Simple induction of (deterministic) probabilistic finite-state automata for phonotactics by stochastic gradient descent

Huteng Dai[1] and Richard Futrell[2]

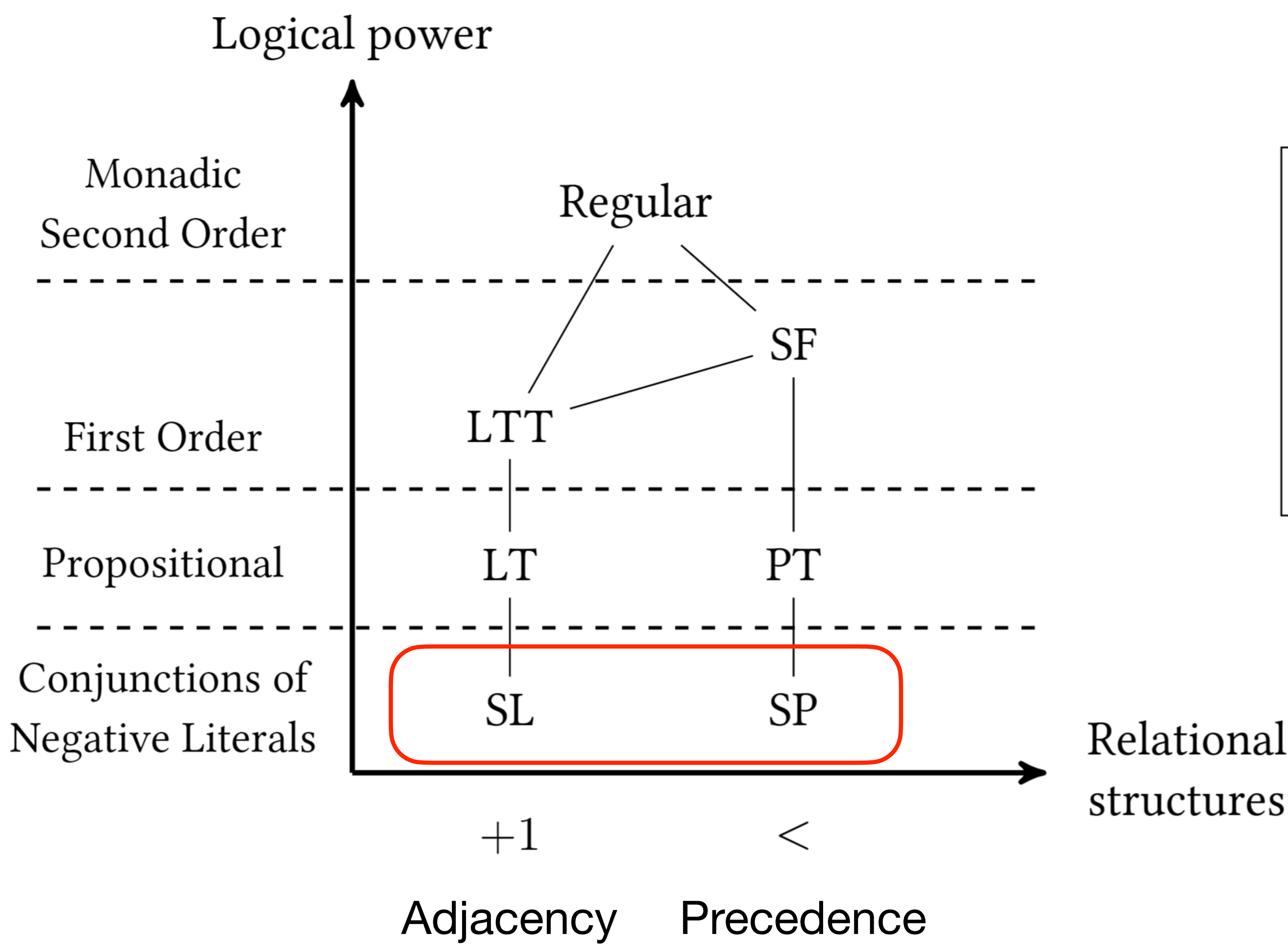
[1] Rutgers University, New Brunswick
[2] University of California, Irvine

Take-home messages

- A **differentiable** probabilistic framework for modeling and inducing phonotactics;

- This facilitates the application of machine learning methods and the comparison of subregular languages.

