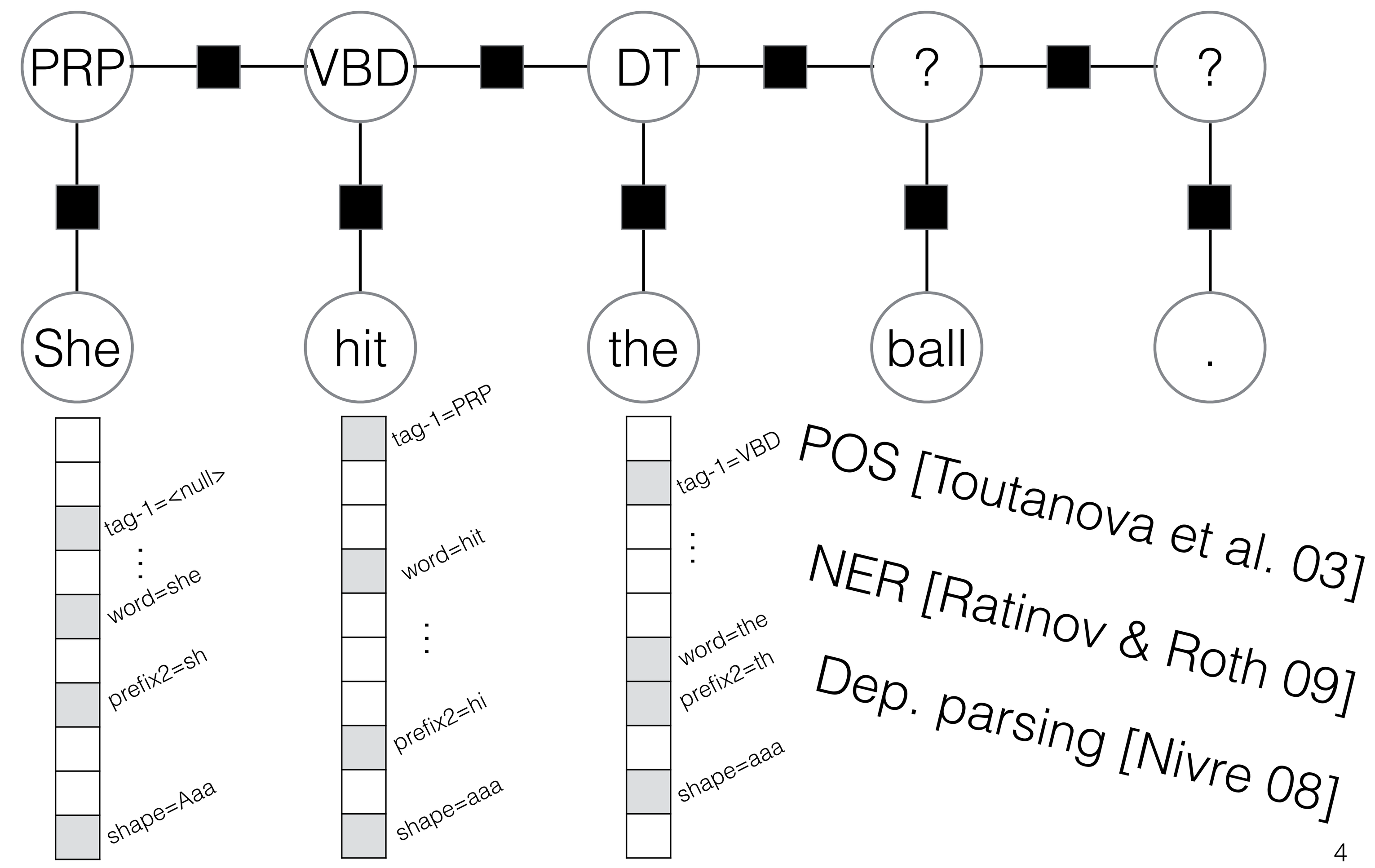
Learning Dynamic Feature Selection for Fast Sequential Prediction

Emma Strubell, Luke Vilnis, Kate Silverstein & Andrew McCallum
College of Information and Computer Sciences, University of Massachusetts Amherst

Problem



- Want fast and accurate NLP
- In many cases, fewer features needed for accurate prediction

