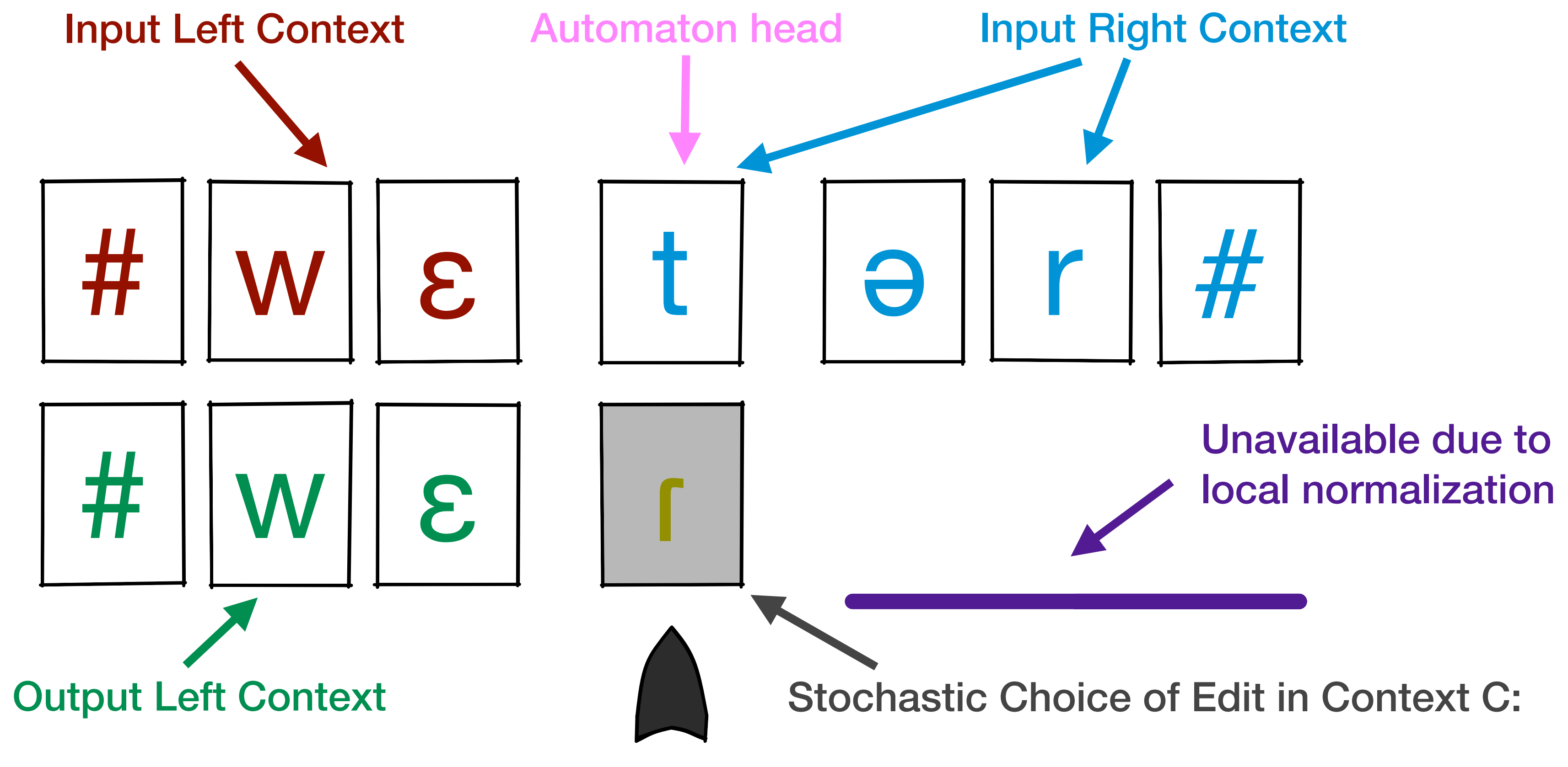
# Stochastic Contextual Edit Distance and Probabilistic FSTs

