A first-order-logic based model for grounded language learning

Leonor Becerra-Bonache[1], Hendrik Blockeel[2,3], María Galván[1], and François Jacquenet[1]

[1] Laboratoire Hubert Curien, Université Jean Monnet
[2] Department of Computer Science, KU Leuven
[3] Leiden Institute of Advanced Computer Science, Leiden University

Abstract. Much is still unknown about how children learn language, but it is clear that they perform "grounded" language learning: they learn the grammar and vocabulary not just from examples of sentences, but from examples of sentences in a particular context. Grounded language learning has been the subject of much research. Most of this work focuses on particular aspects, such as constructing semantic parsers, or on particular types of applications. In this paper, we take a broader view that includes an aspect that has received little attention until now: learning the meaning of phrases from phrase/context pairs in which the phrase's meaning is not explicitly represented. We propose a simple model for this task that uses first-order logic representations for contexts and meanings, including a simple incremental learning algorithm. We experimentally demonstrate that the proposed model can explain the gradual learning of simple concepts and language structure, and that it can easily be used for interpretation, generation, and translation of phrases.

1 Introduction

Despite the complexity of natural languages, children are able to acquire their native language quite easily, efficiently and without any specific training. This human ability is not yet fully understood, even though it has been studied by researchers from many different areas: psychology, linguistics, ...

In computer science, grammatical inference (GI) deals with the learning of grammars and languages from data. Although historically this task has been associated with that of children acquiring their native language, most research in GI reduces the language learning problem to syntax learning, and does not use semantic information in this process [7]. Children, however, do have additional information, derived from the context in which utterances are made, and they learn not only the syntax but also the semantics of utterances. This type of learning is often called *grounded language learning*.

Grounded language learning is a broad research area. Most of this research focuses on particular aspects of the problem, such as semantic parsing (which maps sentences to their semantics), or on specific types of applications. In this paper, we propose a model for grounded language learning that uses a first-order

logic representation of contexts and meanings. We present a learning algorithm that analyzes utterances and the context in which they are produced to create a language model that can be used to map sentences onto meanings and vice versa. An important difference with earlier work [9, 4, 3, 11, 13] is that our learner learns from sentence/context pairs, whereas earlier work requires the meaning, or a set of candidate meanings, to be provided for each example sentence. That is, earlier work used examples of the form (x, y), with $y = f(x)$, or (x, Y) with $f(x) \in Y$, to learn the function f that maps sentences to their meaning; our work uses examples of the form (x, C) with a more complex relationship between the context description C and the meaning $f(x)$.

2 Background and related work

Our work is mainly inspired by Angluin and Becerra-Bonache [1, 2], who present, for the first time in the field of GI, a computational model that takes into account semantics for language learning. Their main goal was to investigate the effects of semantics and meaning-preserving corrections on the language learning process. Our work mainly differs from their work in the type of methods adopted to solve the problem. Instead of using a variety of techniques such as transducers, co-occurrence graphs, decision trees, etc., we use a single model represented in first order logic. Language comprehension and generation are then achieved with a simple query to this model. The simplicity of our approach is appealing from a cognitive point of view.

Within computational linguistics, much research exists on grounded language learning. Different from GI, most of this work does not aim to learn the grammar itself or understand the learning process, but to develop systems that can deal with natural language data in a particular application. Much of it focuses on semantic parsing. Semantic parsers are often learned from a *supervised* corpus containing sentences with their meaning representation [20, 15, 21], but constructing such a corpus is expensive, difficult and time-consuming.

In order to avoid this limitation, researchers have investigated grounded learning from *ambiguous* training data, where each sentence is associated with a small set of candidate meanings. In seminal work on grounded language learning, Siskind [19] focused on learning word meaning, but not grammatical structure. In a series of work, Mooney and colleagues [9, 4, 3, 11] learn to match phrases to elements of a context. Their goals are similar to ours, but an essential difference is that they only map a phrase to a single element in the context; that is, the meaning of the phrase must be a single element of the context. Chen et al. [3] acknowledge this limitation, and mention inductive logic programming (ILP) as a possible approach to learning more complex meanings, to be explored in future work. This work partially fills that gap.