Subregular Hierarchy and Phonotactics



SL: Strictly Local
LT: Locally Testable
LTT: Locally-threshold Testable
SP: Strictly Piecewise
PT: Piecewise Testable
SF: Star Free

Rogers et al. (2013); Heinz (2018)

Strictly Local and Strictly Piecewise Languages

Given the alphabet {a, b, c}

2-SL: *ab

Legal: cac, aac, acbc …

Illegal: ***c*ab**, ***ab**bc …

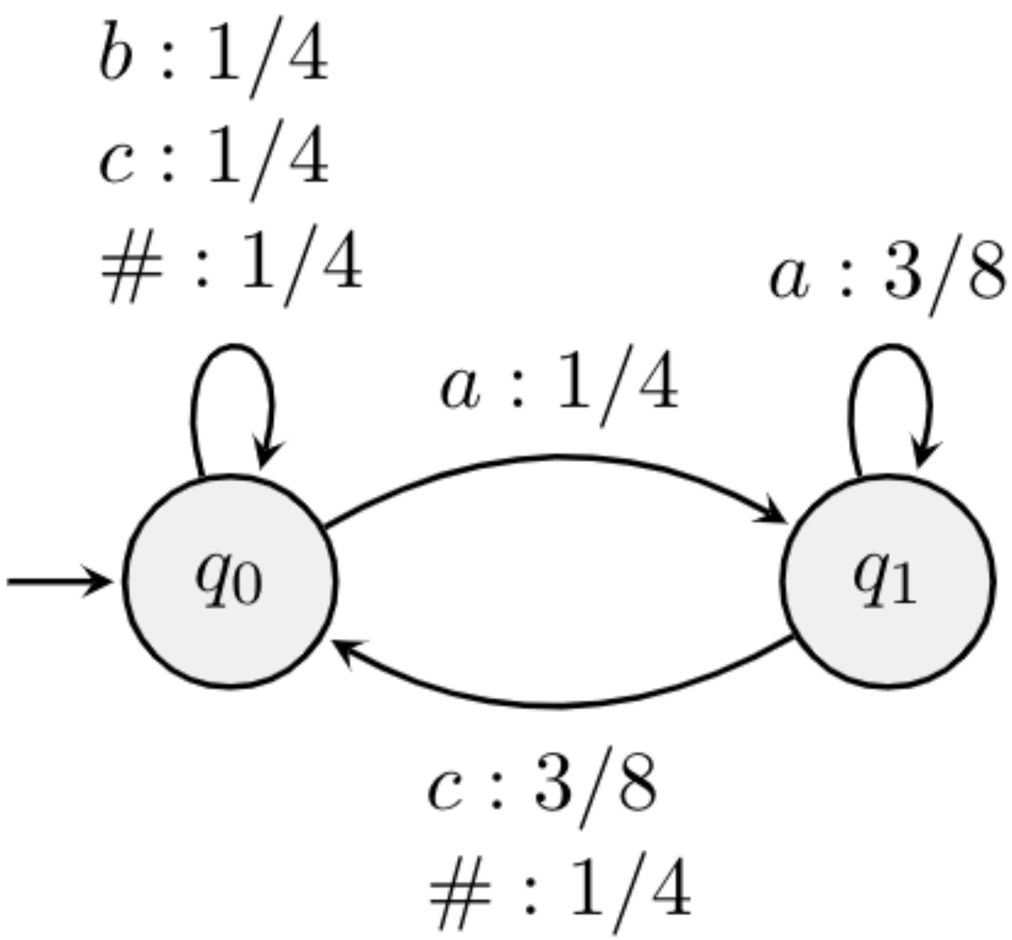
2-SP: *a…b

Legal: bbaaa, baccc …

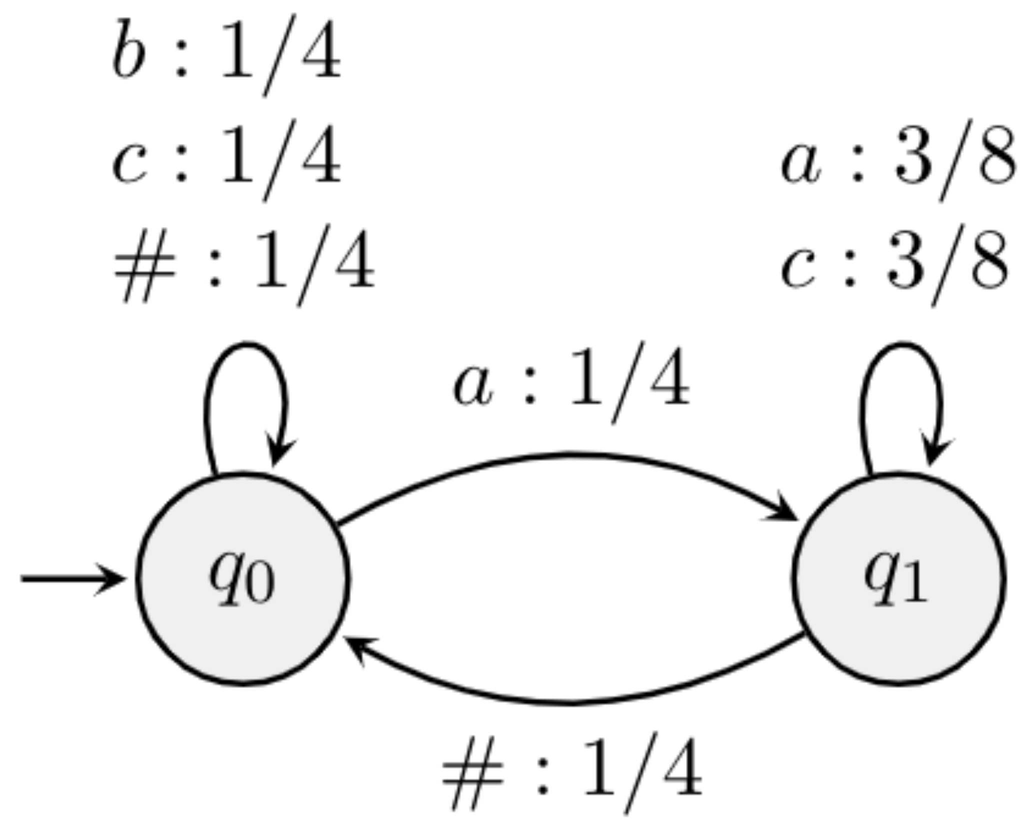
Illegal: ***a*c*b**, ***b*a*ccc*b** …

Grammar and automaton

2-SL: *ab



2-SP: *a…b



Note: 2-SL + 2-SP automata is the product of both

Subregular Program and probabilistic models

A growing interest in linking the study of subregular hierarchy and probabilistic models;

Goals and benefits:

- Handling noisy corpus data;
- Filling the gap in Formal Language Theory;
- Understanding the nature of phonological acquisition.

Wilson & Gallagher (2018); Shibata & Heinz (2019); Dai (2021)

Challenges and contributions

A **unified** framework for studying the induction of subregular phonotactic models:

- Incorporating unrestricted PFAs and restricted Deterministic PFAs of SP, SL, and SP + SL (for now);
- Induction from small datasets in natural languages;
- Showing which classes can learn to predict certain phonotactic patterns most effectively.

The framework

Initializing (D)PFA matrices

Computing word likelihood (loss function)

Probabilistic Finite-State Automata (PFAs)

PFAs are parameterized by **E** and $\mathbf{T}_x$ matrices

Emission probability:
Conditional on state *q*, the probabilistic distribution on symbols;

Transition probability:
Conditional on state *q* and symbol *x*, the probabilistic distribution on next states.

