Class 12 (Week 6, T): Inner workings of the grammar III, Constraint weighting

To do

- ☐ Read **Moore-Cantwell & Pater** for Thursday (Nov. 5)
    - ▪ presenters, if you e-mail me your handout as a PDF by noon Thurs., I can print
- ☐ Prepare at least one **question or point for discussion** on the reading
- ☐ Computing **homework** on harmonic serialism in OT-Help is due Thursday (Nov. 5). Turn in write-up on paper; I enable file upload on CCLE for your 3 text files.

Overview: Last week we experimented with revising Classic OT's assumptions about GEN, as in Harmonic Serialism. What if we revised Classic OT's assumptions about EVAL?

0. Discuss Belfast English data, especially Richness of the Base

1. Review

- o What is a constraint? (There could be many answers to this question!)

- o What are some things you know about EVAL?

- o How does strict domination work?

2. It doesn't have to be this way

- What if instead of a constraint ranking, we just gave each constraint a number.
  - Discuss some plausible ways we could make EVAL work (fill in violations first, as warm-up—leave rightmost column blank):

/hat$_1$ j$_2$oga/	NOGLIDE INITIALSYLLABLE *weight:* 10	ALIGN (Stem,R; Syll, R) *weight:* 8	DEP-C *weight:* 3	*OBSTRUENT-NASAL *weight:* 2	IDENT(nas) *weight:* 1	harmony
a hat$_1$.j$_2$o.ga						
b ha.t$_1$j$_2$o.ga						
c hat$_1$.n$_8$j$_2$o.ga						
d han$_1$.n$_8$j$_2$o.ga						

3. How Harmonic Grammar does it

- A candidate's "harmony" is the weighted sum of its violations
  - Fill in the last column of the tableau above
- The winner is the candidate with the best harmony
- <u>Negative signs</u>: Some people make the weights negative, some make the violations negative, some do neither.
  - Either way, the closer the harmony is to zero, the better the candidate is.
  - Unless you can have constraints that bestow bonuses rather than penalties?

- Some key references: Legendre, Miyata & Smolensky 1990; Legendre, Sorace & Smolensky 2006; Boersma & Pater 2008; Potts et al. 2010

4. Can Harmonic Grammar ever make a candidate win that would be harmonically bounded in Classic OT?

- Fill in violations (inspired by Woleaian—Sohn 1975)
- In OT, which candidates are harmonically bounded and which aren't (could win under some ranking)?
- Is there a weighting of the constraints that could make any of the harmonically bounded candidates win under Harmonic Grammar?
  - Hint: you will have to think back to high school and solve a system of inequalities!

/malamara/	*aCa *weight:*	IDENT(lo) *weight:*	harmony
a malamara			
b melamera			
c melamara			
d malemara			

5. Using only non-harmonically bounded candidates, can Harmonic Grammar produce a different typology?

- o Fill in violations
- o What is the OT typology? (i.e., is *a* & *c* a possible language? *a* & *d?* etc.)

- o What is the Harmonic Grammar typology?

/bla/	*COMPLEXONSET	MAX-C	harmony
a bla			
b ba			

/spli/	*COMPLEXONSET	MAX-C	harmony
c spli			
d pli			
e pi			

6. How to turn this into a theory of variation: Noisy Harmonic Grammar

- We've got a number for every candidate, not just the winner—can't we use that somehow?
- Noisy HG's solution (see Boersma & Pater 2008 for references)
  - Don't use those numbers directly, but add some **noise** to each constraint's weight every time
  - o Let's use a random-number phone app to generate a noise value for each constraint and see what happens.
    - The random number should be drawn from a normal (bell-curve) distribution, centered on 0.

/tɾi/	*ALVEOLARRHOTIC *grammar's weight: 1.5* *noise this time:* *weight this time:*	*DENTAL *grammar's weight: 1* *noise this time:* *weight this time:*	average harmony	harmony on this occasion
a tɾi	*			
b t̪ɾi		*		

- To know each candidate's probability of winning, we can either simulate (easier) or use numerical integration (harder).
- Software
  - OTSoft and Praat both support noisy HG
  - including a way to learn weights (Gradual Learning Algorithm—we'll talk about this next week)

7. How to turn this into a theory of variation: Maximum Entropy

- Instead of using noise, we turn each candidate's harmony directly into a probability of being chosen.
  - Let's fill it in: the last column will be the candidate's probability

/tɾi/	*ALVEOLARRHOTIC *grammar's weight: 1.5*	*DENTAL *grammar's weight: 1*	harmony	$e^{-harmony}$	share of total $e^{-harmony}$
a tɾi	*				
b t̪ɾi		*			
			total:		

- Probability is usually presented using this expression—can we pick it apart and convince ourselves thatt it's equivalent to what we just did?

  $$p(\omega) = \frac{1}{Z} e^{-\Sigma_i w_i C_i(\omega)}$$

  - ω is a candidate
  - $p(\omega)$ is that candidate's probability of being uttered, according to the grammar
  - w_i is the weight of the ith constraint
  - $C_i(\omega)$ is the number of constraints that the ith constraint assigns to candidate ω
  - and $Z = \Sigma_j e^{-\Sigma_i w_i C_i(\omega_j)}$

- Can a candidate have a probability of zero? one?

