Generating Procedural Knowledge Test Items Using Natural Language Processing Techniques

Bridge Eimon, Sowmya Ramachandran, Jeremy Ludwig

Stottler Henke Associates Inc

San Mateo, CA

beimon@stottlerhenke.com, sowmya@stottlerhenke.com, ludwig@stottlerhenke.com

ABSTRACT

Field manuals often contain useful instructions for how to complete certain tasks or procedures. The ability to apply knowledge of these tasks and procedures is typically assessed by hands-on demonstrations; however, testing the recall of such knowledge could be beneficial in advance of hands-on assessment. Questions that test recall of procedural knowledge could also promote mental rehearsals that are known to aid knowledge retention. They can aid personnel in self-assessments – they can test their recall of procedures in between missions. With these potential benefits in mind, we hypothesized that it is possible to accurately identify and extract descriptions of procedures from documents like field manuals, using a combination of rule-based and neural network-based approaches, and to automatically generate questions to test recall of these procedures. As a first step in testing this hypothesis, we developed and implemented an approach based on linguistic pattern recognition. We evaluated the performance of this approach in creating high quality test items that would be useful in practice. This paper will describe the technique used, provides examples, and describes a validation study to evaluate the results. It will also discuss the challenges that need to be addressed to improve accuracy and utility. Successful implementation of procedure extraction from manuals will also pave the way for automatically developing assessment criteria for simulation-based or hands-on performance assessments as well.

ABOUT THE AUTHORS

Bridge Eimon is an AI software engineer at Stottler Henke Associates, Inc. He leads the computer vision group of Stottler Henke, specializing in machine learning and neural networks for solving a wide variety of real-world classification and recognition projects, as well as automating human decision making processes.

Dr. Sowmya Ramachandran is a research engineer at Stottler Henke Associates where her research focuses on the application of artificial intelligence (AI), machine learning, and natural language processing to improve human performance in a broad range of domains such as intelligence analysis and tactical decision-making. Dr. Ramachandran has led the development of Intelligent Tutoring Systems for a diverse set of domains, from training paramedics to training Information Systems Technicians in the U.S. Navy.

Dr. Jeremy Ludwig is a principal engineer at Stottler Henke Associates, Inc. He directs teams of computer scientists and conducts research in artificial intelligence, applying reasoning, knowledge representation, and machine learning to create solutions for complex, real-world, problems.

Generating Procedural Knowledge Test Items Using Natural Language Processing Techniques

Bridge Eimon, Sowmya Ramachandran, Jeremy Ludwig

Stottler Henke Associates Inc

San Mateo, CA

beimon@stottlerhenke.com, sowmya@stottlerhenke.com, ludwig@stottlerhenke.com

INTRODUCTION

The ability to follow procedures to achieve mission and task objectives is critically important for Soldiers in the US Army. Army field manuals, the go-to source for references for Soldiers, contain step-by-step descriptions of the Army procedures to guide Soldiers in their service. Knowledge of procedures and the skills of executing them effectively are important requirements for most Army Military Occupational Specialties (MOSs). As such, it is important for the Army to assess Soldiers' proficiency at performing job-related procedures. While the standard way of assessing procedural knowledge in the Army is through observations of Soldiers performing procedures, assessing recall of procedures offers potential benefits. First, such assessments could serve as a precursor and priming for subsequent hands-on assessments. Second, assessing recall of procedures could be beneficial for refresher training and improve retention of knowledge and skills by serving as opportunities for mental rehearsals (Leonard, 2007). They can be also used by Soldiers for self-assessments of their readiness for hands-on performance assessments. Additionally, they can be used in training programs for evaluating learning and readiness.