Ryan Cotterell, Nanyun Peng, Jason Eisner

## Example from English Phonology

Consider the productive case of intervocalic alveolar flapping in American English e.g., compare the pronunciation of *wet* and *wetter*. We should map the underlying form /**wɛtər**/ to its surface form [**wɛɾər**]. This is predicted by a left-to-right, context sensitive editing process:



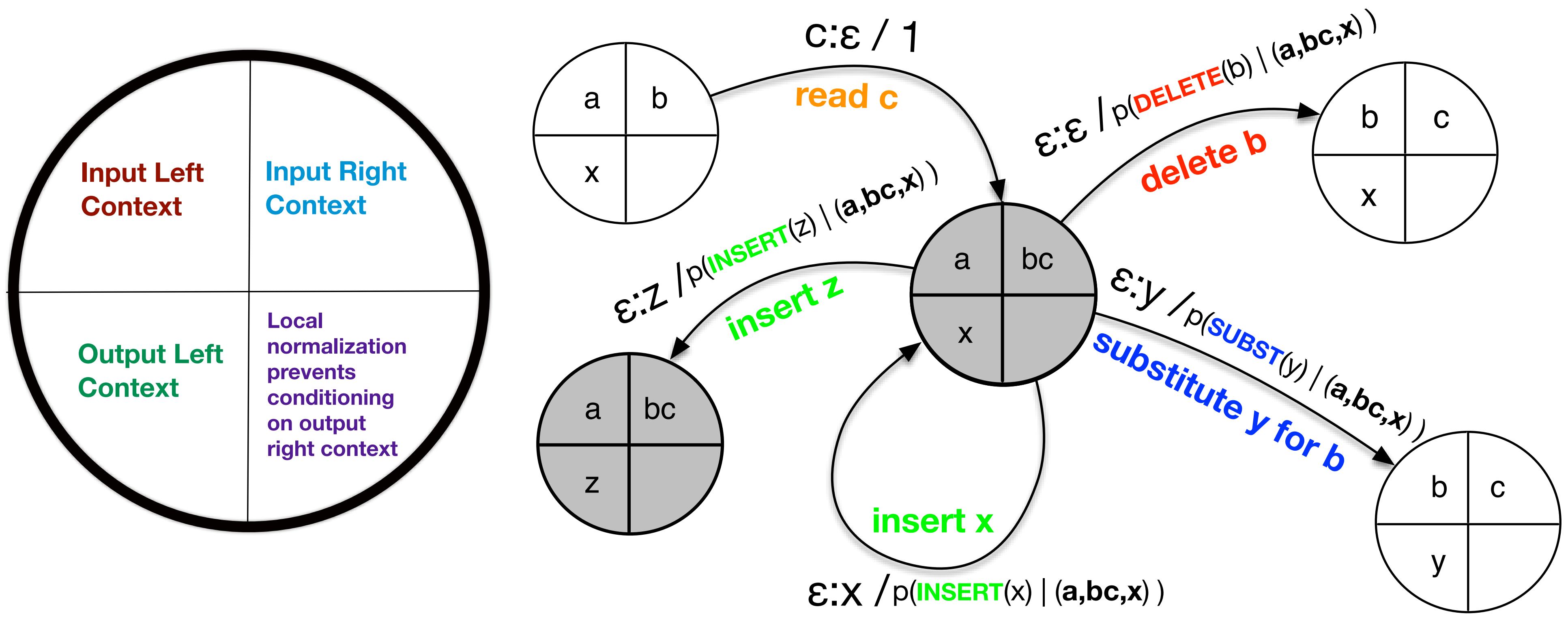
The distribution over possible edits takes the form of a conditional log-linear model:

$$p(e \mid C) \overset{\text{def}}{=\!=} \frac{1}{Z_c} \exp\left(\theta \cdot \vec{f}(C, e)\right)$$

$$p(\textbf{COPY}[\text{t}] \mid \text{C}) = \ldots$$
$$p(\textbf{INS}[\text{z}] \mid \text{C}) = \ldots$$
$$p(\textbf{SUB}[\text{t}, \text{ɾ}] \mid \text{C}) = \ldots$$
$$p(\textbf{DEL}[\text{t}] \mid \text{C}) = \ldots$$
$$\ldots$$

