

**Human Response Data Collection
for the Defense Advanced Research Projects Agency's
(DARPA) Mind's Eye Program**

by Nicholas C. Fung, Laurel C. Sadler, and Robert P. Winkler

ARL-TR-6635

September 2013

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory

Adelphi, MD 20783-1197

ARL-TR-6635

September 2013

Human Response Data Collection for the Defense Advanced Research Projects Agency's (DARPA) Mind's Eye Program

**Nicholas C. Fung, Laurel C. Sadler, and Robert P. Winkler
Computational and Information Sciences Directorate, ARL**

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
<p>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</p>					
1. REPORT DATE (DD-MM-YYYY) September 2013		2. REPORT TYPE Final		3. DATES COVERED (From - To) September 2011	
4. TITLE AND SUBTITLE Human Response Data Collection for the Defense Advanced Research Projects Agency's (DARPA) Mind's Eye Program				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Nicholas C. Fung, Laurel C. Sadler, and Robert P. Winkler				5d. PROJECT NUMBER 1FWDKA	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-CII-A 2800 Powder Mill Road Adelphi, MD 20783-1197				8. PERFORMING ORGANIZATION REPORT NUMBER ARL-TR-6635	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT <p>The Defense Advanced Research Projects Agency's (DARPA) Mind's Eye Project seeks to generate research in computer vision and artificial intelligence. The goal of the project is to create computer systems capable of perceiving humans in action and describe the actions in conversational English. This report documents efforts to create ground truth labeling for video files that can be used as training data for computer systems, as well as efforts to organize these data into a well-structured database. Ground-truth labeling efforts include data gathered through panel studies as well as crowdsourcing efforts through the Amazon Mechanical Turk service.</p>					
15. SUBJECT TERMS Mind's eye mechanical turk, action recognition, task labeling					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 52	19a. NAME OF RESPONSIBLE PERSON Nicholas C. Fung
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER (Include area code) (301) 394-3101

Contents

List of Figures	iv
List of Tables	v
List of Listings	v
1. Introduction	1
2. Videos, Verbs, Exemplars, and Variants	1
3. Baseline Human Response Data Collection	3
3.1 Login and Select Task	4
3.2 Detection Tasks	5
3.2.1 Recognition Task.....	5
3.2.2 Round Table Task	12
3.3 Description Task.....	21
3.3.1 Description	21
3.3.2 User Interface	22
3.3.2 Results and Discussion.....	24
3.4 Gap-Filling Task.....	24
3.4.1 Description	24
3.4.2 User Interface	26
3.3.2 Results and Discussion.....	28
4. Amazon Mechanical Turk Human Response Collection	28
4.1 HIT Creation.....	29
4.2 AMT Recognition Task.....	33
4.2.1 AMT Recognition Task Results and Discussion.....	36
4.3 AMT Description and Gap-Filling Tasks.....	37
4.4 Results and Discussions	42
5. Conclusion	42
6. References	43
Distribution List	44

List of Figures

Figure 1. Videos, verbs, and variants entity-relationship diagram.	3
Figure 2. Human response data collection top-level use cases.	4
Figure 3. Login and task selection.	5
Figure 4. Recognition task use cases.	7
Figure 5. Recognition task entity-relationship diagram.	8
Figure 6. Recognition task GUI.	9
Figure 7. Precision, recall, and F -score for the recognition task.	11
Figure 8. Round Table Task use case.	13
Figure 9. Round Table Task entity-relationship diagram.	14
Figure 10. Round Table Task GUI.	15
Figure 11. Precision, recall, and F -score for the Round Table Task.	17
Figure 12. Recognition Task vs. Round Table Task precision.	18
Figure 13. Recognition Task vs. Round Table Task recall.	19
Figure 14. Recognition Task vs. Round Table Task F -score.	20
Figure 15. Description Task use cases.	22
Figure 16. Description Task entity-relationship diagram.	22
Figure 17. Description Task GUI.	23
Figure 18. Gap-Filling Task use cases.	25
Figure 19. Gap-Filling Task entity-relationship diagram.	25
Figure 20. Gap-Filling Task GUI.	27
Figure 21. AMT Recognition Task.	34
Figure 22. AMT Recognition Task entity-relationship diagram.	35
Figure 23. AMT Recognition Task: precision, recall, and F -score.	37
Figure 24. AMT Description Task.	38
Figure 25. AMT Description Task entity-relationship diagram.	39
Figure 26. AMT Gap-Filling Task.	40
Figure 27. AMT Gap-Filing Task entity-relationship diagram.	41

List of Tables

Table 1. Verbs.....	3
Table 2. “Gold standard” reference recognition task (verb,exemplar) pairings.	6
Table 3. Precision, recall, and F -score for recognition task.	11
Table 4. Precision, recall, and F -score for the Round Table Task.....	17
Table 5. Recognition Task vs. Round Table Task precision.	19
Table 6. Recognition Task vs. Round Table Task recall.	20
Table 7. Recognition Task vs. Round Table Task F -score.....	21
Table 8. In-house vs. AMT Recognition Task.....	37

List of Listings

Listing 1. JSON video metadata.	2
Listing 2. Algorithm for assigning recognition tasks to evaluators.	6
Listing 3. Recognition task algorithm.....	10
Listing 4. Round Table task assignment algorithm.	12
Listing 5. Round Table Task algorithm.	16
Listing 6. Description Task selection algorithm.	24
Listing 7. Gap-Filling Task algorithm.	27
Listing 8. AMT HIT XML for a single question.	30
Listing 9. FormattedContent XML example.....	31
Listing 10. Video.html.	32

INTENTIONALLY LEFT BLANK.

1. Introduction

The Defense Advanced Research Projects Agency’s (DARPA) Mind’s Eye program is aimed at developing a visual intelligence capability for unmanned systems. This program seeks to automatically identify the action between objects in a scene directly from visual inputs, and then reason over those representations. “A key distinction between this research and the state of the art in machine vision is that the latter has made continual progress in recognizing a wide range of objects and their properties - what might be thought of as the nouns in the description of a scene. The focus of Mind’s Eye is to add the perceptual and cognitive underpinnings for recognizing and reasoning about the verbs in those scenes, enabling a more complete narrative of action in the visual experience.” (DARPA)

Twelve research teams have been contracted by DARPA for the Mind’s Eye program. In order to evaluate each of the teams, a common set of test videos were created to depict common actions. Human-generated ground-truth data for these videos were collected for each of the tasks performed by the software algorithms. The visual intelligence algorithms were then evaluated as to how well they performed the identification task on these videos as compared to a human subject.

The ground-truth collection was a two-step process. In the first step, data were collected in a small controlled environment. These data were used as ground truth as well as to create a “gold standard.” The second step was to collect crowdsourced data using Amazon Turk, where a larger and more diverse sample set could be accessed. A major concern with the crowdsourced data were inaccurate data. The “gold standard” was used to help verify the Amazon Turk data and filter workers that did not properly perform the task. This report describes each of the tasks used to collect the ground-truth data as well as the design of the database used to hold the results of each task.

First, we describe the videos, their characteristics, the motivation for the collection process, and the database used to store them. Next, we describe different variants for collecting the ground-truth data, the database used to store them, and the software used for data collection. Finally, we describe the Amazon Turk tasks used to crowdsource the videos and the database used to store them.

2. Videos, Verbs, Exemplars, and Variants

The creation of the trial video collection was driven by a selection of verbs being demonstrated in a variety of lighting and camera conditions. Janus Research Group, Inc.

(<http://www.janusresearch.com>) was contracted by DARPA to produce these videos. Examination of the JavaScript Object Notation (JSON) metadata associated with these videos (listing 1) leads to the following discussion of entities, relationships and attributes for normalizing a relational model of videos for storage in a Relational Database Management System (RDBMS).

```
{
  "action" : "1",
  "actors" : [ "6", "1"],
  "angle" : "front",
  "camera" : "1",
  "contrast" : "high",
  "creation" : "Tue Jan 11 22:24:20 2011",
  "filter" : "none",
  "framing" : "centered",
  "location" : "mc",
  "scene" : "park",
  "setting" : "park",
  "time" : "aftn",
  "url" : "APPROACH5_A1_C1_Act1_6_PARK_MC_AFTN_1a372d08-1e47-11e0-9fc9-
e80688ca39a2.mov",
  "uuid" : "1a372d08-1e47-11e0-9fc9-e80688ca39a2",
  "verbs" : [ "approach5" ]
}
```

Listing 1. JSON video metadata.

The **action**, **actors**, **camera**, **location**, **scene**, **setting**, and **time** fields proved irrelevant to the Mind’s Eye program and are not discussed. We preserved these fields in the database for organizational purposes. The **angle** field indicates the camera angle and is $\in \{\text{front, side}\}$. The **contrast** field indicates the level of contrast in the video and is $\in \{\text{high, low}\}$. The **creation** field indicates the date and time the video was produced. The **filter** field indicates whether or not a filter was applied to the video and is $\in \{\text{dark, none}\}$. The **framing** field indicates where in the camera’s field of view that activity took place and is $\in \{\text{centered, peripheral}\}$. The **url** field indicates the unique name of the video file. The **uuid** field indicates a universally unique identification (uuid) for the video. Taken together, the Cartesian product of *angle* \times *contrast* \times *filter* \times *framing* represents what we term a “variant.” There are 16 different variants representing the 16 elements of this Cartesian product. Forty-eight action verbs (see table 1) were selected to be illustrated by human actors in the videos. Ten exemplars for each action verb were filmed, each with their 16 variants for a total of $16 \times 48 \times 10 = 7680$ videos. The **verbs** field indicates the concatenation of the verb being illustrated with the exemplar represented by the verb-variant pairing. For example “approach5” in listing 1 indicates the fifth exemplar of the verb “approach” under the conditions of front camera angle, high contrast, no filter, and centered framing.

Table 1. Verbs.

Verbs							
Approach	Arrive	Attach	Bounce	Bury	Carry	Catch	Chase
Close	Collide	Dig	Drop	Enter	Exchange	Exit	Fall
Flee	Fly	Follow	Get	Give	Go	Hand	Haul
Have	Hit	Hold	Jump	Kick	Leave	Lift	Move
Open	Pass	Pickup	Push	Putdown	Raise	Receive	Replace
Run	Snatch	Stop	Take	Throw	Touch	Turn	Walk

Given the above discussion, we modeled the videos, verbs, and variants as strong entities. There is a many-to-one relationship between the videos and variants, as well as between the videos and the verbs. We introduced a system-assigned integer as the primary key for all three of the entities to eliminate the need to perform joins on strings. We then split the **verbs** field in the JSON metadata into its verb and exemplar. The entity-relationship diagram shown in figure 1 illustrates this design.

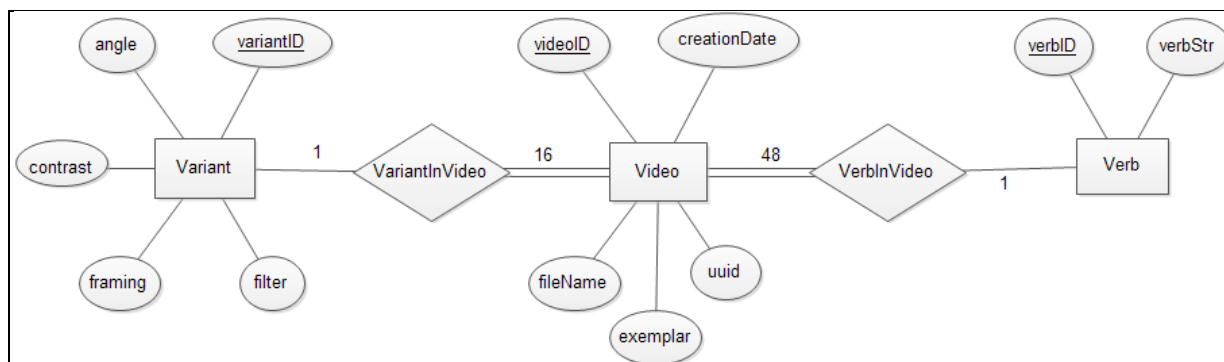


Figure 1. Videos, verbs, and variants entity-relationship diagram.

There are 16 variants associated with each of the videos representing 48 different verbs. A system-assigned integer primary key was introduced for the video, verb and variant entities to avoid having to join on strings. Each variant is defined by a unique combination of the binary-valued angle, contrast, framing, and filter attributes. Each video has an associated uuid, creation date, and filename. The Janus research group collected 10 exemplars for each (verb, variant) pairing, the video entity’s exemplar attribute indicates which one. The verb string is preserved in the verbStr attribute of the verb entity.

3. Baseline Human Response Data Collection

ARL was responsible for collecting the baseline human ground-truth data to label the set of video vignettes for the DARPA Mind’s Eye program. The ground-truth data were collected for three of the Mind’s Eye tasks: a “recognition” task, where individuals watched a video and were presented with one of the 48 action verbs and asked whether or not they observed that action in

the video; a “description” task, where individuals watched a video and were asked to describe what they saw in the video; and a “gap-filling” task, where individuals watched a video that had been modified to include dead space and were asked to describe what they imagined had happened in the gap. An additional task, dubbed the “round table,” was also conducted in which the individuals watched a video, were presented with a list of all 48 verbs, and asked which of these verbs were present in the video. This task is a derivative of the “recognition” task. For each of these tasks, human subjects watched multiple videos and answered various questions about the videos. The top level use case diagram for the Human Response Data Collection System is shown in figure 2. A graphical user interface (GUI) was developed for the data collection exercises for each of the tasks.

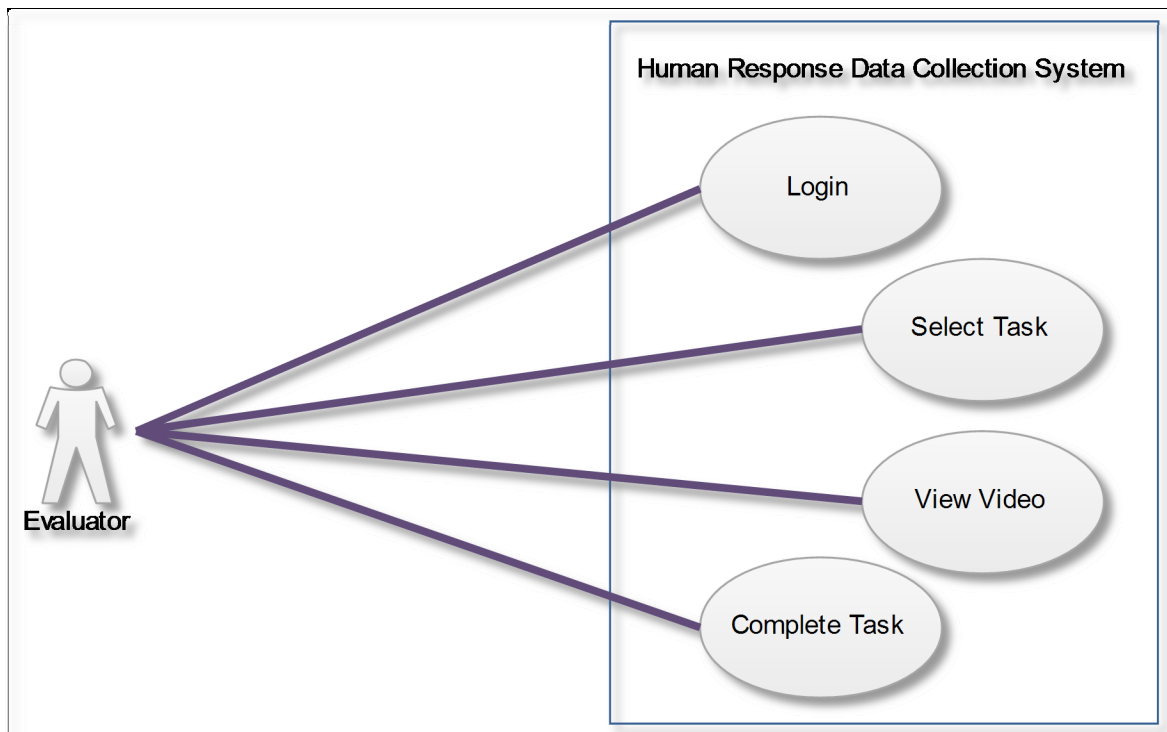


Figure 2. Human response data collection top-level use cases.

