Computational Phonology, class 3: Weighted constraints

Adam Albright

CreteLing 2022 — July 2022

creteling2022.computational.phonology.party

From n-grams and similarity to grammars

- The models discussed last time calculate the “goodness” (probability, wordlikeness) of a string relative to other words
- Phonological grammars are generally formulated to do something different
  - Take an entry from the lexicon (UR) and compute its pronunciation (SR)
  - Decision based on application of rules, or satisfaction of constraints
- We’ll build our way towards models of this form

Some types of models

- Generative
  - Encodes the process of constructing a surface form
  - Probability of a surface form $=$ joint probability of steps needed to generate it
  - Categorical, or probabilistic
- Discriminative
  - Decides between possible outcomes
  - Classification models: decide between classes (languages, word classes, etc.)
  - Another discriminative function: probability of yes/no

Constraint-based classification

A simple model of relative acceptability

	*[bn	*ɹg]	OCP(LAB)	*[bl
a. blɪg				*
b. mɪp			*	
c. smɛɹg		*		
d. bnɪk	*			

- Forms that violate higher constraints (left) are worse than forms that violate lower constraints (right) (Everett and Berent, 1997; Coetzee, 2004)
    - blɪg > mɪp > smɛɹg > bnɪk

Constraint-based classification

From acceptability to probabilities

	*[bn	*ɹg]	OCP(LAB)	*[bl			
Weight:	8	4	2	1	Sum	e^{-Sum}	Prob
a. blɪg				*	1	0.3679	.7049
b. mɪp			*		2	0.1353	.2593
c. smɛɹg		*			4	0.0183	.0350
d. bnɪk	*				8	0.0003	.0006

- Constraints have numeric weights
- Competing strings (candidates) incur violations
  - Total violations = sum of weighted violations ($\Sigma v \cdot w$)
- Score: $\exp(-\mathrm{sum})$
  - A common move, used in Maximum Entropy models (more below)
- Probability of string = Score / Summed scores for all

Why is this useful?

- As it turns out, this formulation makes it straightforward to find the weights that best match a given (trained) probability distribution
- Human learners don’t receive relative acceptability judgments, but they do receive words of different shapes, with different relative frequencies ($\Rightarrow$probabilities)

The learning task

- Given…
  - A set of structural descriptions (constraints)
  - A set of output forms, with their violations
  - A probability distribution over of a set of output forms
- Learn…
  - Weights that will generate the observed distribution

A familiar learning problem!

- Perceptron learning
- Other forms of learning in neural nets
- Maximum Entropy models

Maximum Entropy: Berger et al. (1996)

- Translation device: produce French renditions of English preposition ‘in’
- Hypothetical observation: human translator always uses one of five phrases:
  - {*dans*, *en*, *à*, *au cours de*, *pendant*}
  - p(*dans*) $+$ p(*en*) $+$ p(*à*) … $=$ 1

Maximum Entropy: Berger et al. (1996) (*cont.*)

- Possible hypotheses about the model that produced this data:
  - p(*dans*) = .5, p(*en*) = .2, others = .1
  - p(*dans*) = 1, others = 0
  - p(*dans*) = p(*en*) = p(*à*) … = .2
- Intuitively, the last is the most compatible with the description of the data
  - The others unwarrantedly make stronger assertions
  - As it turns out, this is also the one that maximizes the likelihood of the data

A terminological note

- As we transition to talking about constraint-based models of phonology (Harmonic Grammar, OT), it is convenient to refer to the descriptions (the columns) as constraints
- The rows (possible outputs) are candidates
- Unfortunate terminology clash with more usual terms in machine learning literature: features and constraints

Entropy

Entropy (Shannon 1948)

Entropy (Shannon 1948)

$$H = -\sum_{i=1}^{n} p(i)\ log\ p(i)$$

- Measure of uncertainty, or unpredictability of a set of events
- Events = series of die tosses, words in a text, etc.
- Unpredictability measured by how much information you have to transmit in order to convey the data
  - If everyone's on the same page and your language is sensible, things that are highly predictable don't need much spelling out
  - Things that are surprising may need more elaboration

Entropy example

- Coin tosses: heads or tails
- Assuming fair coin (P(heads) $=$ P(tails) $=$ .5)
    - $\log_2(.5) = -1$
    - H $= -(.5 \times \log(.5) + .5 \times (log(.5))) = -(-1) = 1$
- Corresponds to optimal encoding: 0 or 1 (1 bit)

Entropy example

the farmer in the dell the farmer in the dell hi ho the derry oh the farmer in the dell

- 8 unique words (*the*, *farmer*, *in*, *dell*, *hi*, *ho*, *derry*, *oh*)
- One possible encoding:

the	000	hi	100
farmer	001	ho	101
in	010	derry	110
dell	011	oh	111

- Text:

  000 001 010 000 011 000 001 010 000 011 100 101 000 110 111 000 001 010 000 011

- 3 bits (digits of binary coding) per word

Entropy example

Optimizing encoding: some words more frequent than others

the	$\frac{7}{20}$	hi	$\frac{1}{20}$
farmer	$\frac{3}{20}$	ho	$\frac{1}{20}$
in	$\frac{3}{20}$	derry	$\frac{1}{20}$
dell	$\frac{3}{20}$	oh	$\frac{1}{20}$

- Give more frequent words shorter encodings (e.g., 0), rarer words longer encodings (e.g., 110)
- Entropy H $= -\frac{7}{20} log_2 \frac{7}{20} - 3(\frac{3}{20} log_2 \frac{3}{20}) - 4(\frac{1}{20} log_2 \frac{1}{20}) = 2.626$

