ACS Poster Collection (2018) 1–15 Published 8/2018; Paper 4

Case Retrieval Using Goal Similarity for Knowledge Investigations

Vahid B. Eyorokon EYOROKON.3@WRIGHT.EDU
Venkatsampath R. Gogineni GOGINENI.14@WRIGHT.EDU
Pratyusha Yalamanchili YALAMANCHILI.23@WRIGHT.EDU
Michael T. Cox MICHAEL.COX@WRIGHT.EDU
Computing Science and Engineering, Wright State University, Dayton, OH 45435 USA

Abstract

Goals can be described as the user's desired state of the world and are satisfied when the world has been altered in such a way that the present state matches the desired state. Similarly, a knowledge goal is the user's desired state of knowledge and is satisfied when the missing information that the user seeks to understand has been learned. This process of acquiring the missing information that satisfies a knowledge goal is a knowledge investigation. For complex domains, traditional question answering systems are less effective as they often rely on simple database queries for isolated questions. Without considering conceptual, contextual, and task related information, it becomes difficult to assist the user as their knowledge investigations increase in complexity often spanning multiple questions like in a dialogue. We propose a retrieval method for case-based reasoning systems and evaluate its ability to retrieve not just isolated questions, but entire dialogues by using conceptual, contextual, and task information.

1. Introduction

Question and answer systems like StackOverflow or Quora are powerful tools when it comes to assisting users in knowledge tasks. For simpler domains, these systems easily satisfy the user's requirements when they need answers to an isolated and individual query. Since knowledge goals can sometimes be expressed in a question's utterance, by retrieving the answer to the question, these systems are effective with simple knowledge tasks. However, in more complex domains where *knowledge investigations* consist of multiple knowledge goals in a series, as in dialogues, these basic systems are unable to consider past or other relevant information (Eyorokon et. al, 2016). Without an effective method for retrieving entire dialogues, the system's ability to assist the user is bounded, which can lead to confusion and prolonged knowledge investigations. We propose a retrieval method for case-based systems that can identify not just best matching questions, but entire dialogues useful for domains with complex knowledge investigations.

A *case-based reasoning (CBR)* system (Aamodt & Plaza, 1994) stores problem solution pairs in a case-base where each individual problem solution pair is called a case. These systems are effective where problems have similar structures, thereby allowing CBR systems to leverage their experience. As described by (De Mantaras et. al, 2005) these systems perform four basic functions: retention, retrieval, revision and reuse. Retention is the ability for the system to save cases from new

interactions with users. Retrieval is the process of finding the best matching case from the case-base where the problem is closely related to the current user's problem. Revision is the process by which the system modifies the retrieved case's solution. This revised solution is then reused by the current user to solve their current problem.

We outline knowledge goals and their three main features followed by a case representation of dialogues as goal trajectories within CBR systems. Next, we cover four separate approaches for measuring the similarity between text and how each can be integrated in our system's case retrieval algorithm. Then, we evaluate the performance of our retrieval algorithm when using each similarity method. We review their performance in retrieving dialogues in two complex domains: military and concierge. Lastly, we survey related research and then conclude.

2. Knowledge Goals and Cases

A *knowledge goal* is not the same as the utterance of a question. A knowledge goal is the needed information or knowledge that would satisfy the user's desired state of knowledge. By acquiring this information, the user's state of knowledge transitions to a new state where they have learned the previously missing information and the user's knowledge goal is said to have been satisfied. Knowledge goals often can be expressed in the form of the utterance of the question, where the utterance of the question is the most superficial part of the entire knowledge goal.

Our system is a conversational CBR system (Aha et. al, 1999) and shares fundamental problems found in natural language and text processing (Bengfort & Cox, 2015 ; Eyorokon et. al, 2017). Users interact with our system by posing a series of questions each of which represents the utterance of a knowledge-goal. By posing these questions, or knowledge goals, in a series, users create dialogues (Gu et. al, 2006) that preserve the order in which individual knowledge goals were asked.

However, before our system can perform any CBR related functions (retrieval, revision, reuse, retention) using these natural language utterances, our system must extract useful information from them. First, by understanding concepts and ideas that appear in the user's knowledge goals. Next, by determining the appropriate context in which knowledge goals are asked. Finally, by determining the underlying reason or task that is the user's motive for asking the knowledge goal which is sometimes expressed as the preface to beginning a dialogue. Thus, these components of: concept, context and task; represent the three components of a knowledge goal (Bengfort & Cox 2015).

