

Grounding the Meaning of Words through Vision and Interactive Gameplay

Natalie Parde¹, Adam Hair¹, Michalis Papakostas^{2,3}, Konstantinos Tsiakas^{2,3}
 Maria Dagioglou³, Vangelis Karkaletsis³, and Rodney D. Nielsen¹

¹Department of Computer Science and Engineering, University of North Texas

²Department of Computer Science and Engineering, University of Texas at Arlington

³Institute of Informatics and Telecommunications, N.C.S.R. Demokritos

Abstract

Currently, there exists a need for simple, easily-accessible methods with which individuals lacking advanced technical training can expand and customize their robot’s knowledge. This work presents a means to satisfy that need, by abstracting the task of training robots to learn about the world around them as a vision- and dialogue-based game, *I Spy*. In our implementation of *I Spy*, robots gradually learn about objects and the concepts that describe those objects through repeated gameplay. We show that *I Spy* is an effective approach for teaching robots how to model new concepts using representations comprised of visual attributes. The results from 255 test games show that the system was able to correctly determine which object the human had in mind 67% of the time. Furthermore, a model evaluation showed that the system correctly understood the visual representations of its learned concepts with an average of 65% accuracy. Human accuracy against the same evaluation standard was just 88% on average.

1 Introduction

As technology advances, many people are welcoming interactive robots into their homes to assist with personal needs in a diverse array of applications, including healthcare [Nielsen *et al.*, 2010] and assistive technologies [Galatas *et al.*, 2011]. Very few of these new robot owners possess the advanced technical background traditionally required to train robots. However, it is likely that such robot owners have objects in their homes that are unique and that their robots should learn about for conversational or task-related purposes. Moreover, these robot owners may need to fine-tune their robot’s knowledge regarding other concepts—for instance, color-blind owners may describe a red armchair to their robot quite differently from the average person. For these reasons, among others, non-technical robot owners should still be given the ability to train their robots about their home environments, but in a manner that is natural and intuitive.

One way to achieve this is through interactive gameplay. In this work, we present a game, *I Spy*, that allows a robot

to build visual models for concepts extracted from natural-language object descriptions, and gradually improve those models during gameplay. It does this by continuously updating its knowledge to reflect information gathered from newly-captured training images and positive and negative labels learned from “yes” or “no” responses to questions about objects during the game, as it attempts to determine which object the human player has in mind. This paper is organized as follows: we first provide an overview of prior related work, followed by a technical discussion of the different techniques used to develop the game. We then describe a study designed to gauge the efficacy of the resulting system by tracking the system’s game and model performance. Finally, we conclude with an analysis of the results obtained during the study, and outline some plans for future iterations of the game.

2 Background

Our implementation of *I Spy* learns concepts from human descriptions and grounds those concepts in visual features learned during gameplay, using positive and negative labels also learned during gameplay interactions. Although no prior work has done exactly that, some aspects of the approach have been tackled in previous work. For example, an “I Spy” or “20 Questions” format was previously explored as a means of improving robot vision by [Vogel *et al.*, 2010]. The system in [Vogel *et al.*, 2010] asked questions about hard-coded features to improve vision-based category and attribute knowledge. However, this implementation was tested with only one type of attribute (color), and was tested using only two different colored blocks per game. We have tested our current implementation using 17 objects per game, and as a result of the object descriptions collected during our evaluation our system learned many more concepts.

Some prior work has also explored using other types of gameplay for grounded language learning in robots, such as the object-naming game developed by [Steels, 1995] and specifically applied to human-robot games in [Steels and Kaplan, 2002]. The object-naming game differs from *I Spy* in that it creates a 1:1 mapping between an object and its name, whereas *I Spy* creates many:many vision:language associations, recognizing multiple features common across multiple objects (e.g., clock, basketball, and soccer-ball are round, have black lines, etc.). Furthermore, human input in [Steels and Kaplan, 2002] is scripted. In contrast, *I Spy* learns its

concepts from unscripted, natural-language descriptions that often contain noise.