Assessments to test recall of procedures can be facilitated by automatically generating them from sources such as the field manuals. This will reduce the manual labor involved in creating them. Given the potential benefits, we designed a research study to test the hypothesis that it is possible to accurately identify and extract descriptions of procedures from documents like field manuals using a combination of rule-based and neural network-based approaches, and to automatically generate questions to test recall of these procedures. As a first step in testing this hypothesis, we developed and implemented an approach based on linguistic pattern recognition. We evaluated the performance of this approach in creating high quality test items that would be useful in practice. This paper describes the technique used, provides examples, and describes a validation study to evaluate the results.

This research is situated in the larger context of research into methods for automated generation of multiple-choice assessment items from reference documents using emerging Artificial Intelligence (AI) techniques. Using field manuals and other topic-related documents as source manuals, the objective is to create multiple choice test items to assess Soldiers' knowledge of the topic. Given the importance of procedures-related knowledge and skills and the preponderance of procedural descriptions in field manuals, generating test items to assess knowledge of procedures became a significant component of this research.

PROCEDURE EXTRACTION

We have decomposed the task of automatic question generation for procedures into two steps: 1. Automatically identifying and extracting procedural descriptions, and 2. Generating multiple-choice test items from these extracted descriptions. In this section, we will focus on procedure extraction. The next section will discuss how questions are generated from extracted procedures. As shorthand, we will use the name Procedural Question Generation (PQG) throughout the paper to refer to the techniques presented.

The challenge of identifying and extracting procedure descriptions embedded in documents such as field manuals is that typically there are no explicit indictors or markings of their start and end. Usually there is some text stating that a procedure is going to be described, followed by the description. Often there are also formatting conventions that distinguish procedures from surrounding text. For example, it is common to present procedures as an enumerated or bulleted list of steps. It is common to have other linguistic cues as well. Procedures are usually described as imperative instructions (e.g., Chop the vegetables finely, Turn up the heat, etc.). Humans can easily identify procedural descriptions in texts using such linguistic and visual cues. Furthermore, the common sense understanding of the world helps us recognize when enumerated section of a document is describing an ordered set of steps of a procedure as opposed to an unordered set of directives that may not be a procedural description. Figure 1 shows two enumerated lists; it is easy to see that the one on the right is a description of procedure while the one on the left is not. Both lists include instructions but the list on the left represents an unordered set of suggested guidelines and the list on the right represents a prescriptive set of actions to be performed in the order specified to achieve a goal. The challenge is encoding this knowledge in a computational form that will enable automated identification and extraction of procedural descriptions. This section describes how we use such heuristics to achieves this purpose.

1. Develop a daily writing habit.
2. Try to read every day.
3. Capitalize when you're supposed to.
4. Avoid using exclamation points.
5. Always think about your audience.
6. Cut the filler phrases and buzzwords.
7. Sign up for a free writing course online.
8. Use writing templates.

1. Cross the laces to make an "X."
2. Wrap the top lace under the bottom lace and pull it through.
3. Make a loop with one end of the shoelace.
4. Wrap the other lace around the loop once.
5. Make another small loop with the wrapped lace and pull it through the "hole" in the middle.

Figure 1. Examples Of Enumerated Lists Signifying The Steps On A Procedure On The Right And A Non-Procedure On The Left

As mentioned, procedures are often written in the form of commands describing the steps you should take to complete the procedure, and so procedural sentences are often marked by a verb phrase that starts with a verb preceding the object because the subject of the sentence is implicit (you). Furthermore, procedures are almost always written sequentially: First take step 1, then step 2, and finally step 3.

