Conversational Control Interface to Facilitate Situational Understanding in a City Surveillance Setting

Daniel Harborne[1], **Dave Braines**[2], **Alun Preece**[1], **Rafal Rzepka**[34]
[1] Crime and Security Research Institute, Cardiff University, Cardiff, UK
[2] IBM Emerging Technology, Hursley, UK
[3] Graduate School of Information Science and Technology, Hokkaido University, Japan
[4] RIKEN Center for Advanced Intelligence Project (AIP), Tokyo, Japan
harborned@cardiff.ac.uk

Abstract

In this paper we explore the use of a conversational interface to query a decision support system providing information relating to a city surveillance setting. Specifically, we focus on how the use of a Controlled Natural Language (CNL) can provide a method for processing natural language queries whilst also tracking the context of the conversation with relation to past utterances. Ultimately, we propose our conversational approach leads to a versatile tool for providing decision support with a low enough learning curve such that untrained users can operate it either within a central command location or when operating within the field (at the tactical edge). The key contribution of this paper is an illustration of applied concepts of CNLs as well as furthering the art of conversational context tracking whilst using such a technique.

Keywords: Natural Language Processing (NLP), Conversational Systems, Situational Understanding

1 Introduction

With the continued improvements made to machine-based analytics tools and techniques (such as the rise of Deep Learning), there has been an increase in the extent to which data can be processed autonomously by machines to provide actionable intelligence. We now can harness broader datasets, existing in many modalities and that are collected from many sources. Furthermore, the capability for a system to perform this collection and analysis in real time is increasingly common. For tactical decision makers, such as emergency service incident commanders, this means at the point of formulating a decision, the quantity of information feeds and the variety of the information within those feeds has vastly grown. Having access to the right information at the right time is a key aspect to making the right decision and being overloaded by information can inhibit forming a decision entirely. Due to this change in the information landscape, novel approaches to capitalizing on the vast information available need to be explored. To fulfill this need, we have seen the increased innovation and adoption of novel interaction methods, such as conversational interfaces [Mctear *et al.*, 2016], to access and manipulate information. In this paper we explore an approach to a conversational interface that takes advantage of a Controlled Natural Language(CNL), specifically ITA Controlled English [Mott, 2010]. We first outline the key characteristics of this technology and then move on to discuss the benefits it provides. Finally we include an approach for tracking the context of user queries, furthering the capabilities of the framework. To demonstrate these factors, we use a hypothetical scenario of city-wide surveillance, where data feeds such as traffic cameras, tweets concerning the local area and reports from agents on the ground could be used to build an awareness for the state of the city. This could grant insights with regard to congestion, crimes in progress or emergencies that require response. In this work, we focus on traffic camera data feeds and on the information a surveillance system could plausibly generate when processing such data.

2 Situational Understanding and Decision Support Systems

Situational awareness (SA) [Endsley, 1995] is the ability to build an accurate model of the state of a system, with situational understanding (SU) [Smart *et al.*, 2009] being the ability to reason about it's current and future states. Decision support systems attempt to augment a human user's ability to perform one or both of these tasks. These systems can offer simple aggregation of data and information sources in to a more comprehensible channel and/or can bring together services that can process such data and make inferences from the available information, providing insights, predictions and recommendations to the decision maker.

2.1 Conversational Interfaces

It is not uncommon for a decision maker's primary skill set to be outside the realm of computer science or data analysis. Instead, they take advantage of domain knowledge and related intuition to make decisions within a given scenario, making use of information provided to them on request or preemptively by human or machine analysts. By offering, a conversational interface to a decision support system, decision makers can request information, perform reasoning and action their decisions using natural language rather than through a traditional software interface. Firstly, this can minimize the learning curve for using the system and can speed up the decision

making process. In addition, when combined with speech-to-text and text-to-speech technology, can remove the need conventional input devices. This move away from mice, keyboards and even screens to voice input/output mechanisms not only can increase efficiency by allowing a wide range of actions to be available without using menus but also can often free the user from the requirement for a desk-based system or mobile computational device (such as a laptop or smart device). Instead, the decision maker can form requests and receive information, with minimal change to their operational behavior, including while operating at the tactical edge.