Other prior work with similar goals to this project includes the recent work of [Liu and Chai, 2015], which utilizes conversational dialogue with a small, humanoid robot to ground visual perceptions regarding hard-coded type, color, and position attributes. Another system that has integrated multiple modalities for robot learning in a non-game context is Ripley [Mavridis and Dong, 2012], a non-humanoid robotic system that learns color, size, and weight attributes via a mix of computer vision techniques, motor actions (for weighing objects), and spoken dialogue interactions with users. Like the other systems mentioned though, it was unable to dynamically learn new attribute types, and was tested with a very limited number of objects.

Rather than employing human-robot interaction, some prior systems have focused on automatically learning visual representations of concepts using a purely web-based approach. One example of this is the Never-Ending Image Learner (NEIL), which automatically extracts visual knowledge from internet data [Chen *et al.*, 2013]. Although this is useful for many applications, it does not allow for knowledge acquisition based on unique concepts customized to an individual robot owner’s home environment.

Finally, some other interesting applications designed to connect language and visual perception include those of [Mooney, 2008; Chen and Mooney, 2008], [Gupta *et al.*, 2008], [Mitchell *et al.*, 2012], and [Elliott and Keller, 2013]. In [Mooney, 2008; Chen and Mooney, 2008], a system learned to sportscast RoboCup games by training on commentaries of simulated games. Similarly, the work by [Gupta *et al.*, 2008] focused on developing image and video classifiers by co-training on captioned images and human-commentated videos. [Mitchell *et al.*, 2012; Elliott and Keller, 2013] both worked toward generating natural-sounding descriptions of images.

The work in *I Spy* is most similar in its implementation to that of [Vogel *et al.*, 2010]. However, it goes well beyond color learning based on a pair of simple objects, by utilizing concepts learned from players’ object descriptions rather than by relying on pre-programmed attributes. It also incorporates a larger quantity and greater diversity of objects in its games. By building visual concept models that evolve during gameplay, our implementation of *I Spy* functions at its deepest level as a multimodal interactive learning system.

3 Approach

3.1 Game Overview

I Spy follows a two-stage process: (1) an initial learning phase, in which the robot populates its initial knowledge base by capturing a sequence of images and pairing the extracted visual feature vectors with keywords learned from object descriptions provided by players, and (2) a gaming phase, in which the actual game is played. Both of these stages are further decomposed into sub-tasks requiring natural language processing, computer vision, and machine learning methods. The high-level flow diagrams for these stages are presented in Figures 1 and 2. The robot platform is NAO V4, a humanoid

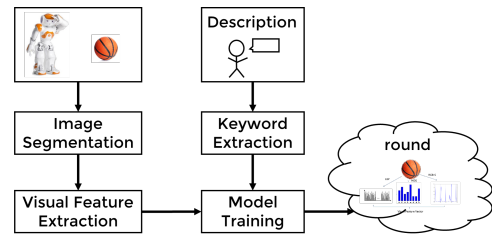


Figure 1: The robot’s initial learning phase.

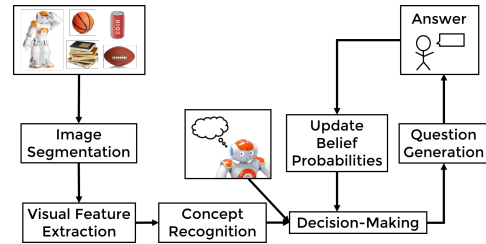


Figure 2: The robot’s gaming phase.

robot created by Aldebaran Robotics (www.aldebaran.com), running the NAOqi v1.14.5 operating system. The JNAOqi SDK is used to integrate the robot with Java for motion control and image capture operations.

Initial Learning Phase.

In the initial learning phase, the robot begins with an empty knowledge base. To learn about an object, the robot captures a series of images of the object from different angles and distances. The captured images are segmented, and visual feature vectors are extracted from the image segments of the object. In parallel, the player supplies a short description of the object, and keywords are automatically extracted from the description. Finally, the extracted visual feature vectors are used to train statistical concept models for each of the keywords.

Gaming Phase.