Entropy example (*cont.*)

- Given optimal encoding in which more frequent items are shorter, words in this text require average 2.626 bits
  - Modest savings over 3, which we got when all words were equally frequent and we used a equal-length encoding scheme
- The more predictable things are, the less we have to say about them → smaller entropy; more uniform distribution → higher the entropy

In a similar vein

Beyoncé: Pray you catch me (2016)

Prayin’ to catch you whispering
I’m prayin’ you catch me listening
I’m prayin’ to catch you whispering
I’m prayin’ you catch me
I’m prayin’ to catch you whispering
I’m prayin’ you catch me listening
I’m prayin’ you catch me

- Also 8 unique words (3 bits per word)

catch	$\frac{7}{39}$	me	$\frac{4}{39}$
you	$\frac{7}{39}$	to	$\frac{3}{39}$
prayin’	$\frac{7}{39}$	whispering	$\frac{3}{39}$
I’m	$\frac{6}{39}$	listening	$\frac{2}{39}$

In a similar vein (*cont.*)

- Entropy H $= -3(\frac{7}{39}log_2\frac{7}{39}) - \frac{6}{39}log_2\frac{6}{39} - \frac{4}{39}log_2\frac{4}{39} - 2(\frac{3}{39}log_2\frac{3}{39}) - \frac{2}{39}log_2\frac{2}{39}$

An exercise for the reader

Determine the entropy of the following text (*Largo al factotum* from *The Barber of Seville*)

Figaro! Figaro! Figaro!
Figaro! Figaro! Figaro!
Figaro! Figaro! Figaro!
Ahime, ahime, che furia!
Ahime, che folla!
Uno alla volta, per carità! per carità! per carità!
Uno alla volta, Uno alla volta, Uno alla volta, per carità!
Ehi, Figaro! Son quà. Ehi, Figaro! Son quà.
Figaro quà, Figaro là, Figaro quà, Figaro là,
Figaro su, Figaro giù, Figaro su, Figaro giù...

- 16 unique words: 4 bits with equal length coding
- What is theoretical minimum avg. length per word, given this frequency distribution? (i.e., entropy)

Or, if you prefer…

Dua Lipa ‘Don’t start now’ (2019)

Oh, oh
Don’t come out, out, out
Don’t show up, up, up
Don’t start now (oh)
Oh, oh
Don’t come out, out
I’m not where you left me at all

- 16 unique words: 4 bits with equal length coding
- What is theoretical minimum avg. length per word, given this frequency distribution? (i.e., entropy)

Or perhaps…

Lady Gaga and Ariana Grande ‘Rain on me’ (2020)

I’d rather be dry, but at least I’m alive
Rain on me, rain, rain
Rain on me, rain, rain
I’d rather be dry, but at least I’m alive
Rain on me, rain, rain
Rain on me
Rain on me
Mmm, oh yeah, baby
Rain on me

- 16 unique words: 4 bits with equal length coding
- What is theoretical minimum avg. length per word, given this frequency distribution? (i.e., entropy)

Back to French translation

- In the translator example, we’re not just interested in the distribution of words in French texts
  - How does distribution change depending on English input?
- Process (translation) produces outputs *y* from among set of outputs Y
- Choice of *y* depends on (is conditioned by) input $x \in X$
- Training data (observations): pairs (*x*,*y*)
- Conditional entropy
  - $$\mathrm{H}(\vec{y}|\vec{x}) = \sum_i p(x_i) H(\vec{y}|x_i)$$
    $$= -\sum_i p(x_i) \sum_j p(y_j|x_i)\ log\ p(y_j|x_i)$$
    $$= -\sum_{x,y} p(x,y)\ log\ p(y|x)$$

Berger, Della Pietra and Della Pietra's observation

Conditional entropy

- $$\mathrm{H}(\vec{y}|\vec{x}) = \sum_i p(x_i) H(\vec{y}|x_i)$$
  $$= -\sum_i p(x_i) \sum_j p(y_j|x_i)\ log\ p(y_j|x_i)$$
  $$= -\sum_{x,y} p(x,y)\ log\ p(y|x)$$

Log likelihood of training data

- $$likelihood(\vec{x},\vec{y}) = \prod_{x,y} p(y|x)^{freq(x,y)}$$
- $$log\ likelihood(\vec{x},\vec{y}) = log \prod_{x,y} p(y|x)^{freq(x,y)}$$
  $$= \sum_{x,y} freq(x,y) \cdot log\ p(y|x) \text{(These are proportional!)}$$

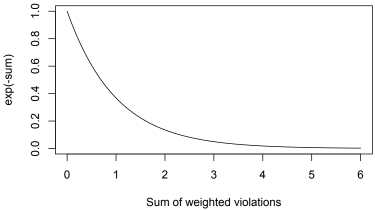
Applied to phonology: Goldwater and Johnson (2003)

- Grammar assigns conditional probability distribution: P(output $y|$input x)
- Probabilities depend on weighted sums of violations
- Fancier decision rule: $\mathrm{P}(\mathrm{y}_i) =$
$$\frac{e^{-\sum(\text{weighted violations}(\mathrm{y}_i))}}{\sum_j e^{-\sum(\text{weighted violations}(\mathrm{y}_j))}}$$
- We’ve already seen this decision rule

	*[bn	*ɹg]	OCP(LAB)	*[bl			
Weight:	8	4	2	1	Sum	e^{-Sum}	Prob
a. blɪg				*	1	0.3679	.7049
b. mɪp			*		2	0.1353	.2593
c. smɛɹg		*			4	0.0183	.0350
d. bnɪk	*				8	0.0003	.0006