In earlier work the concept of conversational interactions to support casual system users without specific ontology or knowledge engineering capabilities has been explored. In [Pizzocaro *et al.*, 2013], the concept of a conversational interaction to support the tasking of sensing assets within a coalition context, in a constrained environment was constructed. The work brought together earlier task-oriented matching of assets and capabilities to requirements, and placing the power of the system and all the complexities within it, behind a simple conversational interface. In [Preece *et al.*, 2014] the work was extended further into the intelligence analysis domain, and articulated using a simple intelligence gathering and tracking scenario with various human and sensor assets providing information relevant to the task. We also formally defined the underlying conversational model using the CE language, enabling formal definition of different speech acts and the pre-defined ways in which conversational interactions can logically flow. In this work the human users were able to provide their "local knowledge" as new facts via the conversational interaction, as well as ask questions using the same interface. As a result of reasoning and other task-oriented activities the machine agents within the system were able to raise alerts and start conversations with the human users via the same conversational interface as well. Finally in [Braines *et al.*, 2016] we extended the conversational interaction to enable development and extension of the underlying models (also known as ontologies) that underpin the system. Through these capabilities we have been able to show support for question answering interactions as well as the addition of local knowledge and the extension of the underlying models, all through natural language conversational interfaces using the Controlled English language.

3 City Surveillance

In this work, we use a scenario based on city surveillance to explore how a dialogue system can facilitate conversational control of many services and how a decision maker can use natural language to make queries and perform reasoning across the range of available information. We imagine hypothetical tasks the agent may need to perform that relate to the monitoring of traffic volumes and assisting the location and tracking of specific vehicles to assist law enforcement.

3.1 Resources Available

In our system we focus on information provided by traffic cameras. Specifically, traffic camera locations, video and imagery available via Transport for London's Jam Cams Application Programming Interface (API)[1]. In our scenario, we imagine the type of services that could be available to process this data, some of which we have been using in related work [Harborne *et al.*, 2018] and others are proposed as hypothetical services that realistically could exist. Using these services would generate information relating to detecting cars in video and imagery as well as refining the car detections to a specific make and color. For the purpose of this paper, we use pre-generated information, rather than that generated from live services as the integration of such services is outside of the scope of our work.

4 Controlled Natural Language and Controlled English

ITA Controlled English (CE) is an example of a controlled natural language (CNL) which aim to reduce the complexity of natural language (NL) to allow for easier human-machine interaction. The benefits of CNL is that by reducing the grammar to a confined subset, the information retains a machine-readable structure whilst also being naturally readable by humans. This is converse to unstructured data, such as natural conversation, typically difficult for machines to process and highly structured data, such as XML, which are less human readable. In previous work, we have shown that controlled English can help a user control smart devices within their home [Braines *et al.*, 2017]. In that work we outline many of the principles of CE, in this paper we will recap the fundamentals and relate them to the functionality required for this specific piece of work. It is recommended to read the previous work for a thorough explanation of CE.

4.1 Concepts, instances, rules

Controlled English allows the maintaining of a knowledge base via concepts and instances and allows for automatic inferences using rules. All three of these can be created before a support system goes live or can be created as part of the operation of the system. Concepts outline the classes of entities, instances are representations of specific known entities and rules allow for the system to perform reasoning with these items. In Figure 1 and Figure 2 we show the definition of the traffic camera concept along with some parent concepts it inherits from in order to further facilitate inference and reasoning. In Figure 3, we show an instance of a traffic camera. Both the concept and instance definition can be automatically generated from the result of querying the traffic camera API.

[1] https://data.london.gov.uk/dataset/tfl-live-traffic-cameras

```
conceptualise a ~ traffic camera ~ C that
     is a spatial thing and
     is a image source and
     is a video source and
     is a geolocation source and
     has the value LO as ~ longitude ~ and
     has the value LA as ~ latitude ~ and
     has the value U as ~ url ~ and
     has the value I as ~ id ~ and
     has the value C as ~ common name ~.
```

Figure 1: Transport for London traffic camera concept definition.

```
conceptualise a ~ displayable thing ~ DT that
     has the value LAT as ~ latitude ~ and
     has the value LON as ~ longitude ~ and
     ~ can show ~ the location LOC and
     ~ can show ~ the region REG and
     ~ is located in ~ the region REGL.

conceptualise a ~ image source ~ ISo that
     is a displayable thing and
     has the value UR as ~ image url ~.

conceptualise a ~ video source ~ VSo that
     is a displayable thing and
     has the value VUR as ~ video url ~.

conceptualise a ~ geolocation source ~ GSo that
     is a displayable thing.
```