In the gaming phase, the robot is placed in front of a set of objects. The robot captures images of the gamespace, and these images are segmented to isolate each object. Visual feature vectors are then extracted for each object, and classified against the concept models created in the initial learning phase. The robot uses the classification scores for each object in the gamespace as input to its decision-making module, along with information regarding the object descriptions and prior gameplay interactions. In turn, the decision-making module uses this information to decide what concept to ask about in the next question. A question is generated based on this concept, and the player responds to the question with a “yes” or “no” answer. The robot then updates its belief probabilities with regard to which object it thinks the player has in mind, until it is confident enough to make a guess.

4 Methods

4.1 Natural Language Processing Methods

Natural language processing is used in *I Spy* primarily to extract keywords from users’ descriptions, and to automatically

generate questions. A brief description of how these tasks are achieved follows.

Keyword Extraction from Object Descriptions.

Keywords are automatically extracted from users' object descriptions using a filter to eliminate common stopwords, prepositions, descriptors indicating relative direction, linking verbs, verbs indicating composition, and quantifiers. Stopwords are taken from the standard Apache Lucene (lucene.apache.org) English stopword list. Prepositions and descriptors indicating spatial proximity are removed due to inconsistency with images acquired by the robot; for example, users viewing a mug from one angle may describe it as having a handle on its left, but if the robot views the mug from its opposite side then the handle will appear on its right. Words indicating composition (e.g., "made of," "includes") are removed because they exist in nearly every description and are not directly associated with a property of the objects. Quantifiers not providing any concrete information (e.g., "it has *some* lines") are removed because their definitions are too relative to individual objects to yield much meaningful content. Thus, the remaining set of extracted concepts from each object description is comprised of keywords capable of providing discrete information. Common keywords extracted from various participants' object descriptions include attribute words (e.g., "red," "small"), action words (e.g., "bounces," "opens"), and words indicating activities one could perform with the object (e.g., "play," "write"), among others.

Natural Language Generation.

To generate natural-sounding questions during gameplay, the question's subject and verb components are selected according to the chosen concept's part-of-speech tag and, if applicable, its subfunction within that particular part-of-speech. Part-of-speech tags are acquired using the Stanford Part-Of-Speech Tagger [Toutanova *et al.*, 2003] and are assigned to concepts based on the concept's most frequent usage in input object descriptions, to avoid generating confusing or ambiguous questions (e.g., the system should generate "Does it have a handle?" rather than "Can you handle it?" since most users' descriptions have used "handle" as a noun). The words included in function-based subcategories were manually selected (partially from DBpedia (dbpedia.org) category lists), in order to quickly increase the clarity of questions for this work. Once a question's subject and verb are selected, they are paired with the concept, which serves as the object of the question for most part-of-speech categories. The question is then constructed using SimpleNLG [Gatt and Reiter, 2009], although custom templates are used for a small number of categories with which SimpleNLG's realizer yields unnatural results.

4.2 Computer Vision Methods

Computer vision is used in *I Spy* to segment images and extract visual feature vectors describing objects' texture, shape, and color, both during the initial learning phase and the gaming phase. Image segmentation is performed using the OpenCV [Bradski, 2000] Watershed Algorithm. Extracted

visual features are used as training data for all concept models associated with an object, even though the meaning of some of those concepts may not be expressed through texture, shape, or color, because the robot has no initial knowledge regarding which concepts define visual attributes. Likewise, it has no inherent notion of what type of visual attribute might be described by a visually-discernible concept (to a naïve robot, "red" is just as likely to describe a shape as it is a color). Our visual feature extractors are described below.

Local Binary Patterns — Texture-based feature extraction.

I Spy obtains texture information from images by extracting local binary patterns (LBP) feature vectors, which have been shown to be illumination invariant and efficient under different environmental setups [Heikkila and Pietikainen, 2006]. LBP feature vectors are extracted by first dividing an image segment into smaller disjoint regions. In these regions, each pixel is labeled with a binary value according to whether its original value is greater than or equal to that of the region's center pixel (1) or less than the central pixel's value (0). The result is a binary "word" describing all of the pixels in the region, and the feature vector is then created by concatenating the words from all smaller regions.

