Efficient Inference for Unsupervised Semantic Parsing

Maxim Rabinovich[*]
Computer Science Division
University of California, Berkeley
Berkeley, CA 94720
rabinovich@berkeley.edu

Zoubin Ghahramani
Department of Engineering
University of Cambridge
Cambridge, UK CB2 1PZ
zoubin@eng.cam.ac.uk

Abstract

Unsupervised semantic parsing algorithms based on Bayesian nonparametric grammars offer a promising way to bootstrap semantic analyses for unsupervised relation/information extraction (RE/IE). Yet the form in which they currently exist makes it difficult to apply them to larger data sets. In this paper, we make progress toward scaling unsupervised semantic parsing by introducing a local, collapsed sampling scheme for a class of nonparametric dependency grammars. We show that our algorithm can rapidly discover the semantic units in a corpus, and we report edifying preliminary results on the semantic clustering task.

1 Model specification

Our work centers on the Bayesian nonparametric dependency grammar introduced for unsupervised semantic parsing in [9]. In this grammar, trees are partitioned into fragments, each corresponding to a particular lexical realization of an abstract concept, which we call a "semantic class." Semantic classes are viewed as values $c \in \mathbb{N}$ associated with parameters dictating their behavior: a distribution ϕ_c over its lexical realizations (i.e. tree fragments); a probability of generating arguments, encoded as a probability ψ_{ct} of generating any arguments of latent type t and a parameter ψ_{ct}^{+} to a geometric distribution on the number of arguments of type t; a distribution $\beta_{ct}^{\mathrm{edge}}$ over edge labels for arcs connecting it to its arguments; and a distribution θ_{ct} over semantic class labels for type t arguments of the class. A special distribution θ^{rt} controls the semantic class labels at the root node.

As the number of semantic classes is unknown (and likely to be large), we sample the base distribution over semantic classes according to

$$\gamma \sim \mathrm{PYP}(\alpha^{\mathrm{prior}},\, d^{\mathrm{prior}}).$$

The root distribution and class-specific distributions over arguments then come from a second-level PYP with base measure γ, viz.

$$\theta^{\mathrm{rt}} \sim \mathrm{DP}(\alpha^{\mathrm{rt}},\, \gamma) \quad \text{and} \quad \theta_{ct} \sim \mathrm{DP}(\alpha,\, \gamma).$$

We construct the prior on tree fragment distributions using a weak base dependency grammar H:

$$\phi_c \sim \mathrm{PYP}(\alpha^{\mathrm{syn}},\, d^{\mathrm{syn}},\, H).$$

The recursive structure of the model makes it difficult to represent in graphical form. We therefore present it as a generative process, explicitly spelling out how nodes with given context are created and expanded according to the model. A tree below a node is generated via: [9]

[*]Work completed while the author was a student at the University of Cambridge.

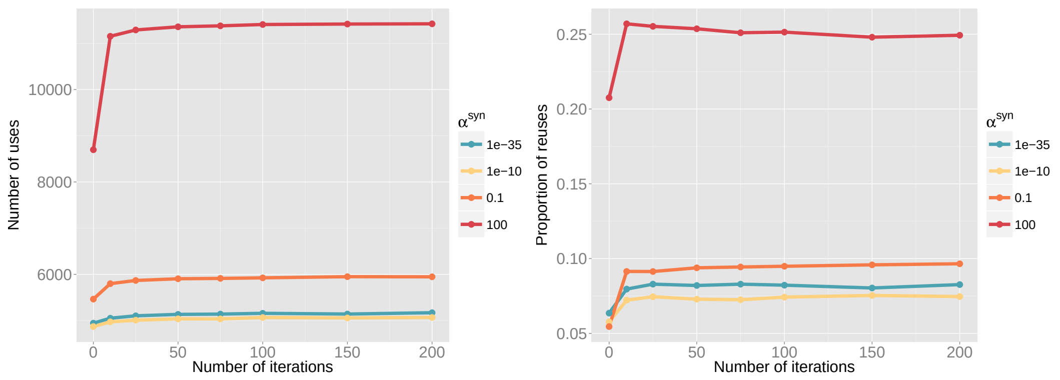

Figure 1: *(Left)* Number of uses of fragments of size ≥ 2 and *(right)* proportion of those uses accounted for by fragments used at least twice.

1. Draw $c \sim \theta^{\text{rt}}$ (if at root) or $c \sim \theta_{c't'}$ if in context $(c',\ t')$.
2. Draw $f \sim \phi_c$.
3. For each $t \in [T]$:
   (a) With probability $1 - \psi_{ct}$, generate no arguments of type t.
   (b) Otherwise, draw $a_{t0} \sim \beta_{ct}^{\text{edge}}$ and generate a subtree with context $(c,\ t)$.
   (c) Draw $m_t^+ \sim \text{Geom}(\psi_{ct}^+)$.
   (d) For $i = 1, \ldots, m_t^+$, draw $a_{ti} \sim \beta_{ct}^{\text{edge}}$ and generate a subtree with context $(c,\ t)$.