Figure 2: Definition of parent concepts used within the knowledge base

```
there is a traffic camera named 'tfl Camera 02151' that
     has '0.00524' as longitude and
     has '51.5421' as latitude and
     has '/Place/JamCams_00001.02151' as url and
     has 'JamCams_00001.02151' as id and
     has 'Romford Rd / Tennyson Rd' as common name and
     has '00001.02151.jpg' as image url and
     has '00001.02151.mp4' as video url and
     can show the location 'Tennyson Road' and
     can show the location 'Romford Road'.
```

Figure 3: Transport for London traffic camera instance definition.

4.2 CE Hudson and Custom Answerers

When a user submits a query to the interface, it is sent to Hudson - an API that interprets natural language into recognized CE components[2]. This interpretation is returned as a JSON output which the interface can use to provide an appropriate responses to the user. In CE terminology, an application that reacts to Hudson API output is called a "custom answer". This approach has both benefits and costs when compared with other machine learning approaches to NLP. A key characteristic is that the interpretation comes from the CE knowledge base, thus interpretations can't be learned based on sentence structure or patterns (like with a deep learning approach). This can make the space of interpretable input sentences smaller. However, CE does allow for synonyms to be assigned to concepts and the approach of CE concept matching is usually powerful and robust enough to recognize user requests in a closed domain (as shown in previous work [Preece *et al.*, 2017b]). To counter this limitation, a hybrid approach that uses deep learning for interpreting from natural language to CE concepts could be used but exploring this is outside the scope of this work.

In our use case, where we explore a control system within a closed domain, CE's power outweighs this drawback. Unlike a user interacting with a general purpose chatbot, a tactical decision maker often will have a higher requirement for consistent and reliable answers and information. Thus, a robust interface with a slightly higher learning curve is more important than covering all possible utterances to achieve a certain goal. This increased learning curve is likely to be quickly overcome during the decision maker's initial interaction with the system and the knowledge base can be designed in such a way that interaction is still intuitive (also shown in [Preece *et al.*, 2017b]). In addition to this consistency of response, this approaches also allows users to update the knowledge base via the conversational interface and for these updates to immediately be utilized in input processing and output generation. This is discussed further in Section 6.

5 Rules Inferencing

Rules in CE are used to provide inherent inferencing that can take place upon the information within the knowledge base. For example, the rule shown in Figure 5 can take advantage of the properties of the region and location concepts (Figure 4) to allow the system to infer which locations are located within defined regions of the city based on the geo-position properties of the location and the boundary of the region. In addition, the rule shown in Figure 6 allows for the system to infer that if a *displayable thing* (such as a video source) can show a road and that road is in a region, then that camera also shows that region.

[2]In this work we used a publicly available open source implementation of Controlled English, named ce-store which implements a number of generic APIs for simple usage. One set of APIs, known as Hudson, enables natural language text processing in the context of a CNL model, returning an "interpretation" of the specified natural language as matches to concepts, properties, instances and more within the CNL model(s) loaded within the ce-store. ce-store, available online at http://github.com/ce-store/ce-store

This inferencing takes place as the knowledge base is updated and so new information provided to the system can lead to further inferences to be made. Like concept definitions and instances, rules can be added by users during standard usage of the interface. This is discussed further in Section 6

```
conceptualise a ~ region ~ REG that
     has the value XONE as ~ x1 ~ and
     has the value XTWO as ~ x2 ~ and
     has the value YONE as ~ y1 ~ and
     has the value YTWO as ~ y2 ~ and
     is a geolocation source.

conceptualise a ~ location ~ LOCA that
     is a geolocation source and
     ~ is located in ~ the region REG.

conceptualise a ~ road ~ ROAD that
     is a location and
     has the value NAME as ~ road name ~.
```