## The Contextual Edit Transducer

- We define a conditional probability distribution of an *edit* given a *context* using a log-linear model.
- An edit is one of four actions: COPY, SUBSTITUTE, DELETE or INSERT.
- The probability of a sequence of edits is a product where each edit's probability is conditioned on the context produced by the previous edits.
  - A context consists of three context windows: input left, input right and output left.
  - Right output context is unavailable in PFSTs, so the model is left/right asymmetric.
- For $x, y \in \Sigma^*$, let $p(y \mid x)$ be the total probability of all edit sequences that map $x$ into $y$. Note that $\sum_y p(y \mid x) = 1, \forall x$.
- We construct a single probabilistic finite-state transducer to compute $p(y \mid x)$.



## Training

Given $(x_k, y_k)$ with unobserved alignments (edit sequences), EM will locally maximize $\sum_k p(y_k \mid x_k)$. The E-step sums over all $x_k$-to-$y_k$ alignment paths in the transducer (forward-backward algorithm). The M-step uses L-BFGS. The gradient takes the following well-known form:

$$\sum_{C,e} c(C,e) \left[ \vec{f}(C,e) - \sum_{e'} p_\theta(e' \mid C) \vec{f}(C,e') \right].$$

When L-BFGS is not run to convergence we recover a generalized EM algorithm, which is more efficient because we do not keep adjusting parameters based on out-of-date counts.

**Algorithm 1** Training a PFST $T_\theta$ by EM.

```
while not converged do
    reset all counts to 0                                  ▷ begin the "E step"
    for $k \leftarrow 1$ to $K$ do                         ▷ loop over training data
        $M = x_k \circ T_\theta \circ y_k$                 ▷ small acyclic WFST
        $\vec{\alpha}$ = FORWARD-ALGORITHM($M$)
        $\vec{\beta}$ = BACKWARD-ALGORITHM($M$)
        for arc $A \in M$, from state $q \to q'$ do
            if $A$ was derived from an arc in $T_\theta$
             representing edit $e$, from edit state $q_C$, then
                $c(C, e) \mathrel{+}= \alpha_q \cdot \text{prob}(A) \cdot \beta_{q'} / \beta_{q_I}$
    $\theta \leftarrow$ L-BFGS($\theta$, EVAL, max_iters=5)  ▷ the "M step"
function EVAL($\theta$)                                    ▷ objective function & its gradient
    $F \leftarrow 0; \nabla F \leftarrow 0$
    for context $C$ such that $(\exists e) c(C, e) > 0$ do
        count $\leftarrow 0$; expected $\leftarrow 0$; $Z_C \leftarrow 0$
        for possible edits $e$ in context $C$ do
            $F \mathrel{+}= c(C, e) \cdot (\theta \cdot \vec{f}(C, e))$
            $\nabla F \mathrel{+}= c(C, e) \cdot \vec{f}(C, e)$
            count $\mathrel{+}= c(C, e)$
            expected $\mathrel{+}= \exp(\theta \cdot \vec{f}(C, e)) \cdot \vec{f}(C, e)$
            $Z_C \mathrel{+}= \exp(\theta \cdot \vec{f}(C, e))$
        $F \mathrel{-}=$ count $\cdot \log Z_C$; $\nabla F \mathrel{-}=$ count $\cdot$ expected$/Z_C$
    return $(F, \nabla F)$
```

## Probabilistic vs. Weighted Finite-State Transducers

PFSTs are locally normalized models. WFSTs, which are globally normalized models, do not suffer from *label bias* and are likely to beat PFSTs as a linguistic model. The distinction is identical to that between a MEMM and a CRF. So why are we interested in PFSTs?