Lists such as this are represented in text as either bullets or numbered/lettered lists. Using these two insights, we identify procedures as an ordered list in the text where verbs come before the objects. PQG first extracts the text using the *PyMuPDF* library (McKie et. al, 2021) and then extracts bullets or numbered lists from a document as well as the block of text preceding the list (which is used as contextual information for the list). PQG does this by first identifying which blocks have an initial list indicator, which is identified by extracting the first sequence of non-alphabetic, non-whitespace characters in the block as well as any alphabetic characters in the same word preceding the sequence. This captures for example bullets, and items like 1., 2), A:, B-, 3c). A list is then identified by sequential identifiers where each item is listed as the previous item with an increment of one (A->B, 1->2) or an identifier that remains the same (in the case of bullets). Once such a list is identified, the text block preceding the list is extracted as the contextual information for the list. Next, PQG breaks each list element down into its individual words and uses the open source Natural Language Toolkit (Bird et. al, 2009) Part-Of-Speech tagger to assign a part of speech to each word (Table 1 shows the different tags), which can then be iterated through to find the position of the first verb and the first noun. If the first verb comes before the first noun, then the list item is procedural.

These rules alone are not sufficient, however. First, because English often uses the Subject-Verb-Object sentence structure, ambiguous words at the beginning of the sentence were resolved to nouns by the part-of-speech tagger. For example, “Place the pen on the table” would have “Place” tagged as a noun, and as a result PQG missed many procedural sentences. To solve this issue, we used the Natural Language Toolkit (NLTK) corpus to get a list of all the meanings of each word; we then counted a word as a verb if there was any meaning for the word, spelled the same way, that was listed as a verb. We then added a threshold for the percentage of procedural list items for the entire list to be marked as a procedure to avoid false positives coming from sentences where words such as “place” or “find” really are being used as nouns. Second, there were many sentences that contained a verb phrase that was procedural but was hidden by a noun phrase earlier in the sentence. For a simple example, “After step 1 is complete, do step 2” was not being identified as a procedural sentence because “After step 1” contains the first noun and the first verb doesn’t appear until the verb phrase “do.” In this verb phrase, however, the phrase structure is procedural as the verb comes before the object and there is no subject. We addressed this issue in PQG by breaking the sentence down into phrases by splitting the sentence on commas, and then processing each phrase to check if it was procedural. Here again, the threshold percentage of procedural list items helps mitigate the problem of false positives being introduced.

Table 1. List Of Part-Of-Speech Tags Produced By NLTK

Number	Tag	Description	Number	Tag	Description
1	CC	Coordinating conjunction	16	PDT	Predeterminer
2	CD	Cardinal number	17	POS	Possessive ending
3	DT	Determiner	18	PRP	Personal pronoun
4	EX	Existential there	19	PRPS	Possessive pronoun
5	FW	Foreign word	20	RB	Adverb
6	IN	Preposition or subordinating conjunction	21	RBR	Adverb, comparative
7	JJ	Adjective	22	RBS	Adverb, superlative
8	JJR	Adjective, comparative	23	RP	Particle
9	JJS	Adjective, superlative	24	SYM	Symbol
10	LS	List item marker	25	YO	To
11	MD	Modal	26	UH	Interjection
12	NN	Noun, singular or mass	27	VB	Verb, base form
13	NNS	Noun, plural	28	VBD	Verb, past tense
14	NNP	Proper noun, singular	29	VBG	Verb, gerund or present participle
15	NNPS	Proper noun, plural	30	VBN	Verb, past participle

QUESTION GENERATION

Once we were able to extract the procedural lists from the text, we explored multiple ways to generate questions from the lists exploiting the "baked in" structure of procedural lists. These question types were "Next Step," "Re-order," and "Grab Bag." We provide further information on each of these question-generation methods below.

Next step questions. For this question type, the first few steps of a procedure can be listed, with a multiple choice for what the next step should be. The easiest way to do this is to shuffle the last four steps of a procedure so that all are semantically relevant, but the ordering is being tested. Algorithmically, this approach is very straightforward: simply print a procedural list omitting the last four steps, shuffle those last four steps, remove any part of the list numbering (which would give away the order of the shuffle), assign each step a choice assignment (1, 2, 3, 4) and print the shuffled steps. Figure 2 is an example of a "Next Step" question**.**