2.1 Decomposing Knowledge Goals

Knowledge goals appear in various domains and can be decomposed into three simpler components of concept, context and task. Fig. 1 shows this decomposition. Here we considered two domains which often have complex knowledge investigations: a military domain and a concierge domain. Within the concierge domain, our CBR system took the role of the concierge at a hotel desk to answer questions from guests. In the military domain, our CBR system assisted users when developing strategies within dynamic environments. These strategies involved patterns of questions and answers, much like dialogues in the concierge domain.

Yet before we proceed, we must consider the following question: "Why should we bother ourselves with decomposing knowledge goals in the first place?". The reason for decomposing knowl-

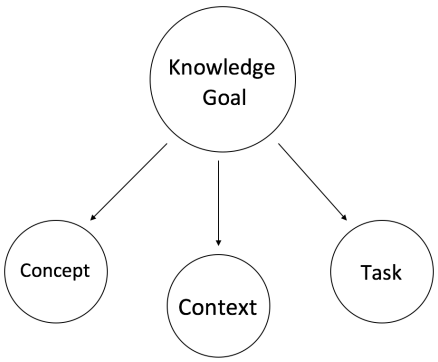

Figure 1. The decomposition of a knowledge goal into concept, context and task.

edge goals can be understood when realizing that the same utterance of a knowledge goal can be made by different people, in different contexts for different reasons and therefore they can have very different answers. Consider this seemingly trivial utterance of a knowledge goal from the concierge domain: "Where can we find something to eat?". When asked by honeymooners at noon, an appropriate response might suggest something romantic like a sit-down restaurant or even a picnic on the beach. However, a completely identical utterance can also be expressed by two bachelors at 2:00 a.m. For the concierge to suggest a picnic at the beach might lead to confusion which would delay the bachelors by prolonging their knowledge investigation. For this reason, knowledge goals should be decomposed into concept, context and task.

2.1.1 Concept

The ideas that appear within the utterance of a knowledge goal represent the *concept*. When considering our example: "Where can we find something to eat?", the word "eat" is part of the verb phrase and would be parsed by our CBR system which uses the Stanford Parser (Schuster & Manning, 2016). This term is a concept that gives our CBR system a clue about the ideas which should appear in the answer as well as when identifying related knowledge investigations. Terms related to eating and food should be given a higher priority.

2.1.2 Context

As part of our CBR system's interactive process, users are required to create profile accounts to better understand the context of a question (Hwang et. al, 2012). These accounts include information regarding their: gender, marital status, sex, location, biography, etc. Attributes that describe characteristics and traits of the user are part of the *context*. Users can create groups based on interests and join them. They can also be assigned to teams based on their skill-set and experience. Our system can then leverage this contextual information to compare user profiles and measure the similarity of individual users themselves

2.1.3 Task

The *task* is the need or motivation for initiating a knowledge investigation and often appears as the preface to a dialogue. In the concierge domain, guests at a hotel will approach the concierge and ask a series of questions. Usually, before the first question is even asked, guests will give a preface that outlines the reason for the questions they are about to ask. If instead guests approached the concierge and promptly began firing away a series of questions, this would be perceived as robotic and socially awkward. For this reason, generally dialogues are preceded by a preface, or task.

In the concierge domain, the guest would first greet the concierge at the desk. Then, the guest would preface their dialogue with their task and state something like: "Our plane was delayed for 6 hours and we finally arrived." This would then be followed by the first question "Where can we find something to eat?" With this information, the concierge would realize that time might be of a higher priority and therefore fast food might be a better response than suggesting a sit-down restaurant that requires a reservation. From here, the dialogue will naturally continue as a series of questions. Yet, if we hadn't considered task, how else would we have known simply from the individual question that time was important for the user?

Often, work with question/answering systems do not consider task. Yet the task we've described clearly provides crucial and relevant information. As the dialogue progresses, clarifying details may be introduced as new tasks and knowledge investigations become increasingly complex. Representing dialogues is crucial for the success of a case-based reasoning system. We outline a case representation for dialogues using a goal trajectory.