Our learner is incremental, which is interesting from a cognitive point of view; apart from Angluin and Becerra-Bonache, the only other incremental learner we know of was proposed by Kwiatkowski et al. [13]. Our approach differs substantially from Kwiatkowski's in that the latter requires a parallel corpus: it can learn

from ambiguous supervision (multiple candidate meanings per sentence), but for each sentence the correct meaning must still be constructed in advance and made available to the learner, whereas our approach construes meanings from scratch. Our context descriptions are motivated by how the learner perceives the world, not by what meanings might look like.

3 A logic-based approach

3.1 Terminology

We assume familiarity with first-order logic and Prolog. We here briefly review the main concepts; for an extensive introduction, see [14].

First-order predicate logic allows us to make statements about objects in some universe U. A *constant* always refers to one and the same object from U. A *variable* may refer to any object. A *term* is a variable or a constant. A *predicate* refers to a relation over U^n for some $n \geq 0$; n is the *arity* of the predicate.

An *atom* is of the form $p(t_1, t_2, \ldots, t_n)$ where p is a predicate symbol and the t_i are terms. A *literal* is of the form A or $\neg A$, with A an atom. A *clause* is a set of literals. A *fact* is a clause with exactly one, positive, literal. An atom, literal or clause is *ground* if it does not contain variables. A *variable substitution* $\{X_1/t_1, X_2/t_2, \ldots, X_n/t_n\}$ is an operation that, applied to a structure, simultaneously replaces each occurrence of variable X_i in that structure by term t_i. An *instantiation* is a substitution that changes all variables into ground terms.

A ground atom $p(t_1, t_2, \ldots, t_n)$ evaluates to true if the tuple denoted by its arguments is in the relation denoted by p, and false otherwise; literal $\neg A$ evaluates to true if and only if A evaluates to false and vice versa; a clause evaluates to true if for each of its instantiations, at least one of its elements evaluates to true (so it represents a universally quantified disjunction).

We use Prolog syntax: predicates and constants start with a lowercase letter, variables with an uppercase letter, and a clause of the form $\{H, \neg B_1, \ldots, \neg B_m\}$ is written as H `:-` B_1`,` `...,` B_m. The meaning of such a clause is equivalent to "if all B_i hold, then H holds" (for each instantiation of the variables). The symbol `_` denotes an anonymous variable occurring in only that position.

In Prolog, clauses can be added dynamically to a knowledge base using `assert`, and removed using `retract`.

Example 1. `father(joseph, hendrik)` is a ground atom. When stated as a fact, it expresses that the entities referred to by the constants `joseph` and `hendrik` are in a relationship referred to as father; or, briefly: "Joseph is the father of Hendrik". The clause `ancestor(adam,X) :- human(X)` expresses that Adam is an ancestor of all humans ("for all X it holds that if X is human, Adam is an ancestor of X").

3.2 Representation

In our learning setting, an example is a pair (C, S) where C is a context and S is a phrase (a sentence or a part of it). A phrase is represented as a sequence of

Category	Constants	Interpretation
Color	`bl, re, gr, ye, or, pu`	blue, red, green, yellow, orange, purple
Shape	`sq, di, tr, he, st, el`	square, disc, triangle, hexagon, star, ellipse
Size	`sm, me, bg`	small, medium-sized, big
Position	`ab, be, lo, ro`	above, below, left of, right of

Table 1. Constants used in context descriptions, and their interpretation.

words, a context as a set of ground facts. Our model assumes that everything a phrase refers to is in the context, but not everything in the context is necessarily referred to in the phrase.

In our examples and experiments, we use a simplified world model with colorful figures, inspired by Feldman et al. [5]. In context descriptions, we use predicates of which it is reasonable to assume that a child, observing the world, has some notion. For instance, we assume that a child recognizes that being green and being red are two properties that describe the same aspect of the visual appearance of an object, and that being square or being round are a different aspect of that appearance; in other words, it recognizes the concepts "color" and "shape", even if it has no word for these (nor for their possible values). In line with this assumption, we use the following predicates: object(x) (x is an object), color(x,y), shape(x,y), size(x,y) (the color/shape/size of x is y), relpos(x,r,y) (the position of x relative to y is r). Constants used for specific colors, shapes and sizes are shown in Table 1. Note that neither constants nor predicates are hard-coded in our model; if a new example contains constants or predicates not seen before, our learner can handle it without any change.