3.1 Login and Select Task

Prior to viewing any of the Task GUIs each participant logged into the Mind’s Eye Tasks viewer, as shown in figure 3, using a pre-assigned evaluator ID. The system uses this evaluator ID to select the set of pre-assigned tasks designated for each evaluator. Once the evaluator has logged onto the system, they select a task by clicking on the appropriate button. The GUI then presents the user with the designated task.

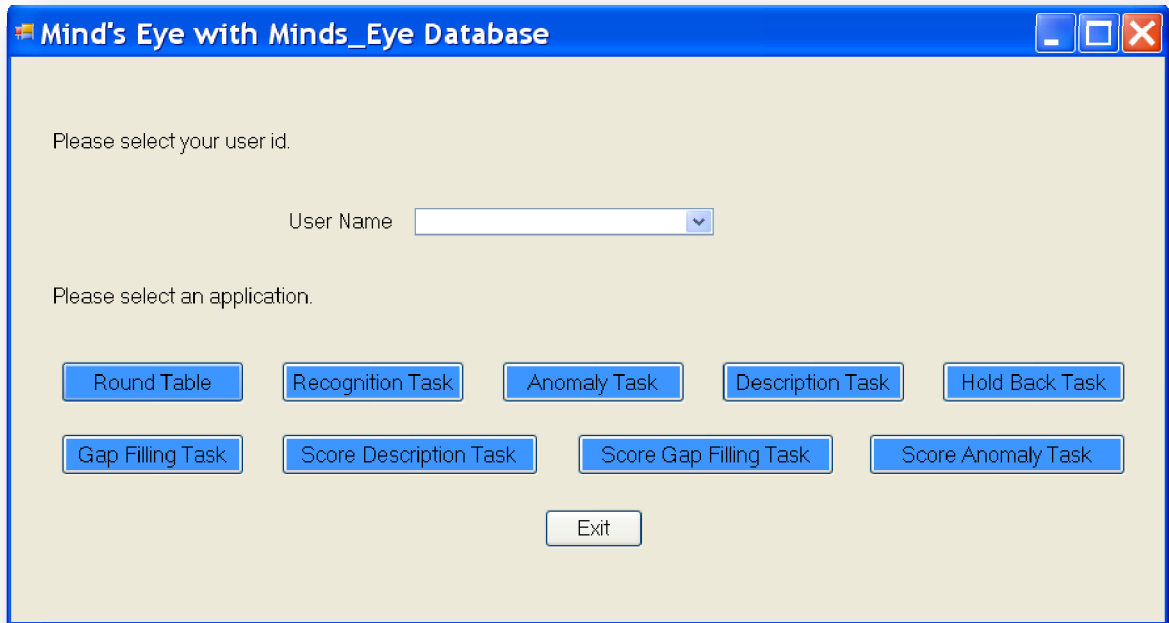


Figure 3. Login and task selection.

3.2 Detection Tasks

The ground-truth data for the Mind’s Eye project for the Detection Tasks was collected in two different exercises: the “round table” task and the “recognition” task. The “recognition” task more closely resembles the task assigned to the automated systems designed by the research team principal investigators (PIs). Data for this task were collected in two stages: an internal panel study and a crowdsourced study. The “round table” task was collected solely from an internal panel. The data responses from the Detection Tasks were used to create a “gold standard” for the Amazon Mechanical Turk crowdsourcing study. As crowdsourcing studies rely on external workers, confidence in their responses can be low. The “gold standard” data were used to check Mechanical Turk evaluators for inaccurate or fraudulent responses as a screen against malicious users. The “recognition” and “round table” tasks are described in greater detail below.

3.2.1 Recognition Task

3.2.1.1 Description

For the “recognition” task, 24 verb/exemplar pairings (shown in table 2) were chosen to be evaluated by the internal panel. For each of these recognition verbs a member of the panel was asked whether or not one of the 48 verbs shown in table 1 was observed in a video intended to illustrate a “gold standard” or reference verb. This task required each evaluator to answer a total of $24 \times 48 = 1152$ yes/no questions. The panel consisted of 16 viewers, each tasked with viewing a different variant of the verb/exemplar video for a given verb asked. The “gold standard” video

shown and verb asked pairings were pre-assigned to each viewer according to the algorithm shown in listing 2. The use case diagram for the Recognition Task is shown in figure 4 and the GUI for this task is shown in figure 6. The system randomly selected a video-verb question task from this set of predefined pairings for each evaluator. Once the evaluator submitted a response, the information was recorded to the database for the corresponding video, verb, and evaluator ID. The observed video only appeared again if paired with a new *verb* question. The entity-relationship diagram for the Recognition Task is shown in figure 5.

Table 2. “Gold standard” reference recognition task (verb,exemplar) pairings.

Gold Standard Reference Recognition Task (Verb, Exemplar) Pairings							
Bounce,4	Bury,2	Carry,5	Catch,1	Chase,9	Close,6	Collide,2	Dig,2
Enter,2	Flee,3	Fly,10	Follow,1	Give,2	Hit,3	Jump,10	Kick,5
Lift,7	Open,4	Pickup,2	Push,2	Raise,9	Run,4	Throw,3	Walk,9

```

ReferenceVideos = {Table 2 (verb,exemplar) pairings}
Variants = {angle X contrast X filter X framing}
VerbsToAsk = {Table 1 verbs}
Evaluators = {16 human evaluators}
Database.DetectionTask = the table in the database used for storing the task
for all pairings videos in VerbsToShow do
  evaluatorIndex := 0
  for all variants variant in Variants do
    evaluatorID := evaluatorIndex+1
    for all verbs verbToAsk in VerbsToAsk do
      // Find the corresponding video
      videoToShow := Find(video.verb,video.exemplar, variant)
      // Assign this evaluator the (videoToShow,verbToAsk) pair and increment evaluatorID
      task := (evaluatorID mod 16,videoToShow, verbToAsk) // evaluatorID will go to 48
      Database.DetectionTask.Insert(task)
      evaluatorID = evaluatorID + 1
  
```

Listing 2. Algorithm for assigning recognition tasks to evaluators.

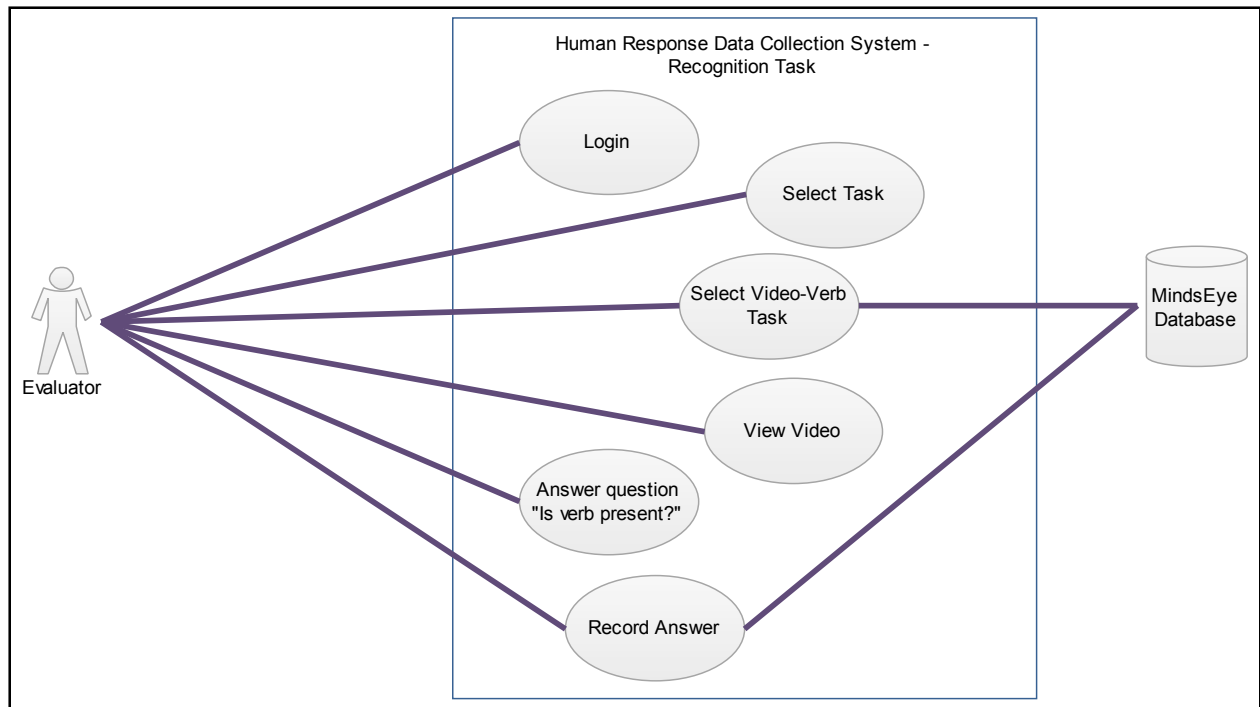


Figure 4. Recognition task use cases.

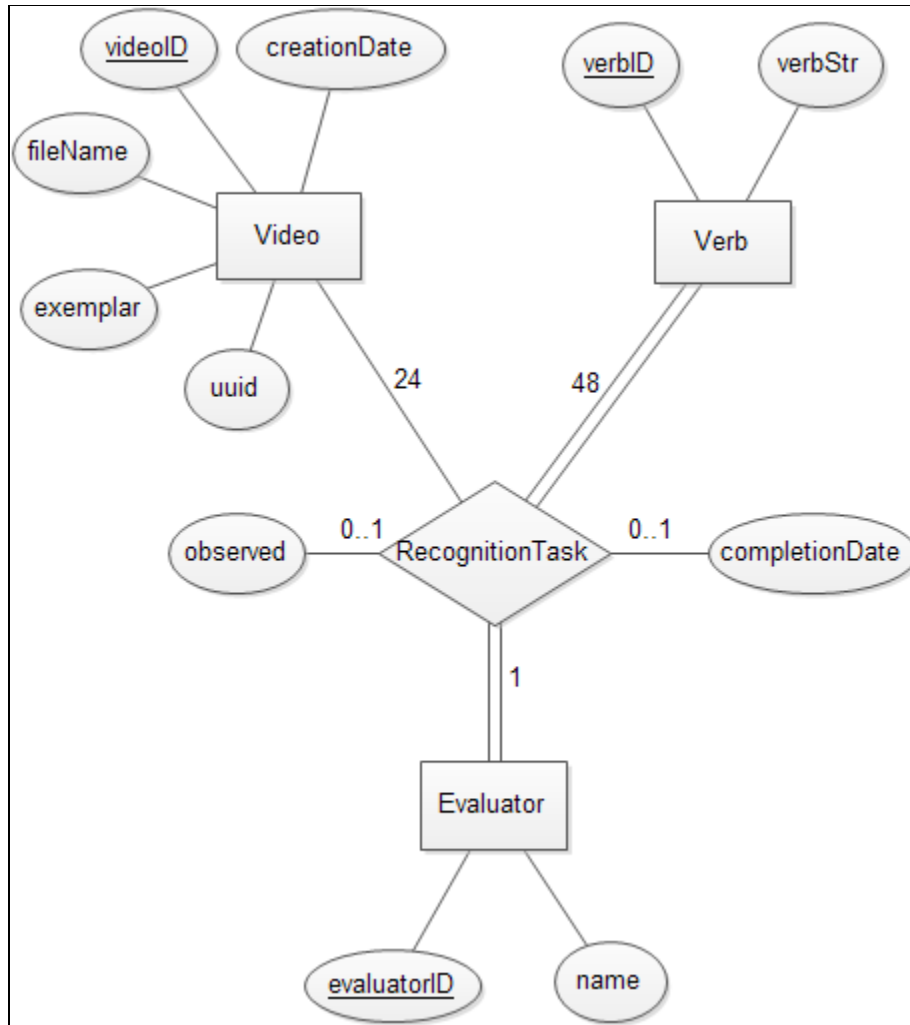


Figure 5. Recognition task entity-relationship diagram.

The RecognitionTask relationship is a ternary relationship between 24 video/exemplar pairings for each of the 48 verbs and an associated evaluator. A system-assigned integral primary key called evaluatorID was introduced in the evaluator entity to avoid having to join on the string-values name. The RecognitionTask's observed and completionDate attributes are null until the evaluator has answered the question as to whether or not they recognized that verb in the video. The observed attribute is a bit where a value of 1 indicates the evaluator did recognize the verb in the video and a value of 0 indicates the evaluator did not.

3.2.1.2 User Interface

The GUI for the Recognition Task (figure 6) consists of a video viewing system and user response buttons. The video viewing system is complete with the ability to pause and replay the video. The GUI also provides the participant with directions instructing the participant to watch the video and complete their response by choosing Yes or No using the interface buttons. The GUI also provides the evaluator with the definition for the verb in question as well as an example

of how the verb is used in a sentence. The inclusion of this information is intended to remove any ambiguity in the circumstance where a verb might have multiple definitions or if the user is unfamiliar with the verb. The task question is displayed below the verb definition with bold lettering used to emphasize the verb. Once the evaluator has selected an answer, this information is committed to the database and the next video and verb pairing question designated for this evaluator is randomly selected from the database. The new video is displayed and automatically begins playing in the video screen with the corresponding question and verb definition reflected on the GUI. The “Skip Video” button allows the evaluator to move on to a different video/verb pairing exercise without committing any data to the database, allowing the user to view the video-verb pairing at a later time. The GUI denotes how many video-verb pairing tasks remain for the current evaluator to give the user an indication of how much progress has been made. Once the evaluator has completed all their designated tasks, a window is displayed informing the evaluator that all the tasks have been completed. The evaluator can then exit the Recognition Task GUI using the “Exit Recognition Task” button or the X in the upper right-hand corner of the application window. The algorithm for selecting and displaying the Recognition Tasks for a given user is shown in listing 3.

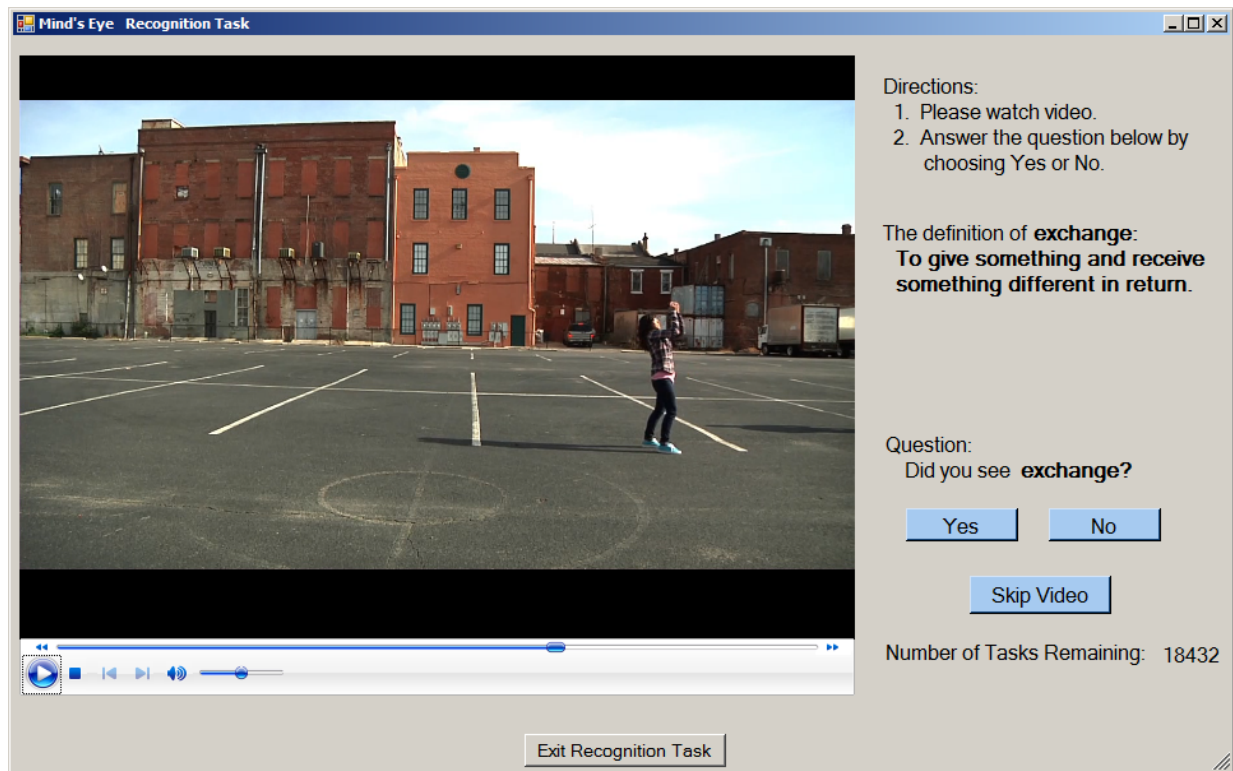


Figure 6. Recognition task GUI.

```

RecognitionTasks = the database table storing the Recognition Task assignments
User = the id of the currently logged on evaluator
GUI = the user interface
Tasks := RecognitionTasks.Select("evaluatorID = User ∧ completionDate ≠ ∅")
while Tasks ≠ ∅ do
    Task := Random(Tasks) // select a random tuple
    GUI.Show(Figure 6: Recognition Task GUI)
    Task.Update(GUI.result(), date())
    Tasks := Tasks − task
    GUI.Update(|Tasks|) // show number of tasks remaining
    Tasks := RecognitionTasks.Select("evaluatorID = User ∧ completionDate ≠ ∅")
GUI.Show("You're all done")

```

Listing 3. Recognition task algorithm.