2.2 Case Representation: Dialogues and Goal Trajectories

Our system uses a dialogue styled interface where the user can ask a series of questions. An interactive dialogue interface is more natural than search engines or textbooks and produces a data structure that is a chain of knowledge goals which we call a goal trajectory (Eyorokon et. al, 2016). The structure of the chain preserves the order in which knowledge goals have been asked, thus goal trajectories have a beginning and an end and we can consider the evolution of ideas by moving in that direction. When the dialogue begins, the user is asked to enter in the preface, or the dialogue's task. Information about the context can be captured by the profile of the user who is engaged in the dialogue. As knowledge goals are posed to our system, each goal's utterance contributes additional conceptual information obtained through parsing.

During this process, the search plan can change as the user discovers new information and forms new questions; indeed, the questions themselves can change. Like attainment goals (i.e., goals to achieve world states), that are subject to transformation (Cox et. al, 2017), we claim that a knowledge goal is also subject to change. Therefore, as interactive reasoning changes a knowledge goal, the path that leads to the final information can be represented by a goal trajectory. This goal trajectory is schematically shown in the case representation illustrated in Fig. 2. Here the dialogue is *D [1...n]*, a sequence of knowledge goals.

During the dialogue, the system tracks goal changes by recording when new questions are posed. The dialogue is completed by a final knowledge-goal, presumably the target of the investigation. A

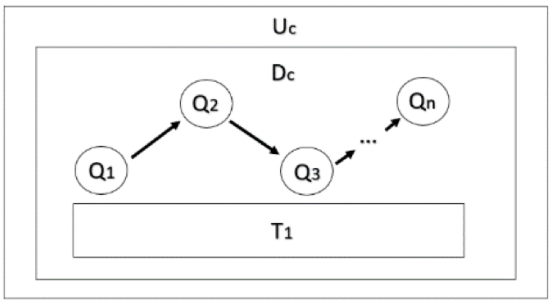

Figure 2. The case representation of a knowledge investigation *I*.

knowledge investigation *I*, is represented with an initial goal g, user *u*, start time t_1, [1] end time t_n, length *n* and a Boolean (= $\perp$ if successful). Here, *name* is a string from the alphabet Σ.

$$D[1...n] = Q[1] \mid D[2...n] = \langle Q[1], Q[2], \ldots Q[n] \rangle \tag{1}$$

$$I = (D_i, u, t_1, t_n, n, succ) \in D \times User \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \times \{\perp, \top\} \tag{2}$$

$$\begin{aligned} User = \{\{(name, age, gender, mstatus)\} \\ \in \Sigma^* \times \mathbb{N} \times \{male, female\} \times \{single, mar, divr, widow\}\} \end{aligned} \tag{3}$$

3. Similarity Measures

Our system can use four different ways of measuring similarity between text. Such similarity measures are also used in case trajectory retrieval as described in (Eyorokon et. al, 2017). The first of these is Term Frequency Inverse Document Frequency which is a statistical, bag-of-words approach (Harris, 1954). The second is Word2Vec which is a neural network that generates vectors for a word by considering its surrounding words. These vector representations can be compared to measure the similarity between words. We will discuss how word vectors can be used to measure similarity between sentences. The third measure uses a semantic net, along with corpus statistics and is an algorithm based on the work done by (Li et. al, 2006) which we refer to as NetSim. The fourth is called SkipThoughts and uses a neural network similar to Word2Vec. We will discuss each similarity measure, their advantages, and their drawbacks.

1. Time is a positive integer representing the number of seconds elapsed since the UNIX epoch (i.e., January 1, 1970)

3.1 Term Frequency Inverse Document Frequency Similarity

Term Frequency Inverse Document Frequency (*TFIDF*) requires a rich corpus of text for the bag-of-words approach to be effective. For this, our system relies on parsing the noun and verb phrases from each utterance and querying the phrases to Wikipedia. This query usually returns a related page from which our system extracts the first three sentences. These sentences are added to the original text of the question's utterance to build a question document. If no Wikipedia pages are returned than this term/phrase is ignored. This process is done for each utterance in the database and TFIDF is performed using the resulting question documents instead of the utterances alone. This allows us to gain a better under-standing of conceptual terms in the question (Huang et. al, 2009). However, the process of querying Wikipedia introduces the issue of disambiguation. Sometimes, Wikipedia will return a list of possible results that are related to the queried phrase instead of a single page. To disambiguate these results would require a sophisticated algorithm or constant human supervision. Given the challenges of disambiguation and since our research focuses on retrieval, we default to the first page in the list of possible pages.

