Learning the Semantics of Manipulation Action

Yezhou Yang[†] and **Yiannis Aloimonos**[†] and **Cornelia Fermüller**[†] and **Eren Erdal Aksoy**[‡]
[†] UMIACS, University of Maryland, College Park, MD, USA
{yzyang, yiannis, fer}@umiacs.umd.edu
[‡] Karlsruhe Institute of Technology, Karlsruhe, Germany
eren.aksoy@kit.edu

Abstract

In this paper we present a formal computational framework for modeling manipulation actions. The introduced formalism leads to semantics of manipulation action and has applications to both observing and understanding human manipulation actions as well as executing them with a robotic mechanism (e.g. a humanoid robot). It is based on a Combinatory Categorial Grammar. The goal of the introduced framework is to: (1) represent manipulation actions with both syntax and semantic parts, where the semantic part employs λ-calculus; (2) enable a probabilistic semantic parsing schema to learn the $lambda$-calculus representation of manipulation action from an annotated action corpus of videos; (3) use (1) and (2) to develop a system that visually observes manipulation actions and understands their meaning while it can reason beyond observations using propositional logic and axiom schemata. The experiments conducted on a public available large manipulation action dataset validate the theoretical framework and our implementation.

1 Introduction

Autonomous robots will need to learn the actions that humans perform. They will need to recognize these actions when they see them and they will need to perform these actions themselves. This requires a formal system to represent the action semantics. This representation needs to store the semantic information about the actions, be encoded in a machine readable language, and inherently be in a programmable fashion in order to enable reasoning beyond observation. A formal representation of this kind has a variety of other applications such as intelligent manufacturing, human robot collaboration, action planning and policy design, etc.

In this paper, we are concerned with manipulation actions, that is actions performed by agents (humans or robots) on objects, resulting in some physical change of the object. However most of the current AI systems require manually defined semantic rules. In this work, we propose a computational linguistics framework, which is based on probabilistic semantic parsing with Combinatory Categorial Grammar (CCG), to learn manipulation action semantics (lexicon entries) from annotations. We later show that this learned lexicon is able to make our system reason about manipulation action goals beyond just observation. Thus the intelligent system can not only imitate human movements, but also imitate action goals.

Understanding actions by observation and executing them are generally considered as dual problems for intelligent agents. The sensori-motor bridge connecting the two tasks is essential, and a great amount of attention in AI, Robotics as well as Neurophysiology has been devoted to investigating it. Experiments conducted on primates have discovered that certain neurons, the so-called mirror neurons, fire during both observation and execution of identical manipulation tasks (Rizzolatti et al., 2001; Gazzola et al., 2007). This suggests that the same process is involved in both the observation and execution of actions. From a functionalist point of view, such a process should be able to first build up a semantic structure from observations, and then the decomposition of that same structure should occur when the intelligent agent executes commands.

Additionally, studies in linguistics (Steedman, 2002) suggest that the language faculty develops in humans as a direct adaptation of a more primitive apparatus for planning goal-directed action in the world by composing affordances of tools and consequences of actions. It is this more primitive

*Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics
and the 7th International Joint Conference on Natural Language Processing*, pages 676–686,
Beijing, China, July 26-31, 2015. ©2015 Association for Computational Linguistics

apparatus that is our major interest in this paper. Such an apparatus is composed of a "syntax part" and a "semantic part". In the syntax part, every linguistic element is categorized as either a function or a basic type, and is associated with a syntactic category which either identifies it as a function or a basic type. In the semantic part, a semantic translation is attached following the syntactic category explicitly.

Combinatory Categorial Grammar (CCG) introduced by (Steedman, 2000) is a theory that can be used to represent such structures with a small set of combinators such as functional application and type-raising. What do we gain though from such a formal description of action? This is similar to asking what one gains from a formal description of language as a generative system. Chomskys contribution to language research was exactly this: the formal description of language through the formulation of the Generative and Transformational Grammar (Chomsky, 1957). It revolutionized language research opening up new roads for the computational analysis of language, providing researchers with common, generative language structures and syntactic operations, on which language analysis tools were built. A grammar for action would contribute to providing a common framework of the syntax and semantics of action, so that basic tools for action understanding can be built, tools that researchers can use when developing action interpretation systems, without having to start development from scratch. The same tools can be used by robots to execute actions.

In this paper, we propose an approach for learning the semantic meaning of manipulation action through a probabilistic semantic parsing framework based on CCG theory. For example, we want to learn from an annotated training action corpus that the action "Cut" is a function which has two arguments: a subject and a patient. Also, the action consequence of "Cut" is a separation of the patient. Using formal logic representation, our system will learn the semantic representations of "Cut":

$$Cut := (AP\backslash NP)/NP : \lambda x.\lambda y.cut(x,y) \rightarrow divided(y)$$

Here $cut(x,y)$ is a primitive function. We will further introduce the representation in Sec. 3. Since our action representation is in a common calculus form, it enables naturally further logical reasoning beyond visual observation.

The advantage of our approach is twofold: 1) Learning semantic representations from annotations helps an intelligent agent to enrich automatically its own knowledge about actions; 2) The formal logic representation of the action could be used to infer the object-wise consequence after a certain manipulation, and can also be used to plan a set of actions to reach a certain action goal. We further validate our approach on a large publicly available manipulation action dataset (MANIAC) from (Aksoy et al., 2014), achieving promising experimental results. Moreover, we believe that our work, even though it only considers the domain of manipulation actions, is also a promising example of a more closely intertwined computer vision and computational linguistics system. The diagram in Fig.1 depicts the framework of the system.