Example 2. A context in which a big red square is to the left of a small green triangle is represented as {object(o1), shape(o1, sq), color(o1, re), size(o1, bg), object(o2), shape(o2, tr), color(o2, gr), size(o2, sm), relpos(o1, lo, o2)}.

3.3 Learning the meaning of specific n-grams

The "meaning" of a sentence, phrase or word is difficult to define [6]. In this work, we use a pragmatic definition: the meaning of an n-gram (a sequence of n words) is "whatever is in common among all contexts where the n-gram can be used".[4] We formalize this as follows.

A *pattern* is an existentially quantified set of atoms (one can think of it as a Prolog query). Note that a context description can be seen as a variable-free pattern. A pattern Q *subsumes* another pattern Q' if there is a variable substitution that turns it into a subset of Q'. Two patterns are *equivalent* if they subsume each other. Given a set of contexts, their *most specific common pattern* is a pattern Q that subsumes all of them and for which no other pattern exists

[4] This is in line with the work by Mooney et al. and with Wittgenstein's views on the meaning of language.

that subsumes all of them and is subsumed by Q. The *meaning* of an n-gram is the most specific common pattern of all the contexts where it can be used.

The most specific common pattern can be computed using Plotkin's lgg ("least general generalization") operator [17], which is intensively used in inductive logic programming [16].

Example 3. The lgg of the following pair of contexts:

{ object(o1), color(o1, re), shape(o1, sq) }
{ object(o2), color(o2, gr), shape(o2, tr), object(o3), color(o3,re), shape(o3,tr) }

is {object(X), color(X, re), shape(X,Y)}. It captures everything that is in common among these two contexts, which is: there is an object that is red and that has some shape.

Example 4. The most specific common pattern in

{ obj(o1), clr(o1,re), shp(o1,sq), obj(o2), clr(o2,gr), shp(o2,tr), relpos(o1,lo,o2) }
{ obj(o3), clr(o3,gr), shp(o3,tr), obj(o4), clr(o4,re), shp(o4,tr), relpos(o3,lo,o4) }

is

{ obj(B), clr(B,re), shp(B,D), obj(E), clr(E,gr), shp(E,tr),
obj(A), clr(A,C), shp(A,D), relpos(A,lo,F), obj(F), clr(F,G), shp(F,tr) }.

It captures that, in both contexts, there is a red object (B), a green triangle (E), and an object (A) to the left of a triangle (F). It may seem strange that this clause refers to four objects, when each context had only two, but note that different variables do not have to refer to different objects. In the first clause, $A = B$ and $E = F$, but in the second clause $A = E$ and $B = F$. Identifying all commonalities between the two contexts cannot be done with fewer than four object references, because they unify differently in both contexts.

Our algorithm incrementally learns the meaning of specific n-grams. Whenever it sees a new example with context C and phrase S, it iterates over all the n-grams in S, and for each n-gram, it replaces the currently stored meaning of the n-gram by the lgg of that meaning and the new context. Pseudocode is shown in Algorithm 1 (main algorithm and UPDATE procedure). For reasons made clear below, the algorithm also remembers the category of each constant (e.g., a constant that occurs as the second argument of color is a color), UPDATE also keeps track of how long ago the meaning of an n-gram was last changed (stability counter), and UPDATE may assert or retract mrf ("may refer to") facts.

3.4 Generalizing n-grams

There is often a relationship between the meaning of an n-gram and that of k-grams with $k < n$ it is composed of. For instance, compare:

"red triangle": there is an object that is both triangular and red
"red": there is an object that is red
"triangle": there is an object that is triangular

This particular relationship is common to all bigrams of the form "color shape". A learner that recognizes the concepts "color" and "shape" may detect this. We try to give our learner this capacity in the following manner.

When the meaning of a 1-gram w seems to have converged (that is, it has not changed for the last s updates, with s a parameter called the *stability threshold*) to a context that contains only one constant c, the learner assumes that the 1-gram refers to that constant. The fact `mrf(`w`, `c`)` is then asserted. (Should w's meaning change anyway later on, it is retracted again, since this implies it was asserted prematurely.)