3.2 Word2Vec Similarity

Word2Vec (Mikolov et. al, 2013) is a neural network trained on a large corpus of textual data collected by Google. This approach creates a vector representation for words that considers surrounding words. When an utterance is provided, each word in the string is converted into a vector (Salton et. al, 1975). After gathering the vector representations for each word in the utterance, we averaged these vectors into a single vector for the entire utterance. That is, given a vector for each individual word in a sentence we compute the sum of all vectors divided by the number of words in the sentence to form a sentence vector (Singhal, 2001). The main insight in this method is to obtain the individual word embedding vectors in a question/sentence and form a sentence embedding by averaging vectors. By using the cosine similarity (Kryszkiewicz, 2014) between two questions vectors we calculate the similarity. We obtain each word embedding from the Word2Vec skip-gram model which was pre-trained on Google News vectors containing a corpus of 3 billion words. Word2Vec's skip-gram algorithm predicts whether a word belongs to the surrounding window of words, from a three-layer neural network with one hidden layer while both input and output layers being the unique bag of words thereby forming a word embedding.

3.3 NetSim Similarity

NetSim uses a semantic net along with corpus statistics to measure the similarity between two sentences. The semantic net factors into account two measures: semantic and syntactic similarity. Semantic similarity looks at the synonyms words have in common, the distance from one word to an-other in the semantic net and the depth of a word in the semantic net. Since depth of a word relates to the specialization of a word, distance alone cannot be used. Fig. 3 shows a simple semantic net.

To understand why depth is important, consider the following example. The word "human" may appear closer to the word "boy" than the word "babysitter" but a knowledge goal about humans may be less relevant than one about "babysitters". Since words become more specialized as

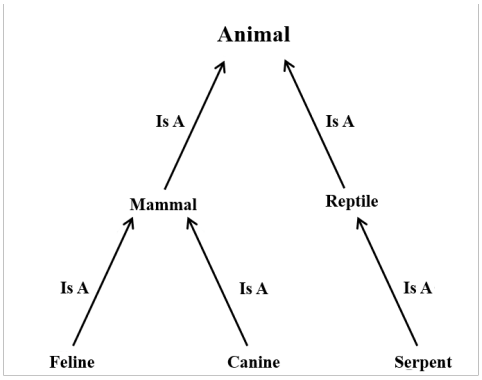

Figure 3. A simple semantic net representing a basic taxonomy of organism types.

we go down the semantic net, depth is factored into the similarity measure. Syntactic similarity for NetSim considers the position of words in one utterance and the distance from related words in another utterance. Additionally, inverse document frequency (IDF) is used to establish an information content of each word. This is a statistical measure where words that are common in the corpus have a low information content and thus a lower IDF score, while less occurring words have a higher IDF score. Finally, the semantic and syntactic scores are multiplied by weights and added together yielding a scalar value for similarity. It should be noted that the use of a semantic net has drawbacks since they only capture is-a relationships and while similarity using NetSim is powerful, it is also bounded as we will see from the evaluation.

3.4 Skip-Thoughts Similarity

Skip-thought is a neural network model that is trained to reconstruct the surrounding sentences that share syntactic and semantic properties (Kiros et. al, 2015). This model is based on an encoder-decoder architecture where encoder maps natural language sentences into fixed length representations and given a vector representation of a sentence, decoder can predict the preceding and subsequent sentences. When we feed the sentence into the encoder model, the sentences that share semantic and syntactic properties are mapped into a similar skip-thought vector. The decoder is another recurrent neural network, that, when given a vector representation of a sentence, predicts the preceding and subsequent sentences.