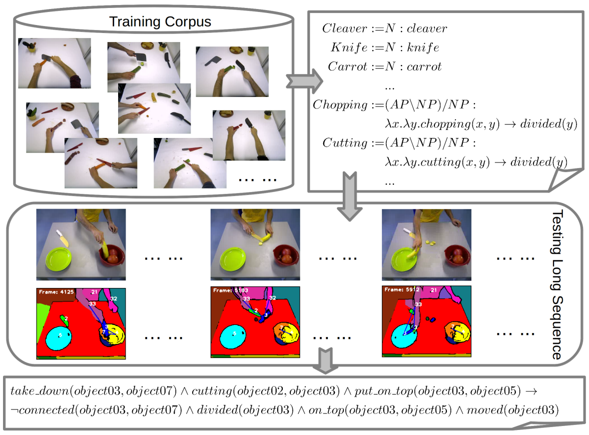

Training Corpus
... ...
$Cleaver := N : cleaver$
$Knife := N : knife$
$Carrot := N : carrot$
...
$Chopping := (AP\backslash NP)/NP :$
$\lambda x.\lambda y.chopping(x,y) \rightarrow divided(y)$
$Cutting := (AP\backslash NP)/NP :$
$\lambda x.\lambda y.cutting(x,y) \rightarrow divided(y)$
...
Testing Long Sequence
Frame: 4125
Frame: 5183
Frame: 5912
$take\_down(object03, object07) \wedge cutting(object02, object03) \wedge put\_on\_top(object03, object05) \rightarrow$
$\neg connected(object03, object07) \wedge divided(object03) \wedge on\_top(object03, object05) \wedge moved(object03)$

Figure 1: A CCG based semantic parsing framework for manipulation actions.

2 Related Works

Reasoning beyond appearance: The very small number of works in computer vision, which aim to reason beyond appearance models, are also related to this paper. (Xie et al., 2013) proposed that beyond state-of-the-art computer vision techniques, we could possibly infer implicit information (such as functional objects) from video, and they call them "Dark Matter" and "Dark Energy". (Yang et al., 2013) used stochastic tracking and graph-cut based segmentation to infer manipulation consequences beyond appearance. (Joo et al., 2014) used a ranking SVM to predict the persuasive motivation (or the intention) of the photographer who captured an image. More recently, (Pirsiavash et al., 2014) seeks to infer the motivation of the person in the image by mining knowledge stored in a large corpus using natural language processing techniques. Different from these fairly general investigations about reasoning beyond appearance, our paper seeks to learn manipulation actions semantics in logic forms through CCG, and further infer hidden action consequences beyond appearance through reasoning.

Action Recognition and Understanding: Human activity recognition and understanding has been studied heavily in Computer Vision recently, and there is a large range of applications for this work in areas like human-computer interactions, biometrics, and video surveillance. Both visual recognition methods, and the non-visual description methods using motion capture systems have been used. A few good surveys of the former can be found in (Moeslund et al., 2006) and (Turaga et al., 2008). Most of the focus has been on recognizing single human actions like walking, jumping, or running etc. (Ben-Arie et al., 2002; Yilmaz and Shah, 2005). Approaches to more complex actions have employed parametric approaches, such as HMMs (Kale et al., 2004) to learn the transition between feature representations in individual frames e.g. (Saisan et al., 2001; Chaudhry et al., 2009). More recently, (Aksoy et al., 2011; Aksoy et al., 2014) proposed a semantic event chain (SEC) representation to model and learn the semantic segment-wise relationship transition from spatial-temporal video segmentation.

There also have been many syntactic approaches to human activity recognition which used the concept of context-free grammars, because such grammars provide a sound theoretical basis for modeling structured processes. Tracing back to the middle 90's, (Brand, 1996) used a grammar to recognize disassembly tasks that contain hand manipulations. (Ryoo and Aggarwal, 2006) used the context-free grammar formalism to recognize composite human activities and multi-person interactions. It is a two level hierarchical approach where the lower-levels are composed of HMMs and Bayesian Networks while the higher-level interactions are modeled by CFGs. To deal with errors from low-level processes such as tracking, stochastic grammars such as stochastic CFGs were also used (Ivanov and Bobick, 2000; Moore and Essa, 2002). More recently, (Kuehne et al., 2014) proposed to model goal-directed human activities using Hidden Markov Models and treat subactions just like words in speech. These works proved that grammar based approaches are practical in activity recognition systems, and shed insight onto human manipulation action understanding. However, as mentioned, thinking about manipulation actions solely from the viewpoint of recognition has obvious limitations. In this work we adopt principles from CFG based activity recognition systems, with extensions to a CCG grammar that accommodates not only the hierarchical structure of human activity but also action semantics representations. It enables the system to serve as the core parsing engine for both manipulation action recognition and execution.

Manipulation Action Grammar: As mentioned before, (Chomsky, 1993) suggested that a minimalist generative grammar, similar to the one of human language, also exists for action understanding and execution. The works closest related to this paper are (Pastra and Aloimonos, 2012; Summers-Stay et al., 2013; Guha et al., 2013). (Pastra and Aloimonos, 2012) first discussed a Chomskyan grammar for understanding complex actions as a theoretical concept, and (Summers-Stay et al., 2013) provided an implementation of such a grammar using as perceptual input only objects. More recently, (Yang et al., 2014) proposed a set of context-free grammar rules for manipulation action understanding, and (Yang et al., 2015) applied it on unconstrained instructional videos. However, these approaches only consider the syntactic structure of manipulation actions without coupling semantic rules using λ expressions, which limits the capability of doing reasoning and prediction.

Combinatory Categorial Grammar and Semantic Parsing: CCG based semantic parsing originally was used mainly to translate natural language sentences to their desired semantic representations as λ-calculus formulas (Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007). (Mooney, 2008) presented a framework of grounded language acquisition: the interpretation of language entities into semantically informed structures in the context of perception and actuation. The concept has been applied successfully in tasks such as robot navigation (Matuszek et al., 2011), forklift operation (Tellex et al., 2014) and of human-robot interaction (Matuszek et al., 2014). In this work, instead of grounding natural language sentences directly, we ground information obtained from visual perception into semantically informed structures, specifically in the domain of manipulation actions.