Histograms of Oriented Gradients — Shape-based feature extraction.

I Spy uses Histograms of Oriented Gradients (HOG) to find the shape of objects. HOG features are created by first counting occurrences of gradient orientation in localized portions of an image segment. Then, the local object shape within an image is described according to the distribution of intensity gradients or edge directions.

RGB-S — Color-based feature extraction.

To extract color-based features, *I Spy* uses RGB and HSV histograms. RGB is an additive color model based on the human perception of colors, and is able to reproduce a broad array of colors by combining red, green, and blue light. HSV is a color model that describes the hue, saturation, and value of colors in terms of their shading and brightness. *I Spy* extracts an RGB histogram for each separate channel (red, green, and blue), as well as a saturation histogram from the HSV color space.

Finally, all of the extracted texture, shape, and color features are concatenated to construct a single visual feature vector containing all of the essential information to describe a given image segment.

4.3 Statistical Modeling

Gaussian Mixture Models (GMMs) are used to build concept models from the extracted visual feature vectors. GMMs are probabilistic models that assume that all data points are generated from a mixture of Gaussian distributions with unknown parameters. They have been used extensively as statistical models for image classification, including for classification and segmentation using color and texture features [Permuter *et al.*, 2006]. Models with tied covariances are used to avoid data overfitting, and object images captured during the initial learning phase are used as training data for

a concept model if at least half of the descriptions for that object include the concept. Models are retrained following every game, to reflect new information from “yes” and “no” player responses and visual features extracted from game photos. GMMs are constructed with two components, and the Expectation-Maximization (EM) algorithm is used for parameter estimation (component weights, means, and covariances). Model training is performed using the Python SciKit-Learn [Pedregosa *et al.*, 2011] library.

4.4 Decision-Making

Decision-making is critical to *I Spy*’s success, since the keywords chosen for the robot’s questions impact the quality of the player’s gaming experience, as well as the efficiency of the robot’s underlying learning process. This subsection describes how the system chooses those keywords, based on probabilities calculated using the robot’s vision and language knowledge at the time of the game. The system calculates probabilities using three different sources of information: its concept models, answers collected during previous games, and initial object descriptions.

For all of the system’s calculations, let \mathbf{O} be the set of all objects $\{o_1, \dots, o_n\}$ in the gamespace and \mathbf{C} be the set of all concepts $\{c_1, \dots, c_m\}$ in the robot’s current knowledge base. $P^{(V)}(c_j|o_i)$ is the classification score output by the concept model for c_j , given the game image of object o_i .

$P^{(QA)}(\alpha(c_j) = yes|o_i)$, shown in Equation 1, is the probability that the answer to a question about c_j is “yes” given that the game’s target object is o_i , based on answers to questions about this c_j, o_i pairing in prior games. $\alpha(c_j)$ is the answer to a question about concept c_j . $\Phi(c_j, o_i, yes)$ is the number of “yes” answers received to questions asked about this c_j, o_i pairing during prior gameplay, and $\Phi(c_j, o_i)$ is the total number of answers received regarding this pairing.

$$P^{(QA)}(\alpha(c_j) = yes|o_i) = \frac{\Phi(c_j, o_i, yes) + 1}{\Phi(c_j, o_i) + 2} \quad (1)$$

$P^{(Q,D)}(\alpha(c_j) = yes|o_i)$, shown in Equation 2, is the probability that the answer to a question about c_j is “yes” given that the game’s target object is o_i , based on information from all the objects, descriptions, and question responses in prior gameplay. Let n be the number of descriptions for o_i in which c_j occurs. Let \mathbf{A}_n be the set of all answers $\{a_{n_1}, \dots, a_{n_p}\}$ to questions previously asked, for all concept-object pairs for which the concept was included in n descriptions of that object. Let $\sigma(a_{n_k})$ be 1 if an answer a_{n_k} is “yes,” and 0 otherwise.

$$P^{(Q,D)}(\alpha(c_j) = yes|o_i) = \frac{1}{|\mathbf{A}_n|} \sum_{a_{n_k} \in \mathbf{A}_n} \sigma(a_{n_k}) \quad (2)$$

$P^{(V)}$, $P^{(QA)}$, and $P^{(Q,D)}$ are combined to calculate $P(\alpha(c_j) = yes|o_i)$, the overall probability for the answer to a question about c_j being “yes.”