Figure 4: Controlled English concept definitions for regions, locations and roads. These concepts, via inheritance, creates a specialisation of the *geolocation source* concept defined earlier in figure 2

```
[DisplayableInRegion]
if
     (the location L has the value _X as longitude) and
     (the location L has the value _Y as latitude) and
     (the region R1 has the value _X1 as x1) and
     (the region R1 has the value _X2 as x2) and
     (the region R1 has the value _Y1 as y1) and
     (the region R1 has the value _Y2 as y2) and
     (the value _X >= _X1) and
     (the value _X <= _X2) and
     (the value _Y >= _Y1) and
     (the value _Y <= _Y2)
then
     (the location L is located in the region R1).
```

Figure 5: Example of a CE rule which infers the city regions that locations are found in based on the location's geolocation data and region boundaries.

```
[ShowRegion]
if
     (the displayable thing C can show the location R)
and
     (the location R is located in the region REG)
then
     (the displayable thing C can show the region REG).
```

Figure 6: CE rule that allows the system to infer that if a *displayable thing* instance can show a location and that location is within a region, the instance can also show the region.

6 Tellability

As outlined in previous work [Preece *et al.*, 2017a], a system's tellability describes how easy it is for a user to inject new or updated information in to a system during operation. This is a strength of a CE solution as not only can a user inject new instances or update those instances through the conversational interface but they can define new entirely new concepts. This does require some level of familiarity with the system but requires no coding and the interface can take advantage of the new information immediately. This is in contrast with deep learning techniques, where the creation of a new class, feature or query type will often require retraining of the model backing the interface, this can require the knowledge of a trained engineer and time before the new information is accounted for within the interface.

To illustrate this, Figure 7 and Figure 8 show the interfaces being used to request a view of a region of the city. As outlined in Section 5, this region (named, 'test region') is defined with a geospatial boundary, and rules are used to infer that any instance of a *displayable thing* that *can show* a *location* within that boundary (such as a traffic camera the can show a *road*) *can show* that region. In Figure 8, we see a hypothetical scenario where the user knows that a camera that is marked in the API as viewing a certain road (not in the test region area) also can indirectly view another road (one that is in the boundary of the test region). The user, via the interface, can tell the system that the camera can show the second road, the knowledge base is updated instantly and future queries will take this in to account, including when answering queries correctly requires rule inferences.

7 Actions and Query Types

To provide decision support, a system must allow a decision maker to query the data and information available within the system. Sometimes, the user simply needs to see a selection of the information or data sources for manual inspection (discussed in Section 7.1). In addition, the decision maker may want to ask a question about the state of the world and receive a computed or inferred answer. In this work, we explore three forms of query response: confirmation of the existence of entities matching desired criteria, a count of entities matching desired criteria and listing all entities matching desired criteria (these response types are detailed in Section 7.2). Identifying the required response type to appropriately answer a user's query is achieved by detecting instances of *question phrases*, examples of which are shown in Figure 9.

To process user queries that contain filter criteria (such as car color), the CE knowledge base contains definition of property categories (Figure 10) which represent attributes that instances can be filtered by (these mirror a subset of the properties found on concepts defined in the knowledge base). Instances of these properties are then created reflecting possible values that can be filtered by (Figure 11). The purpose of these property and value definitions is to allow the Hudson API to identify them within an utterance from the user. These properties can be combined with the detection of other criteria such as concepts (e.g. "car"), instances (e.g. "Romford Road") or being interested in a specific relationships ([car] *"is driving on"* [road])—. The detection of these filter criteria, allows the custom answerer to form a query to be sent to the knowledge base, the response of which can then be formated appropriately and displayed to the user. It is worth noting