3.2.1.3 Results and Discussion

Three metrics were employed to analyze the collected data: precision, recall, and F -measure for the recognition of the action intended to be captured by the video (figure 7 and table 3). Precision (equation 1) is a measure of the predictive value of an instance, representing the fraction of recognized values that are relevant.

$$\text{Precision} = \frac{\# \text{ true positives}}{\# \text{ true positives} + \# \text{ false positives}} \quad (1)$$

For this study, a true positive was recorded if a subject indicated the presence of a verb that the video intended to demonstrate. A false positive was recorded if a subject indicated the presence of a verb that a video did not intend to demonstrate. The average precision of a video was 0.13 ($\sigma = 0.05$). Recall (equation 2), or sensitivity, is a measure of the degree to which relevant instances are correctly classified.

$$\text{Recall} = \frac{\# \text{ true positives}}{\# \text{ true positives} + \# \text{ false negatives}} \quad (2)$$

False negatives are those actions intended to be captured by the video that were not recognized by one or more evaluators. Since the number of evaluators was constant at 16, recall represents that fraction of evaluators who recognized the action the video was intended to capture. The average recall was 0.89 ($\sigma = 0.16$). The high levels of recall indicate that almost all of the videos (with the notable exception of “flee”) did sufficiently capture the intended action. On the other hand, the low levels of precision indicate that none of the videos unambiguously captured the intended action. Only “jump” exceeded 20% precision. This result is understandable given that many of the actions were either synonymous with another verb (“pick up”, “lift”, and “raise”), subsumed one another (“walk”, “run”, and “leave” all subsume “move”), required one another to capture the intended action (“bury” requires “touch,” “stop” requires “go”), or imply one another

(“chase” implies “run”). The implication is that the degree of overlap between the action verbs selected for the Mind’s Eye program were deeply intertwined with extensive overlap. Finally, the F -scores indicate that none of the videos achieved 50% accuracy. The average F -score was 0.22 ($\sigma = 0.07$). How these results were used in the Amazon Turk collection is discussed in that section.

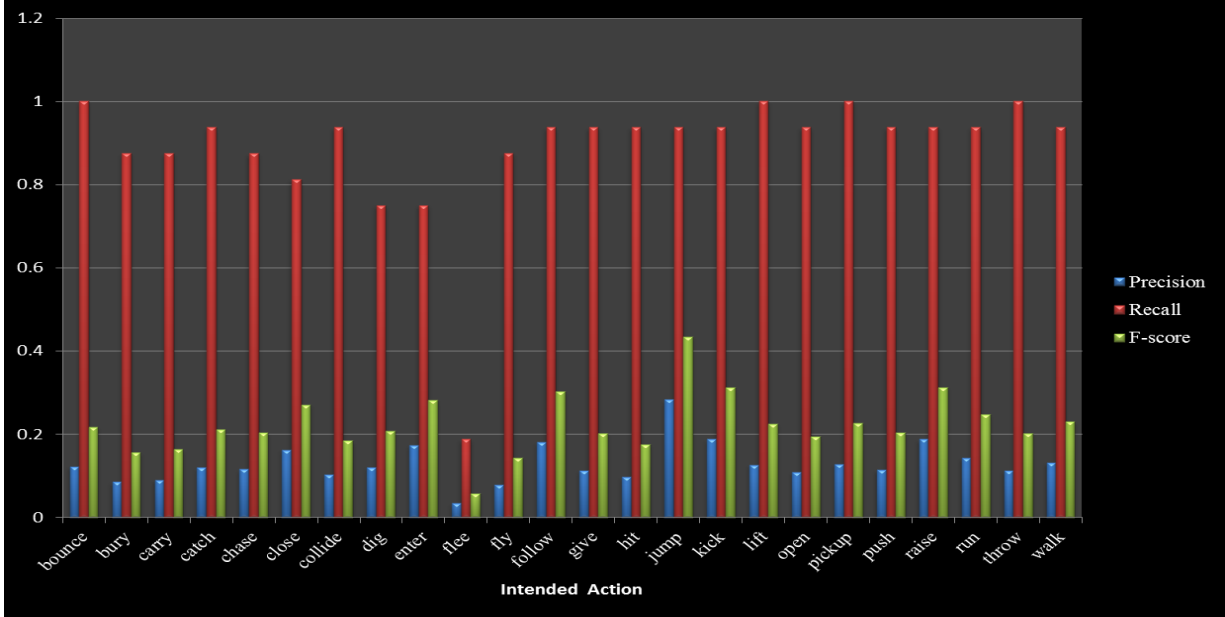


Figure 7. Precision, recall, and F -score for the recognition task.

Table 3. Precision, recall, and F -score for recognition task.

Intended Action	Precision	Recall	F -score
bounce	0.122	1.000	0.218
bury	0.085	0.875	0.156
carry	0.089	0.875	0.163
catch	0.119	0.938	0.211
chase	0.115	0.875	0.204
close	0.163	0.813	0.271
collide	0.102	0.938	0.184
dig	0.120	0.750	0.207
enter	0.173	0.750	0.282
flee	0.034	0.188	0.058
fly	0.078	0.875	0.144
follow	0.180	0.938	0.303
give	0.112	0.938	0.201
hit	0.096	0.938	0.175
jump	0.283	0.938	0.435
kick	0.188	0.938	0.313
lift	0.126	1.000	0.224
open	0.109	0.938	0.195
pickup	0.128	1.000	0.227
push	0.115	0.938	0.204

Table 3. Precision, recall, and F -score for recognition task (continued).

Intended Action	Precision	Recall	F -score
raise	0.188	0.938	0.313
run	0.143	0.938	0.248
throw	0.113	1.000	0.203
walk	0.132	0.938	0.231
μ	0.130 ($\sigma=0.05$)	0.890 ($\sigma=0.16$)	0.220 ($\sigma=0.07$)

3.2.2 Round Table Task

3.2.2.1 Description

The Round Table Task consisted of 5 evaluators, each of which was assigned a random variant of each of the 10 exemplars for all 48 verbs. All of the evaluators watched the same 480 videos. When shown each of these 480 videos, the evaluators were presented with a list of 48 check boxes containing the action verbs shown in table 1 and asked to check all of the verbs present in the video. The algorithm for assigning Round Table tasks to evaluators is shown in listing 4. The use case diagram for the Round Table Task is shown in figure 8 and the entity-relationship diagram for storing the tasks and results is shown in figure 9. It is structurally identical to the entity-relationship diagram for the Recognition Task shown in figure 5 with only the cardinalities changed. The system randomly selected a video from a set of predefined videos for each evaluator. Once the evaluator had performed the task, the response information was submitted to the database for the corresponding video, verb, and evaluator ID. This completed the data collection for the observed video and it never appeared again.

```

ReferenceVideos = {Table 2 (verb,exemplar) pairings}
Variants = {angle X contrast X filter X framing}
Exemplars = {10 exemplars}
VerbsToAsk = {Table 1 verbs}
VerbsToShow = {Table 1 verbs}
Evaluators = {5 human evaluators}
Database.RoundTableTask = the table in the database used for storing the task
for all verbs toShow in VerbsToShow do
  for all exemplars exemplar in Exemplars do
    variant := Random(Variants)
    video := Find(toShow, exemplar, variant)
    for all evaluators evaluator in Evaluators do
      for all verbs toAsk in VerbsToAsk do
        Database.DeterctionTask.Insert(task)

```

Listing 4. Round Table task assignment algorithm.

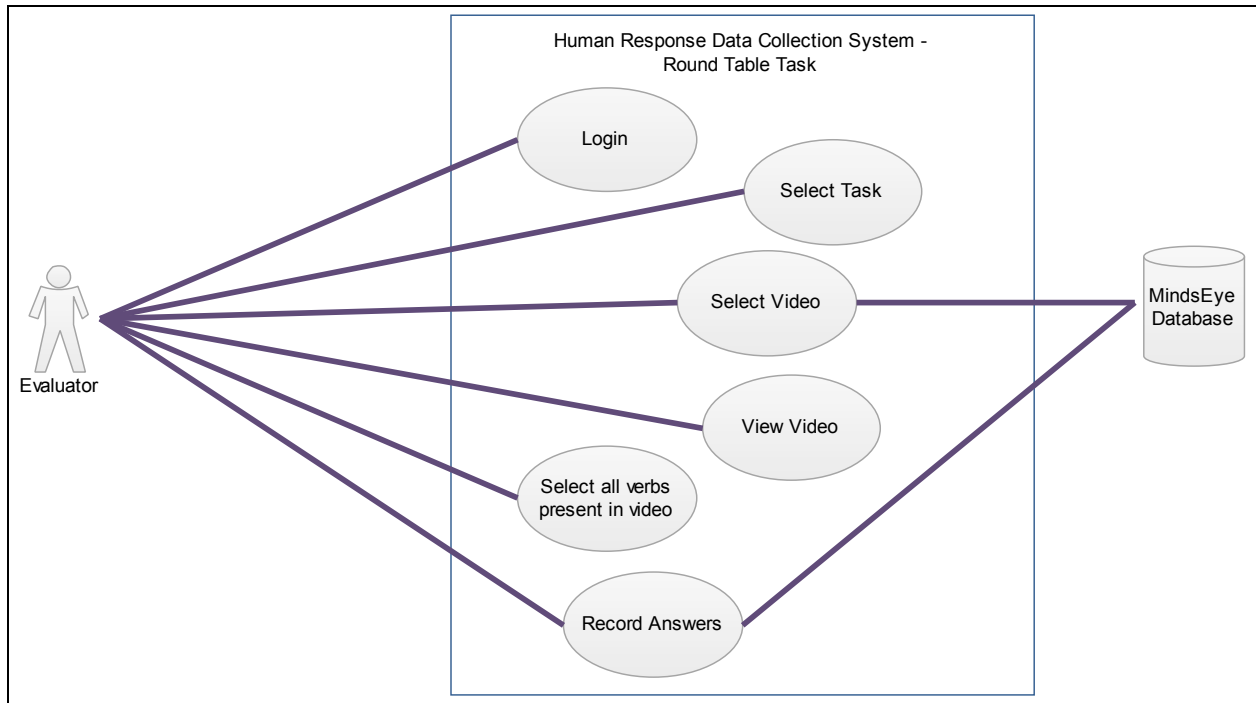


Figure 8. Round Table Task use case.

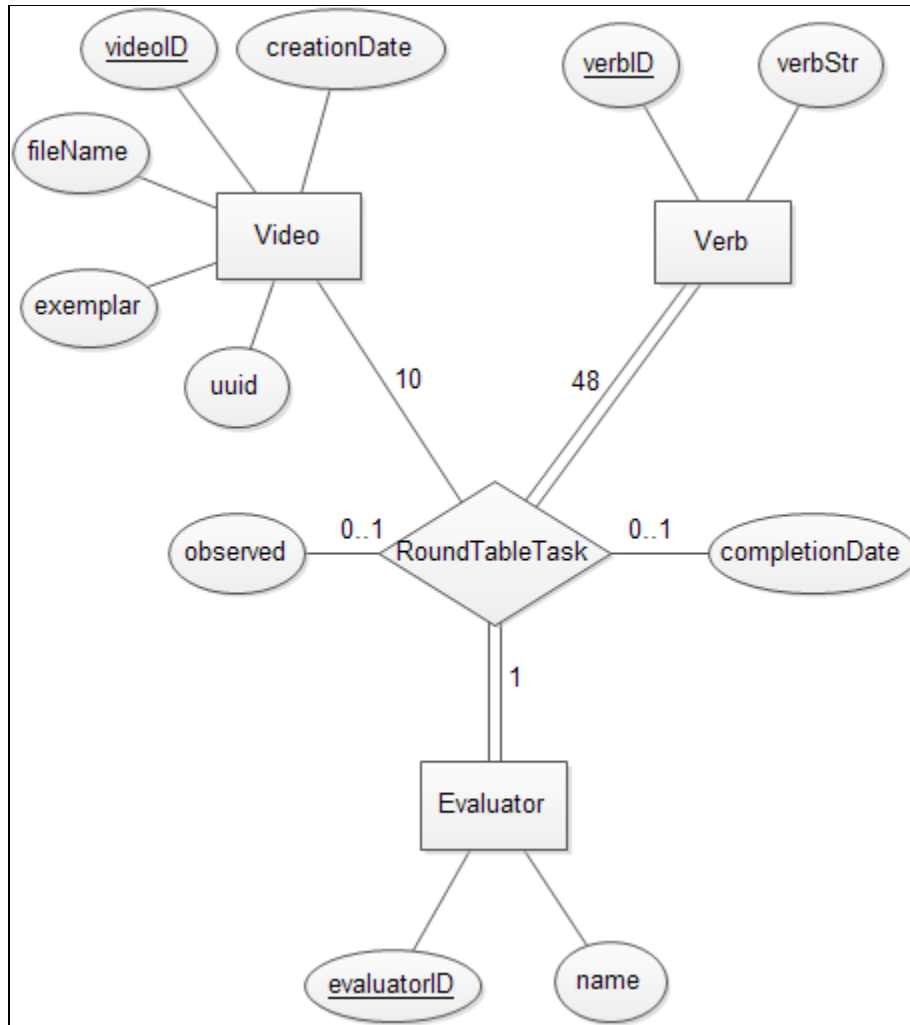


Figure 9. Round Table Task entity-relationship diagram.

The RoundTableTask is a ternary relationship between a video, verb, and evaluator. The video/verb pairing is associated with a random variant for each of the 10 exemplars. The RoundTableTask's completionDate and observed attributes are null until the evaluator has completed the task. The observed attribute is a bit where a value of 1 indicates the evaluator did recognize the verb in the video and a value of 0 indicates they did not.