The learner tries to generalize n-grams for $n > 1$ using this `mrf` mapping by constructing a more general rule, as follows (see also GENERALIZE in Algorithm 1). A word w in the n-gram is turned into a variable X_w if a corresponding mrf(w,c) is available; in the n-gram's meaning, each occurrence of c is then turned into a variable X_c, and the condition mrf(X_w, X_c) is added to the body of the rule. The category of c is also added to the body as a condition; this guarantees "cautious" generalization: everything we know about the word is added so that a maximally specific rule is obtained.

For instance, after mrf(red,re) and mrf(triangle, tr) have been learned, the fact

meaning(ngram(2, [red, triangle]), [object(O), color(O, re), shape(O,tr), size(O,_)])

can be generalized into the rule

```
meaning(ngram(2, [X,Y]) , [object(O), color(O, C), shape(O,S), size(O,_)]) :-
    mrf(X,C), mrf(Y,S), category(C,color,2), category(S,shape,2).
```

This rule essentially generalizes the meaning of the 2-gram "red triangle" into an equivalent meaning for any 2-gram of the type "color shape".

Any n-gram can be generalized in this manner, but the resulting rule is not necessarily correct. To evaluate the rule, we define two criteria: *evidence* (how much evidence is there that the rule is correct?) and *coverage* (how many separate facts could be replaced if we introduced this rule in the knowledge base?).

Let C be the set of all previously observed n-grams for which a meaning is predicted by rule R. Let $S \subseteq C$ contain all n-grams whose (currently stored) meaning is *subsumed* by the meaning predicted by R, and $E \subseteq S \subseteq C$ all n-grams whose meaning is *equivalent* to the one predicted by R. Every n-gram in E is predicted correctly by R. Every n-gram in $S - E$ is *compatible* with R, in the sense that its meaning may still converge to the predicted meaning after seeing more examples. Any n-gram in $C - S$ contradicts the rule: further updates cannot lead to a meaning equivalent to the prediction. We call a rule *valid* if $S = C$, and we call $|C|$ the *coverage* of the rule.

The fact that a rule is valid does not provide strong evidence for its correctness. A rule that predicts an empty pattern is automatically valid, but does not capture any meaning. The n-grams in E, however, do provide evidence: the larger E is, the less likely it is that the rule accidentally predicts all the meanings of all these n-grams correctly. $|E|$ is called the *evidence* for the rule.

The utility of the rule is related to $|C|$, but our confidence in its correctness is related to $|E|$. In practice, it seems reasonable to only consider valid rules whose

Algorithm 1 The learning algorithm

Input: stability threshold s, evidence threshold e, stream of context/phrase examples $\mathcal{D}$
Output: predicate definition for meaning

whenever a new example $(C, S) \in \mathcal{D}$ is presented:
  for each constant c occurring as the i'th argument of a predicate p in C:
    assert category(c,p,i) (if not asserted yet)
  for each n-gram G in S, with n=1, 2, ...
    UPDATE(C, G)
    GENERALIZE(G)

UPDATE(context C, n-gram G):
  if meaning(G, M) **then**
    if lgg(M,C) = M **then**
      increase stability(G) by 1
      if G is a 1-gram, stability(G)=s, and M contains one constant c
      then assert mrf(G,c); CLEANUP
    else
      retract meaning(G, M); assert meaning(G, lgg(M, C)); stability(G)=0
      retract mrf(G, _);
  else assert meaning(G, C); stability(G)=0

GENERALIZE(n-gram G) :
  call meaning(G,M)
  $Ref = \{(w, c) | w \in G \wedge \mathsf{mrf}(w, c)\}$
  $Cat = \{(c, x, i) | (\_, c) \in Ref \wedge \mathsf{category}(c, x, i)\}$
  R = meaning(G, M) :- $\bigwedge_{(w,c)\in Ref} \mathsf{mrf}(w, c), \bigwedge_{(c,x,i)\in Cat} \mathsf{category}(c, x, i)$
  introduce for each w and c in Ref a different variable X_w, X_c
  replace in R each w and c by the corresponding X_w or X_c
  if R is valid and has evidence $\geq e$ **then** assert R; CLEANUP

CLEANUP : retract each meaning(G,M) fact covered by a rule R

$|E|$ is above some threshold, which is what our current algorithm does. Note that when R gets introduced, it replaces not only the previously stored meanings of the n-grams in E, but also those in $S - E$. For the latter, the currently stored meaning gets replaced by what the rule predicts will be their converged meaning; in other words, the rule boosts their convergence.