The effect of this decision rule

- Score $= \exp(-\sum \text{weight}_c \times \text{c(output)})$



- Assumes positive weights and violations (remove '$-$' if violations or weights are negative)
- Tends to make distribution more concentrated on output(s) with low summed violations
- Makes sense to have modest sums of weights (more like 0–10 than 0–100 or more)

Goldwater and Johnson (2003)

- Often, probabilities are conditioned on input
  - Like French translator example, phonology converts input to output
  - Probability distribution of a set of SR’s, given the UR: P($\vec{\mathrm{SR}}|\mathrm{UR}$)

/bnɪk/ Weight:	*[bn 8	CONTIG 4	MAX 2	DEP 1	Sum	e^{-Sum}	Prob
a. bənɪk				*	1	0.3679	.7270
b. nɪk			*		2	0.1353	.2674
c. bɪk		*	*		6	0.0025	.0049
d. bnɪk	*				8	0.0003	.0007

Interlude: Markedness and Faithfulness

/bnɪk/	*[bn	Contig	Max	Dep			
Weight:	8	4	2	1	Sum	e^{-Sum}	Prob
a. bənɪk				*	1	0.3679	.7270
b. nɪk			*		2	0.1353	.2674
c. bɪk		*	*		6	0.0025	.0049
d. bnɪk	*				8	0.0003	.0007

- Tableaus evaluate candidate outputs for a given input
    - Assigns a conditional probability distribution $P(\vec{y}|x|)$
- Constraint-based approaches to phonology have generally adopted a model in which constraints regulate surface forms (Markedness) and the relation between input and output (Faithfulness)

Interlude: Markedness and Faithfulness

/bnɪk/	*[bn	Contig	Max	Dep			
Weight:	8	4	2	1	Sum	e^{-Sum}	Prob
a. bənɪk				*	1	0.3679	.7270
b. nɪk			*		2	0.1353	.2674
c. bɪk		*	*		6	0.0025	.0049
d. bnɪk	*				8	0.0003	.0007

- Markedness
  - *[bn, *[bl, *ɹg], etc.
- Faithfulness:
  - Max: don’t delete (every element in the input must have a correspondent in the output)
  - Dep: don’t epenthesize (every element in the output must have a correspondent in the input)
  - Contig: don’t skip/don’t intrude (etc.)

Interlude: Markedness and Faithfulness

/bnɪk/	*[bn	Contig	Max	Dep			
Weight:	8	4	2	1	Sum	e^{-Sum}	Prob
a. bənɪk				*	1	0.3679	.7270
b. nɪk			*		2	0.1353	.2674
c. bɪk		*	*		6	0.0025	.0049
d. bnɪk	*				8	0.0003	.0007

- Space of candidates is infinite (GEN), but Faithfulness constraints generally reduce competition to those that deviate from the input in limited ways
- Categorical vs. gradient outputs: determined by decision rule

Learning weights

We want to find the weights that maximize the likelihood of the data (or maximize the conditional entropy)

- Training data: a series of N observations
  - Conditioned: $(y_1|x_1)$, $(y_2|x_2)$, $(y_3|x_3)$, …$(y_n|x_n)$
- Probability distribution:
  - $\text{Prob}(d_i) = \frac{\text{Count}(d_i)}{\text{N}}$
- We want to find a model that produces this probability distribution as closely as possible

Learning weights

Constraints are a type of ‘indicator’ function, that tell us whether or not they care about a given situation

- *C[$-$son] =
  $$\begin{cases} 0 & \text{if a consonant followed by a sonorant} \\ 1 & \text{otherwise} \end{cases}$$
- Adjusting weights $\rightarrow$ adjust prediction about probability of violations (i.e., how often forms violating the constraint should occur)
- Pays to do this if the *empirical* distribution in the learning data diverges from the *expected* distribution according to the current model

Learning weights

Some terminology

- Empirical distribution: $\tilde{p}(*\mathrm{C}[-\mathrm{son}])$
  - Probability in the data of a *C[−son] violation
- Expected distribution: $p(*\mathrm{C}[-\mathrm{son}])$
  - Model's predicted probability of a *C[−son] violation

Empirical (observed) violations

- Depends on how many times a given datum (*x*,*y*) occurs, and whether the constraint cares about (*x*,*y*)

$$\tilde{p}(con) \equiv \sum_{x,y} \tilde{p}(x,y)con(x,y)$$

- Count how many times violating sequences occur in the data

Expected violations

- How often do we expect to see violations, given a particular model that assigns a distribution over outputs?
- For simpler unconditional case:
  - $p(con) \equiv \sum_{Y} p(y) con(y)$
- Conditioned on inputs:
  - We don't try to predict how often a given input occurs (just use the empirical distribution)
  - Given those inputs, we use the probability distribution over outputs

$$p(con) \equiv \sum_{x,y} \tilde{p}(x) p(y|x) con(x,y)$$

Learning weights

- Goal: enforce the condition[1] $p(con) = \tilde{p}(con)$
  - $\sum_{x,y} \tilde{p}(x,y) con(x,y) = \sum_{x,y} \tilde{p}(x) p(y|x) con(x,y)$
- At any arbitrary point in learning, there may be a discrepancy
  - $p(con) - \tilde{p}(con) > 0$
- Adjust model probabilities by adjusting weights
  - Changing weight of a constraint in one direction may decrease $p(con) - \tilde{p}(con)$, while changing in other direction increases discrepancy

[1] This is called a *constraint* in the maxent literature.