### Comparative Advantages

#### PFSTs

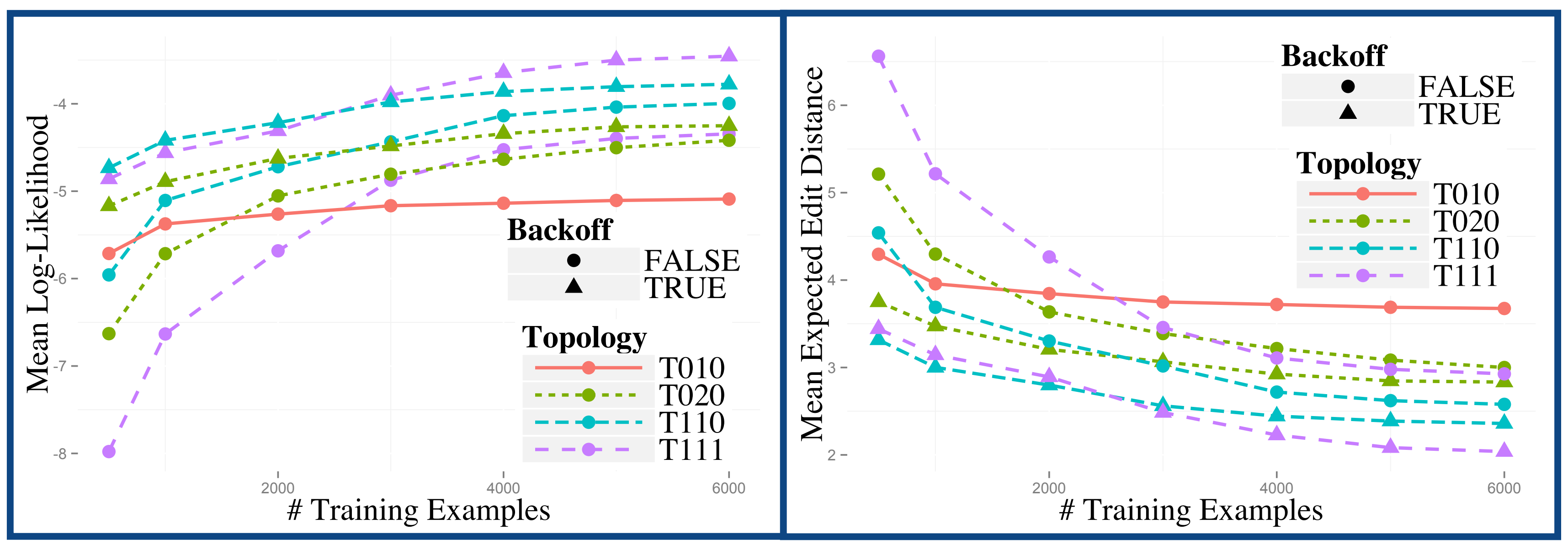
- PFSTs do not require the computation of a separate partition function $Z_x$ for every $x$. This makes them tractable when $x$ is uncertain e.g., in noisy channel models, channel cascades and Bayesian networks.
- PFSTs are more efficient to train under conditional likelihood. It is faster to compute the gradient, since we only have to raise the probabilities of arcs in $x_k \circ T \circ y_k$ relative to competing arcs in $x_k \circ T$.

#### WFSTs

- A WFST's advantage is that the probability of an edit can be indirectly affected by the weights of other edits at a distance.
- One could construct WFSTs where an edit's weight directly considers local right output context.
- WFSTs can also use a simpler topology while retaining determinism, since edits can be scored "in retrospect" after they have passed into the left context.

