Models of Grammar:Training

Cem Bozşahin

Boğaziçi Linguistics, Ling488

- Suppose that we have developed a grammar which is well-formed according to a theory. Say it studies an idea about some NL phenomenon.

- 'Well-formed' means the models of the grammar would be ready for model validation. The stage of model verification has been reached.

- In our particular case, Bozşahin (2025), we do this by studying linguistic analysis using linguistic categories alone.

- These are categories of two command relations: syntactic command (s-command) and semantic command (l-command).

(1) a. **played** :: $(\mathtt{S\backslash NP})/\mathtt{NP} : \lambda x \lambda y.\mathit{practice}'xy \wedge \mathit{master}'xy$

s-command: S \NP /NP

l-command: $\mathit{practice}'$ x y $\wedge$ master' x y

b. **sleep** :: $\mathtt{S\backslash NP} : \lambda x.\mathit{torpid}'x$

S \NP

torpid' x

Examples of grammar in TheBench notation:

(2)

```
likes  | v :: (s\^np[agr=3s])/^np : \x\y.like x y <120, 1.0>
#np-raise np[agr=?x] : lf  --> s/(s\np[agr=?x]) : \lf\p. p lf  <34, 1.0>
runs | tense :: s[t=pres,agr=3s]\np:\x.pres run x  <2, 1.0>
ran | tense :: s[t=past]\np:\x.past run x <76, 1.0>
```

`<key,parameter>` : element's unique key and its parameter's value (added by the system.)

- We now want to put the grammar-idea to experiment. That is, we turn the grammar into a model.
- What is the experiment for? Depends on what you wanted to capture with the grammar. (word order, lang. acq., case, grammatical relations etc.)

- We first initialize the grammar so that all and only the elements (data points) get a parameter (aka. data parameters). There are no intermediaries.
- We obtain form:meaning pairs that we think are correct pairings about the phenomenon we are studying.
- Training will bias the grammar toward certain elements, depending on the experiment.
- Bias and variance control in an experiment is like granma's recipe for cooking: not too much, not too little.
- Bias means different things in modeling and statistics. In modeling, it is essentially a consequent (if not deliberate) error: what assumptions are made in the model to simplify learning. High bias: strong assumptions. Low bias: flexible.

- Because our grammar is well-formed wrt. a theory, its initial model has in fact passed model verification.
- In training, we do model validation, that is, check how well the grammar model fits the world (rep. by training pairs).
- We do this by model training, parameter re-estimation, and model selection.

- Last thing first: Once parameters are re-estimated and a model is chosen, we can assess the quality of the chosen model using a parse-ranking algorithm.

- The one we use is summarized from Zettlemoyer and Collins (2005). This is what the `r-command` of TheBench does (Bozşahin, 2024).

$$\arg\max_L P(L \mid S; \bar{\theta}) = \arg\max_L \sum_D P(L, D \mid S; \bar{\theta}) \tag{1}$$

S is the expression to be parsed, L is the l-command for it,

D is a sequence of derivations for the (S, L) pair,

$\bar{\theta}$ is the n-dimensional parameter vector for a grammar of size n (the total number of elements).

Example: Suppose we have two alternative analyses (Ds) for the same expression:

$$\cfrac{\underset{\textbf{Sincerity}}{\mathtt{NP}_{3s} : \mathit{sincerity}'} \quad \cfrac{\underset{\textbf{admires}}{(\mathtt{S}\backslash\mathtt{NP}_{3s})/\mathtt{NP} : \lambda x\lambda y.\mathit{admire}'xy} \quad \underset{\textbf{John}}{\mathtt{NP} : \mathit{john}'}}{\mathtt{S}\backslash\mathtt{NP}_{3s} : \lambda y.\mathit{admire}'\mathit{john}'y}}{\mathtt{S} : \mathit{admire}'\mathit{john}'\mathit{sincerity}'}
\qquad
\cfrac{\underset{\textbf{Sincerity}}{\mathtt{NP}_{3s} : \mathit{sincerity}'} \quad \cfrac{\underset{\textbf{admires}}{(\mathtt{S}\backslash\mathtt{NP}_{3s})/\mathtt{NP} : \lambda x\lambda y.\mathit{admire}'xy} \quad \underset{\textbf{John}}{\mathtt{NP} : \mathit{cat}'}}{\mathtt{S}\backslash\mathtt{NP}_{3s} : \lambda y.\mathit{admire}'\mathit{cat}'y}}{\mathtt{S} : \mathit{admire}'\mathit{cat}'\mathit{sincerity}'}$$

$$\arg\max_L P(L \mid \text{sincerity admires John}; \bar{\theta}) = \arg\max_L \sum_D P(L, D \mid \text{sincerity admires john}; \bar{\theta}) \tag{2}$$

There are two Ls. Each has one D. It is a simple choice for the analysis that maximizes the l-command prob. of the expression.