2 Inference strategy

Prior work on this model used global split-merge proposals as the basis for their Metropolis-Hastings algorithm [9]. In "compose-decompose" steps, their algorithm chooses a fragment f in a class c, then selects another fragment f' and proposes joining it to f. In "split-merge" steps, on the other hand, the algorithm proposes to split a semantic class or merge two semantic classes into one.

Our proposal method differs from theirs in its handling of both fragments and classes. First, it is based on local updates that iteratively remove a single tree from the state and resample its partition and semantic class labels. The algorithm can therefore produce many new fragments in one step, when warranted, and can more quickly generate fragments combining rare words. Similarly, new classes come into being as a side effect of compartment label sampling, with no need to explicitly propose creating a new class as part of a special move. Likewise, fragments move between classes as a side effect of label sampling, rather than through global split-merge moves. The locality of the algorithm makes it suitable for incorporation into large-scale learning procedures [4, 7, 11, 12].

Our algorithm consists of three major components:

1. **Collapsed state representation.** Unlike prior work on Bayesian nonparametric grammars [3, 9], our algorithm uses a fully collapsed representation of the state space. This makes inference conceptually more difficult, but entirely eliminates the need to maintain multiple counts for each tree fragment [2, 8].

2. **Proposals based on a point-estimated grammar.** Building on the methodology of [3, 5], we use a Bayes estimate of the grammar based on the starting latent variable configuration as our proposal distribution. This is the key idea which makes our local sampling scheme possible.

	Number of reused nontrivial fragments		
	$\alpha^{\text{syn,proposal}} = 10^{-35}$	$\alpha^{\text{syn,proposal}} = 0.1$	$\alpha^{\text{syn,proposal}} = 100$
$\alpha^{\text{syn}} = 10^{-35}$	112	191	**1362**
$\alpha^{\text{syn}} = 0.1$	93	164	**656**
$\alpha^{\text{syn}} = 100$	116	219	**710**

Method	Iteration	Metric		
		Mean coherence	# trivial classes	Mean external similarity
Unpruned	10	0.167	9447	0.430
Pruned	10	0.314	6839	0.387
	100	0.326	5192	0.356

Table 1: *(Top)* Computing point estimates using a larger hyperparameter value leads to increased fragment reuse. *(Bottom)* Comparison of pruned and unpruned variants of the algorithm. The pruned variant leads to more coherent and distinct classes, as well as fewer single-fragment (trivial) classes. It also runs about 40 times faster, allowing us to improve the results further with more sampling.

3. **Semantic class pruning.** Explicitly considering each semantic class at each node in the dependency tree leads to an intractable inference algorithm. Instead, we prune most of them based on a distributional similarity heuristic.

Due to space restrictions, we limit our discussion to the proposal distributions. The overall idea is based on the observation that partition structures under the point-estimated grammar can be sampled exactly by first computing marginal probabilities of all subtrees. Indeed, we first use dynamic programming along the tree structure to compute two types of quantities for each node u: the marginal probability of generating its subtree if u is the root of a compartment labeled c, namely

$$P^{\text{sub}}(u,\ c) = p(x_u \mid \text{compartment root},\ c_u = c),$$

and the marginal probability of the subtree given that the fragment in which u occurs is drawn from the prior, irrespective of whether u is the root of that fragment or not, namely

$$P^{\text{base}}(u,\ c) = p(x_u \mid \text{new draw},\ c_u = c,\ w_u).$$

Given these probabilities, it becomes possible to recursively sample variables indicating whether each node is a compartment root or not—from which we can read off the partition. The key ideas behind this are, first, that the sampling probability for a class assignment c at a node u is only a function of the class c and the table entry $P^{\text{sub}}(u,\ c)$ and, second, that the indicators for whether an out-edge is part of u's compartment can be expressed in terms of binary variables related by a chain MRF, which allows us to sample them efficiently by dynamic programming.

3 Empirical evaluation

Following the literature, we use the GENIA corpus [6]. In order to run our experiments more quickly, we use the first 500 abstracts, which span about 4600 sentences.

We organize our evaluation around the two halves of the semantic parsing task: *semantic segmentation*, or the discovery of semantic units, and *semantic clustering*, or the grouping of equivalent units. In the language of semantic grammars, semantic segmentation corresponds to inference of tree fragments, while semantic clustering corresponds to inference of the fragment clusters defined by the semantic classes.

3.1 Semantic segmentation

We aim to answer two questions with our empirical evaluation. First, does the algorithm identify nontrivial reusable fragments, and does it actually reuse them? We judge how well our algorithms