Finally, the system maintains belief probabilities for each object throughout the course of the game. These are computed using Equations 3 and 4. $P(\alpha(c_j) = yes)$ is the overall

probability that a concept c_j elicits a “yes” answer independent of the object. $P(o_i|\alpha(c_j) = yes)$ is the belief probability for an object o_i given that a question about c_j elicits a “yes” response. $P(o_i)$ is the probability that the player has o_i in mind, independent of any other factors (or $1/|\mathbf{O}|$ before the first question is asked).

$$P(\alpha(c_j) = yes) = \sum_{o_k \in \mathbf{O}} P(\alpha(c_j) = yes|o_k) \times P(o_k) \quad (3)$$

$$P(o_i|\alpha(c_j) = yes) = \frac{P(\alpha(c_j) = yes|o_i) \times P(o_i)}{P(\alpha(c_j) = yes)} \quad (4)$$

To choose the most informative question, $P(o_i|\alpha(c_j) = yes)$ is calculated for all o_i in \mathbf{O} and all unasked c_j in \mathbf{C} . Letting \mathbf{S}_1 be the subset of objects with the highest belief probabilities and \mathbf{S}_2 be the next highest-probability subset of objects, with the subsets split at the point at which the difference between the belief probabilities in the two subsets is greatest, the concept that maximizes the expected difference between the average belief probabilities for objects in \mathbf{S}_1 and \mathbf{S}_2 is selected. The goal in doing this is to reduce the set of plausible objects that the player might have in mind with each question asked.

5 Experiment

5.1 Experiment Overview

A preliminary user study evaluating *I Spy*’s appeal to in-person players has shown that the game is fun and engaging to people of a diverse range of age, gender, and educational demographics [Parde *et al.*, 2015]. To evaluate our current implementation’s gaming and concept model performance, we simulated 255 games using real human responses obtained via Amazon Mechanical Turk (AMT) (www.mturk.com). Our experimental design and results are presented below.

5.2 Data Collection

Initial Learning Phase.

To create initial concept models for the simulated games, the initial learning phase was conducted for 17 objects of varied size, color, shape, and texture, including: digital clock, analog clock, red soccer ball, basketball, football, yellow book, yellow flashlight, blue soccer ball, apple, black mug, blue book, blue flashlight, cardboard box, pepper, green mug, polka dot box, and scissors. Descriptions for these objects were acquired via AMT. Participants provided demographic information (age, gender, educational level, and native language) to control for eligibility (native English speakers age 18 or over). The first six descriptions meeting eligibility requirements were kept for each object. Forty-three AMT workers (22 female) of various educational levels (4 had a high school diploma, 15 had some college experience, 5 had an associate’s degree, 12 had a bachelor’s degree, 3 had some post-bachelor college experience, and 4 had a master’s degree), provided descriptions for the objects.

To acquire visual information for the objects, each object was placed on a white floor with a white background, and a

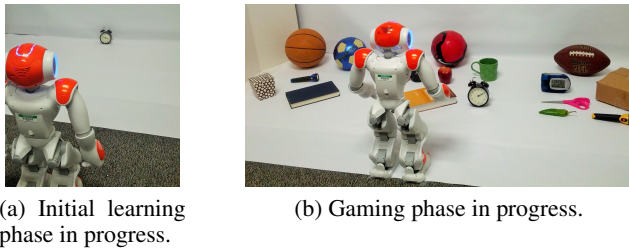


Figure 3: Initial learning phase and gaming phase.

NAO robot walked around the object, capturing photos from different distances and angles (see Figure 3a for the initial learning phase in progress). These images were segmented, and since individual image components are not the focus of this work, non-relevant segments were discarded manually.

Simulated Games.