How do we measure $P(L, D \mid \text{sincerity admires john}; \bar{\theta})$?

- It is induced from the following relation of probabilities and parameters.

$$P(L, D \mid S; \bar{\theta}) = \frac{e^{\bar{f}(L,D,S)\cdot\bar{\theta}}}{\sum\limits_{L}\sum\limits_{D} e^{\bar{f}(L,D,S)\cdot\bar{\theta}}} \tag{3}$$

- $\bar{f}$ is a vector of 3-argument functions $<f_1(L, D, S), \cdots f_n(L, D, S)>$.
- The functions of $\bar{f}$ count local substructure in D. By default, f_i is the number of times the lexical element i (item or rule) is used in D, sometimes called the feature i.

$$\dfrac{\begin{matrix}\mathtt{NP}_{3s}\\ : \mathit{sincerity}'\\ \textbf{Sincerity}\end{matrix} \qquad \dfrac{\begin{matrix}(\mathtt{S}\backslash\mathtt{NP}_{3s})/\mathtt{NP}\\ : \lambda x \lambda y.\mathit{admire}' x y\\ \textbf{admires}\end{matrix} \qquad \begin{matrix}\mathtt{NP}\\ : \mathit{john}'\\ \textbf{John}\end{matrix}}{\mathtt{S}\backslash\mathtt{NP}_{3s} : \lambda y.\mathit{admire}'\,\mathit{john}'\,y}}{\mathtt{S} : \mathit{admire}'\,\mathit{john}'\,\mathit{sincerity}'}
\qquad
\dfrac{\begin{matrix}\mathtt{NP}_{3s}\\ : \mathit{sincerity}'\\ \textbf{Sincerity}\end{matrix} \qquad \dfrac{\begin{matrix}(\mathtt{S}\backslash\mathtt{NP}_{3s})/\mathtt{NP}\\ : \lambda x \lambda y.\mathit{admire}' x y\\ \textbf{admires}\end{matrix} \qquad \begin{matrix}\mathtt{NP}\\ : \mathit{cat}'\\ \textbf{John}\end{matrix}}{\mathtt{S}\backslash\mathtt{NP}_{3s} : \lambda y.\mathit{admire}'\,\mathit{cat}'\,y}}{\mathtt{S} : \mathit{admire}'\,\mathit{cat}'\,\mathit{sincerity}'}$$

$$\arg\max_L P(L \mid \text{sincerity admires John}; \bar{\theta}) = \arg\max_L \sum_D P(L, D \mid \text{sincerity admires john}; \bar{\theta}) \tag{4}$$

There are two Ls. Each has one D. It is a simple choice for the analysis that maximizes l-command prob. of the expression.

$\arg\max_L P(L \mid \text{sincerity admires John}; \bar{\theta}) = Max_L P(: \mathit{admire}'\,\mathit{john}'\,\mathit{sincerity}'$,
$\mathtt{S}\backslash\mathtt{NP}_{3s} : \lambda y.\mathit{admire}'\,\mathit{john}'\,y$ **admires john**, $\mathtt{S} : \mathit{admire}'\,\mathit{john}'\,\mathit{sincerity}'$ **sincerity admires john**
$\cdots\cdots$ $\mathtt{NP} : \mathit{john}'$ **john**
$\mid$ sincerity admires john; $\bar{\theta})$
$P(: \mathit{admire}'\,\mathit{cat}'\,\mathit{sincerity}'$,
$\mathtt{S}\backslash\mathtt{NP}_{3s} : \lambda y.\mathit{admire}'\,\mathit{cat}'\,y$ **admires john**, $\mathtt{S} : \mathit{admire}'\,\mathit{cat}'\,\mathit{sincerity}'$ **sincerity admires john**
$\cdots\cdots$ $\mathtt{NP} : \mathit{cat}'$ **john**
$\mid$ sincerity admires john; $\bar{\theta})$

- Parameters can be re-estimated from training data of (L_i, S_i) pairs where L_i is the meaning associated with sentence S_i.

- This is what the `t-command` of TheBench does.

- sample training data in TheBench format:

```
Mary persuaded Harry to study : persuade (study harry) harry mary
Mary promised Harry to study : promise (study mary) harry mary
Mary expected Harry to study : expect (study harry) mary
```

- The log-likelihood of the training data is:

$$O(\bar{\theta}) = \sum_{i=1}^{n} log\, P(L_i \mid S_i; \bar{\theta}) = \sum_{i=1}^{n} (\sum_{D} P(L_i, D \mid S_i; \bar{\theta})) \qquad (5)$$

(To see how likely our training data is according to our grammar, analyze S_i and add up all analyses (D) that led to L_i).

$$O(\bar{\theta}) = \sum_{i=1}^{n} log\, P(L_i \mid S_i; \bar{\theta}) = \sum_{i=1}^{n} (\sum_{D} P(L_i, D \mid S_i; \bar{\theta})) \qquad (6)$$