8. More on MaxEnt

- Why exponentiate?
  - Because it makes the math of learning weights work (see below)
- Intellectual roots
  - information theory: Jaynes (1957)
  - cognitive science: Smolensky (1986)
  - computer science: Berger, Della Pietra & Della Pietra (1996), Della Pietra, Della Pietra & Lafferty (1997)
  - as an implementation of OT's GEN+EVAL architecture: Goldwater and Johnson (2003)
  - a MaxEnt classifier is basically the same thing as a logistic regression model
    - except that conceptually, in a MaxEnt grammar you don't have to decide what category each outcome belongs to
    - e.g., to do a regression model, you'd have to say that [t̪ɾi] and [lit̪ɚ] both belong to the category "dentalized"
- Software
  - OTSoft and the MaxEnt Grammar Tool both implement MaxEnt grammars
  - The MaxEnt Grammar Tool is probably more accurate, and definitely more customizable

9. How are weights learned in MaxEnt?

- Learning algorithm is trying to maximize *predicted probability of data – penalty for weights*
    - This expression, which we'll spell out below, is the **objective function** that the learner is trying to adjust the weights in order to optimize.

- <u>Predicted probability of data</u>
    - Suppose we have observed 10 utterances, from a variety of inputs

[t̪ɾi]	[litɚ]	[mit̪ɚ]	[lit̪ɚ]	[lit̪ɚ]	[t̪ɾi]	[lit̪ɚ]	[tɾi]	[lit̪ɚ]	[mitɚ]

-
    - Each of those utterances is a candidate in a tableau
        - and our grammar assigns it a probability

[t̪ɾi]	[litɚ]	[mit̪ɚ]	[lit̪ɚ]	[lit̪ɚ]	[t̪ɾi]	[lit̪ɚ]	[tɾi]	[lit̪ɚ]	[mitɚ]
0.62	0.38	0.62	0.62	0.62	0.62	0.62	0.38	0.62	0.38

-
    - We want to maximize the probability that the grammar assigns to the whole series of events that we observed:
        - with current weights: 0.62*0.38*0.62*0.62*0.62*0.62*0.62*0.38*0.62*0.38 = 0.0019
    - That is, adjust the weights until that number gets as big as it can.

    - Because these numbers get very small, let's take the natural logarithm instead
        - currently, ln(0.62*...*0.38)=ln(0.62)+...+ln(0.38) = 7ln(0.62) + 3ln(0.38) = -6.24
        - adjust the weights to bring this number as close to zero as possible
        - (this may remind you of the definition of entropy!)

- <u>Penalty for weights</u>, aka the **prior—we need a digression first**

10. Fitting and overfitting

- o Discuss: there are some differences among the words in our toy data above that prevent us from getting a perfect fit to the data. We could achieve a perfect fit by introducing separate constraints for *tree*, *liter*, and *meter*. Pros and cons?

- In machine learning applications, people worry about **overfitting**. I'll draw some pictures on the board.
    - To summarize what just happened on the board: a model that fits the *existing* data too well could make worse predictions about *new* data.
- One response to overfitting is to do some model comparison to decide if some independent variables (in our case, constraints) should be removed altogether.
- But another response is to (decide how much to) **penalize** weights/coefficients that are large.
    - We want to trade weight/coefficient size off against fit: in order to have a large coefficient, a constraint/variable should do a lot of work in explaining the data.

11. Weight penalty in MaxEnt, first and second approximations

- Just add up the square of every constraint's weight, maybe times a constant (for later convenience, we'll call it $1/\sigma^2$) that determines how much we care about large weights.
  - $\frac{1}{\sigma^2}\sum_{i=1}^{n} w_i^2$
  - Remember that now we'll be subtracting this number from the predicted probability of the data, and adjusting the weights to maximize the result
  - o What happens if $1/\sigma^2$ is really big? Really small?

- What we just did penalizes every weight for deviating from zero.
- We could let in some more generality, and give each constraint *i* its own default value μ_i that it shouldn't deviate from:
  - $\frac{1}{\sigma^2}\sum_{i=1}^{n} (w_i - \mu_i)^2$
  - o Again, what happens if $1/\sigma^2$ is really big? Really small?

- Example: White (2013) gives each *MAP(X,Y) constraint a μ based on how perceptually different *X* and *Y* are, according to confusion experiments.

12. Weight penalty in MaxEnt, for real

- Rather than use a single $1/\sigma^2$, let each constraint *i* have its own σ_i
  - $\sum_{i=1}^{n} \frac{(w_i - \mu_i)^2}{2\sigma_i^2}$
- Some constraints are pretty OK with deviating from their default value (so σ is big or small?), and some really want to stick close to it.
- Example: Wilson (2006) gives each markedness constraint a σ based again on confusability

- This is known as a Gaussian prior, and it's not the only choice

o Supposing μs of zero, what would the Gaussian prior say about these two sets of weights: {1,1,1,99}, {25,25,25,25}?

⇒ This choice of smoothing term prefers to spread responsibility (weight) evenly across constraints as much as possible.