3 A CCG Framework for Manipulation Actions

Before we dive into the semantic parsing of manipulation actions, a brief introduction to the Combinatory Categorial Grammar framework in Linguistics is necessary. We will only introduce related concepts and formalisms. For a complete background reading, we would like to refer readers to (Steedman, 2000). We will first give a brief introduction to CCG and then introduce a fundamental combinator, i.e., functional application. The introduction is followed by examples to show how the combinator is applied to parse actions.

3.1 Manipulation Action Semantics

The semantic expression in our representation of manipulation actions uses a typed λ-calculus language. The formal system has two basic types: entities and functions. Entities in manipulation actions are Objects or Hands, and functions are the Actions. Our $lambda$-calculus expressions are formed from the following items:

Constants: Constants can be either entities or functions. For example, *Knife* is an entity (i.e., it is of type N) and *Cucumber* is an entity too (i.e., it is of type N). *Cut* is an action function that maps entities to entities. When the event *Knife Cut Cucumber* happened, the expression *cut(Knife, Cucumber)* returns an entity of type AP, aka. Action Phrase. Constants like *divided* are status functions that map entities to truth values. The expression $divided(cucumber)$ returns a true value after the event (*Knife Cut Cucumber*) happened.

Logical connectors: The λ-calculus expression has logical connectors like conjunction ($\wedge$), disjunction ($\vee$), negation($\neg$) and implication($\rightarrow$).

For example, the expression

$$connected(tomato, cucumber) \wedge \\ divided(tomato) \wedge divided(cucumber)$$

represents the joint status that the sliced *tomato* merged with the sliced *cucumber*. It can be regarded as a simplified goal status for "making a cucumber tomato salad". The expression $\neg connected(spoon, bowl)$ represents the status after the *spoon* finished stirring the *bowl*.

$$\lambda x.cut(x, cucumber) \rightarrow divided(cucumber)$$

represents that if the *cucumber* is cut by x, then the status of the *cucumber* is divided.

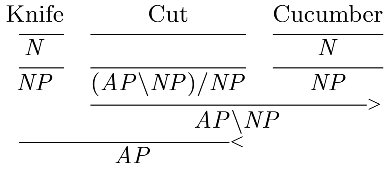
λ **expressions**: $lambda$ expressions represent functions with unknown arguments. For example, $\lambda x.cut(knife, x)$ is a function from entities to entities, which is of type NP after any entities of type N that is cut by *knife*.

3.2 Combinatory Categorial Grammar

The semantic parsing formalism underlying our framework for manipulation actions is that of combinatory categorial grammar (CCG) (Steedman, 2000). A CCG specifies one or more logical forms for each element or combination of elements for manipulation actions. In our formalism, an element of Action is associated with a syntactic "category" which identifies it as functions, and specifies the type and directionality of their arguments and the type of their result. For example, action "Cut" is a function from patient object phrase (NP) on the right into predicates, and into functions from subject object phrase (NP) on the left into a sub action phrase (AP):

$$Cut := (AP\backslash NP)/NP. \tag{1}$$

As a matter of fact, the pure categorial grammar is a conext-free grammar presented in the accepting, rather than the producing direction. The expression (1) is just an accepting form for Action "Cut" following the context-free grammar. While it is now convenient to write derivations as follows, they are equivalent to conventional tree structure derivations in Figure. 3.2.



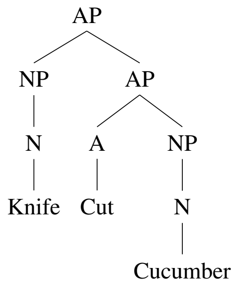


Figure 2: Example of conventional tree structure.

The semantic type is encoded in these categories, and their translation can be made explicit in an expanded notation. Basically a λ-calculus expression is attached with the syntactic category. A colon operator is used to separate syntactical and semantic expressions, and the right side of the colon is assumed to have lower precedence than the left side of the colon. Which is intuitive as any explanation of manipulation actions should first obey syntactical rules, then semantic rules. Now the basic element, Action "Cut", can be further represented by:

$$Cut := (AP\backslash NP)/NP : \lambda x.\lambda y.cut(x,y) \rightarrow divided(y).$$

$(AP\backslash NP)/NP$ denotes a phrase of type AP, which requires an element of type NP to specify what object was cut, and requires another element of type NP to further complement what effector initiates the cut action. $\lambda x.\lambda y.cut(x,y)$ is the λ-calculus representation for this function. Since the functions are closely related to the state update, $\rightarrow divided(y)$ further points out the status expression after the action was performed.

A CCG system has a set of combinatory rules which describe how adjacent syntatic categories in a string can be recursively combined. In the setting of manipulation actions, we want to point out that similar combinatory rules are also applicable. Especially the functional application rules are essential in our system.

3.3 Functional application

The functional application rules with semantics can be expressed in the following form:

$$A/B : f \quad B : g => A : f(g) \tag{2}$$

$$B : g \quad A\backslash B : f => A : f(g) \tag{3}$$

Rule. (2) says that a string with type A/B can be combined with a right-adjacent string of type B to form a new string of type A. At the same time, it also specifies how the semantics of the category A can be compositionally built out of the semantics for A/B and B. Rule. (3) is a symmetric form of Rule. (2).