With $\mathbf{T}_x$, we no longer need to specify state transitions for unrestricted PFAs in learning.

Encoding subregular constraints

We can restrict models to Deterministic PFAs by hard-coding the **T** matrices and inducing only **E**.

e.g. a 2-SL transition matrix:

$$\mathbf{T}_x = \begin{array}{cc} \begin{array}{ccc} \cdots q_{\neq x} \cdots & q_x & \cdots q_{\neq x} \cdots \end{array} & \\ \begin{bmatrix} \vdots & \vdots & \vdots \\ \ldots 0 \ldots & 1 & \ldots 0 \ldots \\ \vdots & \vdots & \vdots \end{bmatrix} & q_x \end{array}$$

state indices

Encoding subregular constraints

Implement 2-SP as the product of factor machines *A*$^{(x)}$, one per segment *x.*

e.g. transition matrices for machine *A*$^{(x)}$ in 2-SP

$$\mathbf{T}_x^{(x)} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}, \mathbf{T}_{y \neq x}^{(x)} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The framework

Initializing (D)PFA matrices

↓

Computing word likelihood (loss function)

↓

Inducing the model by gradient descent

Induction by Gradient Descent: Softmax

Update underlying weight matrices $\tilde{\mathbf{E}}$ and $\tilde{\mathbf{T}}$, which are transformed into probabilistic matrices **E** and **T** by Softmax:

$$E_{ij} = \frac{\exp \tilde{E}_{ij}}{\sum_k \exp \tilde{E}_{ik}}, \; T_{ij} = \frac{\exp \tilde{T}_{ij}}{\sum_k \exp \tilde{T}_{ik}}$$

Induction by Gradient Descent: Objective

Find matrices **E** and **T** to minimize the training objective:

$$J(\tilde{\mathbf{E}}, \tilde{\mathbf{T}}) = \left\langle -\log p(x \,|\, \mathbf{E}, \mathbf{T}) \right\rangle_{x \sim X} + N(\mathbf{E}, \mathbf{T})$$

Average negative log likelihood of data

Regularization

Regularization against nondeterminism

In Deterministic PFA (DPFA), state transition distribution is deterministic.

We penalize nondeterministic automata by using **average nondeterminism** as a regularization term:

$$N(\mathbf{E}, \mathbf{T}) = \mathrm{H}[q' \mid x, q]$$

Conditional entropy of next state given previous state and current symbol.

= 0 for perfectly deterministic PFAs.

The framework

Initializing (D)PFA matrices

↓

Computing word likelihood (loss function)