Convexity

Della Pietra, Della Pietra, and Lafferty (1997): problem of finding weights is convex

- Moves that decrease $p(con) - \tilde{p}(con)$ always move closer to optimal solution
- Various optimization techniques are applicable
  - Berger, Della Pietra and Della Pietra (1996): *improved iterative scaling*
  - Goldwater and Johnson (2003), Hayes and Wilson (2008): *conjugate gradient* methods
  - Jäger (2007): *stochastic gradient ascent*

Learning weights

A convex problem: space of weights for two constraints

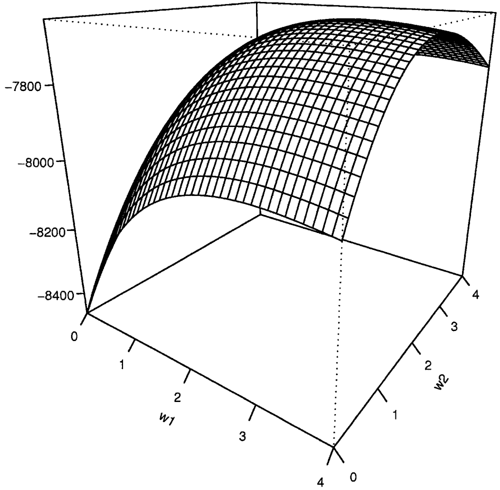


Figure 1
The surface defined by the probability of a representative training set for the grammar given in table 1

A very simple strategy

- Grammar at time t: a vector $\vec{r}$ of weights
  - Ordered list of real numbers: [2, 5, 5, 3, 6, 4]
- On hearing datum (/x$_t$/, [y$_t$]):
  - Apply current grammar to find output: (/x$_t$/, [z$_t$])
  - If $z_t \neq y_t$: adjust weights according to current plasticity η

$$(\vec{r}_{t+1})_k = (\vec{r}_t)_k + \eta \cdot (c_k(x_t, z_t) - c_k(x_t, y_t))$$

A very simple strategy

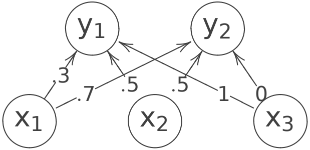
The Perceptron Algorithm (Rosenblatt 1962)

$$(\vec{r}_{t+1})_k = (\vec{r}_t)_k + \eta \cdot (c_k(x_t, z_t) - c_k(x_t, y_t))$$

- Weight of constraint k at time $t+1$, given input /x$_t$/, actual form [y$_t$], current preference [z$_t$]
  $=$ ranking value of constraint k at time t
  $+$plasticity $\times$ difference in violations of y$_t$, z$_t$
- If actual output y$_t$ has more violations, demote this constraint by the difference
- If current favorite z$_t$ has more violations, promote this constraint by the difference
- For discussion, see Jäger (2007), Boersma and Pater (2008)

Perceptron learning

- Perceptrons: single-layered feedforward networks



  - Input layer: $x_1 \ldots x_i$
  - Connections from input to output nodes: weights [-1 …1]
  - Output layer: activated according to weighted sum of input activations
  - Activation(y_i) = $\sum_j weight_{i,j} \cdot x_j$
- Supervised learning: given input and target output activations, adjust weights so actual activation gets closer to target

An example on colab

- The French translator example
  - Weights for unequal probabilities of different outcomes
  - Weights for equal (flat) probability distribution
- A simple phonological example
- Acquisition of Dutch consonant clusters (see the next few slides)

An example: Jäger (2007)

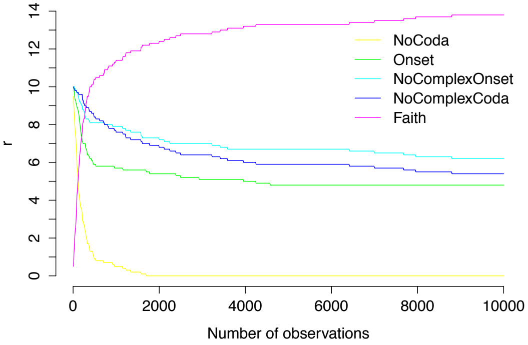
- Representative order of acquisition of syllable types in Dutch (Levelt & al 2000)
  - CV > CVC > VC > CCV > CCVC > CVCC > VCC > CCVCC
- Assume constraints:
  - Markedness: Onset, *Coda, *[CC, *CC]
  - Faithfulness (combines Max/Dep)
- Perceptron learning
  - Learner gets input-output pairs: /CVC/ → [CVC]
  - Hypothesizes output using current grammar: e.g., [CV]
  - If different, changes weights according to plasticity
  - Makes sense to add condition that weights may not cross zero

An example: Jäger (2007) (*cont.*)

- Data fed in proportion to corpus frequencies

Syl type	Freq	*Coda	Onset	*[CC	*CC]
CV	44.90%				
CVC	32.05%	*			
VC	11.99%	*	*		
V	3.85%		*		
CVCC	3.25%	*			*
CCVC	1.98%	*		*	
CCV	1.38%			*	
VCC	.42%	*	*		*
CCVCC	.26%	*		*	*

Jäger (2007): Learning trajectory



- Faithfulness gradually increases to exceed sum of all markedness violations that can co-occur in a form
  - CCVCC: *ComplexOnset + *ComplexCoda + *Coda
  - VCC: Onset + *ComplexCoda + *Coda

Properties of MaxEnt weighting

Another example (see the Colab notebook)

			NoCoda	NoCodaObs	Max	Dep
/pat/		pat	1	1		
	☞	pa			1	
		patə				1
/pak/		pak	1	1		
	☞	pa			1	
		patə				1

- Try finding weights with Stochastic Gradient Ascent
- Why is this an optimal solution?