In the domain of manipulation actions, following derivation is an example CCG parse. This parse shows how the system can parse an observation ("Knife Cut Cucumber") into a semantic representation ($cut(knife, cucumber) \rightarrow divided(cucumber)$) using the functional application rules.

$$\begin{array}{ccc}
\text{Knife} & \text{Cut} & \text{Cucumber}\\
\hline
N & & N\\
\hline
NP & (AP\backslash NP)/NP & NP\\
knife & \lambda x.\lambda y.cut(x,y) & cucumber\\
knife & \rightarrow divided(y) & cucumber\\
\cline{2-3}
 & \multicolumn{2}{c}{AP\backslash NP \qquad >}\\
 & \multicolumn{2}{c}{\lambda x.cut(x, cucumber)}\\
 & \multicolumn{2}{c}{\rightarrow divided(cucumber)}\\
\hline
\multicolumn{3}{c}{AP \qquad <}\\
\multicolumn{3}{c}{cut(knife, cucumber)}\\
\multicolumn{3}{c}{\rightarrow divided(cucumber)}
\end{array}$$

4 Learning Model and Semantic Parsing

After having defined the formalism and application rule, instead of manually writing down all the possible CCG representations for each entity, we would like to apply a learning technique to derive them from the paired training corpus. Here we adopt the learning model of (Zettlemoyer and Collins, 2005), and use it to assign weights to the semantic representation of actions. Since an action may have multiple possible syntactic and semantic representations assigned to it, we use the probabilistic model to assign weights to these representations.

4.1 Learning Approach

First we assume that complete syntactic parses of the observed action are available, and in fact a manipulation action can have several different parses. The parsing uses a probabilistic combinatorial categorial grammar framework similar to the one given by (Zettlemoyer and Collins, 2007). We assume a probabilistic categorial grammar (PCCG) based on a log linear model. M denotes a manipulation task, L denotes the semantic representation of the task, and T denotes its parse tree. The probability of a particular syntactic and semantic parse is given as:

$$P(L,T|M;\Theta) = \frac{e^{f(L,T,M)\cdot\Theta}}{\sum_{(L,T)} e^{f(L,T,M)\cdot\Theta}} \tag{4}$$

where f is a mapping of the triple (L,T,M) to feature vectors $\in R^d$, and the $\Theta \in R^d$ represents the weights to be learned. Here we use only lexical features, where each feature counts the number of times a lexical entry is used in T. Parsing a manipulation task under PCCG equates to finding L such that $P(L|M;\Theta)$ is maximized:

$$\begin{aligned} & argmax_L P(L|M;\Theta) \\ &= argmax_L \sum_T P(L,T|M;\Theta). \end{aligned} \tag{5}$$

We use dynamic programming techniques to calculate the most probable parse for the manipulation task. In this paper, the implementation from (Baral et al., 2011) is adopted, where an inverse-λ technique is used to generalize new semantic representations. The generalization of lexicon rules are essential for our system to deal with unknown actions presented during the testing phase.

5 Experiments

5.1 Manipulation Action (MANIAC) Dataset

(Aksoy et al., 2014) provides a manipulation action dataset with 8 different manipulation actions (cutting, chopping, stirring, putting, taking, hiding, uncovering, and pushing), each of which consists of 15 different versions performed by 5 different human actors[1]. There are in total 30 different objects manipulated in all demonstrations. All manipulations were recorded with the Microsoft Kinect sensor and serve as **training** data here.

The MANIAC data set contains another 20 long and complex chained manipulation sequences (e.g. "making a sandwich") which consist of a total of 103 different versions of these 8 manipulation tasks performed in different orders with novel objects under different circumstances. These serve as **testing** data for our experiments.

(Aksoy et al., 2014; Aksoy and Wörgötter, 2015) developed a semantic event chain based model free decomposition approach. It is an unsupervised probabilistic method that measures the frequency of the changes in the spatial relations embedded in event chains, in order to extract the subject and patient visual segments. It also decomposes the long chained complex testing actions into their primitive action components according to the spatio-temporal relations of the manipulator. Since the visual recognition is not the core of this work, we omit the details here and refer the interested reader to (Aksoy et al., 2014; Aksoy and Wörgötter, 2015). All these features make the MANIAC dataset a great testing bed for both the theoretical framework and the implemented system presented in this work.

5.2 Training Corpus

We first created a training corpus by annotating the 120 training clips from the MANIAC dataset, in the format of observed triplets (subject action patient) and a corresponding semantic representation of the action as well as its consequence. The semantic representations in λ-calculus format are given by human annotators after watching each action clip. A set of sample training pairs are given in Table.1 (one from each action category in the training set). Since every training clip contains one single full execution of each manipulation action considered, the training corpus thus has a total of 120 paired training samples.

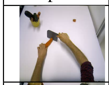
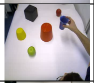
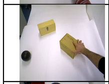
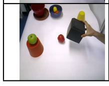
Snapshot	triplet	semantic representation
	cleaver chopping carrot	$chopping(cleaver, carrot)$ $\rightarrow divided(carrot)$
	spatula cutting pepper	$cutting(spatula, pepper)$ $\rightarrow divided(pepper)$
	spoon stirring bucket	$stirring(spoon, bucket)$
	cup take_down bucket	$take\_down(cup, bucket)$ $\rightarrow \neg connected(cup, bucket)$ $\wedge moved(cup)$
	cup put_on_top bowl	$put\_on\_top(cup, bowl)$ $\rightarrow on\_top(cup, bowl)$ $\wedge moved(cup)$
	bucket hiding ball	$hiding(bucket, ball)$ $\rightarrow contained(bucket, ball)$ $\wedge moved(bucket)$
	hand pushing box	$pushing(hand, box)$ $\rightarrow moved(box)$
	box uncover apple	$uncover(box, apple)$ $\rightarrow appear(apple)$ $\wedge moved(box)$

Table 1: Example annotations from training corpus, one per manipulation action category.