4. Retrieval

Retrieval is a key function for a case-based reasoning system. These systems store past cases and their solutions in a data structure called a case. By identifying related problems, the case-based reasoning system can retrieve the best matching case so that the solution can then be adapted and used as a solution to the current problem. For our conversational case-based reasoning system, cases are entire goal trajectories. Retrieving the best matching goal trajectory requires an algorithm

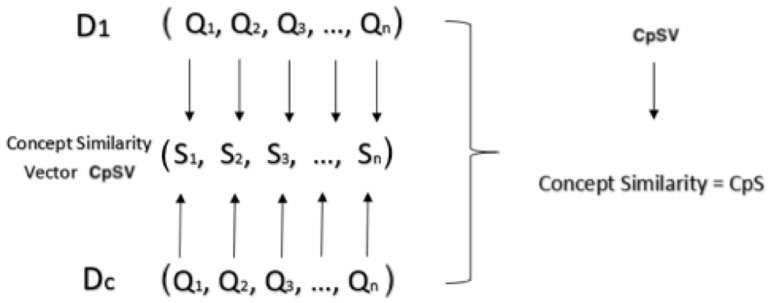

Figure 4. Process for measuring concept similarity.

that considers the contextual, conceptual and task similarities of each goal trajectory's comprising knowledge goals. We begin by determining the conceptual similarity.

4.1 Concept Similarity

Our system performs retrieval by considering the similarity between questions in the user's current dialogue D_c against questions that appear in dialogues from the case-base $D_1 \breve{} D_n$ as shown in Fig. 4. By calculating pair wise similarity between the utterance of questions, we iteratively consider each index in D_c against the corresponding question at the same index in each case $D_1 \breve{} D_n$.

This process produces a vector of similarity scores (CpSV) for each pair of dialogues, whose length is equal to the shorter length of either Dc or a candidate dialogue from the case-base D1 – Dn. From this vector of similarity scores (CpSV) the median value is yielded as the concept similarity which we refer to as CpS. The next step is to capture contextual similarity.

4.2 Context Similarity

Since every dialogue in the case-base has a user, our system can compare the user profiles from the current user x against that from a dialogue in the case-base denoted as y. User profiles are a rich source of attributes that include their age, marital status, gender etc. Specific traits we used were arbitrary and rather we meant to highlight our retrieval algorithm's flexibility when considering contextual information relevant to the problem domain but the aforementioned profile attributes are the only ones currently used. For scaling features like age, a threshold is set. In our system, age had a threshold of 4 years and any difference between the two user profiles under the threshold was considered a match. For binary and enumerable features, matching was straightforward and trivial. Given our representation, each user profile can also be thought of as a set of attributes. Attributes were considered on a matching basis where pairs of attributes from the current user and a case user were passed to a matching function. This function returned a 1 or 0 if the attributes were matched. The sum of total matches is divided by the total number of attributes referred to as n to yield a final contextual score we refer to as CxS. This context scoring function is shown in Eq. 4.

$$C \times S = \frac{\sum_1^n f(x,y)}{n} \tag{4}$$

4.3 Task Similarity

When a user begins a dialogue, our system asks them to provide task information in the form of a single sentence preface. By using one of the four similarity measures for text, we can extract a score for task similarity between the current trajectory D1 and a case trajectory D_c which we refer to as TkS. Finally, we calculate the overall goal trajectory similarity.

4.4 Goal Trajectory Similarity

After each knowledge goal is asked by the user, our system calculates the conceptual, contextual and task similarity to yield a vector (CpS, CxS, TkS) which represents the similarity between the current user's knowledge investigation and a case trajectory. Having a vector representation for goal similarity allows our system to map the user's goal in a 3-dimensional goal space providing visual feedback for the user.

Different domains may have their own need to control the importance of concept, context and task so tunable weights are established for each feature that are then multiplied by their respective score. After the three scores are then weighted, the results are added together to yield a final similarity score between the user's current goal trajectory and a case goal trajectory. These scores are sorted, and goal trajectories are returned in a retrieval set which orders trajectories by most similar to least. Fig. 5 outlines the retrieval algorithm.

5. Evaluation

Evaluation was performed using dialogues from both the concierge and the military domains. The objective was to determine which similarity measure performed the best when used as part of Ronin's retrieval algorithm. As previously mentioned, in the concierge domain, our system took the role of a hotel concierge who answered questions for hotel guests. In the military domain, our system assisted analysts with answering intelligence questions. We iteratively took each dialogue and paraphrased its questions to semantically equivalent questions where utterances were not exact text matches of the original. Retrieval was performed after each question was submitted and the position of the original dialogue was recorded in the retrieval set. The closer the original dialogue was to the first position in the retrieval set, the better.