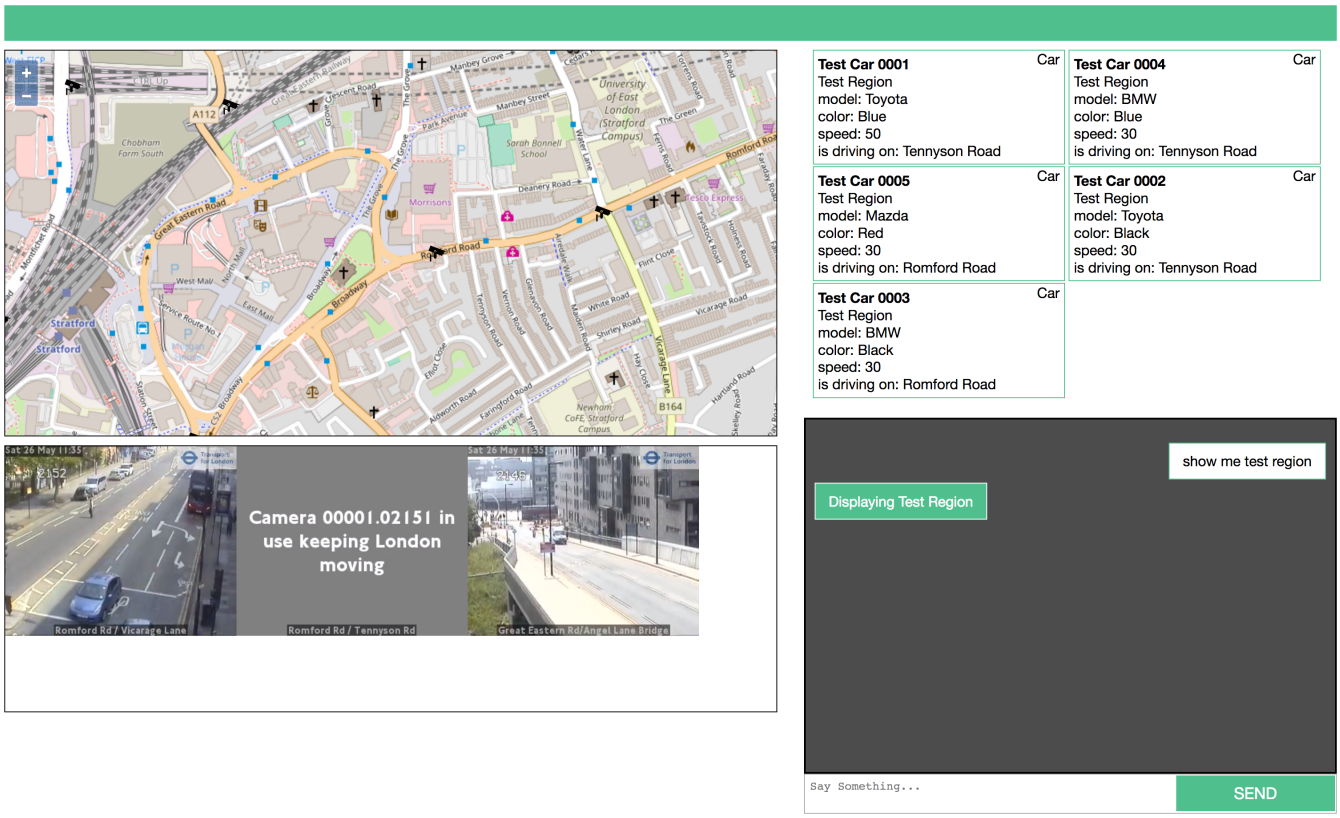


Figure 7: Example of the "show me" type of request with the interface returning *"displayable things"*, In this case a *"video source"* that shows the entity of interest. The map is also focused on the location of interest as it is a *"geolocation source"*. Cameras displaying *"...in use keeping London moving"* have been made temporarily unavailable via the API by TFL.

that the creation of these property and value definitions can be automated based on the concept and instance definitions or from another source and can also be injected by the user during operation.

A further benefit of defining properties and their possible values as concepts is that it makes context tracking possible. This is discussed further in Section 8.

7.1 Actions: "Show me..."

One important benefit to decision support systems is that they can offer one interface for accessing a range of data sources and pieces of information. By offering an efficient and easy to use method for filtering and viewing desired content, a system can ensure a decision maker is not overloaded by being presented with all available data sources simultaneously, instead requesting to see specific resources when they wish to make use of them. As seen in Figure 2, in our system, a concept exists ("*displayable thing*") that indicates that instances of that concept can, in some way, be presented to the user. This parent concept is inherited into further concepts such as *"video source"* which indicate to the interface how the data can be displayed. If the user has screens available a *"video source"* can be shown, if the interface includes a map a *"geolocation source"* can be zoomed in on and become the focus of it. These concepts also allow the user to request to view a list of sources of particular modalities or that feature particular aspects of interest. For example, requesting video sources that can show a specific location.