We also assume the system knows that every "object" involved in the corpus is an entity of its own type, for example:

$$Knife := N : knife$$
$$Bowl := N : bowl$$
$$......$$

Additionally, we assume the syntactic form of each "action" has a main type $(AP\backslash NP)/NP$ (see Sec. 3.2). These two sets of rules form the initial seed lexicon for learning.

5.3 Learned Lexicon

We applied the learning technique mentioned in Sec. 4, and we used the NL2KR implementation from (Baral et al., 2011). The system learns and generalizes a set of lexicon entries (syntactic and semantic) for each action categories from the training corpus accompanied with a set of weights.

[1] Dataset available for download at https://fortknox.physik3.gwdg.de/cns/index.php?page=maniac-dataset.

We list the one with the largest weight for each action here respectively:

$$\begin{aligned}
Chopping :=&(AP\backslash NP)/NP : \lambda x.\lambda y.chopping(x,y)\\
&\rightarrow divided(y)\\
Cutting :=&(AP\backslash NP)/NP : \lambda x.\lambda y.cutting(x,y)\\
&\rightarrow divided(y)\\
Stirring :=&(AP\backslash NP)/NP : \lambda x.\lambda y.stirring(x,y)\\
Take\_down :=&(AP\backslash NP)/NP : \lambda x.\lambda y.take\_down(x,y)\\
&\rightarrow \neg connected(x,y) \wedge moved(x)\\
Put\_on\_top :=&(AP\backslash NP)/NP : \lambda x.\lambda y.put\_on\_top(x,y)\\
&\rightarrow on\_top(x,y) \wedge moved(x)\\
Hiding :=&(AP\backslash NP)/NP : \lambda x.\lambda y.hiding(x,y)\\
&\rightarrow contained(x,y) \wedge moved(x)\\
Pushing :=&(AP\backslash NP)/NP : \lambda x.\lambda y.pushing(x,y)\\
&\rightarrow moved(y)\\
Uncover :=&(AP\backslash NP)/NP : \lambda x.\lambda y.uncover(x,y)\\
&\rightarrow appear(y) \wedge moved(x).
\end{aligned}$$

The set of seed lexicon and the learned lexicon entries are further used to probabilistically parse the detected triplet sequences from the 20 long manipulation activities in the testing set.

5.4 Deducing Semantics

Using the decomposition technique from (Aksoy et al., 2014; Aksoy and Wörgötter, 2015), the reported system is able to detect a sequence of action triplets in the form of (Subject Action Patient) from each of the testing sequence in MANIAC dataset. Briefly speaking, the event chain representation (Aksoy et al., 2011) of the observed long manipulation activity is first scanned to estimate the main manipulator, i.e. the hand, and manipulated objects, e.g. knife, in the scene without employing any visual feature-based object recognition method. Solely based on the interactions between the hand and manipulated objects in the scene, the event chain is partitioned into chunks. These chunks are further fragmented into subunits to detect parallel action streams. Each parsed Semantic Event Chain (SEC) chunk is then compared with the model SECs in the library to decide whether the current SEC sample belongs to one of the known manipulation models or represents a novel manipulation. SEC models, stored in the library, are learned in an on-line unsupervised fashion using the semantics of manipulations derived from a given set of training data in order to create a large vocabulary of single atomic manipulations.

For the different testing sequence, the number of triplets detected ranges from two to seven. In total, we are able to collect 90 testing detections and they serve as the testing corpus. However, since many of the objects used in the testing data are not present in the training set, an object model-free approach is adopted and thus "subject" and "patient" fields are filled with segment IDs instead of a specific object name. Fig. 3 and 4 show several examples of the detected triplets accompanied with a set of key frames from the testing sequences. Nevertheless, the method we used here can 1) generalize the unknown segments into the category of object entities and 2) generalize the unknown actions (those that do not exist in the training corpus) into the category of action function. This is done by automatically generalizing the following two types of lexicon entries using the inverse-λ technique from (Baral et al., 2011):

$$\begin{aligned}
Object\_[ID] :=&N : object\_[ID]\\
Unknown :=&(AP\backslash NP)/NP : \lambda x.\lambda y.unknown(x,y)
\end{aligned}$$

Among the 90 detected triplets, using the learned lexicon we are able to parse all of them into semantic representations. Here we pick the representation with the highest probability after parsing as the individual action semantic representation. The "parsed semantics" rows of Fig. 3 and 4 show several example action semantics on testing sequences. Taking the fourth sub-action from Fig. 4 as an example, the visually detected triplets based on segmentation and spatial decomposition is $(Object\_014, Chopping, Object\_011)$. After semantic parsing, the system predicts that $divided(Object\_011)$. The complete training corpus and parsed results of the testing set will be made publicly available for future research.

5.5 Reasoning Beyond Observations

As mentioned before, because of the use of λ-calculus for representing action semantics, the obtained data can naturally be used to do logical reasoning beyond observations. This by itself is a very interesting research topic and it is beyond this paper's scope. However by applying a couple of common sense Axioms on the testing data, we can provide some flavor of this idea.