Re-order questions. The extension of the "Next Step" question type is the "Re-order" question type, which shuffles all the procedural steps and requires the test taker to re-order all the steps. This is a similar type of question to the "next step" question type, except that there is no context of correct order to use as assistance for the test taker and it requires a more complete understanding of the procedure from the test taker. Once again, PQG removes any part of the list numbering (which would reveal the order of the shuffle), shuffles all the steps, and then assigns each step a choice assignment. Figure 3 is an example of a "Re-order" question**.**

Grab bag questions. The "Grab Bag" question type is a modified version of the "Reorder" question type. Here, instead of requiring the test-taker to reorder all the steps in a procedure, a random subset of steps is omitted and shuffled, and the test-taker must then connect the shuffled steps to the correct order in the procedural sequence. This "Grab Bag" question type is parametrized so that the test generator can easily specify how many steps to omit from the original procedure. Notably, if the number of steps to omit is equal to the number of steps in the procedure, the "Grab Bag" question type is identical to the "Reorder" question type. Omitting only a subset of the procedural steps provides more framework and built-in context to the "Grab Bag" question type, allowing the test generator to make the question either harder, by omitting more steps, or easier by omitting fewer steps. Figure 4 shows a sample "Grab Bag" question with 3 steps omitted**.**

Question:
What is the Next Step in the Sequence?
The straight edge method is used when a compass is not available. When using it—
(1) Orient the map on a flat surface by the terrain association method.
(2) Locate and mark your position on the map.

Answer Choices:

a) Lay a straight edge on the map with one end at the user's position (A) as a pivot point; then, rotate the straight edge until the unknown point is sighted along the edge.
b) The intersection of the lines on the map is the location of the unknown point (C). Determine the grid coordinates to the desired accuracy (Figure 6-17).
c) Draw a line along the straight edge.
d) Repeat the above steps at position (B) and check for accuracy.

Correct Answer: 1: Lay a straight edge on the map with one end at the user's position (A) as a pivot point; then, rotate the straight edge until the unknown point is sighted along the edge.

Figure 2. Example Of A "Next Step" Question

Question:
Reorder the following sequence:

If you do not have a scale or ruler with 300 equal divisions or a map whose interval is other than 5'00", use the proportional parts method. Following the steps determines the geographic coordinates of horizontal control station 141.

1) With a boxwood scale, measure the distance from the bottom line to the top line that encloses the area around the control station on the map (total distance) (Figure 4-7).
2) Find a cross in grid square GL0388 and a tick mark in grid square GL1188 with 25'.
3) Measure the partial distance from the bottom line to the center of the control station (Figure 4-7). These straight-line distances are in direct proportion to the minutes and seconds of latitude and are used to set up a ratio.
4) Locate horizontal control station 141 in grid square (GL0784) (Figure 4-7).
5) Find another cross in grid square GL0379 and a tick mark in grid square GL1179 with 20'.
6) Enclose the control station by connecting the crosses and tick marks. The control station is between 20' and 25' (Figure 4-7).

Correct Sequence: 4, 2, 5, 6, 1, 3

Figure 3. Example Of A "Re-Order" Question

VALIDATION

Methods

To verify the accuracy of the PQG algorithm, we followed a systematic approach. Firstly, we selected a comprehensive manual in PDF format as the reference document. In this example we selected an Army manual titled "MAP READING AND LAND NAVIGATION." Next, we manually read through the entire PDF and carefully counted all the true procedures present in the document. This step ensured that we had a reliable ground truth dataset for comparison. Subsequently, we applied the PQG algorithm to automatically extract procedures from the same PDF. Finally, we compared the procedures extracted by the PQG algorithm to the ground truth procedures that we had manually identified.

We then extracted the number of

1. Hits (ground truth procedures that PQG correctly identified and extracted)
2. Partials (ground truth procedures that PQG identified but did not correctly extract)
3. Misses (ground truth procedures that PQG did not correctly identify)
4. False positives (procedures that PQG extracted that were not ground truth procedures)

By analyzing the similarities and differences between the two sets of procedures, we were able to evaluate the accuracy and effectiveness of the algorithm in accurately identifying and extracting procedures from the document.