3.2.2.2 User Interface

The GUI for the Round Table Task (figure 10) consists of a video viewing system, complete with the ability to pause and replay the video, and user response check boxes. The directions for the task instruct the evaluator to watch the video and select all of the verbs they witness in the video. The evaluator indicates their response by selecting each of the checkboxes corresponding to the verbs observed in the video. To assist in the task, the GUI also provides the evaluator with the definition for each of the verbs. The evaluator can access each definition by hovering the mouse pointer over a verb. The definition is provided to remove any ambiguity in the

circumstance where a verb might have multiple definitions or uses. After selecting all the verbs that describe the video, the evaluator clicks the “Submit” button and the information is committed to the database. The GUI then accesses the database to randomly select the next video for this evaluator from a set of previously assigned candidate videos. This video automatically begins playing in the video screen on the GUI and all checkboxes are cleared for the new task. If the evaluator believes that this video would be a good choice for the Description Task and/or the Gap Filling Task (described in sections 3.3 and 3.4, respectively), the evaluator may check the corresponding checkbox in the bottom right-hand corner of the GUI prior to clicking the “Submit” button. If the evaluator believes that this video is not suitable for any of the other tasks, the evaluator may check the checkbox corresponding to “Vignette not suitable” prior to clicking the “Submit” button. Allowing the users to provide these data was meant as a time-saving endeavor to obtain video feedback without an additional study. The “Skip Video” button allows the evaluator to move on to a different video exercise without committing any data to the database and allowing the user to complete this video exercise at a later time. The GUI also denotes how many remaining video tasks remain for the current evaluator to give the user an indication of the amount of time remaining for the exercise. Once the evaluator has completed all their designated tasks, a window is displayed informing the evaluator that all the tasks have been completed. The evaluator can then exit the Round Table Task using the “Exit Recognition Task” button or the X in the upper right-hand corner of the GUI. The algorithm for randomly selecting the specific Round Table tasks for a specific evaluator is shown in listing 5.

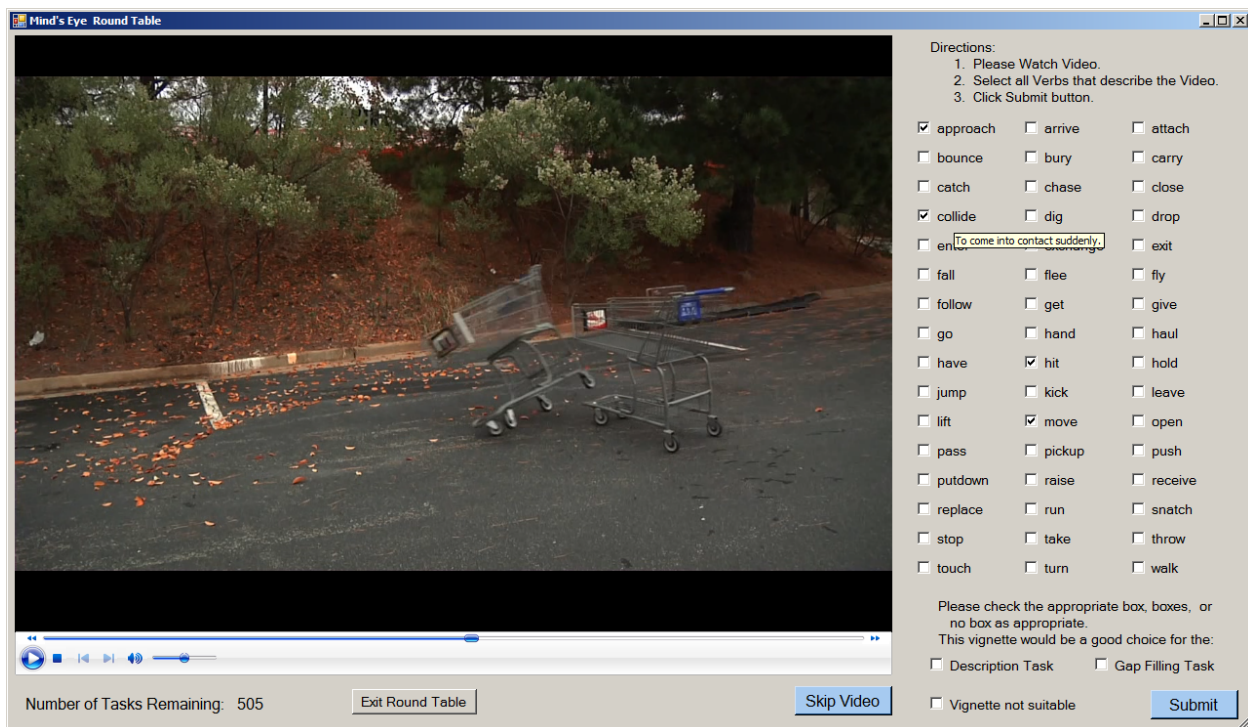


Figure 10. Round Table Task GUI.

```

RoundTableTasks = the database table storing the Round Table task assignments
User = the id of the currently logged on evaluator
GUI = the user interface
Tasks := RoundTableTasks.Select("evaluatorID = User  $\wedge$  completionDate  $\neq \emptyset$ ")
while Tasks  $\neq \emptyset$  do
    | Task := Random(Tasks) // select a random tuple
    | GUI.Show(Figure 10 Round Table Task GUI)
    | Task.Update(GUI.result(), date())
    | Tasks := Tasks - task
    | GUI.Update(|Tasks|) // show number of tasks remaining
    | Tasks := RoundTableTasks.Select("evaluatorID = User  $\wedge$  completionDate  $\neq \emptyset$ ")
| GUI.Show("You're all done")

```

Listing 5. Round Table Task algorithm.

3.2.2.3 Results and Discussion

The results for the Round Table Task are shown in figure 11 and table 4. The average precision was 0.156 ($\sigma = 0.06$), the average recall was 0.74 ($\sigma = 0.20$) and the average F -score was 0.26 ($\sigma = 0.09$). To compare the results of the Recognition Task vs. the Round Table task, we only looked at the 24 intended actions used for the Recognition Task (figure 12 and table 5). The average precision increase in the Round Table Task ($\mu = 0.17$, $\sigma = 0.04$) versus the Recognition Task ($\mu = 0.13$, $\sigma = 0.05$) was significant ($p = 0.006 < 0.05$). Likewise, the average increase in the Round Table Task F -score ($\mu = 0.28$, $\sigma = 0.07$) versus the Recognition Task ($\mu = 0.22$, $\sigma = 0.07$) was also significant ($p = 0.008 < 0.05$) (figure 14 and table 7). The drop in average recall of the Round Table Task ($\mu = 0.82$, $\sigma = 0.13$) versus the Recognition Task ($\mu = 0.89$, $\sigma = 0.16$), however, was not significant ($p = 0.011 > 0.05$) (figure 13 and table 6). The increase in the average precision and F -score of the Round Table Task vs. the Recognition Task may be attributable to the fact that when the evaluator was presented with all possible actions for a particular video, the intended action became more apparent in comparison to the other verbs. The fact that the average recall value did not change might be due to the fact that the recall scores were already high. This result indicates that the videos did in fact capture their intended action, but the low precision scores indicate they did not capture it unambiguously. Details on how these results were used to discount outlying Amazon Turk workers are discussed in section 4.

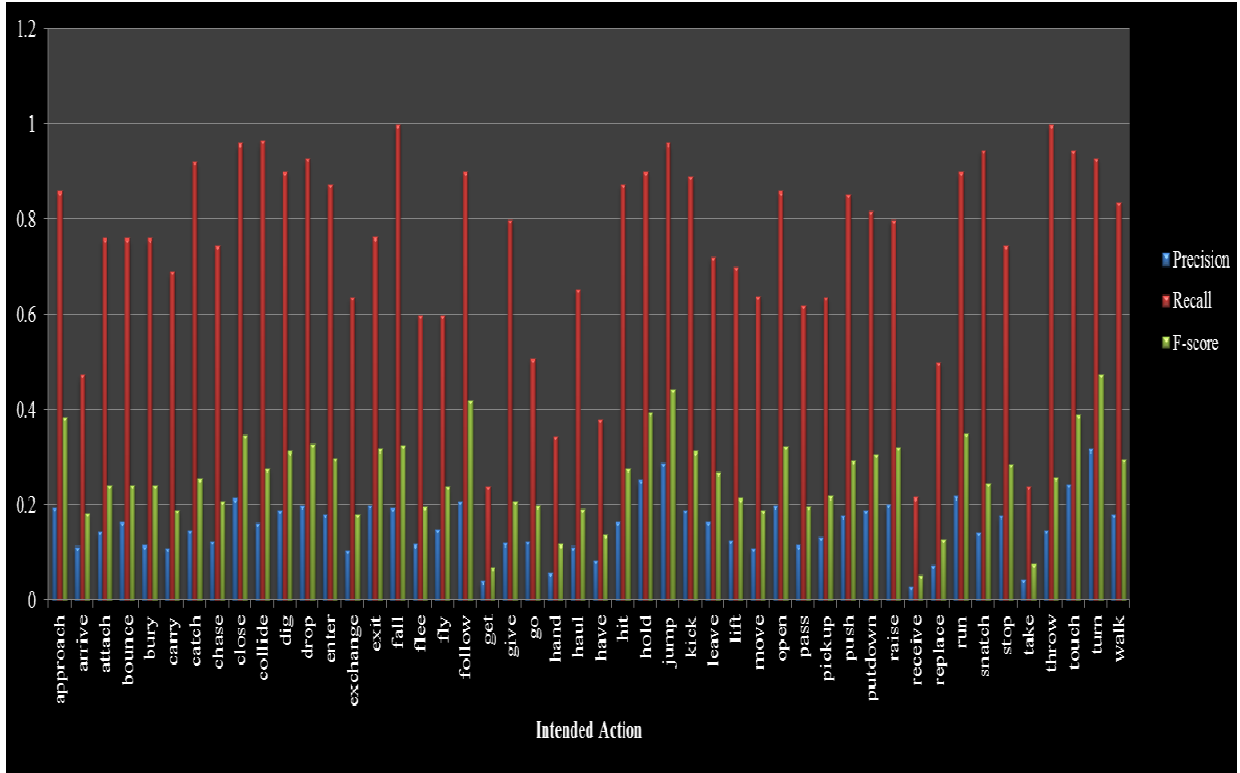


Figure 11. Precision, recall, and F -score for the Round Table Task.

Table 4. Precision, recall, and F -score for the Round Table Task.

Intended Action	Precision	Recall	F -score
approach	0.194	0.860	0.386
arrive	0.114	0.473	0.183
attach	0.144	0.760	0.243
bounce	0.164	0.760	0.270
bury	0.115	0.760	0.191
carry	0.111	0.691	0.191
catch	0.148	0.920	0.256
chase	0.122	0.745	0.210
close	0.213	0.960	0.349
collide	0.162	0.964	0.277
dig	0.190	0.900	0.314
drop	0.200	0.927	0.329
enter	0.180	0.873	0.298
exchange	0.105	0.636	0.180
exit	0.201	0.763	0.318
fall	0.194	1.00	0.324
flee	0.118	0.600	0.197
fly	0.149	0.600	0.239
follow	0.210	0.900	0.420
get	0.041	0.240	0.070
give	0.121	0.800	0.210
go	0.124	0.509	0.199
hand	0.059	0.345	0.118

Table 4. Precision, recall, and F -score for the Round Table Task (continued).

Intended Action	Precision	Recall	F -score
haul	0.114	0.655	0.194
have	0.084	0.380	0.138
hit	0.164	0.873	0.277
hold	0.253	0.900	0.395
jump	0.289	0.960	0.444
kick	0.190	0.891	0.313
leave	0.166	0.720	0.270
lift	0.126	0.700	0.213
move	0.111	0.640	0.189
open	0.199	0.860	0.323
pass	0.117	0.620	0.196
pickup	0.132	0.636	0.219
push	0.178	0.854	0.295
putdown	0.190	0.818	0.308
raise	0.201	0.800	0.321
receive	0.030	0.218	0.053
replace	0.073	0.500	0.127
run	0.218	0.900	0.352
snatch	0.142	0.945	0.247
stop	0.177	0.745	0.286
take	0.045	0.240	0.075
throw	0.149	1.00	0.259
touch	0.245	0.945	0.390
turn	0.319	0.927	0.474
walk	0.179	0.836	0.295
μ	0.155 ($\sigma=0.06$)	0.740 ($\sigma=0.21$)	0.260 ($\sigma=0.09$)

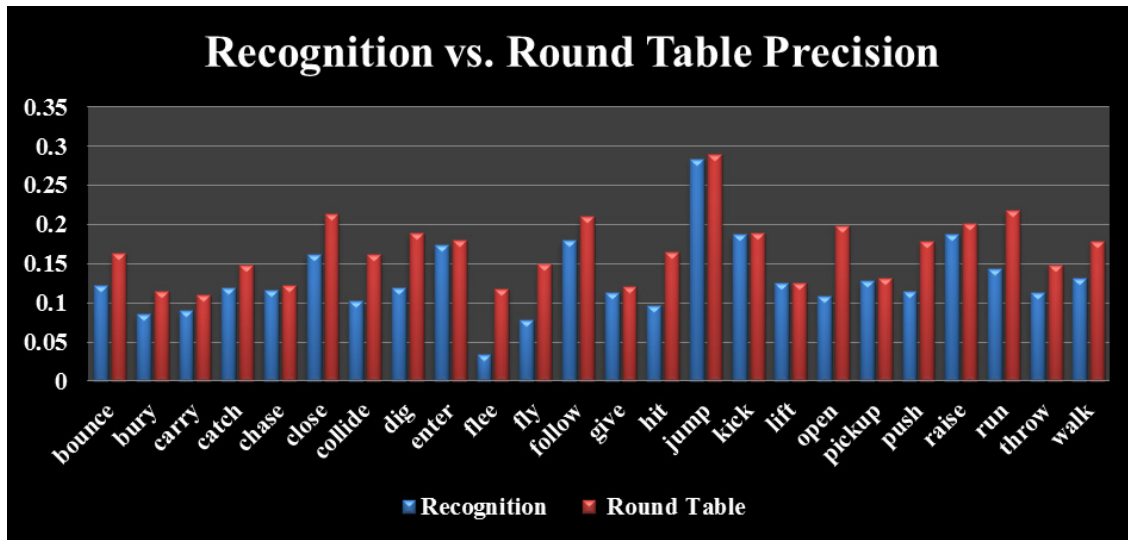


Figure 12. Recognition Task vs. Round Table Task precision.

Table 5. Recognition Task vs. Round Table Task precision.

Intended Action	Recognition	Round Table
bounce	0.122	0.164
bury	0.085	0.115
carry	0.090	0.111
catch	0.119	0.148
chase	0.116	0.122
close	0.163	0.213
collide	0.102	0.1617
dig	0.120	0.190
enter	0.174	0.180
flee	0.034	0.118
fly	0.078	0.149
follow	0.181	0.210
give	0.113	0.121
hit	0.097	0.164
jump	0.283	0.289
kick	0.188	0.190
lift	0.126	0.126
open	0.109	0.199
pickup	0.128	0.132
push	0.115	0.178
raise	0.188	0.201
run	0.143	0.218
throw	0.113	0.149
walk	0.132	0.179

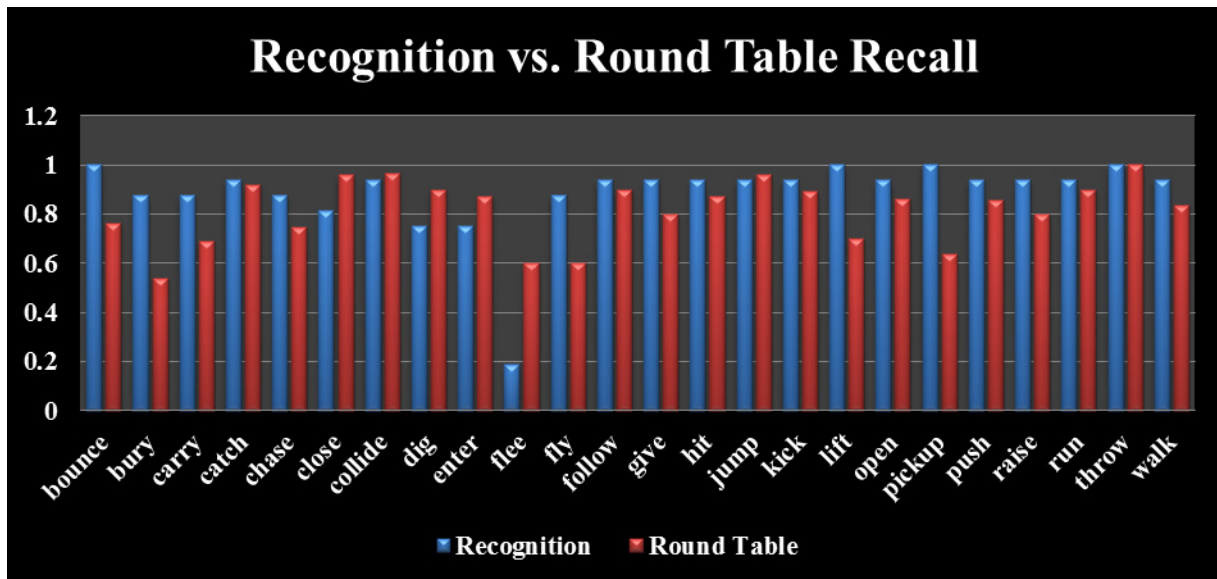


Figure 13. Recognition Task vs. Round Table Task recall.