Case study one: See the "final action consequence and reasoning" row of Fig. 3 for case one. Using propositional logic and axiom schema, we can represent the common sense statement ("if an object x is contained in object y, and object z is on top of object y, then object z is on top of object x") as follows:

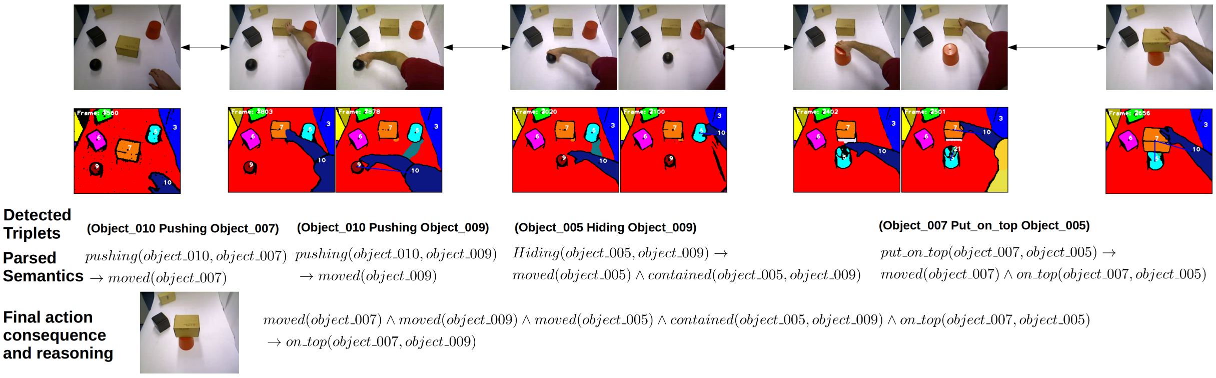

Detected Triplets
(Object_010 Pushing Object_007)
(Object_010 Pushing Object_009)
(Object_005 Hiding Object_009)
(Object_007 Put_on_top Object_005)
Parsed Semantics
pushing(object_010, object_007) → moved(object_007)
pushing(object_010, object_009) → moved(object_009)
Hiding(object_005, object_009) → moved(object_005) ∧ contained(object_005, object_009)
put_on_top(object_007, object_005) → moved(object_007) ∧ on_top(object_007, object_005)
Final action consequence and reasoning
moved(object_007) ∧ moved(object_009) ∧ moved(object_005) ∧ contained(object_005, object_009) ∧ on_top(object_007, object_005) → on_top(object_007, object_009)

Figure 3: System output on complex chained manipulation testing sequence one. The segmentation output and detected triplets are from (Aksoy and Wörgötter, 2015)

.

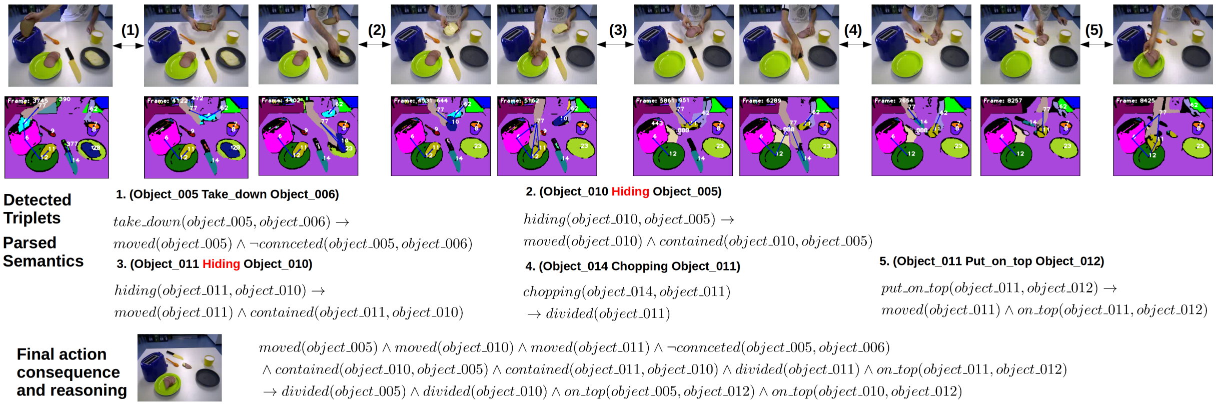

(1)
(2)
(3)
(4)
(5)
Detected Triplets
Parsed Semantics
1. (Object_005 Take_down Object_006)
take_down(object_005, object_006) → moved(object_005) ∧ ¬connceted(object_005, object_006)
2. (Object_010 Hiding Object_005)
hiding(object_010, object_005) → moved(object_010) ∧ contained(object_010, object_005)
3. (Object_011 Hiding Object_010)
hiding(object_011, object_010) → moved(object_011) ∧ contained(object_011, object_010)
4. (Object_014 Chopping Object_011)
chopping(object_014, object_011) → divided(object_011)
5. (Object_011 Put_on_top Object_012)
put_on_top(object_011, object_012) → moved(object_011) ∧ on_top(object_011, object_012)
Final action consequence and reasoning
moved(object_005) ∧ moved(object_010) ∧ moved(object_011) ∧ ¬connceted(object_005, object_006) ∧ contained(object_010, object_005) ∧ contained(object_011, object_010) ∧ divided(object_011) ∧ on_top(object_011, object_012) → divided(object_005) ∧ divided(object_010) ∧ on_top(object_005, object_012) ∧ on_top(object_010, object_012)

Figure 4: System output on the 18th complex chained manipulation testing sequence. The segmentation output and detected triplets are from (Aksoy and Wörgötter, 2015)

.

Axiom (1): $\exists x, y, z, contained(y, x) \wedge on\_top(z, y) \rightarrow on\_top(z, x)$.

Then it is trivial to deduce an additional final action consequence in this scenario that $(on\_top(object\_007, object\_009))$. This matches the fact: the yellow box which is put on top of the red bucket is also on top of the black ball.

Case study two: See the "final action consequence and reasoning" row of Fig. 4 for a more complicated case. Using propositional logic and axiom schema, we can represent three common sense statements:

1) "if an object y is contained in object x, and object z is contained in object y, then object z is contained in object x";

2) "if an object x is contained in object y, and object y is divided, then object x is divided";

3) "if an object x is contained in object y, and object y is on top of object z, then object x is on top of object z" as follows:

Axiom (2): $\exists x, y, z, contained(y, x) \wedge contained(z, y) \rightarrow contained(z, x)$.