<u>Question:</u>
Fill in the Missing Steps:
The following steps will determine the geographic coordinates of Wilkinson Cemetery (northwest of the town of Cusseta) on the Columbus map.
(1) Draw the parallels and meridians on the map that enclose the area around the cemetery.
(2) __________________
(3) __________________
(4) Select a scale that has 300 small divisions or multiples thereof (300 divisions, one second each; 150 divisions, two seconds each; 75 divisions, four seconds each, and so forth).
(5) __________________
(6) Determine the longitude, repeat the same steps but measure between lines of longitude and use E and W. The geographic coordinates of Wilkinson Cemetery should be about 32° 19'06"N and 84° 47'32"W (Figure 4-5).

<u>Answer Choices:</u>
1: Determine the geographic interval (5'00" = 300").
2: Determine the values of the parallels and meridians where the point falls.
3: To determine the latitude—

<u>Correct Answer:</u> [(2):2, (3):1, (5):3]

Figure 4. Example Of A "Grab-Bag" Question

Results

In reading through the navigation manual PDF, we identified 42 ground truth procedures. These included numbered procedures, lettered procedures, and inline procedures as shown in Figures 5, 6, and 7 below.

To locate a point on the Columbus map (Figure 4-6) when knowing the geographic coordinates, many of the same steps are followed.

(1) Place the 0 of the scale on the 32° 25'00" line and the 300 on the 32° 30'00". Make a mark at the number 28 on the scale (the difference between the lower and higher latitude).
(2) Place the 0 of the scale on the 84° 50′00″ line and the 300 on the 84° 50′55″. Make a mark at the number 56 on the scale (the difference between the lower and higher longitude.
(3) Draw a vertical line from the mark at 56 and a horizontal line from the mark at 28; they intersect at 32 25'28"N and 84 50'56"W.

Figure 5. Example Numbered Procedure

To plot an azimuth from a known point on a map (Figure 6-7)—

(a) Convert the azimuth from magnetic to grid, if necessary. (See paragraph 6-6.)
(b) Place the protractor on the map with the index mark at the center of mass of the known point and the base line parallel to a north-south grid line.
(c) Make a mark on the map at the desired azimuth.
(d) Remove the protractor and draw a line connecting the known point and the mark on the map. This is the grid direction line (azimuth).

Figure 6. Example Lettered Procedure

Using the Compass-to-Cheek Technique. Fold the cover of the compass containing the sighting wire to a vertical position; then fold the rear sight slightly forward. Look through the rear-sight slot and align the frontsight hairline with the desired object in the distance. Then glance down at the dial through the eye lens to read the azimuth

Figure 7. Example Inline Procedure

After running PQG on the entire pdf, PQG identified 83 procedures. We will breakdown the analysis of its performance according to its ability to *identify* procedures and to extract an identified procedure completely and correctly. Identification refers to whether PQG recognizes a block of text as describing a procedure. Extraction means finding all the steps of a procedure without omissions or additions. PQG correctly *identified* 36 of the 42 ground truth procedures. Of the 36 procedures it correctly identified, it correctly extracted information from 22. Its identification performance is shown in Table 2. Its extraction performance is shown in Table 3. Where green represents hits, red represents misses, blue represents partials, and yellow represents false positives. As PQG is designed to suggest Procedural Questions to a human, we tuned the PQG algorithm to search for more ground truths even if that included more false positives as a human could quickly and easily discard the false positives. Note that the manual contains 209 pages and therefore the true negative rate (or sensitivity) is quite high. Roughly assuming a procedure spans not more than 1 page of the document, the true negative rate would be 72%.

