Some background for understanding Adaptor Grammars

Annie Louis

September 4, 2014

1 Bayesian inference and cache models

Bayesian inference can be alternatively viewed as cache-based modeling which encourages reuse of analyses.

This perspective can be understood by looking at the predictive distributions of a few models.

1.1 Parametric case: Dirichlet-Multinomial Language Model

Consider a lexicon of $K$ word types. The language model parameter is a multinomial distribution over the $K$ types. The parameter itself is drawn from a symmetric Dirichlet prior with concentration parameter $\beta$.

$$P(\theta | \beta) = \frac{\Gamma(\sum_{k=1}^{K} \beta_k)}{\prod_{k=1}^{K} \Gamma(\beta_k)} \prod_{k=1}^{K} \theta_k^{\beta_k - 1}$$

Let $n_k$ be the number of times word type $w_k$ appears in the data $D$. The data likelihood is:

$$P(D | \theta) = \prod_{k=1}^{K} (\theta_k)^{n_k}$$

The posterior distribution of $\theta$ is:

$$P(\theta | D, \beta) = \frac{p(D | \theta)p(\theta | \beta)}{p(D)} = \frac{p(D | \theta)p(\theta | \beta)}{\int_{\theta} p(D | \theta)p(\theta | \beta) d\theta}$$

$$= \frac{\Gamma(\sum_{k=1}^{K} n_k + \beta_k)}{\prod_{k=1}^{K} \Gamma(n_k + \beta_k)} \prod_{k=1}^{K} \theta_k^{n_k + \beta_k - 1}$$
Since the prior Dirichlet is conjugate to multinomial, the posterior over parameters is also a dirichlet distribution.

The prediction distribution gives the probability that the next token is a particular word type given the words seen so far. The probability that token $w_i$ is word type $j$ after observing the previous $i-1$ tokens.

$$p(w_i = j|w_{-i}, \beta) = \int_\theta p(w_i = j|\theta)p(\theta|w_{-i}, \beta) d\theta$$

$$= \frac{n_j^{(w_{-i})} + \beta_j}{i - 1 + \sum_{k=1}^{K} \beta_k}$$

$$\propto n_j^{(w_{-i})} + \beta_j$$

Here $n_j^{(w_{-i})}$ is the number of times word type $j$ appears in $w_{-i}$.

### 1.2 Non-parametric case: Dirichlet Process Language Model

$P_\phi$ is a base distribution that gives non-zero probability to infinitely many outcomes. The concentration parameter $\alpha$ explains the variance in using outcomes from base distribution.

The predictive distribution is:

$$p(w_i = j|w_{-i}, \alpha, P_\phi) = \frac{n_j^{(w_{-i})} + \alpha P_\phi(j)}{i - 1 + \alpha}$$

$$= \left(1 - \frac{\alpha}{i - 1 + \alpha}\right) \frac{n_j^{(w_{-i})}}{i - 1} + \left(\frac{\alpha}{i - 1 + \alpha}\right) P_\phi(j)$$

With probability $\frac{\alpha}{i - 1 + \alpha}$ the word is generated from the base distribution and the remaining probability mass is for drawing from the cache of previous analyses. This model can also be called as a Chinese Restaurant Process language model.

- We can see that as the history size increases, the base distribution will be consulted less often.
- But often language phenomena exhibit a power law distribution where more types keep appearing in new data.
• So to model that we desire some more control over how the tail of the distribution looks like. The Pitman Yor process allows this flexibility.

The Pitman Yor process generalizes the Chinese restaurant process. It has two parameters \((a, b)\), where \(a\) is a discount parameter.

Let \(z_{-i}\) denote the table assignments for the \(i - 1\) previously seen tokens. The predictive distribution of the PY language model is:

\[
p(w_i = j|w_{-i}, a, b, P_\phi) = \frac{n_j^{(w_{-i})} - K_w(z_{-i})a + (b + K(z_{-i})a)P_\phi(j)}{i - 1 + b}
\]

\(K_w(z_{-i})\) is the number of tables labelled with \(w_j\) so far. \(K(z_{-i})\) is the total number of tables so far.

• So the number of times the base distribution has been used so far influences the probability of its next use.

• The discount parameter \(a\) controls the weight of this influence to create different power law distributions.

• When \(a = 0\), the model is the same as a CRP with concentration parameter \(b\).

An alternative view by Goldwater et al. 2011 [1] proposes a two-stage framework for bayesian models:

1. A generator which corresponds to the base distribution. Can be its own generative model.

2. An adaptor which assigns frequencies to these outcomes such that the frequencies correspond to a distribution of interest, eg. power laws. This split results in two explanations for a token—produced by the generator or by the adaptor.

## Probabilistic Context Free Grammars

Can be represented as \((N, W, R, S, \theta)\) where:

• \(N\) and \(W\) are the set of non-terminals and terminals respectively

• \(S\) is the start symbol.

• \(R\) is a set of productions of the form \(A \rightarrow \beta\) where \(A \in N\) and \(\beta \in (N \cup W)^*\)

• \(\theta\) are rule probabilities of the form \(p(A \rightarrow \beta|A)\)
2.1 Previous approaches for learning PCFGs

When a treebank is available, we can estimate the rule probabilities directly by counting the rule occurrences in the trees.

No treebank: We can use the inside-outside algorithm which is a EM method. Starting from a random initialization of rule probabilities, the training sentences are parsed and counts accumulated from the parses (and weighted by the probability of the parse). But can return a trivial solution.

Bayesian inference (parametric): Put a dirichlet prior on the rule expansions from a non-terminal to encourage sparsity in the expansions that can happen. Collapsed gibbs sampling can be done by integrating out the rule probabilities. This approach is parametric, the rules are given beforehand and fixed. See Johnson et al. 2007 [3]. Still not very good because CFG is often insufficient to capture complex syntactic structure. There are strong independence assumptions: rules are selected independently at random.

2.2 Adaptor grammars (non-parametric Bayesian inference)

See Johnson et al. 2006 [2].

- Focuses on learning extra rules that are useful. Number of rules is unbounded (non-parametric)
- The new rules are meta-rules ie. combinations of existing rules in the form of subtrees. A non-terminal can be re-written as an entire subtree compared to only expanding based on a production rule.
- The meta-rules allow non-terminals to be rewritten depending on how they were expanded in the past. This introduces dependence between rewrites of a non-terminal.
- The probability of a subtree is computed from the cache of analyses produced so far. This probability can be greater than the combination of probabilities of the CFG rules that the subtree actually contains.
- Viewed as a two-stage model, the generator is a PCFG and the adaptor is a PYP
- One of the usefulness of the AG framework is that general-purpose inference algorithms can be used.

References