Axiom (3): $\exists x, y, contained(y, x) \wedge divided(y) \rightarrow divided(x)$.

Axiom (4): $\exists x, y, z, contained(y, x) \wedge on\_top(y, z) \rightarrow on\_top(x, z)$.

With these common sense Axioms, the system is able to deduce several additional final action consequences in this scenario:

$$\begin{aligned} & divided(object\_005) \wedge divided(object\_010) \\ & \wedge on\_top(object\_005, object\_012) \\ & \wedge on\_top(object\_010, object\_012). \end{aligned}$$

From Fig. 4, we can see that these additional consequences indeed match the facts: 1) the bread and cheese which are covered by ham are also divided, even though from observation the system only detected the ham being cut; 2) the divided bread and cheese are also on top of the plate, even though from observation the system only detected the ham being put on top of the plate.

We applied the four Axioms on the 20 testing action sequences and deduced the "hidden" consequences from observation. To evaluate our system performance quantitatively, we first annotated all the final action consequences (both obvious and "hidden" ones) from the 20 testing sequences as ground-truth facts. In total there are 122 consequences annotated. Using perception only (Aksoy and Wörgötter, 2015), due to the decomposition errors (such as the red font ones in Fig. 4) the system can detect 91 consequences correctly, yielding a 74% correct rate. After applying the four Axioms and reasoning, our system is able to detect 105 consequences correctly, yielding a 86% correct rate. Overall, this is a 15.4% of improvement.

Here we want to mention a caveat: there are definitely other common sense Axioms that we are not able to address in the current implementation. However, from the case studies presented, we can see that using the presented formal framework, our system is able to reason about manipulation action goals instead of just observing what is happening visually. This capability is essential for intelligent agents to imitate action goals from observation.

6 Conclusion and Future Work

In this paper we presented a formal computational framework for modeling manipulation actions based on a Combinatory Categorial Grammar. An empirical study on a large manipulation action dataset validates that 1) with the introduced formalism, a learning system can be devised to deduce the semantic meaning of manipulation actions in λ-schema; 2) with the learned schema and several common sense Axioms, our system is able to reason beyond just observation and deduce "hidden" action consequences, yielding a decent performance improvement.

Due to the limitation of current testing scenarios, we conducted experiments only considering a relatively small set of seed lexicon rules and logical expressions. Nevertheless, we want to mention that the presented CCG framework can also be extended to learn the formal logic representation of more complex manipulation action semantics. For example, the temporal order of manipulation actions can be modeled by considering a seed rule such as $AP\backslash AP : \lambda f.\lambda g.before(f(\cdot), g(\cdot))$, where $before(\cdot,\cdot)$ is a temporal predicate. For actions in this paper we consider seed main type $(AP\backslash NP)/NP$. For more general manipulation scenarios, based on whether the action is transitive or intransitive, the main types of action can be extended to include $AP\backslash NP$.

Moreover, the logical expressions can also be extended to include universal quantification $\forall$ and existential quantification $\exists$. Thus, manipulation action such as "knife cut every tomato" can be parsed into a representation as $\forall x.tomato(x) \wedge cut(knife, x) \rightarrow divided(x)$ (the parse is given in the following chart). Here, the concept "every" has a main type of $NP\backslash NP$ and semantic meaning of $\forall x.f(x)$. The same framework can also extended to have other combinatory rules such as **composition** and **type-raising** (Steedman, 2002). These are parts of the future work along the line of the presented work.

$$
\begin{array}{cccc}
\text{Knife} & \text{Cut} & \text{every} & \text{Tomato} \\
\hline
N & & & N \\
\hline
NP & (AP\backslash NP)/NP & NP\backslash NP & NP \\
knife & \lambda x.\lambda y.cut(x,y) & \forall x.f(x) & tomato \\
knife & \rightarrow divided(y) & \forall x.f(x) & tomato \\
\cline{3-4}
 & & \multicolumn{2}{c}{NP \quad >} \\
 & & \multicolumn{2}{c}{\forall x.tomato(x)} \\
\cline{2-4}
 & \multicolumn{3}{c}{AP\backslash NP \quad >} \\
 & \multicolumn{3}{c}{\forall y.\lambda x.tomato(y) \wedge cut(x,y) \rightarrow divided(y)} \\
\hline
\multicolumn{4}{c}{AP \quad <} \\
\multicolumn{4}{c}{\forall y.tomato(y) \wedge cut(knife,y) \rightarrow divided(y)}
\end{array}
$$

The presented computational linguistic framework enables an intelligent agent to predict and reason action goals from observation, and thus has many potential applications such as human intention prediction, robot action policy planning, human robot collaboration etc. We believe that our formalism of manipulation actions bridges computational linguistics, vision and robotics, and opens further research in Artificial Intelligence and Robotics. As the robotics industry is moving towards robots that function safely, effectively and autonomously to perform tasks in real-world unstructured environments, they will need to be able to understand the meaning of actions and acquire human-like common-sense reasoning capabilities.

7 Acknowledgements

This research was funded in part by the support of the European Union under the Cognitive Systems program (project POETICON++), the National Science Foundation under INSPIRE grant SMA 1248056, and by DARPA through U.S. Army grant W911NF-14-1-0384 under the Project: Shared Perception, Cognition and Reasoning for Autonomy.

References

E E. Aksoy and F. Wörgötter. 2015. Semantic decomposition and recognition of long and complex manipulation action sequences. *International Journal of Computer Vision*, page Under Review.

E.E. Aksoy, A. Abramov, J. Dörr, K. Ning, B. Dellen, and F. Wörgötter. 2011. Learning the semantics of object–action relations by observation. *The International Journal of Robotics Research*, 30(10):1229–1249.

E E. Aksoy, M. Tamosiunaite, and F. Wörgötter. 2014. Model-free incremental learning of the semantics of manipulation actions. *Robotics and Autonomous Systems*, pages 1–42.