4 Experiments

To evaluate our model and learning algorithm, we made an example generator that generates random contexts and for each context a random phrase that describes (part of) it in English, Dutch or Spanish. Each context consists of one or two objects with a shape, color, size, and relative position. The phrases

object(11),shape(11,tr),color(11,re),size(11,me), object(12),shape(12,he),color(12,pu),size(12,bg), rel_pos(11,lo,12)	a red triangle to the left of the big hexagon
object(52), shape(52,he), color(52,ye), size(52,sm)	the hexagon

Table 2. Some examples of contexts and relevant phrases.

are generated using a simple probabilistic grammar and are surrounded by the markers $start and $stop, so that n-grams can refer to the beginning and ending of a phrase. For phrases referring to two objects, the format is the same as that used by Angluin and Becerra-Bonache [2]. Table 2 shows a few representative examples. We have generated three corpora (one for each language) of 1000 examples each. The two parameters of the learner, stability threshold and evidence threshold, were both set to 5.

4.1 Learning meanings

Running the learning algorithm on the corpora led to the following observations about the learning process.

It is known from ILP that the size of an lgg can grow exponentially in the number of instances generalized (see, e.g., [10]). In our experiments, this behavior was indeed observed. Eventually, meanings always converge to a relatively simple pattern, but intermediate patterns can be very complex due to accidental similarities among contexts. As subsumption testing is exponential in the length of clauses, this slows down the system unacceptably. We solved this problem by simply not updating the meaning of an n-gram if the new meaning is too complex (specifically, lggs containing over 15 literals were not stored). Such behavior is in fact cognitively plausible: children are unlikely to discover such complex commonalities. Better ways of controlling the complexity of lggs exist, but for our experiments, this simple method worked well.

For English, a learning curve was observed with (among other) the following "milestones" (the number indicates the number of examples seen at this point):

79 the "*color shape*" generalized bigram is learned. The mrf map at this point contains 4 colors and 5 shapes, hence the rule predicts the meaning of 20 combinations. 5 predictions are equivalent to the stored meaning, 14 generalize it (boosting convergence), 1 is for a bigram not seen before.

85 mrf(to,bg) is retracted. Apparently, the system had earlier concluded that "to" means bg (big), because that was the only constant common in all its contexts and it remained present in the next 5 contexts. When finally a context for "to" without a big object is seen, bg disappears from the meaning; it is then clear that mrf(to,bg) was added prematurely, and it is retracted.

86 mrf(disc,di) is added. The meanings of "red disc" and "blue disc" are retracted, as they are now subsumed by the color-shape rule.

89 the "*size shape*" generalized bigram is learned.

102 mrf(yellow,ye) is added. All colors and shapes have now been learned.
165 the "*size color*" generalized bigram is learned.
188 "$start the *color shape* $stop" is learned (this pattern forms a full phrase)
416 the "*size color shape*" generalized trigram is learned.
664 "*shape* to the *relpos* of" is learned. This is an overgeneralization: it correctly covers the words "left" and "right", but incorrectly also "above", "below" and "under", all of which are associated with relative positions.

The learning curves for Dutch and Spanish are similar. However, due to the inflection of adjectives in these languages (and the fact that our representation does not express morphological structure), some forms occur less frequently and get generalized later than in English. Further, more rules overgeneralize because they do not impose, for instance, gender correspondence. This leads to the important observation that our model categorizes in the physical world, but not in the language world (it distinguishes color and shape as difference kinds of properties, but does not categorize words for shapes into masculine and feminine because these are syntactic, not physical, properties).

Retraction of an mrf fact happened only twice: the word "to" in English and "van" in Dutch. There are also incorrect mrf facts that are not retracted. Most notably, the definite article "het" in Dutch is believed to refer to a square. The reason is that among 6 shapes, only the Dutch word for square, "vierkant", uses this article. Thus, "het" only occurs when a square is present, and its meaning is indistinguishable to that of "vierkant". This mistake has dire consequences: as "het" is assumed to refer to a shape, "het oranje vierkant" is seen as an example of the pattern *shape color shape*, and generalized. Because there are 6 examples of such trigrams (for 6 different colors), and all have the same meaning as the one predicted by the rule, the system finds the evidence sufficient for introducing the generalized rule.

