynchronous setting of the Google Speech API for conducting the experiments. As a more realistic approach we plan to use the real-time streaming feature of the API and evaluate its performance in terms of execution time.

The concept lists used in this work are extended using MetaMap and manual observation. To make the process more comprehensive, we plan to develop word embedding models trained on EMS and clinical corpora to expand our EMS ontology and extract more relevant contextual concepts from

text. Also, the results indicate that the performance of existing medical NLP tools vary based on different types of data and contents of the calls and reports. We will extend the functionality of these tools by integrating them with domain specific information extraction algorithms. In addition, we will extend this analysis using larger and more diverse datasets for all stages of the pipeline.

VI. Conclusion

In an emergency medical situation the first responders need to collect, aggregate, filter, and interpret information from different static and real time sources within a short interval. This demands significant amount of human cognitive effort that is better spent on critical decision making and effective response. In this paper, we present *CognitiveEMS* that aims at improving first responders' situational awareness and safety in the incident scene and reduce their cognitive overload. We present preliminary implementation and performance evaluation of three modules in the processing pipeline of *CognitiveEMS* using multiple real datasets: (i) speech recognition under nine different noise profiles, (ii) medical and EMS concept extraction, and (iii) EMS protocol guidelines modeling and execution. The experimental results indicate the robustness of the state-of-the-art speech recognition tool, Google Speech API, under low and medium noise levels. We find that in the presence of high levels of noise the overall recall in medical concept annotation is reduced. Finally, the effect of noise often propagates to the final decision making stage and results in generating misleading feedback to the responders.

VII. Acknowledgement

This work was supported by the award 60NANB17D162 from the U.S. Department of Commerce, National Institute of Standards and Technology (NIST).

References

[1] S. M. Preum, S. Shu, J. Ting, V. Lin, R. Williams, J. A. Stankovic, and H. Alemzadeh, "Towards a cognitive assistant system for emergency response," in *To appear in the poster session of the 9th ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018.

[2] A. R. Aronson and F.-M. Lang, "An overview of metamap: historical perspective and recent advances," *Journal of the American Medical Informatics Association*, vol. 17, no. 3, pp. 229–236, 2010.

[3] E. Soysal, J. Wang, M. Jiang, Y. Wu, S. Pakhomov, H. Liu, and H. Xu, "Clamp–a toolkit for efficiently building customized clinical natural language processing pipelines," *Journal of the American Medical Informatics Association*, 2017.

[4] N. L. Caroline, *Nancy Caroline's emergency care in the streets*. Jones & Bartlett Learning, 2007, vol. 2.

[5] K. Ha, Z. Chen, W. Hu, W. Richter, P. Pillai, and M. Satyanarayanan, "Towards wearable cognitive assistance," in *Proceedings of the 12th annual international conference on Mobile systems, applications, and services*. ACM, 2014, pp. 68–81.

[6] ImageTrend, "Critical Care Solutions, ImageTrend," http://www.imagetrend.com/solutions-ems-critical-care/, 2017, [Online; accessed 10-Jan-2018].

[7] M. Fleshman, I. Argueta, C. Austin, H. Lee, E. Moyer, and G. Gerling, "Facilitating the collection and dissemination of patient care information for emergency medical personnel," in *Systems and Information Engineering Design Symposium (SIEDS), 2016 IEEE*, 2016, pp. 239–244.

[8] F. Meneguzzi, J. Oh, N. Chakraborty, K. Sycara, S. Mehrotra, J. Tittle, and M. Lewis, "A cognitive architecture for emergency response," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*. International Foundation for Autonomous Agents and Multiagent Systems, 2012, pp. 1161–1162.

[9] P. C. B. Khong, L. N. Lee, and A. I. Dawang, "Modeling the construct of an expert evidence-adaptive knowledge base for a pressure injury clinical decision support system," in *Informatics*, vol. 4, no. 3. Multidisciplinary Digital Publishing Institute, 2017, p. 20.

[10] Y. Shang, Y. Wang, L. Gou, C. Wu, T. Zhou, and J. Li, "Development of a service-oriented sharable clinical decision support system based on ontology for chronic disease." *Studies in health technology and informatics*, vol. 245, pp. 1153–1157, 2017.

[11] V. Body, "Access a wealth of critical emergency reference information and helpful tools," https://www.visiblebody.com/anatomy-and-physiology-apps/informeds-emergency-critical-care-guide, 2018, [Online; accessed 10-Jan-2018].

[12] . T. J. E. Council, "Thomas Jefferson EMS Council app," https://itunes.apple.com/us/app/tjems/id594746889?mt=8, 2018, [Online; accessed 1-Feb-2018].

[13] H.-G. Hirsch and D. Pearce, "The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*, 2000.

[14] V. Këpuska and G. Bohouta, "Comparing speech recognition systems (microsoft api, google api and cmu sphinx)," *Int. J. Eng. Res. Appl*, vol. 7, pp. 20–24, 2017.

[15] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, and J. Woelfel, "Sphinx-4: A flexible open source framework for speech recognition," 2004.

[16] F. Morbini, K. Audhkhasi, K. Sagae, R. Artstein, D. Can, P. Georgiou, S. Narayanan, A. Leuski, and D. Traum, "Which asr should i choose for my dialogue system?" in *Proceedings of the SIGDIAL 2013 Conference*, 2013, pp. 394–403.

[17] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky, "Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 1. IEEE, 2006, pp. I–I.

[18] pyDatalog. [Online]. Available: https://pypi.python.org/pypi/pyDatalog.

[19] "LabVIEW User Manual," *National Instruments, Austin, TX*, 1998.

[20] UMLS Metathreasurus API. [Online]. Available: "https://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus/".

[21] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," in *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 2014, pp. 55–60.