7.2 Query Types: "...exists?", "Count...", "List..."

Another important feature that can be offered by a decision support system is for a user to be able to ask questions of the information available. This information may have been present in the initialization (such as the location of traffic cameras) of the knowledge base or may have been generated by services processing data sources over time (such as cars within traffic camera video). To do this, filter criteria are identified as outlined at the beginning of this section, a query is formed that will filter to instances that meet the criteria. The result of this query is then returned in three possible formats based on the nature of the questions asked by the user. This response format is based on the detection of question phrases by Hudson API and the custom answerer's reaction to those detected phrases. Examples of these query types are show in Figures 12, 13 and 14.

8 Tracking Context

Tracking context within conversational interfaces can be considered a challenging but important task. The ability for a decision maker to ask a query of the system and then subsequently refine that query creates a much more efficient work

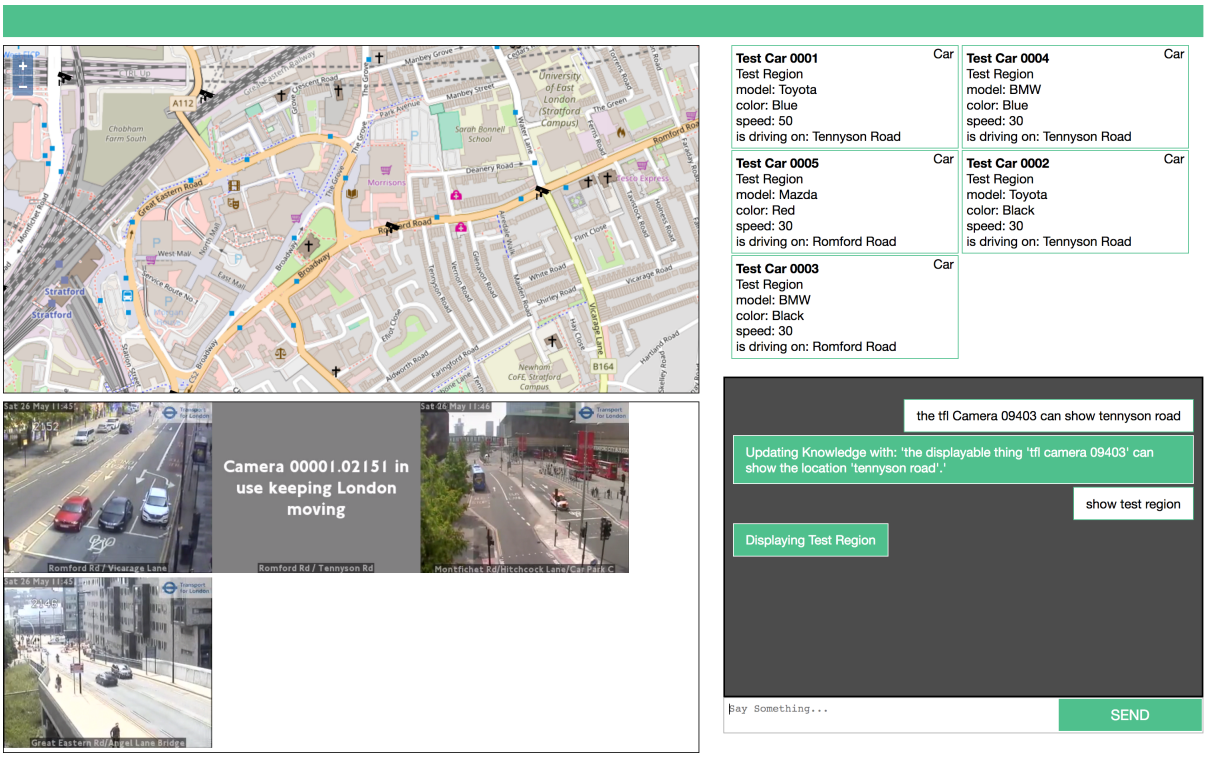


Figure 8: Example of updating the knowledge base via the conversational interface. Once updated with the knowledge a particular camera can indirectly show a road located within test region, the "show test region" request now shows the new camera feed due to inference.

```
there is an action named 'show'.

there is a question type named 'exists'.
there is a question type named 'count'.
there is a question type named 'list'.

there is a question phrase named 'are there any' that
    refers to the question type 'exists'.

there is a question phrase named 'are there' that
    refers to the question type 'exists'.

there is a question phrase named 'how many' that
    refers to the question type 'count'.
```