To some extent, this problem is an artifact of the fact that we have only one shape that uses that article: it would disappear if we had at least two. But it still points to at least two opportunities for improving our learning model. First, the fact that "het" occurs in positions where other shapes do not occur should be an indication that it is not a shape at all. Again, this would require analyzing the sentence structure when categorizing words, which our algorithm currently does not do. Second, our method for evaluating evidence (just counting the number of n-grams in E) assumes that that evidence is statistically independent. This is not the case here: all evidence for the incorrect pattern has "het" for the first word, not just any shape. Our evidence criterion needs to be refined.

4.2 Using the model

Until now, we have mostly focused on the learning process, but the proposed model representation also makes a variety of inferences very easy to perform. We illustrate a few of these. All inferences are made using a model learned from the full corpus, and performed on the context {object(90), shape(90,tr), color(90,re), size(90,me), object(91), shape(91,he), color(91,pu), size(91,bg), rel_pos(90,lo,91)},

which states that a medium-sized red triangle is to the left of a big purple hexagon.

Generating relevant n-grams and phrases. The model can generate n-grams that are relevant in some context, by simply checking which n-grams have a meaning that matches part of the context. By chaining such n-grams, we can produce complete phrases that in principle could be grammatically or semantically incorrect, but in practice work well for the simple contexts here provided. Chaining English 3-grams (with 2 words of overlap) gives 24 phrases for our running context, all of which are correct; they include: "the big hexagon"; "a big purple hexagon"; "the red triangle to the left of the big hexagon"; etc.

For the same context, generated Spanish phrases include "el hexagono purpura y grande", "un triangulo rojo", but also the incorrect "un triangulo roja". Dutch phrases produced for this context are all correct; they include, e.g., "de grote driehoek naast een grote paarse zeshoek".[5]

Identifying objects. We can easily ask which part of a given context, if any, matches the meaning of a given n-gram. For the n-gram "purple hexagon", the system returns {object(91),shape(91,he),color(91,pu),size(91,bg)}, correctly identifying the big purple hexagon as an object matching the description.

Producing denoting phrases. A phrase is denoting if it uniquely identifies one object in a context. The ability to denote objects is important because it is one of the main purposes of using language. Our system can easily produce denoting phrases for some context by generating a phrase, finding what it identifies, and returning it if it identifies exactly one object. In the running context, the following denoting phrases (among others) are returned: "a big hexagon", "the hexagon", "a big purple hexagon", "the red triangle", ...

Note that our approach could be useful in the Referring Expression Generation domain, which concerns with how to produce a description that identifies an specific entity in a given context [12]. It could also have interesting implications for the field of Computer Vision, for example, by providing semantic connections between different objects detected in a concrete context.

Translating sentences. Meaningful n-grams can be translated simply by asking for an n-gram in another language with an equivalent meaning. E.g.:

```
?- meaning(ngram(4, [the, big, blue, triangle]), _, C1), meaning(L, spanish, C2),
equiv(C1,C2).
```

(where equiv tests equivalence) gives as possible answers for L the 5-gram ngram(5, [el,triangulo,azul,y,grande]), but also versions with un, la, una instead of el. Translations are only correct insofar the system understands the phrases (it has not learned the meaning of articles, and therefore cannot distinguish the meanings of "the red triangle" and "a red triangle"), and as said before the system currently does not learn gender correspondence.

[5] With a model learned from 2000 examples, the incorrect phrase "driehoek rode driehoek" was produced; this is a consequence of the belief that "het" is a shape, and the construction of a rule for *shape color shape* as a consequence.

It is worth noting that most work in Machine Translation focuses on syntactic-based approaches, but their limitations to preserve meaning structures across languages have motivated research on semantic-based machine translation (e.g., [18, 8]). Our approach mainly differs from these semantic approaches in that we do not use a parallel corpus consisting of sentence/meaning pairs. Moreover, it can be viewed as a first step towards systems that are able to use the context to translate sentences from one language to another, while preserving the meaning.

5 Conclusions and future work

We have presented a simple model for grounded language learning that uses a first-order logic representation of contexts and meanings. Our system learns to understand and generate simple natural language utterances, from pairs consisting of utterances and the context in which these utterances are produced. In contrast to other approaches, a context is a description of what the learner can see in the world, and not a set of candidates meanings for that utterance; our system constructs all candidate meanings itself. It does not require any prior language-specific knowledge and learns incrementally. Experiments with three different languages show that our system learns a language model that can easily be used to understand, generate and translate utterances.