Table 2. PQG Identification Performance

	PQG Procedure	PQG Non Procedure
Ground Truth Procedure	36 (True Positive)	6 (False Negative)
Ground Truth Non Procedure	47 (False Positive)	120 pages (72%)
True positive rate (sensitivity): 86% False positive rate (defined as percentage of false positives among the positives) = 57%		

Table 3. PQG Extraction Performance

	Correctly Identified, Correctly Extracted	Correctly Identified, Incorrectly Extracted
PQG Procedure	22 (61%)	14 (39%)

A leading cause of false positives were enumerations such as the one shown in Figure 8. Here three of the five items start with verbs, and PQG incorrectly interprets the enumeration as a procedure. Of the missed procedures, many were correctly identified by PQG, but incorrectly extracted, primarily due to images being present in between steps and breaking the procedural list extracted by PQG, causing an incomplete procedure where only the first set of steps were extracted. Of the remaining procedures that were missed, they were all due to inline procedures, which PQG does not currently address.

Overlays of photographs should show the following information:

(1) North Arrow
(2) Title and Objective
(3) Time and Date
(4) Photo Reference
(5) Scale

Figure 8. Example False Positive (Enumeration)

DISCUSSION

Approach Challenges

Though the Procedure Extraction algorithm in PQG works well in a wide variety of situations, there are still multiple challenges that make it difficult to extract procedures. There are 3 main cases in which PQG has difficulty in extracting procedures: Enumerations, Sublists, and Popups.

We discussed earlier the challenge of differentiating between a list of actions that together describe a procedure and those that do not. The list shown on the left hand side of Figure 1 is not a procedure and the suggested actions can be performed in any order. It would not make sense to ask “what is the next step to becoming a better writer after trying to read every day?” Enumerations therefore do not make sense as next step questions, grab bag questions, or reorder questions. Right now, PQG does not attempt to address this challenge and extracts enumerations as procedures and relies on human input to filter out enumerations.

Sublists in which a single high-level step is broken into smaller steps, are a challenge for both procedure extraction and question generation. The primary challenge to procedure extraction is that there are many ways that sub steps can be indicated in text: there might be an extra indent, different numbering conventions might be used (e.g., bullets, alphabetic characters, or hierarchical enumeration numbering like 2.1, 2.2 etc.). Figure 9 below shows an example of a procedure with sub steps.

(1) Draw the parallels and meridians on the map that encloses the area around the cemetery.
(2) Determine the values of the parallels and meridians where the point falls.
(3) Determine the geographic interval (5'00" = 300").
(4) Select a scale that has 300 small divisions or multiples thereof.
(5) To determine the latitude—
    (a) Place the scale with the 0 of the scale on the latitude of the lowest number value and the 300 of the scale on the highest numbered line.
    (b) While keeping the 0 and 300 on the two lines, slide the scale along the parallels until the Wilkinson Cemetery symbol is along the edge of the numbered scale.
    (c) Read the number of seconds from the scale about 246.
    (d) Convert the number of seconds to minutes and seconds (and add to the value of the lower numbered line.
(6) Determine the longitude, repeat the same steps but measure between lines of longitude and use E and W.

Figure 9. Example Of A Procedure With A Sublist

PQG attempts to recognize sublists by keeping track of the marker of the original list (LIST1) item, and if the new marker is not a single increment over the previous item (as in a sequence 1,2,3 or a,b,c) or the same marker (in the case of bullet points) it starts a new list (LIST2) while keeping track of the last marker of LIST1. In the above figure, LIST1 is [1,2,3,4,5,6] and LIST2 is [a,b,c,d]. If after the LIST2 ends, the next marker continues the last marker of LIST1, then LIST2 is treated as a sublist for the last marker of LIST1 and LIST1 proceeds with procedure extraction. There is still more work needed to make sublist extraction more robust, and there is no current method for extracting sublists that are marked by differing indentations instead of some method of numbering. In addition, sublists create challenges for question generation as well. Having multiple levels of detail in sublists makes them distinguishable between LIST1 and LIST2 and identifies which list level they belong in. This makes sublist items poor candidates for next step questions, grab bag, and reorder questions. To tackle this challenge, PQG treats sublists as separate lists for the sake of question generation, creating questions for LIST1 and LIST2 separately.