Beyond perceptron

- Perceptron learning is guaranteed to converge, but it’s slow
  - Possibly useful—the Jäger example shows that we can scrutinize the learning path
- There are more efficient optimization strategies on the market
  - E.g., Hayes and Wilson (2008) use Conjugate gradient descent

Summary of what we have now

- Set of weighted constraints can assign a probability distribution over outcomes

	*C]	H	exp(H)	Prob
Weight:	-5			
pa		0	1	.331
pat	1	-5	.006738	.002
ta		0	1	.331
tat	1	-5	.006738	.002
ka		0	0	.331
kat	1	-5	.006738	.002

Summary of what we have now (*cont.*)

- Probabilities are conditioned on the input

/pa/	*C]	Ident(place)	H	exp(H)	Prob
Weight:	-5	-5			
pa	0	0	0	1.000000	0.980099
pat	1	0	-5	0.006738	0.006604
ta	0	1	-5	0.006738	0.006604
tat	1	1	-10	0.000045	0.000044
ka	0	1	-5	0.006738	0.006604
kat	1	1	-10	0.000045	0.000044

Summary of what we have now (*cont.*)

- Marginalized over inputs

	/pa/ .3333	/ta/ .3333	/ka/ .3333	Prob
pa	0.980	0.007	0.007	0.331
pat	0.007	0.000	0.000	0.002
ta	0.007	0.980	0.007	0.331
tat	0.000	0.007	0.000	0.002
ka	0.007	0.007	0.980	0.331
kat	0.000	0.000	0.007	0.002

Finding weights

- Infinite number of possible (sets of) weights
- An intuitive hypothesis: find the one the maximixes the (log) likelihood of the data
- Easily found by adjusting weights so that each step increases the log likelihood
    - Objective function: $L(w) = -logP(D|w)$

Beyond log likelihood

- Matching data is not the only possible objective
  - Priors: conditions not imposed by the data
- Breakout discussion
  - What other types of conditions might we want to impose on the learned grammar?
  - What are some sources of evidence that human learners care about more than just likelihood?

Infinite weights

/pat/	*C]	H	exp(H)	Prob
Weight:	-5			
pat	1	-5	0.006738	0.006693
pa	0	0	1.000000	0.993307

/pat/	*C]	H	exp(H)	Prob
Weight:	-8			
pat	1	-8	0.000335	0.000335
pa	0	0	1.000000	0.999665

/pat/	*C]	H	exp(H)	Prob
Weight:	-10			
pat	1	-10	0.000045	0.000045
pa	0	0	1.000000	0.999955

Infinite weights (*cont.*)

- Regularization: $L(w) = -logP(D|w) + \frac{(w-\mu)^2}{2\sigma^2}$
    - Likelihood term: $logP(D|w)$
    - Regularization term (prior): $\frac{(w-\mu)^2}{2\sigma^2}$
- Regularization penalizes weights that deviate from some prespecified value μ
    - Common to assume that fitted values should fall within some distribution around μ
    - Prespecified distribution: mean μ, standard deviation σ
    - “L2 regularization”: probability of w drops off with square of distance from μ
- Balances fit vs. magnitude of weights

Seeing regularization in action

- Google Colab interlude: simple paC.txt language
- One effect of regularization:
  - Stabilization to keep weights from going to infinity

Restrictiveness prior

Restrictiveness

- Faithfulness constraints favor contrast (allow more surface structures)
- Markedness constraints favor neutralization (fewer surface structures)
- $\mathcal{M} \gg \mathcal{F}$: reduces number of surface forms
    - Concentrates probability mass on ($=$increases likelihood of) attested structures
- Objective: w($\mathcal{M}$) $>$ w($\mathcal{F}$)
    - Frequently: $\mu(\mathcal{M})$ high, $\mu(\mathcal{F})$ low
    - Better: maximize w($\mathcal{M}$) $-$ w($\mathcal{F}$)

Substantive priors

Wilson (2006) Substantive bias

The empirical problem: velar palatalization

{k, g} → {tʃ, dʒ} / ___ {i, e}

- Palatalization and vowel frontness
  - CV coarticulation means that velars are produced farther forward before front vowels
  - The fronter the vowel, the fronter the stop
  - Perceptual consequence: *ki* and *tʃi* are quite confusable

Wilson (2006) Substantive bias (*cont.*)

  - Mirrored typologically: palatalization /___ a → /___ e → palatalization/___ i, but not vice versa
- Palatalization and voicing
  - Frication and aspiration during release of /k/ is very similar (or identical) to palatal fricative
    - kʰi ≈ kçi/cçi → tʃi
  - Short VOT of voiced /g/ creates less of a ʒ-like period
  - Also reflected typologically: palatalization of /g/ implies palatalization of /k/

A universalist solution

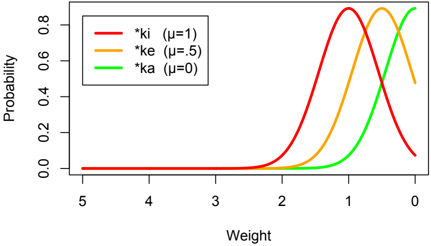
- A priori fixed ranking: *ki $\gg$ *ke $\gg$ *ka
- Typology:

ka, ke, ki	Ident(Strid) $\gg$ *ki, *ke, *ka
ka, ke, *ki	*ki $\gg$ Ident(Strid) $\gg$ *ke, *ka
ka, *ke, *ki	*ki, *ke $\gg$ Ident(Strid) $\gg$ *ka
*ka, *ke, *ki	*ki, *ke, *ka $\gg$ Ident(Strid)