To capture images for the simulated games, all 17 objects were placed on the same white surface used for the initial learning phase. The robot took photos of the game space from three different locations, such that each object was included in one of the images. The items were then rearranged so that they were in different positions and orders, and the process was repeated. This was done for 30 unique gamespace configurations (see Figure 3b for the gaming phase in progress).

To obtain human responses for the simulated games, a list was generated containing all 289 keywords extracted from the user descriptions, and a question was automatically generated for each keyword in this list. AMT workers were provided one pre-captured gamespace configuration's images, and were asked to click "yes" or "no" for each possible question for a specified object. Thirty sets of answers (one set per gamespace configuration) were acquired for each of the 17 objects, leading to 510 sets of answers.

Once the answer sets were collected, they were screened for spam. To do this, questions were first analyzed to determine whether they had a correct answer. Questions tagged as nonsensical by a third-party annotator (8 of the 289 questions) were ignored regardless. The remaining questions were determined to have a "gold standard" answer for a given object if the number of their 30 responses having the majority answer (either "yes" or "no") exceeded chance (15) by two standard deviations. Then, the subset of questions having correct answers for a given object was used to determine the average accuracy of all AMT workers for that object. A worker's answers were discarded if his or her accuracy over the full set of concepts was two standard deviations below the average accuracy of all workers. This resulted in replacing all of the responses for 38 workers. On seven occasions, the same worker submitted more than one set of answers for the same object; these duplicate sets were also replaced. The final collection of 510 sets of answers was gathered from 432 AMT workers. 184 were male, 243 were female, and 5 declined to indicate a gender. Once again, educational experience varied.

5.3 Simulation Design

To simulate the games, the 510 sets of answers were first divided into two groups (training and testing). After complet-

ing the initial learning phase for each object, the 15 training sets of answers for each object and the image features extracted from 15 gameplay images of the object were given to the system so that the simulated games could consider prior gameplay knowledge. The remaining 15 test sets of answers for each object were used for the actual simulations.

At the onset of a simulated game, its gameplay images were imported to the system and visual features were extracted for each object in the gamespace. Then, answers to each possible question for the game's target object (taken from one of the 15 sets of test answers for that object) were loaded. Gameplay proceeded identically to live gameplay, with the only difference being that the system simply checked the answer that had been loaded for the question it asked, rather than waiting for an in-person player to respond. Answers collected during the simulated gameplay were added to the system's knowledge as each game progressed. This process was repeated until all test answer sets for all objects had been played, for a total of 255 simulated games.

5.4 Simulation Results

The system was able to correctly determine the object that the human player had in mind in 67% of the simulated games. One object (the digital clock) was not guessed correctly in any of the games in which it was the target object (although in 80% of those games, the system guessed the other clock); without this outlier, the system guessed correctly 72% of the time. It was extremely accurate when guessing the pair of scissors (100%), the yellow flashlight (93%), the apple (93%), and the black mug (93%).

5.5 Concept Model Evaluation

To further analyze the performance of the system's concept models, test images for each object were classified against each model in the system's knowledge, and the averaged results were compared with the gold standard set of answers for that object. Initial models were created using the same initial learning stage as in the simulated games, and training images (from the same 15 training sets used in the simulated games) were added as additional positive instances to the models for which their individual corresponding answer sets gave "yes" answers. The concept models were then evaluated using the remaining 15 test sets per object. To do this, a new test image was classified against each concept model in the system's knowledge, and the classification scores were recorded. Following this, the image's corresponding answer set was loaded, and a game was simulated. At the end of the game, the image and only the answers collected during the game itself were added to the system's knowledge.

