A Framework for Lexicalized Grammar Induction Using Variational Bayesian Inference

Chris Bruno*, Eva Portelanceb, Daniel Harasimc, Leon Berenga, Timothy J. O’Donnella

(a) McGill University; (b) Stanford University; (c) École Polytechnique Fédérale de Lausanne; (d) UC San Diego; *Co-first authors

L2HM: Learning Language in Humans and in Machines 2018

Introduction

We introduce a probabilistic learning model for a class of lexicalized grammar formalisms. We use these tools to develop a computational framework for investigating ideas in theoretical syntax, by assessing their learnability via compactness studies, similar to the methodology in [1, 2]

We use Minimalist Grammars, designed for being suitable in deriving long distance dependencies via movement.

as well as non-context free dependencies such as crossing dependencies [3]

Minimalist Grammars

A Directional Minimalist Grammar (DMG, [4, 5]) is a tuple (L, R) where

L = {merge, move} is the set of structure building operations. The key what did he do.

R = {d | c} selects the syntactic categories

L = {merg, mov} selects argument constituent to the right.

R = {left select | r, l | pr} selects argument constituent to the left.

L = {dist | r, l | str} selects moving constituent.

R = {prev | l, l | b} selects moving constituent.

We use DMGs, which can merge either to the right or to the left of a node (eq. the key vs key the). Movement is always to the left.

A probabilistic MG also contains:

θ: where each θ ∈ θ is a probability distribution over all lexical items l with the category feature c such that

We sample each θ, from a Dirichlet prior parameterized by α.

The Generative Model

We implement a head-out generative model for derivations in a Merge-only variant of the Directional MG.

The generative model allows us to forward sample derivations, or sentences, given a Lexicon.

Let l be a derivation headed by the lexical item l.

Let cat(l) be the category feature of l.

Let sel(l) give the subderivations d0, . . . , dk that are selected by the head l in derivation d.

This returns the probability of the derivation d0, which is the product of its lexical items.

Variational Bayesian Inference

In order to learn probabilities to a grammar, we calculate the posterior P(D, θ|S, α), where D is a sequence of derivations over a corpus S.

Approximate by minimizing the KL distance between the true posterior P and the variational approximation Q.

Variational independence assumption (where d ∈ D, 1 ≤ n ≤ N is a derivation):

The optimal variational distributions are

Algorithm. Update each wα and each θα until the KL converges, where wα is initialized to α.

This algorithm is guaranteed to find a posterior which is at least a local minimum.

Experiment 1: Grammar Recoverability

We start with a recursive grammar of 24 lexical items without movement and sample a corpus of 1000 sentences.

We run our inference algorithm for 10 iterations starting from a uniform prior conditioning on the corpus to test the learning algorithm.

We compare the probability distribution over 159 unique newly sampled sentences given the ground truth grammar to the retrieved learnt distribution and the prior distribution.

Results:

<table>
<thead>
<tr>
<th></th>
<th>Prior</th>
<th>Learn</th>
<th>KL Divergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Earth mover</td>
<td>28.62</td>
<td>45.2</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Experiment 2: Learning Movement

Grammar of 49 lexical items including wh-words, nouns, determiners, and transitive verbs. Each sentence of containing a wh-word is ambiguous between 2 parses, with and without movement. Its language contains 21888 sentences.

Training set is a random sample of 2188 sentences (1% of the language).

The learned grammar uses movement for wh-objects, but not wh-subjects.

Experiment 3: Recovering English Dependencies

Universal Dependencies English ParTUT corpus, with given train/test splits, with 1754/151 sentences respectively, averaging 24/22 words per sentence.

Trained for 2 iterations with two grammars, semi-supervised conditioned on gold dependencies:

1. G SIMPLE: Lexical, hand-built grammar inspired by Minimalist Theory, respecting the DP hypothesis.
2. G COMP: The same, but respecting the NP-hypothesis.

The best results pick the best performing parse for each sentence, giving an approximate upper bound, showing that our grammars are not capable of recovering around half of the gold dependencies, but our learning algorithm improves upon the uniform grammar approximating the best results.

The NP grammar has higher accuracy, likely because the gold dependencies are Noun-headed.

Experiment 4: Compactness comparisons

Given a grammar G, the grammar of rank k is the subset of G containing the k highest scored lexical items in each category.

A more compact grammar is expected to be more successful at parsing for lower values of k and is expected to have more complex parses.

The DP grammar does better at both properties:

Accuracy results:

<table>
<thead>
<tr>
<th></th>
<th>train</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>G SIMPLE</td>
<td>0.63</td>
<td>0.38</td>
</tr>
<tr>
<td>G SIMPLE best</td>
<td>0.62</td>
<td>0.40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>train</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>G COMP</td>
<td>0.75</td>
<td>0.57</td>
</tr>
<tr>
<td>G COMP best</td>
<td>0.76</td>
<td>0.57</td>
</tr>
</tbody>
</table>

References


Contact chris.bruno@mcgill.ca; portelan@stanford.edu