- You can see how syntax is marginalized by summing over all derivations D of (L_i, S_i). This needs a PARSER. (next lecture)
- For individual parameters we look at the partial derivative of (6) with respect to parameter θ_j.
- The local gradient of θ_j with feature f_j for the training pair (L_i, S_i) is the difference between two expected values:
  $E^i_{\text{true},f_j} = E(L_i, D, S_i)$ $\qquad$ $E^i_{\text{false},f_j} = E(L_k, D, S_i), k \neq i$

$$\frac{\partial O_i}{\partial \theta_j} = \text{ if } E^i_{\text{false},f_j} \approx 0 \text{ then } 0 \text{ else } (E^i_{\text{true},f_j} - E^i_{\text{false},f_j}) \qquad (7)$$

$$\frac{\partial O_i}{\partial \theta_j} = \text{ if } E^i_{\text{false},f_j} \approx 0 \text{ then } 0 \text{ else } (E^i_{\text{true},f_j} - E^i_{\text{false},f_j})$$

- The gradient will be negative if feature f_j contributes more to incorrect parses than it does to the correct parses of (L_i, S_i).
- It will be zero if all parses are correct,
- and positive otherwise.

- Expected values of f_j are therefore calculated under the distributions $P(D \mid S_i, L_i; \bar{\theta})$ and $P(L_k, D \mid S_i; \bar{\theta}), k \neq i$.

- For the overall training set, using sums, the partial derivative is:

$$\frac{\partial O}{\partial \theta_j} = \text{ if } E_{\text{false},f_j} \approx 0 \text{ then } 0 \text{ else } (E_{\text{true},f_j} - E_{\text{false},f_j}) \tag{8}$$

$$E_{\text{true},f_j} = \sum_{i=1}^{n} \sum_{D} f_j(L_i, D, S_i) P(D \mid S_i, L_i; \bar{\theta}) \tag{9}$$

$$E_{\text{false},f_j} = \sum_{i=1}^{n} \sum_{D} f_j(L_k, D, S_i) P(D \mid S_i, L_k; \bar{\theta}), k \neq i \tag{10}$$

- We can think of this gradient search as a way to investigate the Continuity Hypothesis of Crain and Thornton (1998) in linguistics.

- Every candidate model of grammar would be a possible grammar if the model follows from a theory of NL grammar.

- Once we have the derivative, we use Stochastic Gradient Ascent to re-estimate the parameters:

Initialize $\bar{\theta}$ to some value. (11)

$$
\begin{aligned}
&\text{` for } k = 0 \cdots N-1 \\
&\quad \text{for } i = 1 \cdots n \\
&\qquad \bar{\theta} = \bar{\theta} + \frac{\alpha_0}{1 + c(i + kn)} \frac{\partial log\, P(L_i|S_i;\bar{\theta})}{\partial \bar{\theta}}
\end{aligned}
$$

- N is the number of passes over the training set,

- n is the training set size,

- α_0 and c are learning-rate parameters.[1]

[1] Specified in experiment files; see TheBench Guide, §7.5, §7.6.

- This is gradient *ascent*, so initialize $\bar{\theta}$ accordingly. Default is 1.0. (Note: This is not a probability).

- Stochastic gradient search? Are our grammars stochastic?

- No. Every grammar is a proxy for categorial understanding of the form-meaning relation. Linguistic grammars are symbolic empirical species. Formal grammars are, ehm, formal species.

- What is stochastic is the space of all (and hopefully only) human grammars.

- After model training and development, we can do model selection.
- During training, we tend to generate many models, depending on training parameters (data and hyperparameters).
- This is what the experiment facility of TheBench's `t-command` is designed for. There are as many experiments as the number of lines in an experiment file. See TheBench Guide §7.5, §7.6.
- Unlike LLMs, scientific models do not tweak their response

  so that model choice can be independently replicable.

- Model selection can be
  - performance-based (e.g. accuracy, precision, recall, log-likelihood)
  - cross validation (e.g. split the data into N subsets, train on N-1 subsets and test on 1)
  - generalized testing (check with really unseen data, cf. cross-validation)
  - Bias check (e.g. overfitting: high variance, low bias, too little complexity in data for finding patterns, poor generalization to unseen patterns)

    (underfitting: high bias, low variance, too much complexity already in data to allow discovery)
- Model selection has not been streamlined in TheBench. We leave it to the experimenter (for now).

Bozşahin, Cem. 2024. *TheBench Guide*.
`https://github.com/bozsahin/thebench`.

Bozşahin, Cem. 2025. *Connecting Social Semiotics, Grammaticality and Meaningfulness: The Verb*. Newcastle upon Tyne: Cambridge Scholars.

Crain, Stephen, and Rosalind Thornton. 1998. *Investigations in Universal Grammar*. Cambridge MA: MIT Press.

Zettlemoyer, Luke, and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proc. of the 21st Conf. on Uncertainty in Artificial Intelligence*. Edinburgh.