  - If there are two constraints that could both explain the data, weight them equally rather than just picking one.

o Can you dream up a smoothing term that would have the opposite preference—prefer to pick just one constraint and load all the weight onto it?

13. Why is the smoothing term (aka regularization term) also called a prior?

- Bayes' Law:

"posterior": how probable is model given observed data

"likelihood": how probable are data according to model

"prior": how probable is model to begin with?

$$prob(model \mid data) = \frac{prob(data \mid model) * prob(model)}{prob(data)}$$

"evidence": usually irrelevant, since same for all models—just needed to ensure probabilities sum to 1

posterior likelihood

- Taking the log, $\ln p(model|data) = \ln p(data|model) + \ln p(model) - \ln p\,(data)$

  - Compare and contrast this to our MaxEnt objective function with smoothing.

14. Summing up the smoothing bias

- Smoothing (a.k.a. regularization) is a way to avoid overfitting:
  - Tell your software to find a model that compromises between fitting the data and staying close to default parameter values (constraint weights, in our case)
- OTSoft essentially has no prior—it just fits every weight as closely as possible
  - which is why you need to tell it what the maximum weight is, in case a constraint wants to have infinite weight (default: 50)
- The MaxEnt Grammar Tool has zero for all μs and a huge value for all σs by default, but you can customize all of those values.
- This is all well and good for modeling, but do people do it when learning variation?
- That is, beyond any substantive biases (which Bruce will discuss Thurs.), do human learners have a "smoothing bias" to keep weights small?
- Interesting studies of how smoothing itself (with plain-vanilla 0 μs and every constraint having the same σ) may capture important aspects of learning:
  - <u>Martinian leakage</u> (Martin 2011): how phonotactics of monomorphemes can leak into compounds, because learners spread the responsibility for, eg., lack of geminates over both specific constraints (NoGeminateWithinMorpheme) and general constraints (NoGeminateAnywhere)
  - <u>Ryanian variationogenesis</u> (Ryan 2010): frequencies of minor variants (in Tagalog morpheme order) can be predicted from learning just the major variants, plus smoothing.

15. Next time

- On Thursday we probably continue this handout, and have presentation on Moore-Cantwell & Pater (submitted).
- Next week: We move to a closely related topic, learning models.

References

Berger, Adam L, Stephen A Della Pietra & Vincent J Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics* 22(1). 39–71.

Boersma, Paul & Joe Pater. 2008. Convergence properties of a Gradual Learning Algorithm for Harmonic Grammar. Manuscript. University of Amsterdam and University of Massachusetts, Amherst, ms.

Della Pietra, Stephen, Vincent J Della Pietra & John D Lafferty. 1997. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19. 380–393.

Goldwater, Sharon & Mark Johnson. 2003. Learning OT Constraint Rankings Using a Maximum Entropy Model. In Jennifer Spenader, Anders Eriksson & Östen Dahl (eds.), *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, 111–120. Stockholm: Stockholm University.

Jaynes, Edwin T. 1957. Information theory and statistical mechanics. *Physical Review, Series II* 106(4). 620–630.

Legendre, Geraldine, Yoshiro Miyata & Paul Smolensky. 1990. Harmonic Grammar – A formal multi-level connectionist theory of linguistic well-formedness: An Application. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, 884–891. Mahwah, NJ: Lawrence Erlbaum Associates.

Legendre, Géraldine, Antonella Sorace & Paul Smolensky. 2006. The Optimality Theory–Harmonic Grammar Connection. In Paul Smolensky & Géraldine Legendre (eds.), *The Harmonic Mind*, 339–402. Cambridge, MA: MIT Press.

Martin, Andrew. 2011. Grammars leak: modeling how phonotactic generalizations interact within the grammar. *Language* 87(4). 751–770.

Moore-Cantwell, Claire & Joe Pater. submitted. Gradient exceptionality in Maximum Entropy Grammar with lexically specific constraints.

Potts, Christopher, Joe Pater, Karen Jesney, Rajesh Bhatt & Michael Becker. 2010. Harmonic Grammar with linear programming: from linear systems to linguistic typology. *Phonology* 27(01). 77–117.

Ryan, Kevin M. 2010. Variable affix order: grammar and learning. *Language* 86(4). 758–791.

Smolensky, Paul. 1986. Information processing in dynamical systems: Foundations of harmony theory. In D. Rumelhart, J. McClelland, the PDP Research Group, D. Rumelhart, J. McClelland & the PDP Research Group (eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 1: Foundations, 194–281. Cambridge, Mass.: Bradford Books/MIT Press.

Sohn, Ho-min. 1975. *Woleaian Reference Grammar*. Honolulu: University of Hawaii Press.

White, James. 2013. Bias in phonological learning: evidence from saltation. UCLA PhD dissertation.

Wilson, Colin. 2001. Consonant Cluster Neutralisation and Targeted Constraints. *Phonology* 18(1). 147–197.

Wilson, Colin. 2006. Learning Phonology with Substantive Bias: An Experimental and Computational Study of Velar Palatalization. *Cognitive Science* 30(5). 945–982.