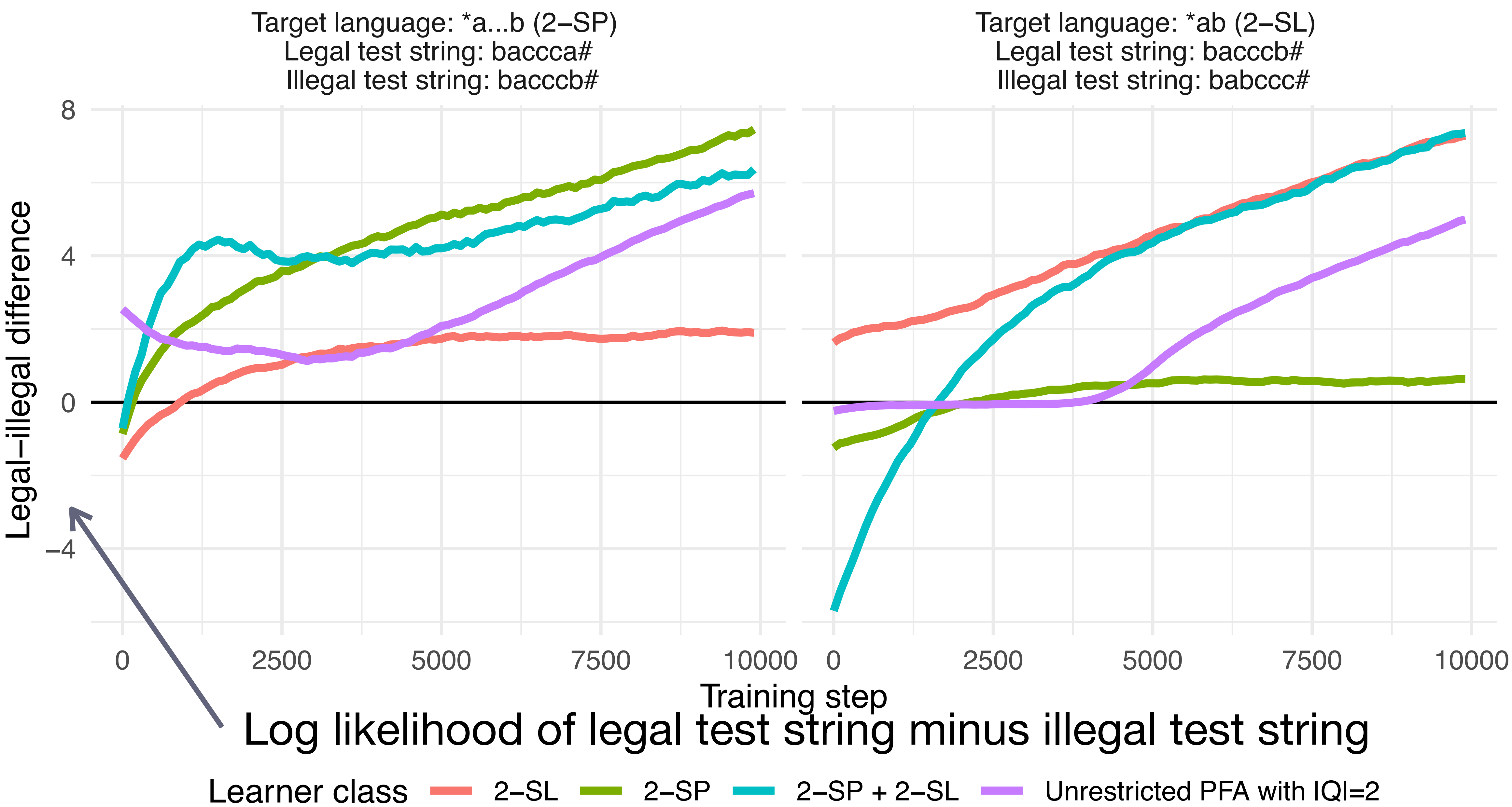
↓

Inducing the model by gradient descent

↓

Evaluation

Evaluation: Toy languages



Evaluation: Navajo and Quechua

Navajo: The co-occurrence of alveolar and palatal strident is illegal;

s o s	*s o ʃ
s o r o s	*s o r o ʃ

Quechua: no stop can be followed by an ejective or aspirated stop;