Here we evaluated the effectiveness of four separate measures: Word2Vec, NetSim TFIDF and SkipThoughts when used in our goal trajectory retrieval algorithm. Each dialogue was at least 5 questions long and paraphrased questions were entered sequentially up to the first 5 questions. After each question, retrieval was performed. We refer to the length of the dialogue at the time retrieval was performed as the probe size and appears on the X axis. After retrieval sets were generated for each iteration of our probe size, we averaged the position of the desired dialogue across all retrieval

1.
```
function CONCEPT-SIMILARITY(old-case, current-case)
    CpSV ← []
    for i ← 1 to length(current-case) do
        Si ← similarity(old-case[i], current-case[i])
        CpSV.append(Si)
    return(median(CpSV))
function CONTEXT-SIMILARITY(old-case, current-case)
    CxS ← 0
    matches ← 0
    possible ← 1
    if (|old-case.user.age − current-case.user.age| < 4) then
        inc(matches)
    Uc ← current-case.user
    Ui ← old-case.user
    intersection ← (Uc.attributes ∩ Ui.attributes)
    union ← (Uc.attributes ∪ Ui.attributes)
    CxS ← (length(intersection) + matches)/(length(union) + possible)
    return(CxS)
function CASE-BASE-SIMILARITY(case-base, current-case)
    sequence ← []
    for case in case-base do
        CpS ← CONCEPT-SIMILARITY(case, current-case)
        CxS ← context − similarity(case,current-case)
        similarity ← average(CpS, CxS)
        sequence.append((similarity,case.id))
    return(sorted(sequence, key=lambda x: x[0]))
CASE-BASE-SIMILARITY(case-base, current-case)
```

Figure 5. Formalization of dialogue retrieval. Refer to Fig. 4 and Eq. 4 for concept and context similarity.

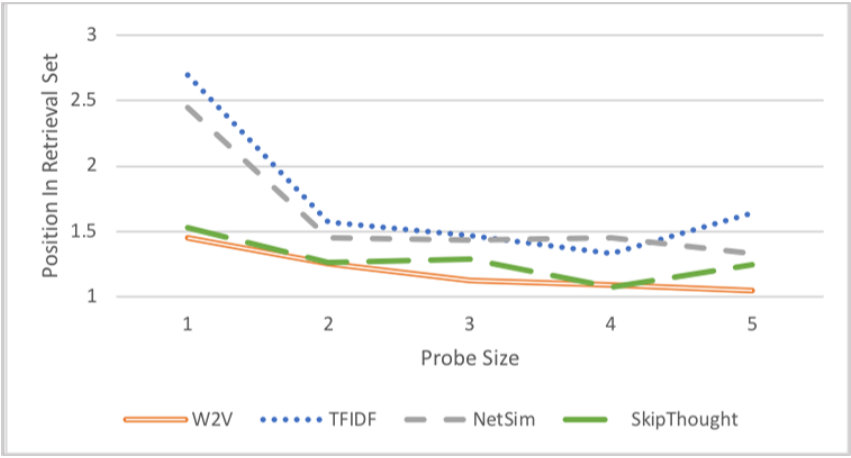

Figure 6. Retrieval evaluation in the military domain.

sets for that probe. This averaged value for the position in the retrieval set is shown on the Y axis. The best any similarity measure could do, was an average position in the retrieval set of 1.

For this evaluation, no task information was used and we only used concept and context. In the military domain, 21 dialogues were paraphrased and evaluated at 5 probes for a total of 105 retrievals each using Word2Vec, NetSim, TFIDF and SkipThoughts for a total of 420 retrievals. The results are shown in Fig. 6.

In the concierge domain 19 dialogues were paraphrased and evaluated at 5 probes for a total of 95 retrievals each for Word2Vec, NetSim TFIDF and SkipThoughts for a total of 380 retrievals. Overall, retrieval was performed 800 times in both domains. In the military domain, both TFIDF and NetSim performed about the same. TFIDF's performance worsened on probe size 5, but aside from this, the performance of all four similarity measures generally improved over time as the position of the desired dialogue approached the first position in the retrieval set. The results are shown in Fig. 7.