Table 6. Recognition Task vs. Round Table Task recall.

Intended Action	Recognition	Round Table
bounce	1.00	0.764
bury	0.875	0.540
carry	0.875	0.691
catch	0.938	0.920
chase	0.875	0.745
close	0.813	0.960
collide	0.938	0.964
dig	0.750	0.900
enter	0.750	0.873
flee	0.188	0.600
fly	0.875	0.600
follow	0.938	0.900
give	0.938	0.800
hit	0.938	0.873
jump	0.938	0.960
kick	0.938	0.891
lift	1.00	0.700
open	0.938	0.860
pickup	1.00	0.636
push	0.938	0.855
raise	0.938	0.800
run	0.938	0.900
throw	1.00	1.00
walk	0.938	0.836

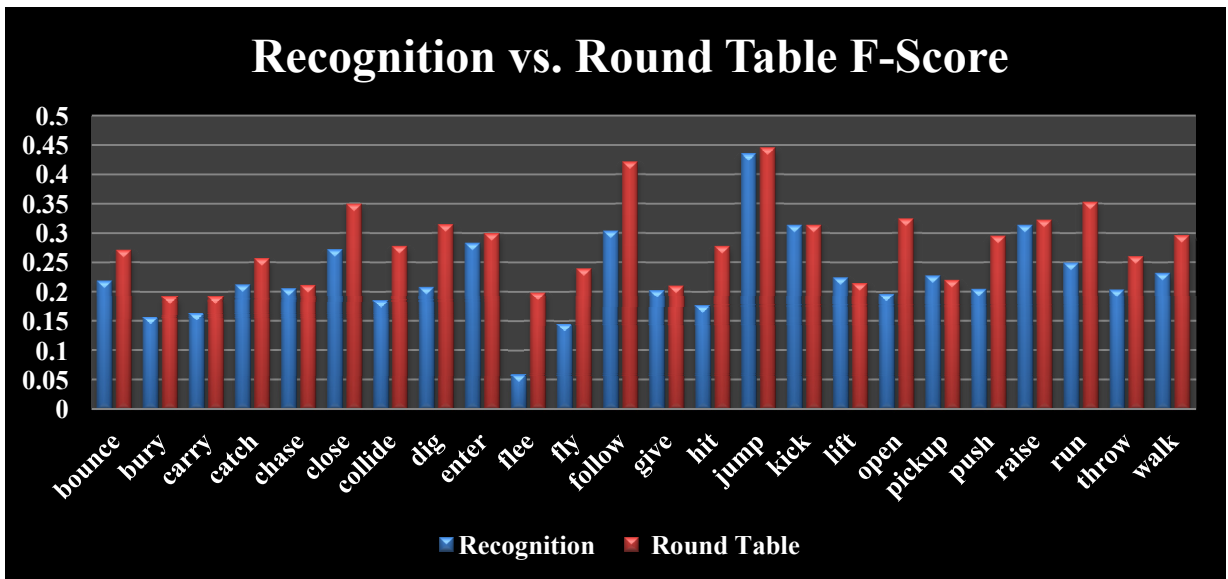


Figure 14. Recognition Task vs. Round Table Task *F*-score.

Table 7. Recognition Task vs. Round Table Task *F*-score.

Intended Action	Recognition	Round Table
bounce	0.218	0.270
bury	0.156	0.190
carry	0.163	0.191
catch	0.211	0.256
chase	0.204	0.210
close	0.271	0.349
collide	0.184	0.277
dig	0.207	0.314
enter	0.282	0.298
flee	0.058	0.197
fly	0.144	0.239
follow	0.303	0.421
give	0.201	0.209
hit	0.175	0.277
jump	0.435	0.444
kick	0.313	0.313
lift	0.224	0.213
open	0.195	0.323
pickup	0.227	0.219
push	0.204	0.295
raise	0.313	0.321
run	0.248	0.352
throw	0.203	0.259
walk	0.231	0.295

3.3 Description Task

3.3.1 Description

The Description Task consists of 480 hand-selected videos evaluated by 5 participants, each being tasked to provide a brief textual description (140 characters or less) of what they observed. The subjects were instructed to use the Mind’s Eye verbs to guide these descriptions. The intent of this exercise was to capture natural language descriptions for use in comparing to the Mind’s Eye automated systems natural language. Such a comparison was inspired by the “Turing Test” (Turing, 1950). The system randomly selected a video from a set of predefined videos for each evaluator. Once the evaluator had performed the task, these response data were committed to the database under the corresponding video and evaluator ID. The observed video never appeared again for this evaluator. The Description Task was designed to closely resemble the exercise presented as part of the Amazon Turk crowdsourcing data collection, as well as the task that will be performed by the automated Mind’s Eye systems. The use case diagram for the Description Task is shown in figure 15 and the entity-relationship diagram for storing the results in a relational database is shown in figure 16.

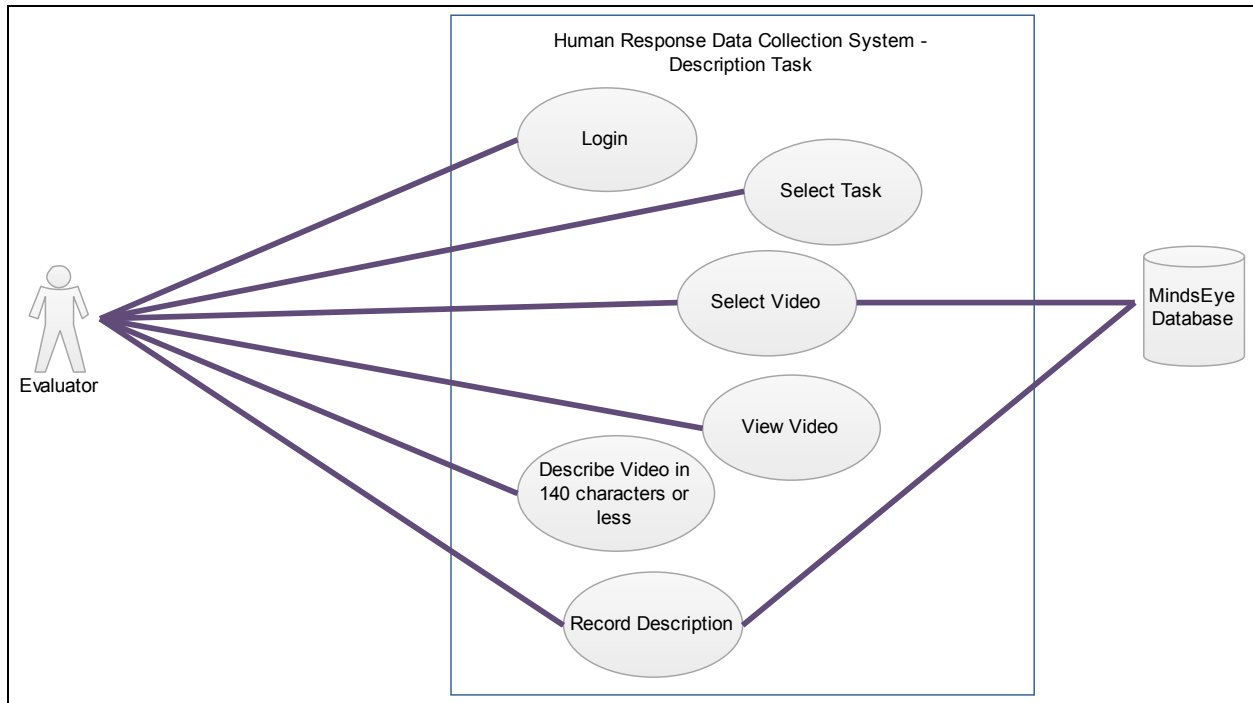


Figure 15. Description Task use cases.

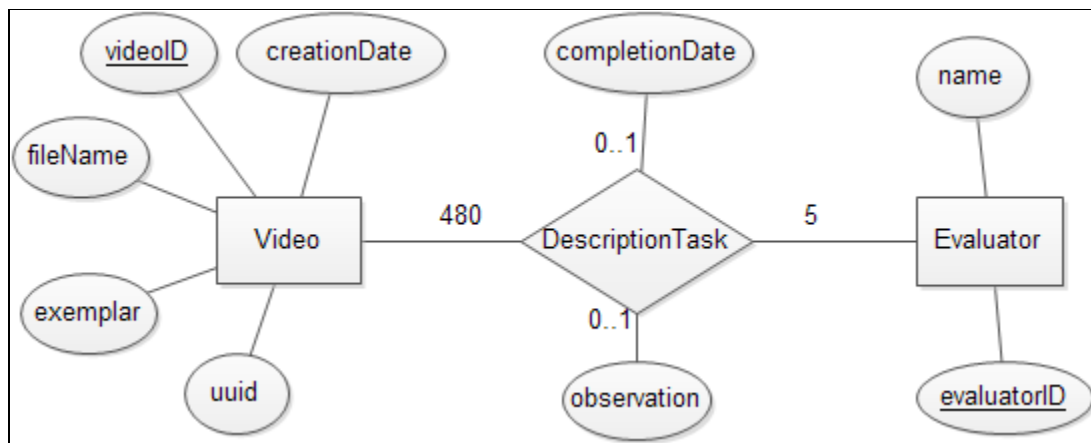


Figure 16. Description Task entity-relationship diagram.

The DescriptionTask is a binary relationship associating 480 pre-selected videos with each of 5 pre-selected evaluators. The DescriptionTask's completionDate and observed attributes are null until the evaluator has completed the task. The description attribute is a string representing the evaluator's summary of the action in the video.

3.3.2 User Interface

The GUI consists of a full feature video viewing system with pause and replay capabilities. The GUI also provides the participant with directions requesting that the evaluator describe what has occurred in the video using 140 characters or less while incorporating the Mind's Eye designated

verbs. The evaluator types the description into the incorporated GUI textbox. To assist the evaluator, the GUI provides definitions for each of the verbs listed in the Verb Box. These definitions can be accessed by hovering the mouse pointer over a verb. Once satisfied with their description of the video, the evaluator clicks the “Submit” button, committing the description to the database according to the video and evaluatorID. A random video assigned to this evaluator is then selected from the database and displayed by the GUI. This video automatically begins playing in the video screen the text box is cleared for the new task. The “Skip Video” button allows the evaluator to move on to a different video exercise without committing any data to the database, allowing the user to view the current video task at a later time. The GUI denotes how many video tasks remain for the current evaluator. Once the evaluator has completed all their designated tasks, a window is displayed informing the evaluator that all the tasks have been completed. The evaluator can then exit the Description Task using the “Exit Recognition Task” button or the X in the upper right-hand corner of the GUI. The GUI for this task is shown in figure 17 and the algorithm is shown in listing 6.

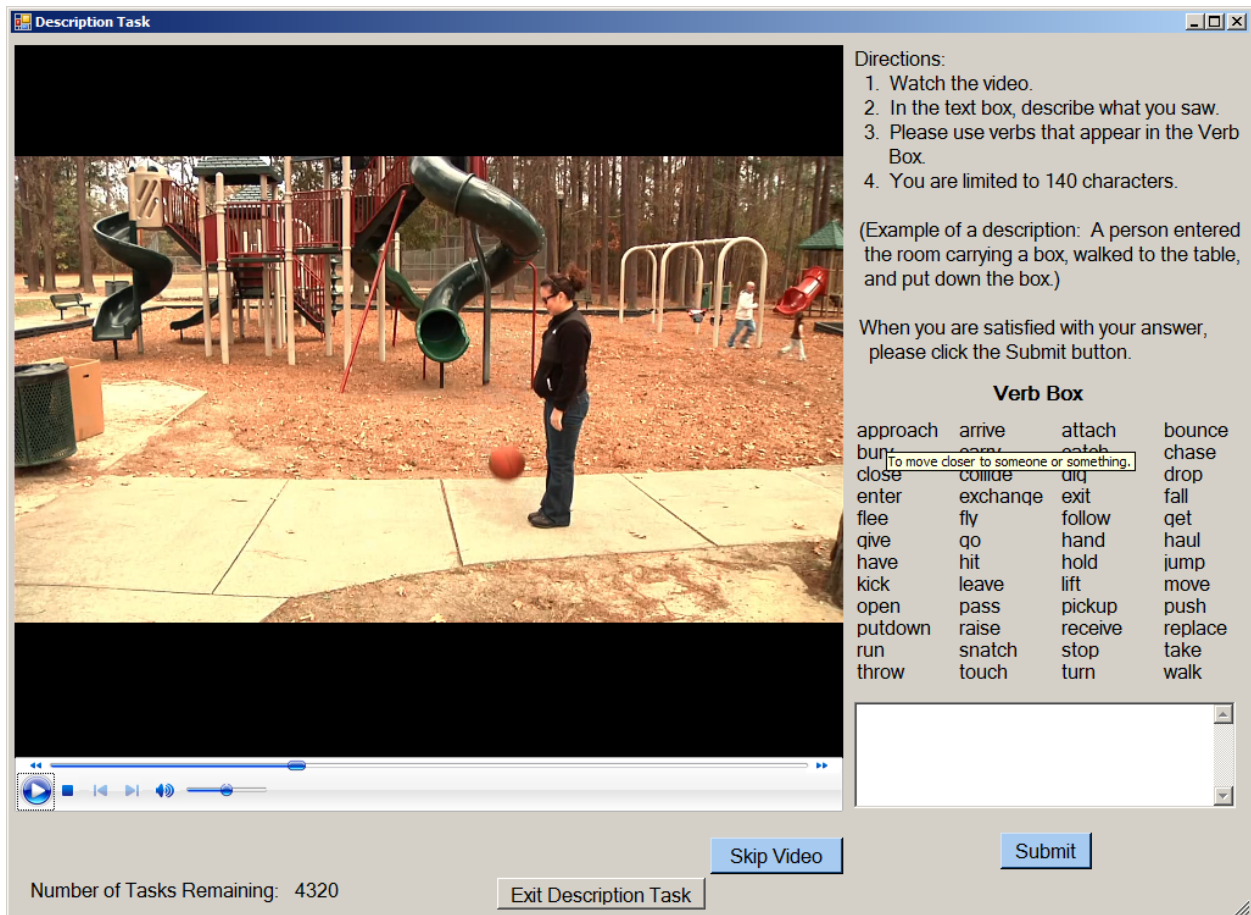


Figure 17. Description Task GUI.

```

DescriptionTasks = the database table storing the Description Task assignments
User = the id of the currently logged on evaluator
GUI = the user interface
Tasks := DescriptionTasks.Select("evaluatorID = User  $\wedge$  completionDate  $\neq \emptyset$ ")
while Tasks  $\neq \emptyset$  do
    | Task := Random(Tasks) // select a random tuple
    | GUI.Show(Figure 17: Description Task GUI)
    | Task.Update(GUI.result(), date())
    | Tasks := Tasks – task
    | GUI.Update(|Tasks|) // show number of tasks remaining
    | Tasks := DescriptionTasks.Select("evaluatorID = User  $\wedge$  completionDate  $\neq \emptyset$ ")
| GUI.Show("You're all done")

```

Listing 6. Description Task selection algorithm.

3.3.2 Results and Discussion

The analysis of the Description Task results involves comparing different natural language responses and is beyond the scope of this document. For a detailed analysis see Thomson (2012a) and Thompson (2012b).