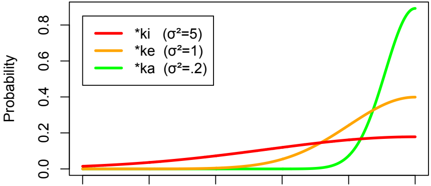
- Or, another option to ponder
  - Markedness: *k
  - Faithfulness: $\mathcal{F}$(k∼tʃ/a) $\gg$ $\mathcal{F}$(k∼tʃ/e) $\gg$ $\mathcal{F}$(k∼tʃ/i)

Prior bias: P(*ki ≫ *ke) > P(*ke ≫ *ki)

- One possibility: different target weights (not explored)



- Another possibility: different strictness (Wilson’s approach)



An experiment testing these biases

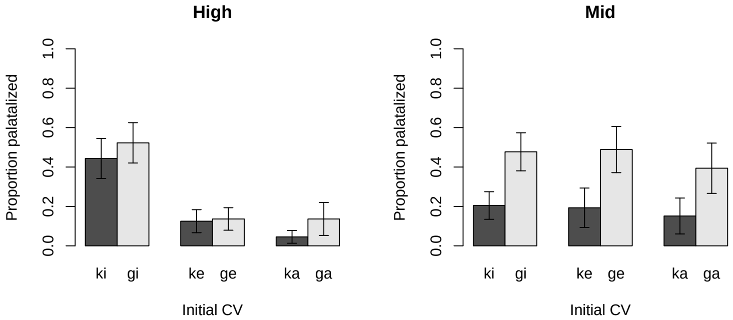
- Taught subjects artificial languages with palatalization

I say…	You say…
pimə	pimə
baʃə	baʃə
kifə	tʃifə
galə	galə

An experiment testing these biases (*cont.*)

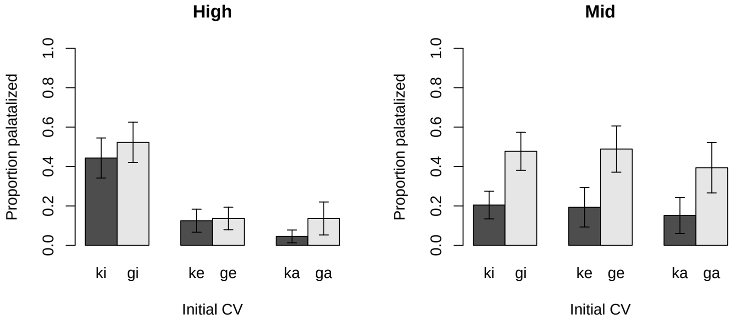
- Training: subjects heard both versions, and repeated second (32 trials)
    - High group: heard palatalized /ki/, /gi/, but no /ke/, /ge/ examples
    - Mid group: heard palatalized /ke/, /ge/, but no /ki/, /gi/ examples
    - Both groups heard /ka/, /ga/ (no palatalization)
- Testing: presented with “I say” versions, had to produce appropriate responses
    - Tested on low, mid and high vowels (80 items)

Results



- Overall bias *not to apply* alternation
  - Even subjects trained on [kifə] ∼ [tʃifə] frequently replied [kifə]
  - Higher rate for /gV/: practice examples had only /g/

Results



- Trained on /ki/ → [tʃi] (High condition): little generalization to mid, low contexts
- Trained on /ke/ → [tʃe] (Mid condition): generalize completely to high, also strongly to low

Wilson (2006) Substantive bias

Modeling goals:

- Capture generalization: training on palatalization in one context can extend to other contexts
- Capture asymmetry:
  - Training on ke→tʃe leads to generalization
  - Training on ki→tʃi does not

Constraints to be assumed

- Markedness constraints motivating palatalization

*ki	(= *k$\begin{bmatrix}-\text{back}\\+\text{high}\end{bmatrix}$)	*gi	(= *g$\begin{bmatrix}-\text{back}\\+\text{high}\end{bmatrix}$)
*ke	(= *k$\begin{bmatrix}-\text{back}\\-\text{high}\\-\text{low}\end{bmatrix}$)	*ge	(= *g$\begin{bmatrix}-\text{back}\\-\text{high}\\-\text{low}\end{bmatrix}$)
*k{i,e}	(= *k$\begin{bmatrix}-\text{back}\\-\text{low}\end{bmatrix}$)	*g{i,e}	(= *g$\begin{bmatrix}-\text{back}\\-\text{low}\end{bmatrix}$)
*ka	(= *k[+low])	*ga	(= *g[+low])
*k{e,a}	(= *k[−high])	*g{e,a}	(= *g[−high])
*kV	(= *k[+syl])	*gV	(= *g[+syl])

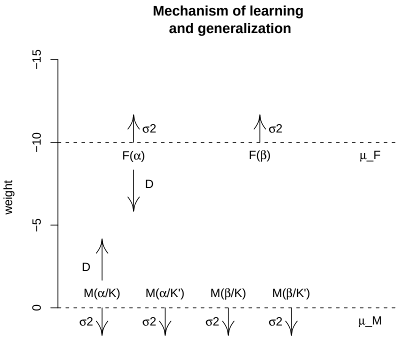
- Faithfulness constraints: separate for /k/, /g/
    - $\mathcal{F}$(k), $\mathcal{F}$(g)