## Experiments

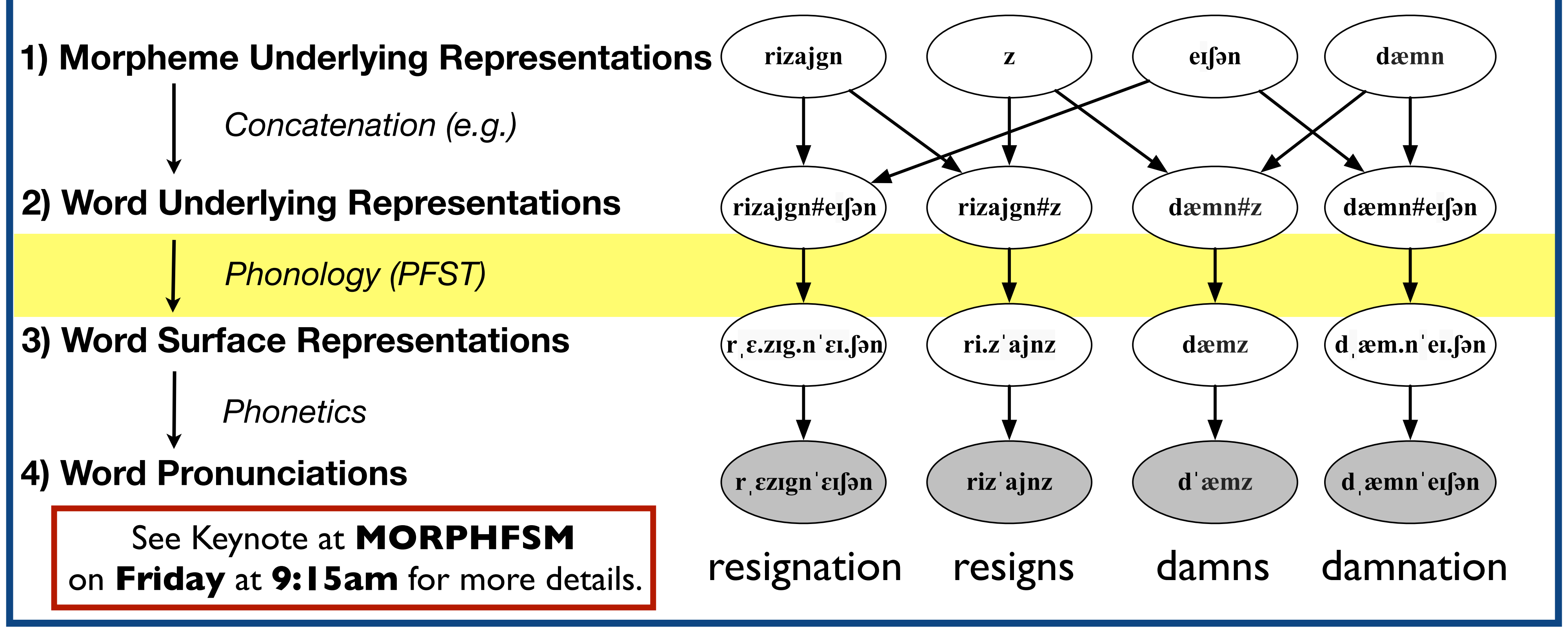
To demonstrate the utility of *contextual* edit transducers, we examine spelling errors in social media data. We report on test data how much probability mass lands on the true $y_k$. We also report how much mass lands "near" $y_k$, by measuring the expected edit distance of the predicted $y$ to the truth. The graphs show that more context improves the performance under both metrics on test data.



We use four different **topologies** (context configurations). Note that (0,1,0) is standard weighted edit distance. We also use **backoff** features that each context shares with other contexts and $L_2$ regularization.

## Future Work - Inferring Underlying Forms

We will use a PFST with features inspired by linguistic theory to model phonology within a Bayesian network. Observed pronunciations are often explained as arising from the "underlying forms" of morphemes. Linguists try to reconstruct these latent strings. Our technique involves loopy belief propagation in a generative (directed) graphical model whose variables are unknown strings and whose factors are finite-state machines with unknown weights.



See Keynote at **MORPHFSM**
on **Friday** at **9:15am** for more details.