3.4 Gap-Filling Task

3.4.1 Description

For the Gap-Filling Task, 120 hand-selected videos were evaluated by 5 participants. Each participant evaluated all 120 videos. The set of videos were drawn from sets used in the previous tasks and were edited to introduce a gap by overwriting a section of the video either at the beginning, middle, or the end with a blank segment 5 s long. Each of the participants was asked to view the same 120 videos and write a description of what they believe to have happened during the “gap” or blank portion of each video in 140 characters or less. The subjects were instructed to use the Mind’s Eye verbs to guide these descriptions. The use case diagram for the Gap-Filling Task is shown in figure 18 and the entity-relationship diagram used to store the results in a relational database is shown in figure 19.

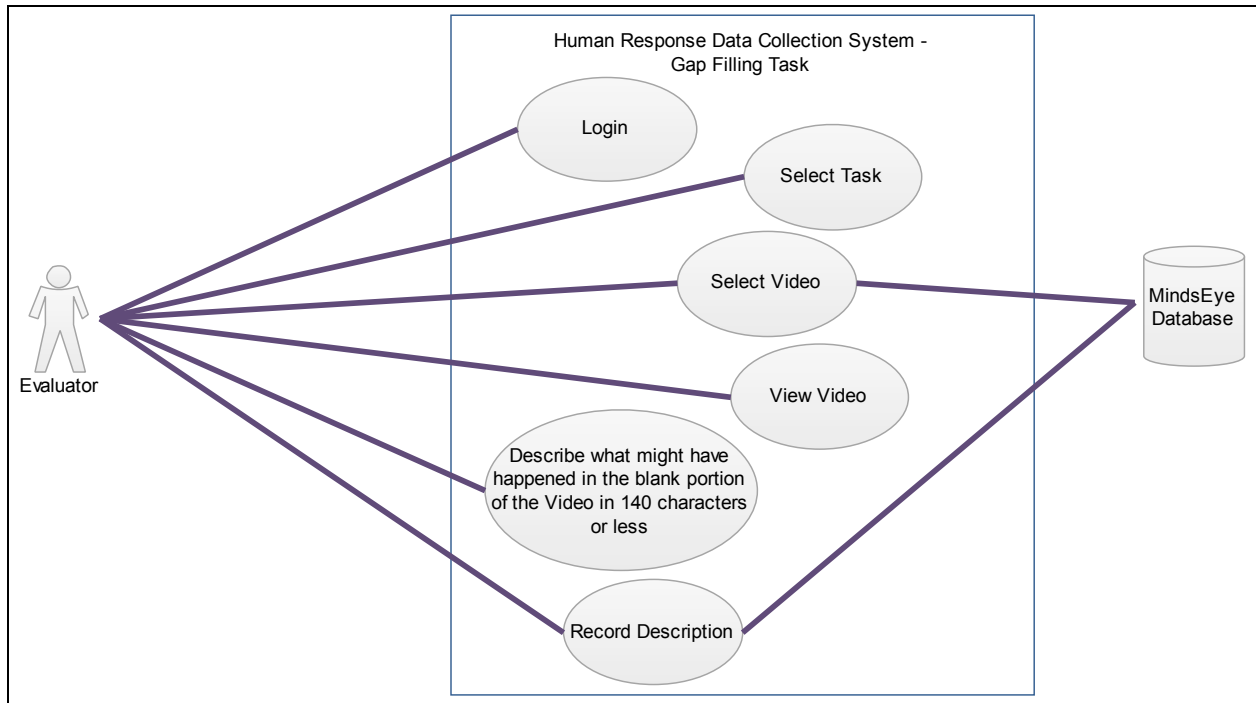


Figure 18. Gap-Filling Task use cases.

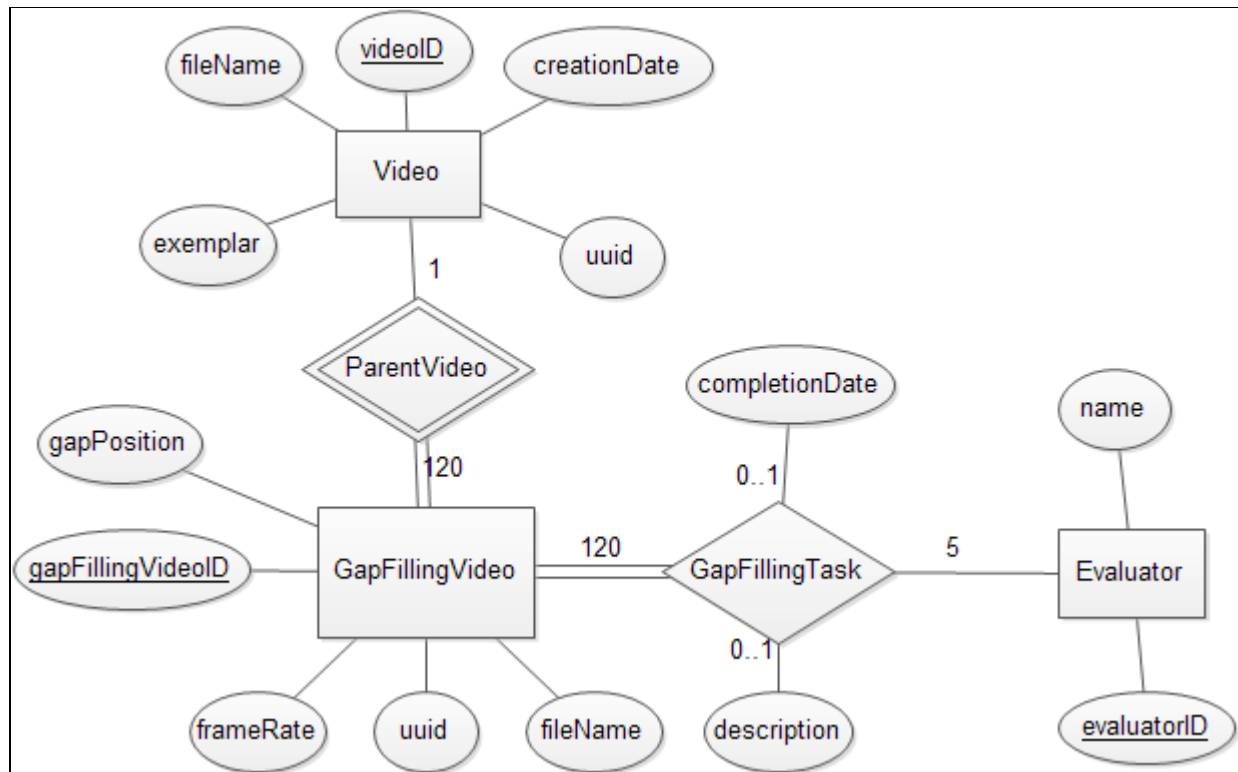


Figure 19. Gap-Filling Task entity-relationship diagram.

The GapFillingVideo is a weak entity dependent on a pre-selected video which has been edited to introduce a 5-s gap at the beginning, middle, or end of the parent video. A system-assigned integer primary key was introduced in the GapFillingVideo entity to avoid having to join on the string-valued unique filename or uuid. The gap position and the frame rate are preserved as attributes in the GapFillingVideo entity. The GapFillingTask element is a binary relationship between the 120 GapFillingVideos and 5 pre-selected evaluators. The completionDate and description attributes are null until the evaluator has completed the task. The description attribute is a string representing what the evaluator thinks happened during the gap.

3.4.2 User Interface

The GUI for this task is shown in figure 20. The GUI for the Gap-Filling Task is fundamentally the same as the Description Task except the video being viewed has a gap of 5 s introduced at the beginning, middle, or end of the video. The evaluator is asked to describe what they believe to have happened during the gap instead of describing the entire video. The system randomly selects a video from a set of predefined videos for each evaluator. Once the evaluator has described what is believed to have occurred during the gap in the video, this information is stored in the database for the corresponding video and evaluator ID. A different video task will then be presented to the evaluator. The algorithm for selecting the Gap-Filling Task videos for a specific user is shown in listing 7.



Figure 20. Gap-Filling Task GUI.

The algorithm for selecting the video task is as follows:

```

GapFillingTasks = the database table storing the Gap-Filling Task assignments
User = the ID of the currently logged on evaluator
GUI = the user interface
Tasks := GapFillingTasks.Select("evaluatorID = User ∧ completionDate ≠ ∅")
while Tasks ≠ ∅ do
    Task := Random(Tasks) // select a random tuple
    GUI.Show(Figure 20: Gap-Filling Task GUI)
    Task.Update(GUI.result(), date())
    Tasks := Tasks − task
    GUI.Update(|Tasks|) // show number of tasks remaining
    Tasks := GapFillingTasks.Select("evaluatorID = User ∧ completionDate ≠ ∅")
GUI.Show("You're all done")
  
```

Listing 7. Gap-Filling Task algorithm.

3.3.2 Results and Discussion

The analysis of the Gap-Filling Task results involves comparing different natural language responses and is beyond the scope of this report. As far as we are aware, no in-depth analysis of the results of the Gap-Filling Task has been performed to date.

4. Amazon Mechanical Turk Human Response Collection

The Mind's Eye Project required a large amount of data from human sources in order to have comprehensive coverage of all videos and verbs. While substantial data were collected internally from U.S. Army Research Laboratory (ARL) test subjects (see section 3), it was decided that comprehensive data needed to be collected from other sources. Some reasons behind this decision included considerations that ARL test subjects have similar educational backgrounds and professional experiences in addition to having knowledge of the project goals. This new data gathering effort endeavored to collect data from a wider and more diverse population in order to be considered ground-truth data for the automated systems that are the overall goal of the Mind's Eye Project.

Amazon Mechanical Turk (AMT) was identified as a viable medium through which to collect these data. AMT is a service offered by Amazon that allows for users, known as Requesters, to solicit data from other users, known as Workers. Typically, Requesters create simple tasks or questionnaires, called Human Intelligence Tasks (HITs), which Workers will volunteer to complete for a small monetary compensation. Each HIT has a number of Assignments, designated by the Requester, which indicates the number of unique Requesters that can respond to a specific HIT.

There are a number of advantages to using this service. Of primary importance, AMT allows for the collection of a very large data set within a relatively short time frame. Because Workers are only paid a small sum per question, data collection is also relatively inexpensive. AMT also provides convenient methods for creating a large number of HITs to account for the large amount of data required by the Mind's Eye project. AMT defines HITs, including the number of assignments and how much a Worker can earn, through extensive markup language (XML) code. Because XML strings can be formed and transmitted to AMT within C# code using AMT libraries, forming a large number of HITs is possible within a single program.

One of the unique aspects in using AMT is the relationship between the Requester and the Workers. While it is important for Workers to be diligent and accurate in their tasks, there are also expectations that the Requester must meet or risk losing Workers to other tasks and having work requests unfulfilled. To this end, Workers and Requesters are encouraged to communicate through e-mail. In addition, a third party Web site known as Turker Nation allowed for an open forum where Workers and Requesters could communicate. Through these mediums, Workers

could give feedback to Requesters in order to make their HITs more attractive or speed up data collections. Requesters can query Workers as to reward amounts and other potential issues.

4.1 HIT Creation

The HITs for this study were created using a C# program. The technical aspects of the program largely involve interacting with a relational database and forming the question XML strings. Because each HIT is identical in structure, varying only in the specific videos and question verbs, a template was created to be used for each HIT XML string. The template employs keyword tags such as “[url]” in place of the question specific string such as the exact uniform resource locator (URL) for a specific video file. The program properly formats each HIT for this task through a query to the database to obtain video and verb pairing information and replaces the tags in the XML template through a substring replacement call using the C# string library. The example code the follows (listing 8) demonstrates a portion of the XML template with keyword tags. This example illustrates how the unique keyword tags are embedded into the XML string and can subsequently be located and replaced using common string manipulation techniques. Multiple question XML strings can then be concatenated to form a complex HIT with more than one question. Listing 8 creates a single question and demonstrates the usage of the keyword tags that can be replaced with specific values in order to form a unique question. The entire AMT XML QuestionForm.xsd schema may be found at <http://mechanicalturk.amazonaws.com/AWSMechanicalTurkDataSchemas/2005-10-01/QuestionForm.xsd> .

```

<Question>
  <QuestionIdentifier>Q_[identifier]</QuestionIdentifier>
  <IsRequired>true</IsRequired>
  <QuestionContent>
    <FormattedContent>
      <![CDATA[
        <p>Definition of <b>[verb]</b>:<br></br>[definition]</p>
        Do you see <b>[verb]</b> in the above video?
      ]]>
    </FormattedContent>
  </QuestionContent>
  <AnswerSpecification>
    <SelectionAnswer>
      <StyleSuggestion>radiobutton</StyleSuggestion>
      <Selections>
        <Selection>
          <SelectionIdentifier>A_[identifier]_Y</SelectionIdentifier>
          <Text>Yes</Text>
        </Selection>
        <Selection>
          <SelectionIdentifier>A_[identifier]_N</SelectionIdentifier>
          <Text>No</Text>
        </Selection>
      </Selections>
    </SelectionAnswer>
  </AnswerSpecification>
</Question>

```

Listing 8. AMT HIT XML for a single question.

The AMT XML schema allows for blocks of extensible hypertext markup language (XHTML) code within the FormattedContent XML element (listing 9). This XHTML is similar to hypertext markup language (HTML) used in typical Web development; however, the FormattedContent tag supports only a limited subset of XHTML. Notably, it does not support JavaScript. However, JavaScript was needed to display the vignettes to the workers. In order to embed this functionality within the HIT, an inline frame (iframe) was used to load an external HTML file hosted on Amazon Simple Storage Service (S3).

```

<Overview>
<FormattedContent>
  <![CDATA[
    <p><b>[Video]:</b></p>
    <iframe src="[url]video.html?video='[movie]'"
      width="1000" height="580">
      If you can see this, your browser does not support IFRAME
      and you should not respond to this HIT!
    </iframe><br />
  ]]>
</FormattedContent>
</Overview>

```

Listing 9. FormattedContent XML example.

This external HTML page is able to use JavaScript because it is a separate page that does not need to parse according to the AMT XML schema. Instead, this page is free to use JavaScript to embed JW Player (<http://www.longtailvideo.com/jw-player/>), a Flash-based streaming video player. In order to load the appropriate vignette, the iframe code in the XML passes a variable to the external HTML file via a URL variable. In the code block above, the iframe source page contains two tags, “[url]” and “[movie]”. The [url] tag is replaced with the root Amazon S3 location where the video loading html page is located, and the [movie] tag is the specific filename of the vignette that is to be loaded. The HTML file (listing 10) can parse the filename from the URL using JavaScript string manipulation.

```

<SCRIPT LANGUAGE="JavaScript">
    urlstring = window.location.search
    urlstring = unescape(urlstring)
    videoIndex = urlstring.indexOf("video")
    video = urlstring.substring(videoIndex+7,urlstring.indexOf("'",videoIndex+7))
    video = "http://s3.amazonaws.com/ARL_ME/videos/"+video

    document.write("
        <object classid=\"clsid:D27CDB6E-AE6D-11cf-96B8-444553540000\"
            width=\"960\" height=\"540\" id=\"player1\" name=\"player\">
            <param name=\"movie\" value=\"jwplayer/player.swf\">
            <param name=\"allowfullscreen\" value=\"true\">
            <param name=\"allowscriptaccess\" value=\"always\">
            <param name=\"flashvars\" value=\"file="+video+"&autostart=false\">
            <embed id=\"player1\" name=\"player1\" src=\"jwplayer/player.swf\"
                width=\"960\" height=\"540\" allowscriptaccess=\"always\"
                allowfullscreen=\"true\" flashvars=\"file="+video+"&autostart=false\"/>
        </object>
        <br></br>")
</SCRIPT>

```

Listing 10. Video.html.

Listing 10 is the video.html file located on the Amazon S3 server. The JavaScript first parses the URL string and assigns appropriate variable values. The script then embeds the JW Player using the variable values. JW Player creates a simple user interface including play/pause button and movie timeline. This interface is similar to that used by YouTube, providing a familiar interface for the workers.