Model training

- Same input data as training exposure in experiment
  - 32 items; 8 show palatalization in relevant context
- Initial weighting for modeling adult English speakers
  - $\mathcal{F} \gg \mathcal{M}$ (English does not have velar palatalization)
  - w($\mathcal{M}$) $= 0$, w($\mathcal{F}$) $= -10$
- Learning from data
  - Promote relevant $\mathcal{M}$ constraints and demote $\mathcal{F}$
  - Promote $=$ make weight stronger; demote $=$ move closer to 0
- Regularization: stay as close to initial state as possible
  - Target μ value for $\mathcal{M} = 0$
  - Target μ value for $\mathcal{F} = -10$
  - Strength of “pull” towards target values is a function of σ^2; set to 10^2 for now, and adjust in a more sophisticated way later

The model set-up, summarized

Wilson (2006), Fig. 1:



- $\mu =$ target values
- D $=$ direction of pull of data
- $\sigma^2 =$ direction of pull of regularization (towards μ)

Implementing bias

- Model so far starts with weights at initial states ($\mathcal{M} = 0$, $\mathcal{F} = 10$) but then allows them to move as far as they like in response to the data
  - Model objective: maximize log likelihood$_{\vec{w}}(\vec{x}, \vec{y})$
  - Adjust weights so P(winner) $>$ P(loser)
- What we want is to allow some constraints to move more freely than others
  - Recall regularization: term to minimize distance between weights from targets
  - Maximize: log likelihood$_{\vec{w}}(\vec{x}, \vec{y}) - \sum_{i=1}^{k} \frac{(w_i - \mu_i)^2}{2\sigma_i^2}$
    - μ determines target value for weights
    - σ^2 determines strictness

Building in bias

- What we want to happen
  - Palatalization of /ke/→[tʃe] should favor increasing weight of *k[−low], as much or more than weight of *ke
- Wilson’s proposal:
  - Constraints should be more freely adjustable if the changes that they motivate/allow are perceptually less salient
  - E.g., *ki should be free to be highly ranked, since it motivates palatalization of /ki/ → tʃi (perceptually inconsequential)
  - Conversely, *ka should have a lower weight, since /ka/ → [tʃa] is perceptually more salient

Implementation

- Set the σ^2 for each Markedness constraint according to the most radical change that constraint could compel
  - *ki can compel /ki/ $\rightarrow$ [tʃi] (perceptually very similar)
  - *ka can compel /ka/ $\rightarrow$ [tʃa] (perceptually less similar)
  - *kV can compel both (including perceptually less similar /ka/ $\rightarrow$ [tʃa])
  - See **?** for details
- Effect: less penalty for promoting *ki than *kV
- This is definitely not the only way one could imagine biasing the grammar!

Concretely

- σ^2 values: Wilson, Table 4

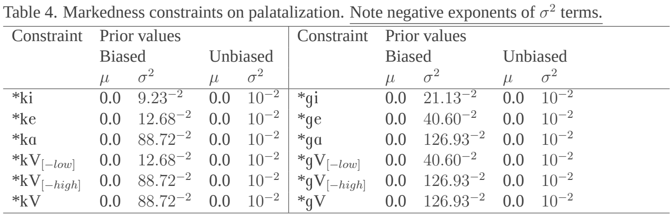


Table 4. Markedness constraints on palatalization. Note negative exponents of σ^2 terms.

Constraint	Prior values				Constraint	Prior values			
	Biased		Unbiased			Biased		Unbiased	
	μ	σ^2	μ	σ^2		μ	σ^2	μ	σ^2
*ki	0.0	9.23^{-2}	0.0	10^{-2}	*ɡi	0.0	21.13^{-2}	0.0	10^{-2}
*ke	0.0	12.68^{-2}	0.0	10^{-2}	*ɡe	0.0	40.60^{-2}	0.0	10^{-2}
*kɑ	0.0	88.72^{-2}	0.0	10^{-2}	*ɡɑ	0.0	126.93^{-2}	0.0	10^{-2}
*kV$_{[-low]}$	0.0	12.68^{-2}	0.0	10^{-2}	*ɡV$_{[-low]}$	0.0	40.60^{-2}	0.0	10^{-2}
*kV$_{[-high]}$	0.0	88.72^{-2}	0.0	10^{-2}	*ɡV$_{[-high]}$	0.0	126.93^{-2}	0.0	10^{-2}
*kV	0.0	88.72^{-2}	0.0	10^{-2}	*ɡV	0.0	126.93^{-2}	0.0	10^{-2}

- Smaller σ values for more dissimilar changes → forced to stay closer to 0

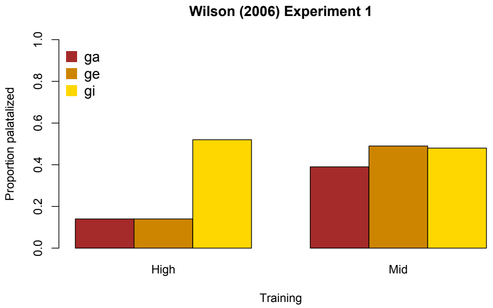
Perceptron learning with regularization

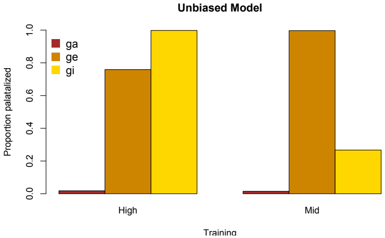
- Add small nudge (‘loss’) to update rule, using σ^2 values
  - Amount proportionate to $\frac{\text{Distance from } \mu}{\sigma^2}$
- Each times weights are adjusted, all constraints move towards their μ, by an amount determined by their σ^2
- Amount of loss is a parameter of such models