t' o r o k	*t o r o k'
t^h o r o k	*t o r o k^h
t o r o k	*t^h o r o k

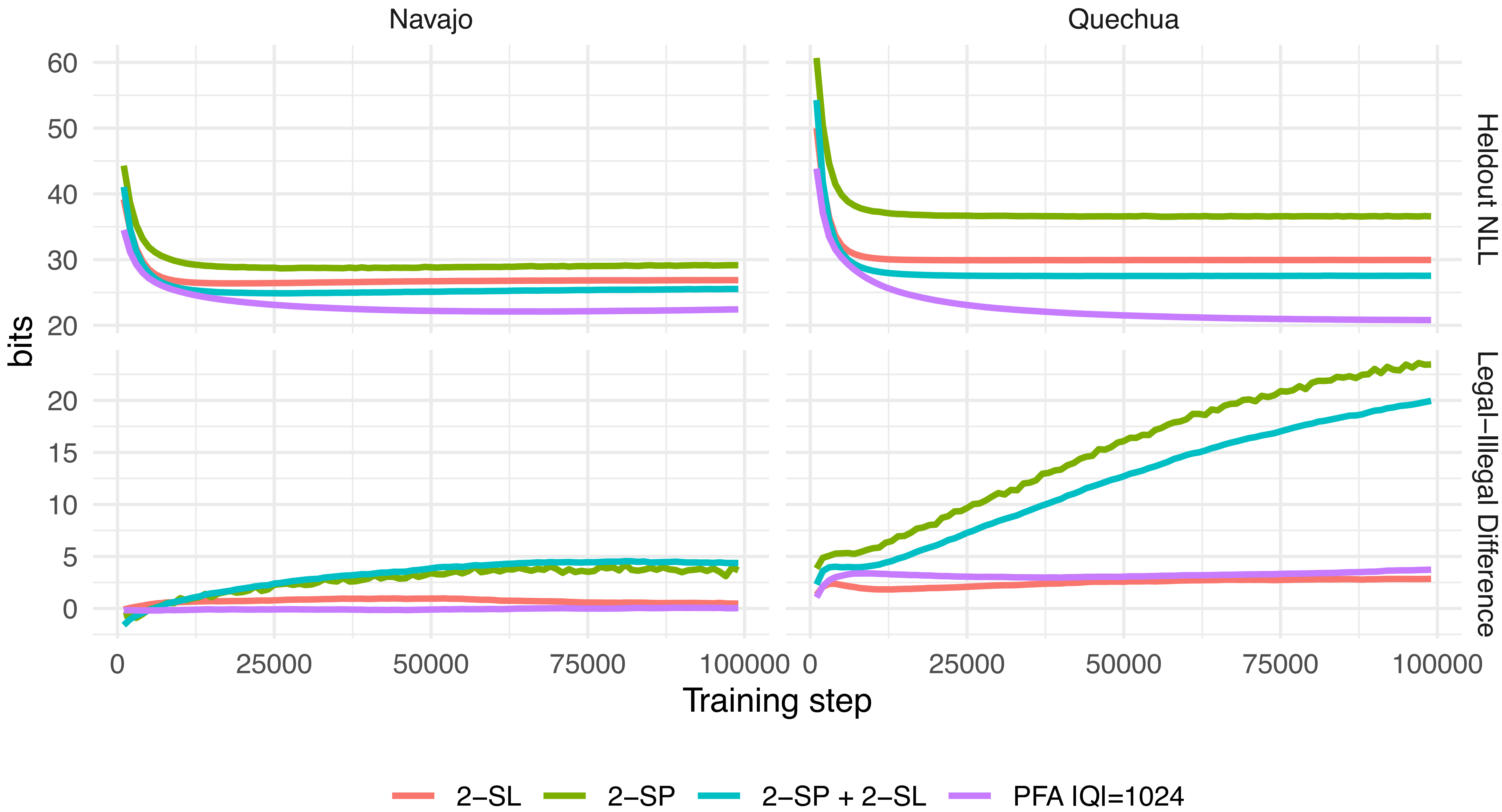
Datasets (Gouskova & Gallagher, 2020)

- Learning data
  - Phonological words;
  - Navajo: 6279; Quechua: 10804;
  - 80% to training set, 20% to held-out set;
- Testing data
  - Nonce words, labelled as legal vs illegal based on nonlocal constraints;
  - Navajo: 5000; Quechua: 24352.