Here we see that our retrieval method when using SkipThoughts and Word2Vec outperformed TFIDF and NetSim in the military domain. In the concierge domain, Word2Vec and TFIDF were closer to the same in performance and both were considerably better than NetSim. As the probe size reached a size of 5, NetSim's performance significantly worsened. Similarly, Word2Vec's performance also worsened on probe size 5 in the concierge domain.

When using Word2Vec our retrieval method consistently returned the correct dialogue in the first position of the retrieval set and was often able to do so on probe size 1. Word2Vec also did not have an average position above 1.5 at any probe size in either domain with SkipThoughts having one occurrence above 1.5 in the military domain. When used in our retrieval algorithm, Word2Vec proved to be the most reliable and consistent of the four similarity measures with SkipThoughts following closely.

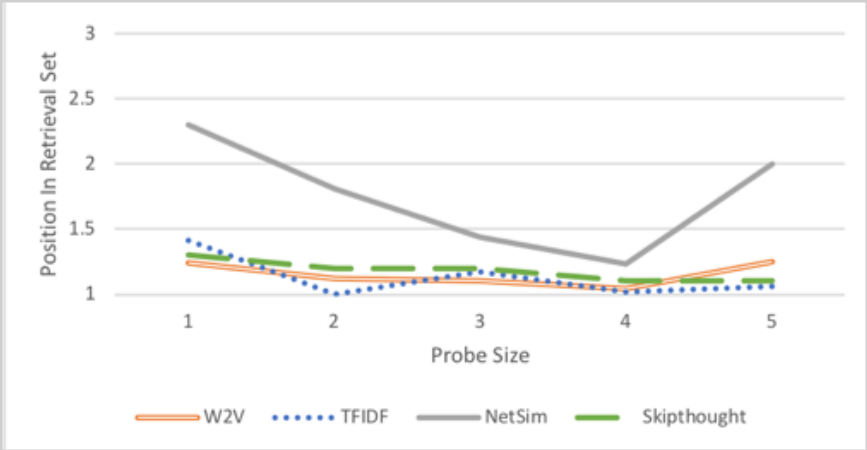

Figure 7. Retrieval evaluation in the concierge domain.

6. Related Research

Our next evaluation will measure the effect of adding task information to the retrieval process to understand the contribution of task information. Additionally, having a richer corpus with more dialogues will allow us to produce more robust results and gain a deeper understanding of KGs and the contributions of concept, context and task.

Additionally, we must consider the length of dialogues in retrieval. Our current retrieval algorithm can potentially be biased towards shorter dialogues if they also contain closely matching question utterances. A dialogue of two questions that are an identical match may be given preference to a longer dialogue that has more similar questions overall. For this reason, the length of dialogues must also be factored into retrieval.

Our approach builds upon previous work, particularly a taxonomy of goals (Bengfort & Cox, 2015) to create a multidimensional representation of a knowledge goal. This representation is defined by a knowledge goal space with which we can compare goal similarity using distance metrics. This implementation therefore allows us to use a simple nearest neighbor algorithm to provide guidance to the user; a simplification that improves upon many challenges regarding case-based learning.

When information is shared between team members on a project, their text is usually within a domain or along the lines of a specific topic of pursuit. The work done by (Hwang et. al, 2012) presents a context awareness system that learns about the user's needs and understands the domain the problem has appeared within. By collecting other data about their users, these systems can establish a context to better tailor their results and information to the user. They use a technology called ubiquitous learning. With this technology, alongside k nearest neighbors their system understands the appropriate context for learners in a learning environment.

The work of (Aha et. al, 2005) is highly relevant due to the conversation-based interface. Their work highlighted the importance of refining the user's question to solve a problem. Such problems

faced by the user are usually vaguely or briefly defined and lack adequate detail for the system to provide a meaningful solution. Their method of implementing a conversational style interface to extracting details of a target goal was proven to be effective and close to the natural way humans communicate problems. Our work builds on this style of conversation-based interface and also tracks the user's decomposition of goals into sub-goals. By allowing users to map out a 'plan' to solve their target goal, our system can better understand the context of why goals change and identify false tangents to better provide guidance.

7. Future Research and Conclusion

An area for future research is to address an issue that arises for concept similarity. Since Ronin uses indices between questions in the current dialogue and questions in a case dialogue, it may be that some questions may instead have a greater similarity to nearby questions. We would like to make our retrieval algorithm more robust to these situations and move beyond this basic assumption that questions will match directly.