This paper describes a simple proof of concept. Opportunities for further work include: learning from more complex contexts (which may include actions), learning more complex languages, categorizing also n-grams (as opposed to only context elements), controlling the lgg complexity in a more principled manner, experimenting with other inductive inference methods known from inductive logic programming, robust learning in noisy environments (probabilistic logics may be useful for this), and much more.

It is worth noting that the development of computational approaches such as this one can help obtain insight into how children build a model of language and make connections between utterances and contexts. It can also help understand how language developed from its origins, since the observation of the world and the intention to communicate were crucial for the development of language.

Acknowledgements

Part of this work was done while HB was visiting Laboratoire Hubert Curien.

References

1. Angluin, D., Becerra-Bonache, L.: A model of semantics and corrections in language learning. Tech. Rep. YALE/DCS/TR1425, Yale University (2010)
2. Angluin, D., Becerra-Bonache, L.: Effects of meaning-preserving corrections on language learning. In: CoNLL. pp. 97–105 (2011)
3. Chen, D.L., Kim, J., Mooney, R.J.: Training a multilingual sportscaster: Using perceptual context to learn language. JAIR 37, 397–435 (2010)

4. Chen, D.L., Mooney, R.J.: Learning to sportscast: a test of grounded language acquisition. In: ICML. pp. 128–135 (2008)
5. Feldman, J., Lakoff, G., Stolcke, A., Weber, S.: Miniature language acquisition: A touchstone for cognitive science. In: CogSci, pp. 686–693 (1994)
6. Harnad, S.: Symbol grounding problem. Scholarpedia 2(7), 2373 (2007)
7. de la Higuera, C.: Grammatical Inference, Learning Automata and Grammars. Cambridge University Press (2010)
8. Jones, B., Andreas, J., Bauer, D., Hermann, K.M., Knight, K.: Semantics-based machine translation with hyperedge replacement grammars. In: COLING 2012. pp. 1359–1376 (2012)
9. Kate, R.J., Mooney, R.J.: Learning language semantics from ambiguous supervision. In: AAAI. pp. 895–900 (2007)
10. Kietz, J.U.: A comparative study of structural most specific generalizations used in machine learning. In: In Proc. Third International Workshop on Inductive Logic Programming. pp. 149–164 (1993)
11. Kim, J., Mooney, R.J.: Generative alignment and semantic parsing for learning from ambiguous supervision. In: COLING. pp. 543–551 (2010)
12. Krahmer, E., van Deemter, K.: Computational generation of referring expressions: A survey. Computational Linguistics 38(1), 173–218 (2012)
13. Kwiatkowski, T., Goldwater, S., Zettlemoyer, L.S., Steedman, M.: A probabilistic model of syntactic and semantic acquisition from child-directed utterances and their meanings. In: EACL. pp. 234–244 (2012)
14. Lloyd, J.W.: Foundations of Logic Programming, 2nd Edition. Springer (1987)
15. Lu, W., Ng, H.T., Lee, W.S., Zettlemoyer, L.S.: A generative model for parsing natural language to meaning representations. In: EMNLP. pp. 783–792 (2008)
16. Muggleton, S., Raedt, L.D.: Inductive logic programming: Theory and methods. Journal of Logic Programming 19/20, 629–679 (1994)
17. Plotkin, G.D.: Machine Intelligence 5, chap. A note on inductive generalization, pp. 153–163. Edinburgh University Press (1970)
18. Pust, M., Hermjakob, U., Knight, K., Marcu, D., May, J.: Using syntax-based machine translation to parse english into abstract meaning representation. CoRR abs/1504.06665 (2015)
19. Siskind, J.: A computational study of cross-situational techniques for learning word-to-meaning mappings. Cognition 61, 39–61 (1996)
20. Wong, Y.W., Mooney, R.J.: Learning synchronous grammars for semantic parsing with lambda calculus. In: ACL. pp. 960–967 (2007)
21. Zettlemoyer, L.S., Collins, M.: Learning context-dependent mappings from sentences to logical form. In: ACL. pp. 976–984 (2009)