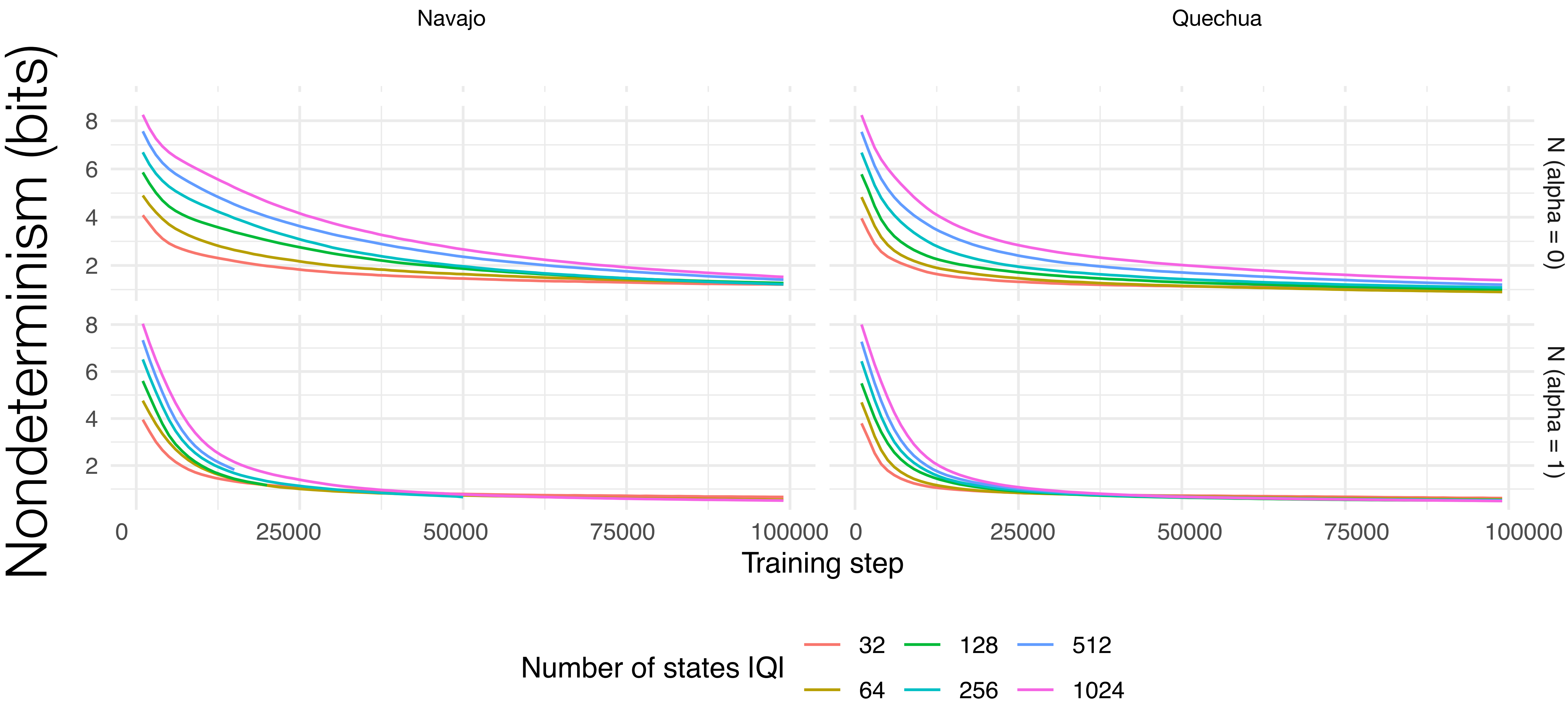
Running the program

```
python new_pfa.py --model_class sp_sl --lang navajo --activation softmax --print_every 1000 --lr 0.01
Training set size = 5023
Dev set size = 1256
Segment inventory size =  47
Model class = sp_sl
0
nondeterminism_penalty,memory_mi_penalty,init_temperature,activation,batch_size,lr,model_class,epoch,train_nll,dev_nll,
0.0,0.0,1,softmax,5,0.01,sp_sl,0,41.609500885009766,41.73338317871094,0.0,nan,39.65355246848905,39.63214659916812,
1000
0.0,0.0,1,softmax,5,0.01,sp_sl,1000,17.554428100585938,18.200313568115234,0.0,nan,34.43270141530961,34.7243515103281,
2000
0.0,0.0,1,softmax,5,0.01,sp_sl,2000,16.844873428344727,17.5587215423584,0.0,nan,35.587077134740774,36.448850729093024,
3000
0.0,0.0,1,softmax,5,0.01,sp_sl,3000,16.608205795288086,17.3580379486084,0.0,nan,36.85474407665988,37.88865142421087,
4000
0.0,0.0,1,softmax,5,0.01,sp_sl,4000,16.509923934936523,17.308490753173828,0.0,nan,38.25863582154379,39.46197283052953,
5000
0.0,0.0,1,softmax,5,0.01,sp_sl,5000,16.455812454223633,17.3162784576416,0.0,nan,39.804874700366454,41.104307318710475,
```

Primary results



The emergence of determinism



Conclusions

It's possible to compare the induction of various (sub)regular languages in a unified framework:

- Inducing unrestricted Probabilistic Finite-state Automata (PFAs) produces the best fit to naturalistic held-out forms;
- However, a restricted subregular model (Strictly Piecewise) is superior in capturing nonlocal constraints as evidenced in nonce data.