After simulating all 255 test games, the classification results were averaged across all test images for a concept-object pair, and the averages were compared with the gold standard answer for that pair. Thresholds were set at greater than 75% ("yes") and less than 25% ("no") to determine the system's answer. Concept models for which the average classification scores for an object did not meet either threshold were excluded from the evaluation for that object, as were concept models for which no gold standard answer existed for that object. The average accuracy of the system's concept models

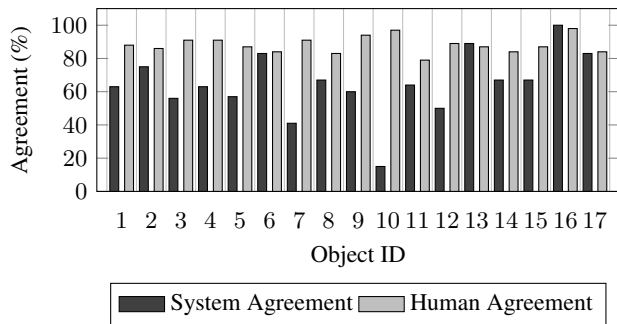


Figure 4: Agreement with the gold standard.

was then compared with the average accuracy of individual human players against the gold standard.

5.6 Concept Model Evaluation Results



The results of the concept model evaluation are shown in Figure 4. Overall, average system agreement with the gold standard was 65% and average human agreement with the gold standard was 88%. A sample of the correctly-identified and incorrectly-identified concepts for two objects are displayed in Table 1 (concepts for which the system correctly identified a “no” answer are in parentheses). The system performed well with concepts having predominately visual meanings, particularly if their models were trained using images of a variety of objects (e.g., “blue,” “brown,” “round”). The incorrectly-identified concepts in Table 1 provide examples of models that perform poorly due to either a lack of visual salience, limited training data, or a combination of both factors. For instance, the concept “signs” is fairly abstract and existed only in the descriptions for a round green mug (covered in peace signs), which may have led the system to conflate the visual representation for “signs” with that of “round.” Interestingly, the first object in Table 1 actually does have tape on it; however, this was undetected by most human responders (hence the gold-standard “no” answer). Regardless, it is likely that the system has just confused the visual representation for “tape” with that of some other concept (such as “rectangular”), rather than having a well-defined “tape” model, due to the limited training data for that concept.

6 Discussion

The overall game performance and the results of the concept model evaluation indicate that the system is reasonably capable of learning the visual representations of concepts over the course of repeated gameplay. The system’s agreement with the gold standard for each of the game objects was fairly close in many cases to human agreement, with the overall average agreement being lower than human agreement.

In addition to the game performance and concept model evaluation, it is interesting to observe the average agreement of human players with their own gold standard. From among the 30 sets of responses per object used to create the gold standard, individual responders only agreed with 88% of the gold standard answers on average. This highlights the relatively large number of perceptual differences among humans,

Table 1: Model Output for Objects

Object	Correctly Identified	Incorrectly Identified
	(light), box, brown, (flashlight), cardboard, (button), (blue), (switch)	tape
	ball, round, soccer, blue	metallic, signs

even regarding concepts that may seem “obvious” to many. These perceptual differences are one of the motivators in developing this game, with the goal that through playing this game a personal robot may perceive the world with a view more closely aligned with its user.

7 Conclusion and Future Work

In conclusion, this paper has shown that *I Spy* is a feasible approach for enabling robot owners who have not had extensive technical training to teach their robots about the world around them. In the study presented, the system was able to determine which object a human player had in mind 67% of the time out of 255 simulated games. Moreover, after only the initial learning phase and 15 sets of training data, the system’s concept models agreed with the gold standard on average 65% of the time over all objects. It should be noted that the average human agreement with this same gold standard was only 88%. The agreement between the system’s concept model classifications and the gold standard is expected to increase over time as a larger variety of descriptions and image samples are added to its training datasets.

The potential avenues for future expansion with this project are manifold. One area currently under development involves the introduction of previously-unseen objects in gameplay, allowing the robot to transfer its existing learned knowledge to new objects. Other plans include supplementing the system’s knowledge with images and descriptions retrieved from the web, to expand the volume and diversity of training samples based on information learned during gameplay. For instance, the system could search the web for images similar to one that the robot had captured locally, adding the most relevant results to the same concept models to which the original, local image had been added. Similarly, it could search text resources for synonyms of the concepts extracted from descriptions during the initial learning phase, saving the synonyms as aliases for the original concept model.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive comments and suggestions. This material is based upon work supported by the National Science Foundation under Grants No. IIS-1262860, CNS-1035913, IIS-1409897, and CNS-1338118. Initial work for the research described in this paper was carried out during the 2014 International Research-centered Summer School, organized by the Software and Knowledge Engineering Lab at N.C.S.R. Demokritos.