Similar to sublists breaking up a procedure in a text, there are numerous other "Popups" that can occur inline with the text that cause challenging interruptions. Examples of these "Popups" include inline graphics, tables, warnings or disclaimers, headers/footers etc. The resulting challenge is adjusting a threshold that will not only ignore these "Popups" but also know whether the "Popup" interrupts a procedure or marks the end of the procedure. For numbered procedures this is a simple challenge, but for bulleted procedures this is much more difficult.

Consider Figure 10 as an example. In this procedure for cooking pasta, there are 6 steps, interrupted by a note rather than two separate lists. PQG attempts to handle inline notes or warnings by allowing a single block of text to interrupt any two steps in a procedure without ending the procedure. This value was chosen empirically as a tradeoff between filtering out interrupting notes, while also correctly ending bulleted procedures that are separated by short amounts of text. PQG currently ignores images and headers/footers when reading documents, so they won't interrupt any potential procedures. Tables are ignored as long as they are marked in the pdf, but if they are not marked, PQG often mistakenly ends procedures interrupted by tables.

- Brown meat in a large pan.
- Once cooked, add salt, pepper, tomato sauce and paste, water, sugar, basil, oregano and garlic.
- Simmer on low for an hour.
- Shortly before the hour is done, cook noodles as directed on package.

NOTE: noodles can be cooked more or less based on preference

- Once the noodles are cooked, drain and add to spaghetti sauce.
- ENJOY!

Figure 10. Example Of A Procedure With An Interruption

Alternative Approaches

Recently the introduction of ChatGPT has demonstrated the remarkable power of Large Language Models as a neural network-based approach to natural language processing. These approaches can generate incredibly appropriate responses to questions, summaries of text (Liu et. al, 2017), translations (Vaswani et. al, 2017), write poetry and songs, and in fact are very accurate at identifying procedures in a clean environment. We performed some preliminary investigations on the ability of such LLMs to extract procedures, using ChatGPT 3.5. Being trained on such a large corpus of text data enables ChatGPT to identify procedures even better than PQG. The drawback of using ChatGPT stems from the generative nature of the model. As a simple comparison, we took the ground truth procedures from the land navigation manual pdf and hand fed them in to ChatGPT 3.5 and attained the results shown in Tables 4 and 5.

Table 4. ChatGPT 3.5 Identification Performance

	PQG Procedure	PQG Non Procedure
Ground Truth Procedure	41 (True Positive)	1 (False Negative)
True positive rate (sensitivity): 98%		

Table 5. ChatGPT 3.5 Extraction Performance

	Extracted Correctly	Extracted Incorrectly
PQG Procedure	13 (32%)	28 (68%)

It is interesting to note that ChatGPT correctly identifies and extracts fewer procedures than PQG (13 for ChatGPT 3.5 vs 22 for PQG). However, combining the counts of correct and partially correct extractions, ChatGPT fares better (41 for ChatGPT 3.5 vs 36 for PQG). The primary issue is that ChatGPT 3.5 identifies more of the procedures (including the inline procedures), but does not extract them correctly, often creating new steps in the procedure or rewriting certain steps. It would be an interesting exercise to compare the entire performance of PQG's procedure extraction approach and ChatGPT 3.5 head-to-head (including the false positives generated by PQG) but that is beyond the scope of the current research project. Anecdotally, we find that ChatGPT 3.5 can identify, and extract procedure descriptions embedded in text but has a few quirks. For example, when extracting procedures from text, ChatGPT 3.5 often adjusts or modifies the procedures in small ways that are significant to the knowledge assessment. As an illustration, when asked what the steps are in the spaghetti cooking procedure, ChatGPT 3.5 turns the 6 steps into 7 by splitting up draining the noodles and adding to spaghetti sauce (Figure 11).