Chitta Baral, Juraj Dzifcak, Marcos Alvarez Gonzalez, and Jiayu Zhou. 2011. Using inverse λ and generalization to translate english to formal languages. In *Proceedings of the Ninth International Conference on Computational Semantics*, pages 35–44. Association for Computational Linguistics.

Jezekiel Ben-Arie, Zhiqian Wang, Purvin Pandit, and Shyamsundar Rajaram. 2002. Human activity recognition using multidimensional indexing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(8):1091–1104.

Matthew Brand. 1996. Understanding manipulation in video. In *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, pages 94–99, Killington,VT. IEEE.

R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal. 2009. Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions. In *Proceedings of the 2009 IEEE Intenational Conference on Computer Vision and Pattern Recognition*, pages 1932–1939, Miami,FL. IEEE.

N. Chomsky. 1957. *Syntactic Structures*. Mouton de Gruyter.

Noam Chomsky. 1993. *Lectures on government and binding: The Pisa lectures*. Walter de Gruyter.

V Gazzola, G Rizzolatti, B Wicker, and C Keysers. 2007. The anthropomorphic brain: the mirror neuron system responds to human and robotic actions. *Neuroimage*, 35(4):1674–1684.

Anupam Guha, Yezhou Yang, Cornelia Fermüller, and Yiannis Aloimonos. 2013. Minimalist plans for interpreting manipulation actions. *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5908–5914.

Yuri A. Ivanov and Aaron F. Bobick. 2000. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):852–872.

Jungseock Joo, Weixin Li, Francis F Steen, and Song-Chun Zhu. 2014. Visual persuasion: Inferring communicative intents of images. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 216–223. IEEE.

A. Kale, A. Sundaresan, AN Rajagopalan, N.P. Cuntoor, A.K. Roy-Chowdhury, V. Kruger, and R. Chellappa. 2004. Identification of humans using gait. *IEEE Transactions on Image Processing*, 13(9):1163–1173.

Hilde Kuehne, Ali Arslan, and Thomas Serre. 2014. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 780–787. IEEE.

Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. 2011. A joint model of language and perception for grounded attribute learning. In *International Conference on Machine learning (ICML)*.

Cynthia Matuszek, Liefeng Bo, Luke Zettlemoyer, and Dieter Fox. 2014. Learning from unscripted deictic gesture and language for human-robot interactions. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

T.B. Moeslund, A. Hilton, and V. Krüger. 2006. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2):90–126.

Raymond J Mooney. 2008. Learning to connect language and perception. In *AAAI*, pages 1598–1601.

Darnell Moore and Irfan Essa. 2002. Recognizing multitasked activities from video using stochastic context-free grammar. In *Proceedings of the National Conference on Artificial Intelligence*, pages 770–776, Menlo Park, CA. AAAI.

K. Pastra and Y. Aloimonos. 2012. The minimalist grammar of action. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 367(1585):103–117.

Hamed Pirsiavash, Carl Vondrick, and Antonio Torralba. 2014. Inferring the why in images. *arXiv preprint arXiv:1406.5472*.

Giacomo Rizzolatti, Leonardo Fogassi, and Vittorio Gallese. 2001. Neurophysiological mechanisms underlying the understanding and imitation of action. *Nature Reviews Neuroscience*, 2(9):661–670.

Michael S Ryoo and Jake K Aggarwal. 2006. Recognition of composite human activities through context-free grammar based representation. In *Proceedings of the 2006 IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1709–1718, New York City, NY. IEEE.

P. Saisan, G. Doretto, Y.N. Wu, and S. Soatto. 2001. Dynamic texture recognition. In *Proceedings of the 2001 IEEE Intenational Conference on Computer Vision and Pattern Recognition*, volume 2, pages 58–63, Kauai, HI. IEEE.

Mark Steedman. 2000. *The syntactic process*, volume 35. MIT Press.

Mark Steedman. 2002. Plans, affordances, and combinatory grammar. *Linguistics and Philosophy*, 25(5-6):723–753.

D. Summers-Stay, C.L. Teo, Y. Yang, C. Fermüller, and Y. Aloimonos. 2013. Using a minimal action grammar for activity understanding in the real world. In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4104–4111, Vilamoura, Portugal. IEEE.

Stefanie Tellex, Pratiksha Thaker, Joshua Joseph, and Nicholas Roy. 2014. Learning perceptually grounded word meanings from unaligned parallel data. *Machine Learning*, 94(2):151–167.

P. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea. 2008. Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1473–1488.

Dan Xie, Sinisa Todorovic, and Song-Chun Zhu. 2013. Inferring "dark matter" and "dark energy" from videos. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2224–2231. IEEE.

Yezhou Yang, Cornelia Fermüller, and Yiannis Aloimonos. 2013. Detection of manipulation action consequences (MAC). In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2563–2570, Portland, OR. IEEE.

Y. Yang, A. Guha, C. Fermuller, and Y. Aloimonos. 2014. A cognitive system for understanding human manipulation actions. *Advances in Cognitive Sysytems*, 3:67–86.

Yezhou Yang, Yi Li, Cornelia Fermuller, and Yiannis Aloimonos. 2015. Robot learning manipulation action plans by "watching" unconstrained videos from the world wide web. In *The Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI-15)*.

A. Yilmaz and M. Shah. 2005. Actions sketch: A novel action representation. In *Proceedings of the 2005 IEEE Intenational Conference on Computer Vision and Pattern Recognition*, volume 1, pages 984–989, San Diego, CA. IEEE.

Luke S Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI*.

Luke S Zettlemoyer and Michael Collins. 2007. Online learning of relaxed ccg grammars for parsing to logical form. In *EMNLP-CoNLL*, pages 678–687.