Figure 9: Controlled English instance definitions for phrases indicating the aim of a user's query —(defined as the *question type*).

```
conceptualise a ~ property category ~ PROPC.

conceptualise a ~ color ~ COL that
     is a property category.

conceptualise a ~ model ~ MODEL that
     is a property category.

conceptualise a ~ direction ~ DIR that
     is a property category.
```

Figure 10: CE concept definition for the concept "property category" and child concepts that facilitate instance filtering.

```
there is a color named 'black'.
there is a color named 'white'.
there is a color named 'blue'.
there is a color named 'red'.
there is a color named 'green'.

there is a model named 'Toyota'.
there is a model named 'BMW'.
there is a model named 'Ford'.
there is a model named 'Range Rover'.
there is a model named 'Renault'.
there is a model named 'Mazda'.

there is a direction named 'North'.
there is a direction named 'South'.
there is a direction named 'East'.
there is a direction named 'West'.
```

Figure 11: CE instance definitions for values the *property categories* can take.

Figure 12: Example of using the interface to check if any instances of a specified criteria exist.

Figure 13: An example of using the interface to retrieve a count of instances meeting a desired criteria.

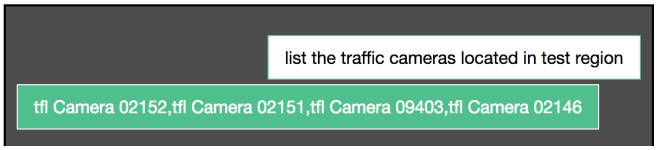


Figure 14: An example of using the interface to retrieve a list of all instances meeting certain criteria.

flow in contrast to forcing the user to re-ask the same question and adding the desired additional parameters. In our approach to adding this functionality to a CE-based interface, the first step is to define query expansion phrases. These are phrases that are used in user utterances that indicate the intent to use the previous query as a base for building the next query (definition for these phrases are shown in Figure 15). The second step is to maintain a store within the custom answerer of the values for properties, the concepts , the instances and the relationships involved in the most recent query. With this information, a query can be formed using the following approach:

- If a query expansion phrase is found, use all parameters from the last query. If a value in the current utterance is from a property category that had a value(s) stipulated in the previous query, then use the ”and” operator for the values.
- If no query expansion phrase is found and the current utterance contains only property values (no concepts, instances or relationships), use the previous query’s concepts, instances and relationships.
- Finally, if neither of the above are true, clear the query store and form a new query.

The query store can also be used with actions. In Figure 17, we see the conversation ends with ”show me” without any stipulation of what to show. Here the custom answerer is able to infer that it should show the *displayable things* from the results of the last query.

```
there is a query expansion phrase named 'of those'.
there is a query expansion phrase named 'of these'.
```

Figure 15: CE definitions for the concept and instances of query expansion phrases.

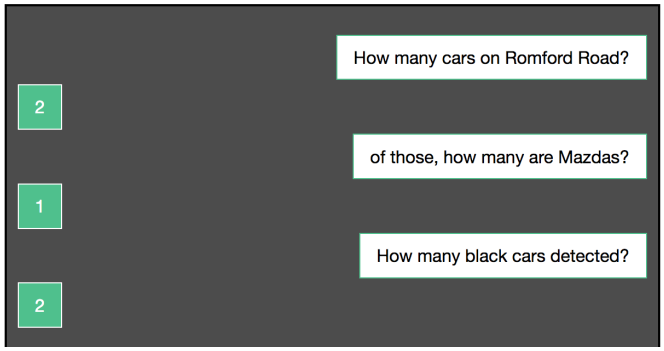
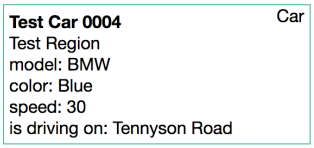


Figure 16: An example of using the interface to make an initial query, then filtering the result rather with additional perimeters and then make a new query.