- Brown meat in a large pan.
- Once cooked, add salt, pepper, tomato sauce and paste, water, sugar, basil, oregano and garlic.
- Simmer on low for an hour.
- Shortly before the hour is done, cook noodles as directed on package.

NOTE: noodles can be cooked more or less based on preference

- Once the noodles are cooked, drain them.
- Add the cooked noodles to the spaghetti sauce.
- ENJOY!

Figure 11. Example Of A ChatGPT 3.5 Extracted Procedure

While in a generative format this would be fine, it is a layer of ambiguity that is detrimental to the knowledge assessment domain. In addition, ChatGPT 3.5 often adds notes or warnings in as steps of the procedure they reference. Furthermore, we have observed that ChatGPT 3.5 also fails at differentiating unordered enumerations from procedure descriptions. A joint approach may therefore offer benefits over the individual linguistic based approach of PQG or the LLM approach. Using an LLM to identify procedures, and a further linguistic approach to extract the exact procedures would allow the two approaches to complement each other and address the deficiencies in each approach.

FUTURE RESEARCH

There are two main pathways for future research and improvements. The first is, as mentioned above, combining LLM approaches with the linguistic approach of PQG to gain improvements in recognizing and extracting procedures, while the second is recognizing more diverse forms of procedures. In the challenges section, we mentioned that PQG ignores graphics and tables, but there are many examples in which graphics or tables include procedures or are referenced by procedures. Processing text out of images and tables, as well as linking references within procedures would all be beneficial next steps. In addition, PQG only recognizes bulleted or numbered lists and further work would be needed to identify and extract paragraph form procedures where contiguous sentences are procedural steps. Another form of procedure that PQG does not currently recognize are descriptive procedures as opposed to prescriptive procedures. The linguistic heuristics of PQG only recognize command form steps, and not descriptions of a model subject performing the procedure. These descriptive procedure forms while less common than prescriptive procedures, still offer a wealth of knowledge that can be extracted and tested on for material comprehension. Coincidentally, both paragraph form and descriptive form procedures are areas in which LLMs excel, and provide an excellent opportunity to combine the two approaches.

ACKNOWLEDGEMENTS

The research described herein was sponsored by the U.S. Army Research Institute for the Behavioral and Social Sciences, Department of the Army (Contract No. W911NF-20-C-0067). The views expressed in this presentation are those of the author and do not reflect the official policy or position of the Department of the Army, DOD, or the U.S. Government.

REFERENCES

Adams, N. E. (2015). Bloom's taxonomy of cognitive learning objectives. *Journal of the Medical Library Association : JMLA*, *103*(3), 152–153. Retrieved from https://doi.org/10.3163/1536-5050.103.3.010

Bird, S., Loper, E., & Klein, E. (2009). *Natural language processing with Python*. O'Reilly Media Inc.

Branson, R. K. (1978). The Interservice Procedures for Instructional Systems Development. *Educational Technology*, *18*(3), 11–14. JSTOR.

FM 3-25.26, Map Reading and Land Navigation. (2001). Department of the Army.

Leonard, B. (2007). *Literature Review on Skill Fade*. Human Factors Integration Defence Technology Centre.

Liu, L., Lu, Y., Yang, M., Qu, Q., Zhu, J., & Li, H. (2017). Generative Adversarial network for abstractive text summarization. *ArXiv:1711.09357 [Cs]*. http://arxiv.org/abs/1711.09357N

Manandhar, S., & Yuret, D. (2013). *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (S em E val 2013)*.

McKie, J., & Liu, R. (2021). *PyMuPDF: Python bindings for the PDF toolkit and renderer MuPDF (1.18.14) [C, Python; MacOS, Microsoft :: Windows, POSIX :: Linux]*. https://github.com/pymupdf/PyMuPDF

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 5998–6008.