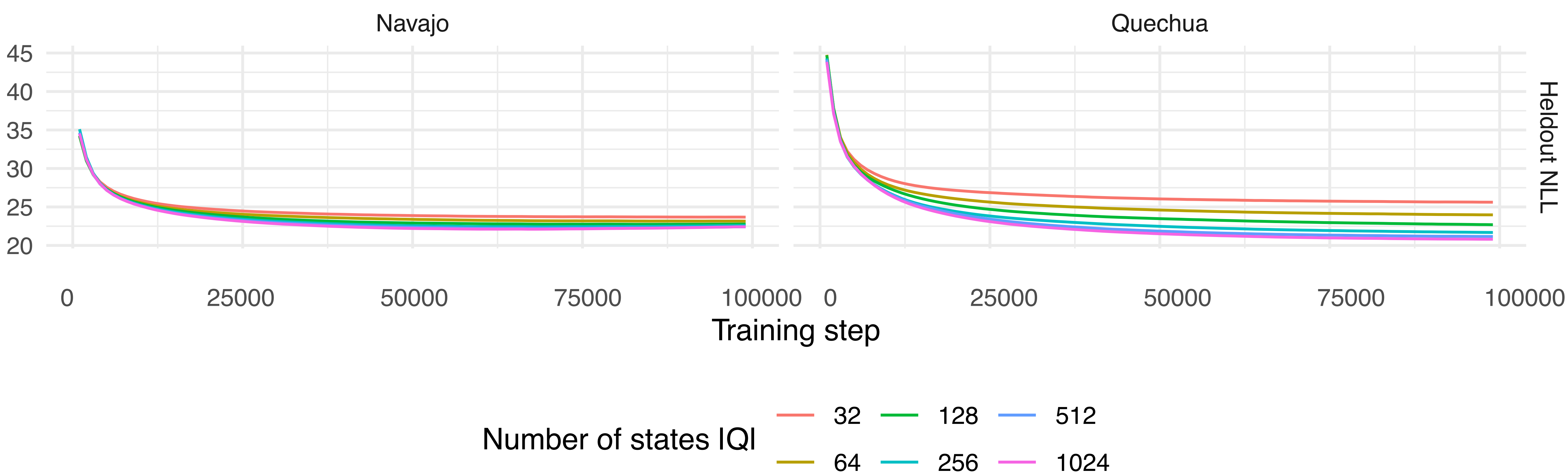
Acknowledgement

- Thanks to Adam Jardine, Jeff Heinz, and three anonymous reviewers for helpful comments.

- This work was supported by an NVIDIA GPU Grant to RF.

- All our code is available at http://github.com/hutengdai/pfa-learner

Unrestricted PFA Induction Results



Probabilistic Finite-State Automata (PFAs)

PFAs are parameterized by matrices **E** and $\mathbf{T}_x$.

Emission probability:
Conditional on state *q*, the probabilistic distribution on symbols

Transition probability:
Conditional on state *q* and symbol *x*, the probabilistic distribution on next states

$$p(\,\cdot\,|\,\mathbf{q}) = \mathbf{q}^\top \mathbf{E}$$

$$p(\,\cdot\,|\,\mathbf{q}, x) = \mathbf{q}^\top \mathbf{T}_x$$

State distribution: e.g.
[q_0: 0.45, q_1: 0.50, q_2: 0.05]

With $\mathbf{T}_x$, we no longer need to specify state transitions for PFAs in learning.