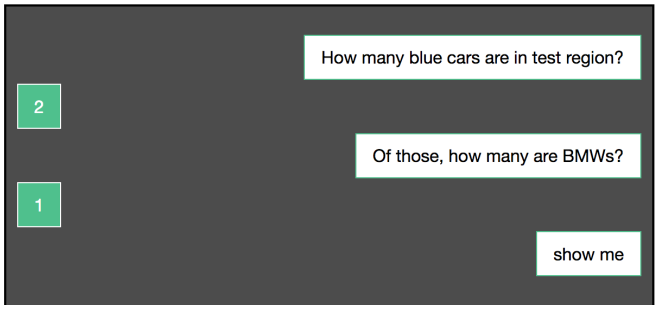


Figure 17: An example of using the interface to make an initial query, then filtering the result rather with additional parameters and finally using ”show me” to display the results from the previous query without having to re-specify and of it’s parameters.

9 Conclusion

In this work we have outlined characteristics and methodology for a conversational interface backed by the controlled natural language, ITA Controlled English. We have shown the benefits such an interface can provide to a decision maker and discussed the implications of using this approach over other techniques. In this work, we have also proposed a method for context tracking using a controlled language to allow for intuitive and efficient query of the system's knowledge base. We have also identified the possibility of future work, that explores the integrating of deep learning techniques performing natural language processing from user utterances to Controlled English. This may lead to an increase in versatility and robustness of interpretation, whilst maintaining the consistency of response, tellability and inferencing provided by CE that is important for tactical decision makers.

Acknowledgments

This research was sponsored by the U.S. Army Research Laboratory and the UK Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the UK Ministry of Defence or the UK Government. The U.S. and UK Governments are authorised to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon. We'd like to thank Cardiff University Global Opportunities Centre for partially funding the collaboration between Crime and Security Research Institute (Cardiff University) and Graduate School of Information Science and Technology (Hokkaido University).

References

[Braines *et al.*, 2016] Dave Braines, Amardeep Bhattal, Alun D. Preece, and Geeth De Mel. Agile development of ontologies through conversation. *Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR VII*, Dec 2016.

[Braines *et al.*, 2017] Dave Braines, Nick O'Leary, Anna Thomas, Dan Harborne, Alun Preece, and Will Webberley. Conversational homes: a uniform natural language approach for collaboration among humans and devices. *International Journal on Advances in Intelligent Systems*, 10:223–237, 2017.

[Endsley, 1995] Mica R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[Harborne *et al.*, 2018] Dan Harborne, Ramya Raghavendra, Chris Willis, Supriyo Chakraborty, Pranita Dewan, Mudhakar Srivatsa, Richard Tomsett, and Alun Preece. Reasoning and learning services for coalition situational understanding. volume 10635, pages 10635 – 10635 – 9, 2018.

[Mctear *et al.*, 2016] Michael Mctear, Zoraida Callejas, and David Griol. Conversational interfaces: Past and present. *The Conversational Interface*, page 51–72, 2016.

[Mott, 2010] D Mott. Summary of ITA Controlled English. *NIS-ITA science library*, 2010.

[Pizzocaro *et al.*, 2013] Diego Pizzocaro, Christos Parizas, Alun Preece, Dave Braines, David Mott, and Jonathan Z. Bakdash. Ce-sam: a conversational interface for isr mission support. *Next-Generation Analyst*, 2013.

[Preece *et al.*, 2014] Alun Preece, Dave Braines, Diego Pizzocaro, and Christos Parizas. Human-machine conversations to support multi-agency missions. *ACM SIGMOBILE Mobile Computing and Communications Review*, 18(1):75–84, Dec 2014.

[Preece *et al.*, 2017a] Alun Preece, Federico Cerutti, Dave Braines, Supriyo Chakraborty, and Mani Srivastava. Cognitive computing for coalition situational understanding. In *First International Workshop on Distributed Analytics InfraStructure and Algorithms for Multi-Organization Federations*, 2017.

[Preece *et al.*, 2017b] Alun Preece, William Webberley, Dave Braines, Erin G. Zaroukian, and Jonathan Z. Bakdash. Sherlock: Experimental evaluation of a conversational agent for mobile information tasks. *IEEE Transactions on Human-Machine Systems*, 47(6):1017–1028, 2017.

[Smart *et al.*, 2009] Paul R Smart, Trung Dong Huynh, David Mott, Katia Sycara, Dave Braines, Michael Strub, Winston Sieck, and Nigel R Shadbolt. Towards an understanding of shared understanding in military coalition contexts. Event Dates: 23rd - 24th, September 2009.