Once verified, the XML strings were uploaded to the AMT service using Amazon C# libraries over an HTTPS connection. These HITs are immediately published and made available for Workers to accept and complete. Workers were permitted to respond to any number of HITs, although only one assignment per HIT is allowed. Allowing Workers to respond to an unlimited number of HITs not only expedites the data gathering process, but is a necessity when considering the large amount of data points necessary for the experiment. As Workers complete the HITs, they become familiar with the instructions and procedure. Because the HITs are all structured in identical fashion (differing only in videos shown and verbs asked), Workers can complete subsequent HITs without reading the instructions or refamiliarizing themselves with the HIT procedure. This encourages Workers to respond to multiple HITs and allows them to be more efficient. Because it was important to collect the data within a specific amount of time, allowing Workers to continuously and rapidly respond to HITs was a design goal.

Obtaining the results, verifying the integrity, and rewarding the Workers is a similar process to creating and publishing HITs. Results for a given HIT assignment are obtained by querying the AMT servers on a specific HIT identification string. The AMT service returns an array of assignment result structures, each of which contains an XML string. The XML string can then be parsed according to the schema provided within the AMT libraries, and the response data can be extracted, organized, and stored in the same relational database used for the ARL internal Human Collection Tasks. The results are then compared against the internal ARL baseline ground-truth data to make a determination as to the accuracy of the Worker's performance. If this performance exceeded a minimum threshold, a HIT response was categorized as acceptable and the AMT servers can be called with the appropriate response identification string so that the assignment can be accepted and the Worker rewarded. However, in the event that the performance did not meet this minimum threshold, the assignment can be rejected and the Worker is not rewarded.

4.2 AMT Recognition Task

The first task implemented on AMT was the recognition task. The task called for collection of data from every video-verb pairing. This is identical to the Recognition Task described in section 3.2.1, with the exception that there was no subsampling. All video-verb pairings were sampled. Each video-verb pairing consists of a video being displayed and a true/false question being asked if the verb was demonstrated in the video. With 7676 stimulus videos and 48 verbs, the data pairings totaled 368,448 questions. With such a large number of data points to be collected, the AMT HITs designed for this task divided the total into sets of 18 distinct video-verb pairings. By putting multiple questions into a single HIT, it is easier for Workers to respond, reducing the amount of overhead produced by selecting and accepting tasks to answer.

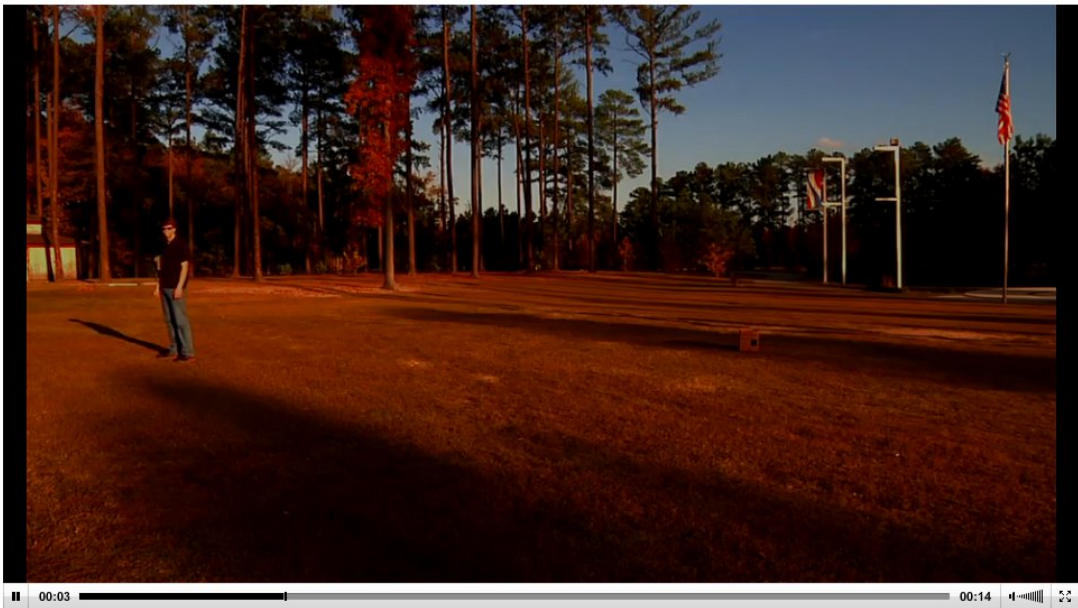
Included in each HIT were two "gold standard" questions. Each of these video-verb questions were selected based on data collected internally from ARL personnel. The specific video-verb pairings were specifically targeted as achieving consensus or near consensus results from the ARL responses, thus they had an expected "true" or "false" response of very high confidence. Because each gold standard question achieved consensus or near consensus results among ARL responses, it was expected that the video-verb pairing were unambiguous in their expected results. One gold standard question expecting a true response and one gold standard question expecting a false response were included in each HIT for a total of 20 video-verb pairing questions. The purpose of these check questions were to create a minimum performance standard that the Workers must meet in order for their response to be considered acceptable. If a particular Worker's response failed to reconcile with both check questions, the response was rejected and the Worker was not awarded a monetary reward.

The two "gold standard" questions with the 18 data collection questions created HITs with 20 different video-verb pairing questions. Figure 21 depicts the beginning of a sample HIT as it would be displayed on AMT through a Web browser such as Internet Explorer or Mozilla

Firefox. The instructions are presented at the top of the HIT page, followed by 20 videos, each with a corresponding yes or no question relating to a specific verb. The videos are displayed using JW Player with an interface similar to many popular Internet video streaming services such as YouTube. After all 20 yes or no questions are answered, the Worker could use the submit button at the end of the HIT to lock in the responses.

Identify the action in these short videos
Guidelines:

- View the short video from start to finish.
- Decide if the indicated action (according to the given definition) is taking place in the video.
- Please view and answer all 20 videos and questions.
- Note: a small number of hits will have fewer than 20 videos

Video 1 of 20:


A video player interface showing a scene of a person standing on a grassy field. The video player has a progress bar at the bottom indicating 00:03 out of 00:14. The scene shows a person standing on a grassy field with trees in the background. The video player interface includes a play button, a progress bar, and a volume icon.


Definition of **move**:

To change the position or location of something.

Do you see **move** in the above video?

☐ Yes

☐ No

Video 2 of 20:


A video player interface showing a scene of a person standing next to a table in front of a building. The video player has a progress bar at the bottom indicating 00:03 out of 00:14. The scene shows a person standing next to a table in front of a building. The video player interface includes a play button, a progress bar, and a volume icon.

Figure 21. AMT Recognition Task.

Through e-mail and forum posts on the Turker Nation Web page, Workers gave very positive feedback to the recognition task HITs. They felt that the offer of \$0.50 per HIT response was a very fair reward amount and that the HIT design was both fun and interesting. For these reasons, the HITs were quickly answered by the community and the goal of obtaining data at a rapid pace was successfully achieved.

However, the reward offer also attracted some negative aspects. Because AMT offers monetary rewards for online work, malicious Workers can disrupt data collection by responding with fraudulent answers. Using automated scripts or similar methods, a malicious Worker can scam a Requester by quickly responding to a large number of HITs using random or otherwise inaccurate answers. To combat this problem, the “gold standard” check questions were included in each HIT in addition to a minimum Worker rating. A Worker was required to have a 95% approval rating from previous AMT work.

Figure 22 shows the entity-relationship diagram used to design the relational database used to store AMT Recognition Task assignments and results.

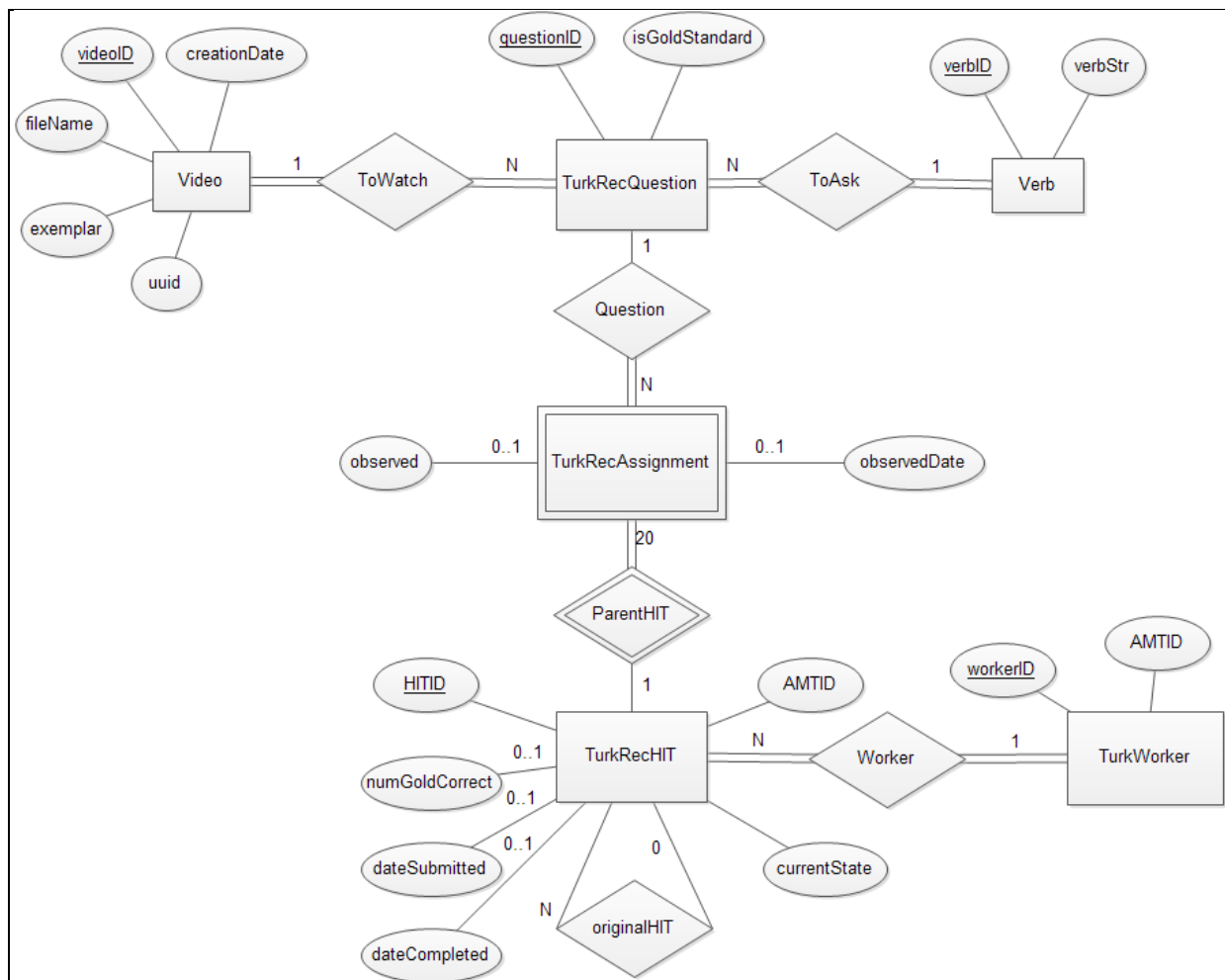


Figure 22. AMT Recognition Task entity-relationship diagram.

Each TurkRecHIT entity is associated with 20 TurkRecAssignment weak entities, which consist of a TurkRecQuestion, which is a pairing of a video to watch and a verb to ask. The TurkRecQuestion was modeled as a separate entity instead of a binary relationship so that it could be reused by more than one Turk worker. A system-assigned primary key is introduced to each TurkRecQuestion to avoid having to join on the natural composite primary key of videoID and verbID. An additional attribute isGoldStandard is a bit-valued attribute, where a value of 1 indicates that the question represents a “gold standard” as described and a value of 0 indicates it is not. The observed and observedDate attributes are null until the Turk worker associated with the parent HIT has answered the question as to whether or not they recognized the verb in the video. A system-assigned integer primary key was introduced for the TurkRecHIT and TurkWorker entities to avoid having to join on the natural unique string identifiers provided by AMT. Once a TurkWorker has accepted a TurkRecHIT, the dateSubmitted attribute is recorded. Once all associated 20 TurkRecQuestions have been answered by the TurkWorker the TurkRecHIT is complete and the completion date are recorded along with the number of correct “gold standard” responses. If the number of correct “gold standard” responses is below the threshold described above, a new identical copy of the HIT is created and the original HIT is recorded. The currentState attribute indicates whether the HIT is unassigned, pending, or completed.

4.2.1 AMT Recognition Task Results and Discussion

The results for precision, recall and F -score for the Amazon Turk Recognition Task are shown in figure 23. The average precision was 0.12 ($\sigma = 0.10$), average recall was 0.79 ($\sigma = 0.18$), and the average F -score was 0.19 ($\sigma = 0.14$). To compare the results of the AMT Recognition Task vs. the in-house Recognition Task used to create the “gold standard” questions, we only looked at the 24 intended actions used for the in-house Recognition Task (figure 12 and table 5). None of the differences for precision, recall, or F -score were significant (table 8). This is not surprising given that the in-house Recognition Task results were used as a criterion for accepting a HIT.

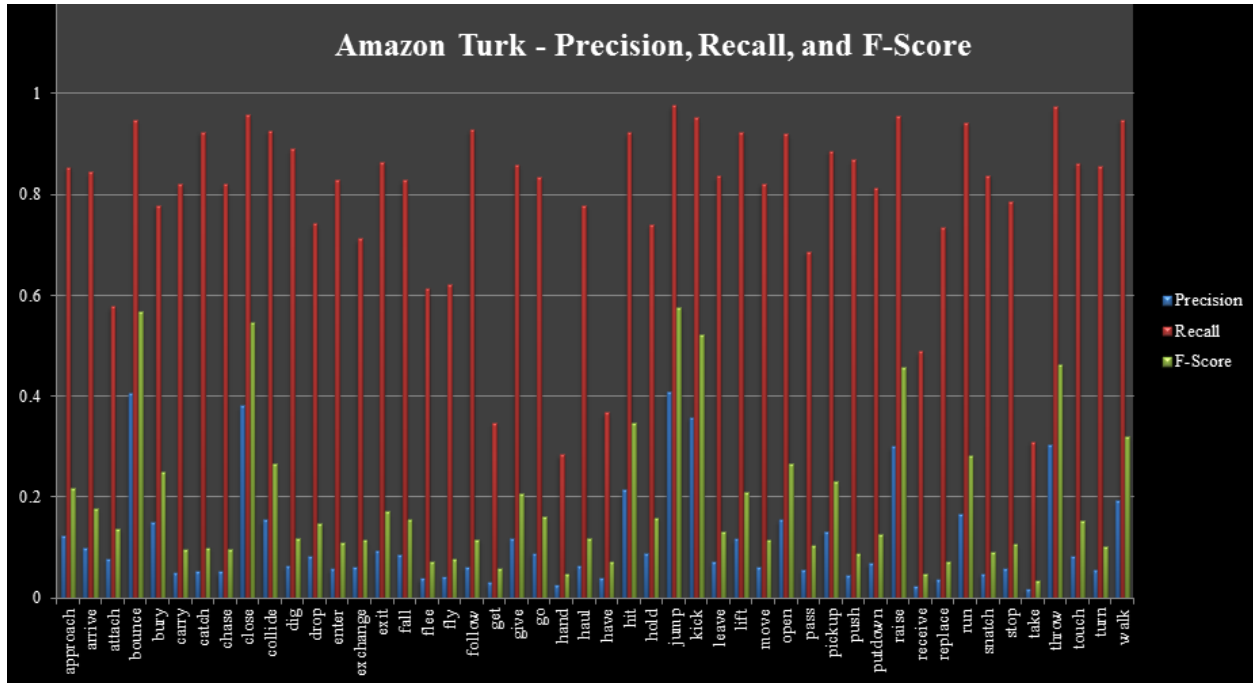


Figure 23. AMT Recognition Task: precision, recall, and F -score.

Table 8. In-house vs. AMT Recognition Task.