Case-based reasoning systems are suitable for domains where knowledge goals are similar across many users. The interaction between users and case-based reasoning systems can capture increasingly complex knowledge investigations and retain them as goal trajectories within their case-base. In order to improve retrieval of cases for these systems, it is important to consider additional conceptual, contextual and task related information. Our method for retrieval demonstrates an effective way to retrieve goal trajectory cases for knowledge investigations across four different ways of measuring similarity. Because of the flexibility of our system, it can be applied to many different domains so long as knowledge goals within those domains can be decomposed into their most basic conceptual, contextual and task components.

Acknowledgements

This material is based on research sponsored by the Air Force Research Laboratory, under agreement number FA8650-16-C-6763. This research was also supported by ONR grant N00014-18-1-2009. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government.

Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, *7*, 39–59.

Aha, D., Breslow, L., & Muñoz-Avila, H. (1999). Conversational case-based reasoning. *Applied Intelligence*, *14*, 1–25.

Aha, D., McSherry, D., & Yang, Q. (2005). Advances in conversational case-based reasoning. *The Knowledge Engineering Review*, *20*, 247–254.

Bengfort, B., & Cox, M. (2015). *Interactive reasoning to solve knowledge goals*. (D. W. Aha, Ed., Technical Report GT-IRIM-CR-2015-001). Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA.

Cox, M., Dannenhauer, D., & Kondrakunta, S. (2017). Goal Operations for Cognitive Systems. *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence* (pp. 4385–4391). San Francisco, California: AAAI Press.

Eyorokon, V., Bengfort, B., Panjala, U., & Cox, T. (2016). Goal trajectories for knowledge investigations. *Workshops Proceedings for the Twenty-fourth International Conference on Case-Based Reasoning* (pp. 202–211). Atlanta, GA: CEUR Workshop Proceedings.

Eyorokon, V., Panjala, U., & Cox, T. (2017). Case-based goal trajectories for knowledge investigations. *Proceedings of the Thirtieth International Florida Artificial Intelligence Research Society Conference* (pp. 477–482). Palo Alto, CA: AAAI Press.

Huang, A., Milne, D., Frank, E., & Witten, I. (2009). Clustering documents using a Wikipedia-based concept representation. *Proceedings of the Thirteenth Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 628–636). Bangkok, Thailand: Springer.

Hwang, G., Tsai, C., Chu, H., Kinshuk K., & Chen, C. (2012). A context-aware ubiquitous learning approach to conducting scientific inquiry activities in a science park. *Australasian Journal of Educational Technology*, *28*, 931–947.

Kolodner, J. (1993). *Case-based reasoning*. San Mateo: Morgan Kaufmann.

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-thought vectors. *Proceedings of the 2015 Conference on Neural Information Processing Systems* (pp. 3294–3302). Montreal, Canada: NIPS Proceedings.

Kryszkiewicz, M. (2014). The cosine similarity in terms of the Euclidean distance. In J. Wang (Ed.), *Encyclopedia of Business Analytics and Optimization* (pp. 2498–2500). Hershey, PA: Business Science Reference.

Li, Y., McLean, D., Bandar, Z., O'shea, J., & Crockett, K. (2006). Sentence similarity based on semantic nets and corpus statistics. *IEEE transactions on knowledge and data engineering*, *18*, 1138–1150.

Mantaras, R., et al. (2005). Retrieval, reuse, revision and retention in case-based reasoning. *The Knowledge Engineering Review*, *20*,215–240.

Mingyan, G., & Agnar, A. (2006). Dialog learning in conversational CBR. *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference* (pp. 358–363). Melbourne Beach, FL: AAAI Press.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Proceedings of the 2013 Conference on Neural Information Processing Systems* (pp. 3111–3119). Nevada: NIPS Proceedings.

Salton, G., Wong, A., & Yang, C. (1975). A vector space model for automatic indexing. *Communications of the ACM*, *18*, 613–620.

Schuster, S., & Manning, C. D. (2016). Enhanced English universal dependencies: An improved representation for natural language understanding tasks. *Proceedings of the Tenth International Conference on Language Resources and Evaluation* (pp. 2371–2378). Portorož, Slovenia: ELRA.

Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, *24*, 35–43.

Harris, Z, S. (1954). Distributional structure. *Word*, *10*, 146–162.