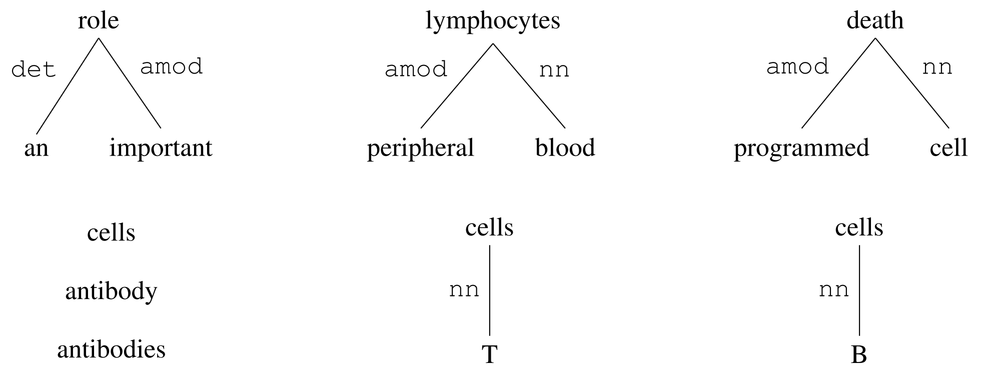

Figure 2: *(Top)* Examples of larger learned units. All of these are in the top 10 most frequent for their size and are used at least five times in the corpus. *(Bottom)* Top 5 fragments in a semantic class whose most frequent fragment is "cells." As the class includes frequent occurrences of "T cells" and "B cells," which are elements of the human immune system, the presence of the semantically related—but not equivalent—"antibody" and "antibodies" makes sense.

fulfil this criterion by computing the proportion of tree compartments corresponding to reusable tree fragments of size ≥ 2, as well as the number of uses of fragments of size ≥ 2 used at least twice. Second, are the learned fragments semantic units? This question is difficult to answer without access to gold annotations. We nonetheless argue on the basis of examples that the answer is affirmative.

Figure 1 shows the evolution during sampling of the number of uses of fragments of size ≥ 2 and the proportion of those uses accounted for by fragments used at least twice. The values stabilize after about 10 sweeps through the corpus (about 3 minutes). With $\alpha^{\text{syn}} = 100$, about half of all nodes fall in compartments of size ≥ 2, and about 25% of the corresponding fragments are applied more than once. Table 1 shows the results of inference when Bayes estimates of ϕ_{cf} are computed with an altered concentration parameter. Larger values of α^{syn} encourage exploration of new partitions and lead to greater fragment reuse.

3.2 Semantic clustering

We focus our evaluation on three questions. First, are semantic classes internally coherent? We use cosine similarity between distributional representations of a class's fragments [1, 9, 10] to give a quantitative answer. Second, does the algorithm cluster similar fragments together? To answer this question, we use the average cosine similarity between fragments in pairs of classes as a measure of between-class similarity. The external similarity of a class (cf. Table 1) of a class c is then its maximum similarity under this measure to any other class c'. Finally, does the semantic clustering have the right granularity? We would like semantic classes to correspond to individual concepts, which means that their member tree fragments should possess nearly equivalent semantics. Not having access to gold standard annotations, we instead limit ourselves to a qualitative assessment of granularity, which shows that some further refinement in the clusterings would be desirable.

Table 1 shows distributional similarity metrics for classes when using unpruned and pruned proposals. Pruned proposals lead both to more internally coherent and externally distinctive classes, and to faster inference (by a factor of 40). We see moreover that running the sampler longer does pay off somewhat in terms of the quality of the semantic clustering.

Figure 2 shows the most likely fragments for a semantic class whose most common fragment is a single node labeled "cells." In this case, the algorithm has clearly identified a group of semantically related terms, but it seems that the granularity of the clustering corresponds to topical relatedness rather than a stringent notion of semantic equivalence. Similar effects are observed in other semantic classes, with the difference—in some cases—that the observed clustering correspond to fine-grained syntactic interchangeability instead of thematic relatedness.

References

[1] Marco Baroni, Raffaela Bernardi, and Roberto Zamparelli. Frege in space: A program of compositional distributional semantics. *Linguistic Issues in Language Technology*, 9, 2014.

[2] Phil Blunsom, Trevor Cohn, Sharon Goldwater, and Mark Johnson. A note on the implementation of hierarchical Dirichlet processes. In *ACL/IJCNLP (Short Papers)*, pages 337–340, 2009.

[3] Trevor Cohn, Phil Blunsom, and Sharon Goldwater. Inducing tree-substitution grammars. *Journal of Machine Learning Research*, 11:3053–3096, Nov 2010.

[4] Matthew D. Hoffman, David M. Blei, Chong Wang, and John William Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[5] Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. Adaptor grammars: A framework for specifying compositional nonparametric Bayesian models. In *NIPS*, pages 641–648, 2006.

[6] J-D Kim, Tomoko Ohta, Yuka Tateisi, and Junichi Tsujii. GENIA corpusa semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl 1):i180–i182, 2003.

[7] David M. Mimno, Matthew D. Hoffman, and David M. Blei. Sparse stochastic inference for latent Dirichlet allocation. In *ICML*, 2012.

[8] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[9] Ivan Titov and Alexandre Klementiev. A Bayesian model for unsupervised semantic parsing. In *ACL*, pages 1445–1455, 2011.

[10] Peter D Turney and Patrick Pantel. From frequency to meaning: vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, 2010.

[11] Chong Wang and David M. Blei. Truncation-free online variational inference for Bayesian nonparametric models. In *NIPS*, pages 422–430, 2012.

[12] Chong Wang, John William Paisley, and David M. Blei. Online variational inference for the hierarchical Dirichlet process. In *AISTATS*, pages 752–760, 2011.