	In-House	AMT	P-Value
Precision	$\mu = 0.13, \sigma = 0.05$	$\mu = 0.17, \sigma = 0.12$	$0.18 > 0.05$
Recall	$\mu = 0.89, \sigma = 0.16$	$\mu = 0.88, \sigma = 0.10$	$0.94 > 0.05$
F -Score	$\mu = 0.22, \sigma = 0.07$	$\mu = 0.27, \sigma = 0.17$	$0.28 > 0.05$

4.3 AMT Description and Gap-Filling Tasks

The Description and Gap-Filling tasks were closely associated and developed in parallel because they are similar in structure and data collection. The Description Task displays a short stimulus video and asks the subject to type their interpretation of what happened in the video. Similarly, the Gap-Filling Task displays a stimulus video, which contains a 5-s segment that is blacked out. The subject is asked to describe what they suspect happened during the blank segment of the video. The description task consisted of 480 vignettes, specifically selected from the vignette collection to demonstrate typical and clear usage of the specific verbs. Each vignette was used in a single HIT and each HIT was created with 10 assignments. For a given hit, Workers were instructed to provide a description of the video using 140 characters or less while using a list of actions as a guideline. Workers were also provided with a sample description unrelated to the video. Instruction was deliberately kept as minimal as possible in order to avoid possible biasing. A sample HIT is shown in figure 24. The entity-relationship diagram used to store the AMT Description Task results is shown in figure 25.

Describe what is occurring in this short video

Guidelines:

- View the short video from start to finish.
- Review the list of actions and definitions via the url link.
- In the text box provided, describe what has occurred in the video (use the list of actions as a guideline).
- Note if you are doing multiple HITs: the list of actions is always the same.

Video:



Actions and Definitions

Please describe the above video using the list of actions as a guideline:

Limit your response to 140 characters (approximately 2 sentences):

Sample Response: A person entered the room carrying a box, walked to the table, and put down the box.

A large empty text box for the user to enter their description of the video.

Figure 24. AMT Description Task.

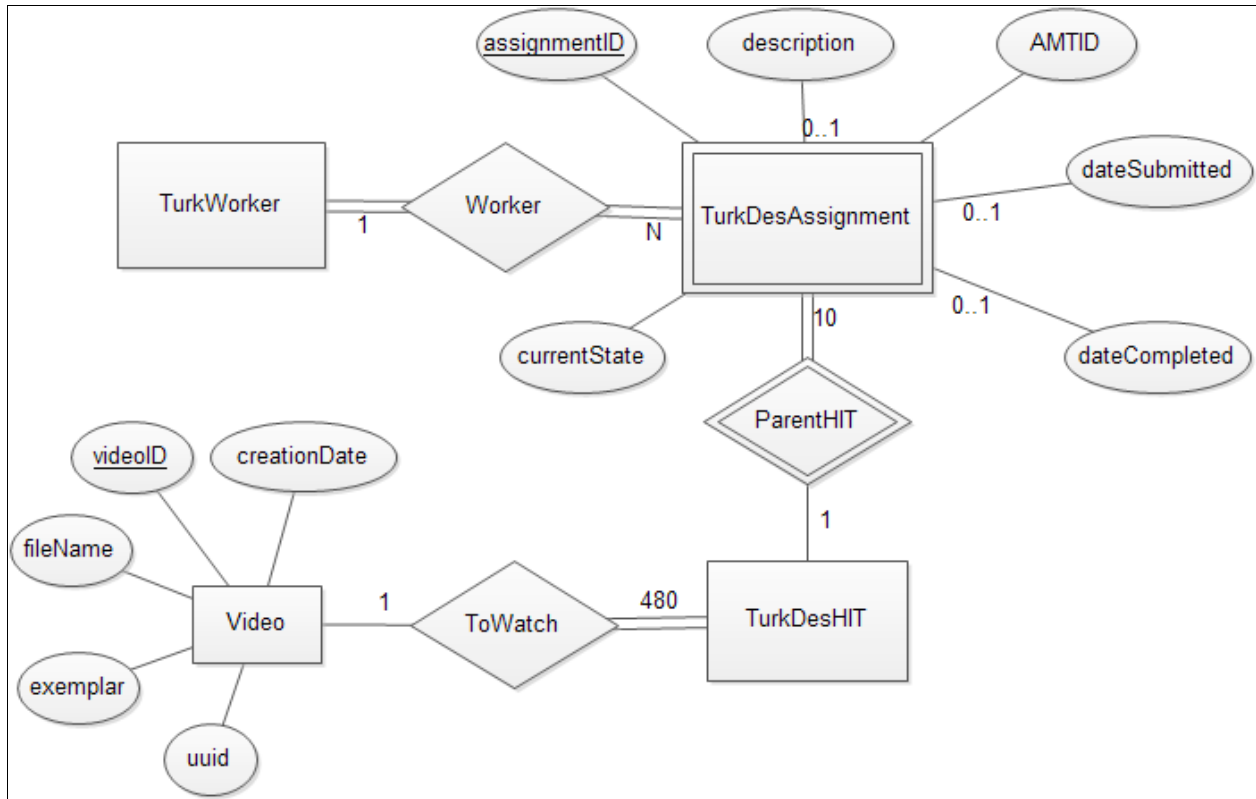


Figure 25. AMT Description Task entity-relationship diagram.

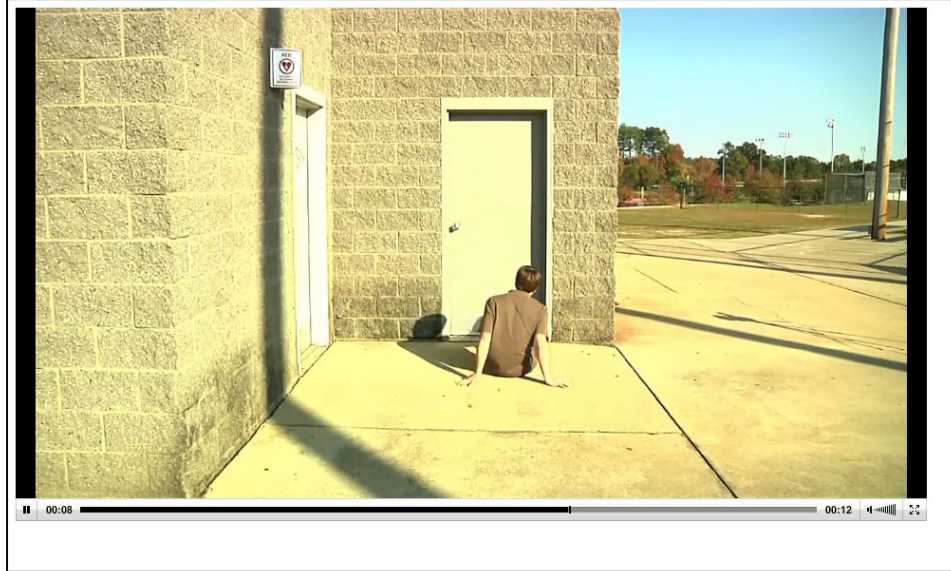
The Gap-Filling Task was similar in format to the Description Task. The Worker was instructed to view a short vignette that has been altered to have a portion of the action removed and replaced with a black screen. The Worker was then instructed to describe what they suspect happened during the blank segment of video in 140 characters or less using a list of verbs as a guide. The Gap-Filling Task consisted of 120 vignettes, each completed by 10 HIT assignments. An example HIT is shown in figure 26. The entity-relationship diagram for the AMT Gap-Filling Task is shown in figure 27.

Describe what occurred in this short video during the blacked out portion

Guidelines:

- View the short video from start to finish.
- A portion of the video will be blacked out. This can happen in the beginning, middle, or end of the video.
- Review the list of actions and definitions via the url link.
- In the text box provided, describe what has occurred in the video during the blacked out portion (use the list of actions as a guideline).
- Note if you are doing multiple HITs: the list of actions is always the same.

Video:



[Actions and Definitions](#)

Please describe what happened in the above video during the missing portion using the list of actions as a guideline:

Limit your response to 140 characters (approximately 2 sentences):

Sample Response: The man continues running away from the woman who hit him.

Sample Response: The car drove past the man and gave him some object which he can be seen to be carrying at the end.

Figure 26. AMT Gap-Filling Task.

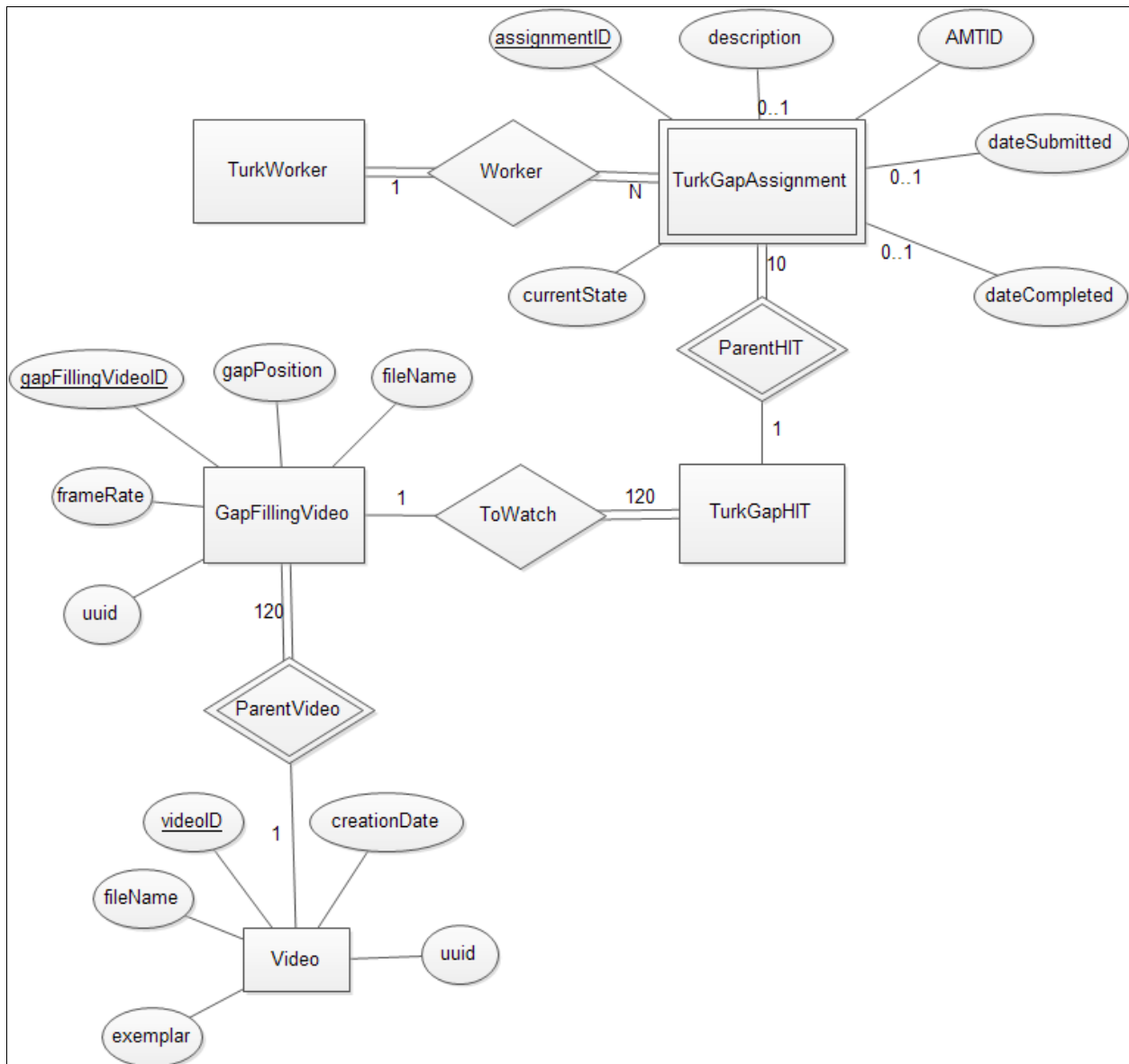


Figure 27. AMT Gap-Filing Task entity-relationship diagram.

There were several challenges regarding the design and implementation of the Description and Gap-Filling tasks. Unlike the multiple choice answers of the Recognition Task, these tasks do not produce a response that can be easily checked using a computer algorithm. For this reason, most responses were accepted and only obviously incorrect responses, such as blank answers, were rejected. In addition, it was difficult to determine an appropriate reward amount. While each assignment only requires a single vignette view and response, the effort to analyze and type a full response is significantly more than selecting a multiple choice response of yes or no. After some communication with Workers through the Turker Nation Web site, a sufficient price point was identified and implemented.

4.4 Results and Discussions

Response data for the Recognition Task collected from the AMT effort proved difficult to analyze. The monetary reward proved popular for AMT Workers, but also attracted a number of malicious Workers that did not put forth a fair effort. Upon analysis, one Worker in particular submitted a large number of responses with a yes response for almost all questions. Investigating responses on a by Worker basis revealed that a subset of Workers performed particularly poorly on the gold standard questions. The responses from Workers that failed to agree with 60% of gold standard questions were disqualified and the responses for these HITs were put back on AMT for additional data.

Data from the Description and Gap-Filling tasks were not formally analyzed. Because this response data consists of language statements, automated analysis would be very difficult. To perform such an evaluation, advanced methods using natural language processing would be required. Alternatively, the data would have to be evaluated manually in a qualitative manner. However, such an analysis would inherently incorporate the bias of the evaluator. Instead, the results from these tasks were screened for blank or obviously erroneous data points and stored for later comparison with the automated systems.

5. Conclusion

In this report, we have described the system for collecting and recording the human response data for the DARPA Mind’s Eye program. Theses human response data were used as “ground truth” for analyzing the system responses of the DARPA Mind’s Eye PIs.

The data collection involved three different tasks: recognition, description, and gap-filling. Data for each task were collected using both internal, trusted sources and broad external sources. Internal data collection showed that the crafted videos did well in demonstrating specific verbs. However, there is a large amount of overlap among the verbs and the videos contained a large amount of ambiguity in verb depiction. As an example, “run” videos were also largely labeled as “move.” In addition, collecting these data proved to be very time consuming.

Data collection through AMT was efficient and inexpensive. The crowdsourced study completed in significantly less time than the internal study. However, such techniques require additional care, as there are several difficulties associated with collecting crowdsourced data. In particular, insincere effort from malicious users can taint the data. Gold standard check questions were embedded into the HITs and proved to be an important safeguard. However, additional effort should be made to analyze the data on a per Worker basis as the HITs are rolled out in batches. Malicious Workers should be identified and banned from HITs to prevent inaccurate data and minimize the work of repeating data collection for inaccurate HITs.

6. References

- AWS, A. Amazon Mechanical Turk, *QuestionForm.xsd*, 1 October 1 2005.
<http://mechanicalturk.amazonaws.com/AWSMechanicalTurkDataSchemas/2005-10-01/QuestionForm.xsd> (accessed 23 May 2013).
- DARPA. Mind's Eye. http://www.darpa.mil/Our_Work/I2O/Programs/Minds_Eye.aspx
(accessed August 2013).
- Thompson, A. A. *The Word-Based Pyramid Method*; ARL-TR-5912; U.S. Army Research Laboratory: Adelphi, MD, 2012a.
- Thompson, A. A. *Interval Scales From Paired Comparisons*; ARL-TR-6003; U.S. Army Research Laboratory: Adelphi, MD, 2012b.
- Turing, A. M. Computing Machinery and Intelligence. *Mind* **1950**, 59, 433-460.

1 DEFENSE TECHNICAL
(PDF) INFORMATION CTR
DTIC OCA
8725 JOHN J KINGMAN RD

1 GOVT PRNTG OFC
(PDF) A MALHOTRA

2 DIRECTOR
(PDFS) US ARMY RESEARCH LAB
RDRL CIO LT
IMAL HRA MAIL & RECORDS MGMT

5 DIRECTOR
(PDFS) US ARMY RESEARCH LAB
RDRL CII B
L SADLER
L TOKARCIK
R WINKLER
RDRL-CII-A
N FUNG
S YOUNG

1 DIRECTOR
(PDF) US ARMY RESEARCH LAB
1 RDRL CII
(HC) B BROOME

3 DARPA/I20
(PDF) J FITZGERALD
P MICHELUCCI
J DONLON
3701 N FAIRFAX DR
ARLINGTON VA 22203