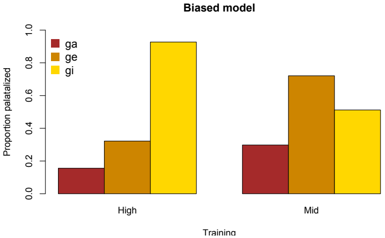
A simulation

- Used SGA (with regularization), with initial weights $=$ target weights
  - Faithfulness $= -10$, Markedness $= 0$
  - NB: not a very good default assumption for learning, for reasons we'll discuss more next time!
- Biased learning
  - σ^2 values as given by Wilson
  - Overall difference between /k/ and /g/, different amounts of data required
  - Assumed 3x as many /gi/ or /ge/ as /ki/ or /ke/
  - We'll just focus here on voiced results

Results


Wilson (2006) Experiment 1
Proportion palatalized
0.0
0.2
0.4
0.6
0.8
1.0
ga
ge
gi
High
Mid
Training


Unbiased Model
Proportion palatalized
0.0
0.2
0.4
0.6
0.8
1.0
ga
ge
gi
High
Mid
Training


Biased model
Proportion palatalized
0.0
0.2
0.4
0.6
0.8
1.0
ga
ge
gi
High
Mid
Training

Summary of substantive priors

- Regularization is a powerful tool for biasing the model to prefer some constraint weightings over others
    - Perceptual similarity (contrast constraints, faithfulness)
    - Articulatory difficulty
    - Generality
- Tension: fine-grained fitting vs. coarser fitting with fewer constraints
    - Modeling of nonce words suggested fairly close fitting of bigrams
    - Perhaps amount of data available to learners for small differences among similar clusters is enough to overcome bias?

From weights to rankings

Advantages of being probabilistic

- Easily stated objective
- Linear models are easy to fit with general purpose learning techniques (e.g., gradient descent), even though the hypothesis space is infinite
- Easy to compare grammars
  - Direct parallel: poisson regression of corpora, experimental data
  - Learning the constraints: we'll come back to Hayes and Wilson (2008)
- Can model gradient patterns

A cost of being probabilistic

- Greater expressive power
- Infinitely many distinct grammars
- Though definitely *not* any possible pattern

Two different decision rules

- Harmonic Grammar (HG)
  - Assign 100% probability to the candidate with the least severe weighted violations

/bnɪk/	*[bn	Contig	Max	Dep		
Weight:	8	4	2	1	Sum	Prob
☞a. bənɪk				*	1	1
b. nɪk			*		2	0
c. bɪk		*	*		6	0
d. bnɪk	*				8	0

- Optimality Theory (OT)
  - Assign 100% probability to the candidate which best satisfies higher ranked constraints

/bnɪk/	*[bn	Contig	Max	Dep	
☞a. bənɪk				*	1
b. nɪk			*!		0
c. bɪk		*!	*		0
d. bnɪk	*!				0

More restrictive theories…

- “Classical” HG and OT both restricted to languages with categorical (100%/0%) distributions
    - But: both can be augmented to allow *some* gradient distributions
- OT further restricted by *strict domination*

Strict domination

- HG: weights of lower constraints matter (ganging up)

/UR/	CON1	CON2	
w	3	1	Sum
☞ a. cand1		**	2
b. cand2	*		3

/UR/	CON1	CON2	
w	3	2	Sum
a. cand1		**	4
☞ b. cand2	*		3

- OT: number of lower violations is irrelevant (strict domination)

/UR/	CON1	CON2
☞ a. cand1		**
b. cand2	*	

Implications for learning

- More restrictive $\Rightarrow$ smaller hypothesis space
- However, this doesn't necessarily make learning easier
- Learning $=$ satisfying an objective function
  - Maximize the likelihood of the attested outputs
  - Maximize the likelihood of the attested outputs while minimizing deviations from priors
  - Ensure that every constraint that is violated more times by an attested output than by a competing output is ranked below at least one constraint that is violated more by the competing output

Two lessons in the upcoming sessions

- Additional conditions imposed by Optimality Theory make learning more complex
  - Especially: integrating priors like Markedness $\gg$ Faithfulness
  - Smaller typology $\neq$ simpler learning
- For all approaches, priors can explain why certain grammars are unlikely, even if they are allowed by the framework
  - Learnability-oriented approach to typology

References

Berger, Adam L.; Stephen A. Della Pietra; and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. Computational Linguistics 22(1), 39–71. URL https://www.aclweb.org/anthology/J96-1002.

Coetzee, Andries W. 2004. What it means to be a loser: Non-optimal candidates in Optimality Theory. Ph.D. thesis, University of Massachusetts, Amherst.

Everett, Daniel and Iris Berent. 1997. The comparative optimality of Hebrew roots: An experimental approach to violable identity constraints. ROA 235.

References (*cont.*)

GOLDWATER, SHARON and MARK JOHNSON. 2003. Learning OT constraint rankings using a maximum entropy model. In Proceedings of the Workshop on Variation within Optimality Theory, Stockholm University, ed. by Jennifer Spenader; Anders Eriksson; and Östen Dahl, 111–120. Stockholm: Stockholm University.

HAYES, BRUCE and COLIN WILSON. 2008. A maximum entropy model of phonotactics and phonotactic learning. Linguistic Inquiry 39, 379–440.