References

- [Bradski, 2000] Gary Bradski. The opencv library. *Doctor Dobbs Journal*, 25(11):120–126, 2000.
- [Chen and Mooney, 2008] David L Chen and Raymond J Mooney. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135. ACM, 2008.
- [Chen *et al.*, 2013] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Neil: Extracting visual knowledge from web data. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1409–1416. IEEE, 2013.
- [Elliott and Keller, 2013] Desmond Elliott and Frank Keller. Image description using visual dependency representations. In *EMNLP*, pages 1292–1302, 2013.
- [Galatas *et al.*, 2011] Georgios Galatas, Christopher McMurrough, Gian Luca Mariottini, and Fillia Makedon. eyedog: an assistive-guide robot for the visually impaired. In *Proceedings of the 4th International Conference on Pervasive Technologies Related to Assistive Environments*, page 58. ACM, 2011.
- [Gatt and Reiter, 2009] Albert Gatt and Ehud Reiter. Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation*, pages 90–93. Association for Computational Linguistics, 2009.
- [Gupta *et al.*, 2008] Sonal Gupta, Joohyun Kim, Kristen Grauman, and Raymond Mooney. Watch, listen & learn: Co-training on captioned images and videos. In *Machine Learning and Knowledge Discovery in Databases*, pages 457–472. Springer, 2008.
- [Heikkila and Pietikainen, 2006] Marko Heikkila and Matti Pietikainen. A texture-based method for modeling the background and detecting moving objects. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):657–662, 2006.
- [Liu and Chai, 2015] Changsong Liu and Joyce Chai. Learning to mediate perceptual differences in situated human-robot dialogue. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [Mavridis and Dong, 2012] Nikolaos Mavridis and Haiwei Dong. To ask or to sense? planning to integrate speech and sensorimotor acts. In *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2012 4th International Congress on*, pages 227–233. IEEE, 2012.
- [Mitchell *et al.*, 2012] Margaret Mitchell, Xufeng Han, Jesse Dodge, Alyssa Mensch, Amit Goyal, Alex Berg, Kota Yamaguchi, Tamara Berg, Karl Stratos, and Hal Daumé III. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 747–756. Association for Computational Linguistics, 2012.
- [Mooney, 2008] Raymond J Mooney. Learning to connect language and perception. In *AAAI*, pages 1598–1601, 2008.
- [Nielsen *et al.*, 2010] Rodney D Nielsen, Richard M Voyles, Daniel Bolanos, Mohammad H Mahoor, Wilson D Pace, Katie A Siek, and Wayne Ward. A platform for human-robot dialog systems research. In *AAAI Fall Symposium: Dialog with Robots*, 2010.
- [Parde *et al.*, 2015] Natalie Parde, Michalis Papakostas, Konstantinos Tsiakas, Maria Dagioglou, Vangelis Karkaletsis, and Rodney D Nielsen. I spy: An interactive game-based approach to multimodal robot learning. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [Pedregosa *et al.*, 2011] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [Permuter *et al.*, 2006] Haim Permuter, Joseph Francos, and Ian Jermyn. A study of gaussian mixture models of color and texture features for image classification and segmentation. *Pattern Recognition*, 39(4):695–706, 2006.
- [Steels and Kaplan, 2002] Luc Steels and Frederic Kaplan. Aibos first words: The social learning of language and meaning. *Evolution of communication*, 4(1):3–32, 2002.
- [Steels, 1995] Luc Steels. A self-organizing spatial vocabulary. *Artificial life*, 2(3):319–332, 1995.
- [Toutanova *et al.*, 2003] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.
- [Vogel *et al.*, 2010] Adam Vogel, Karthik Raghunathan, and Dan Jurafsky. Eye spy: Improving vision through dialog. In *AAAI Fall Symposium: Dialog with Robots*, 2010.