Solution

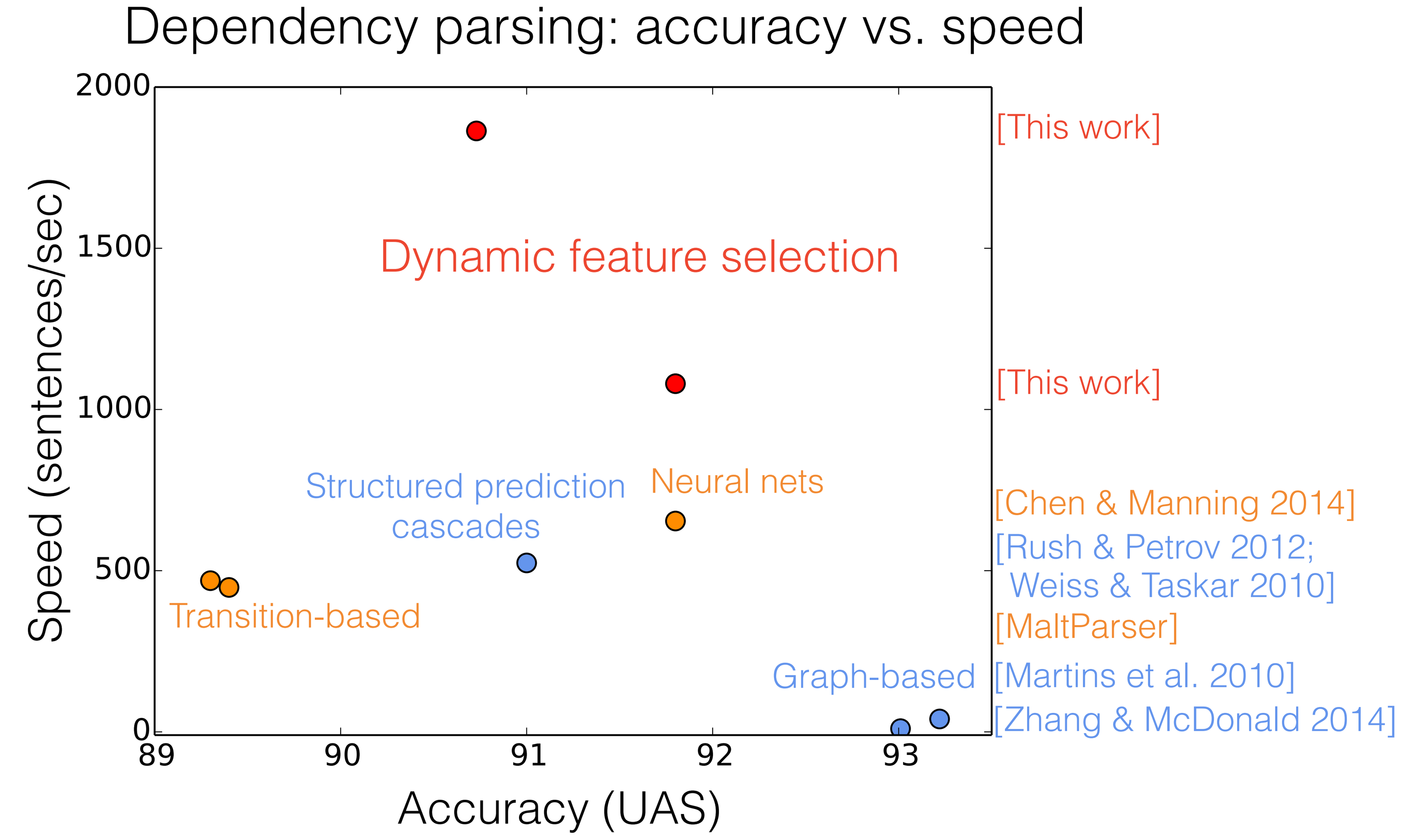
Define a linear model over *feature templates* $\{\Phi_j(x,y)\}$:

$$y^* = \arg\max_{y\in\mathcal{Y}} \mathbf{w}\cdot\Phi(x,y) \quad (1)$$

$$\mathbf{w}\cdot\Phi(x,y) = \sum_j \mathbf{w}_j\cdot\Phi_j(x,y) \quad (2)$$

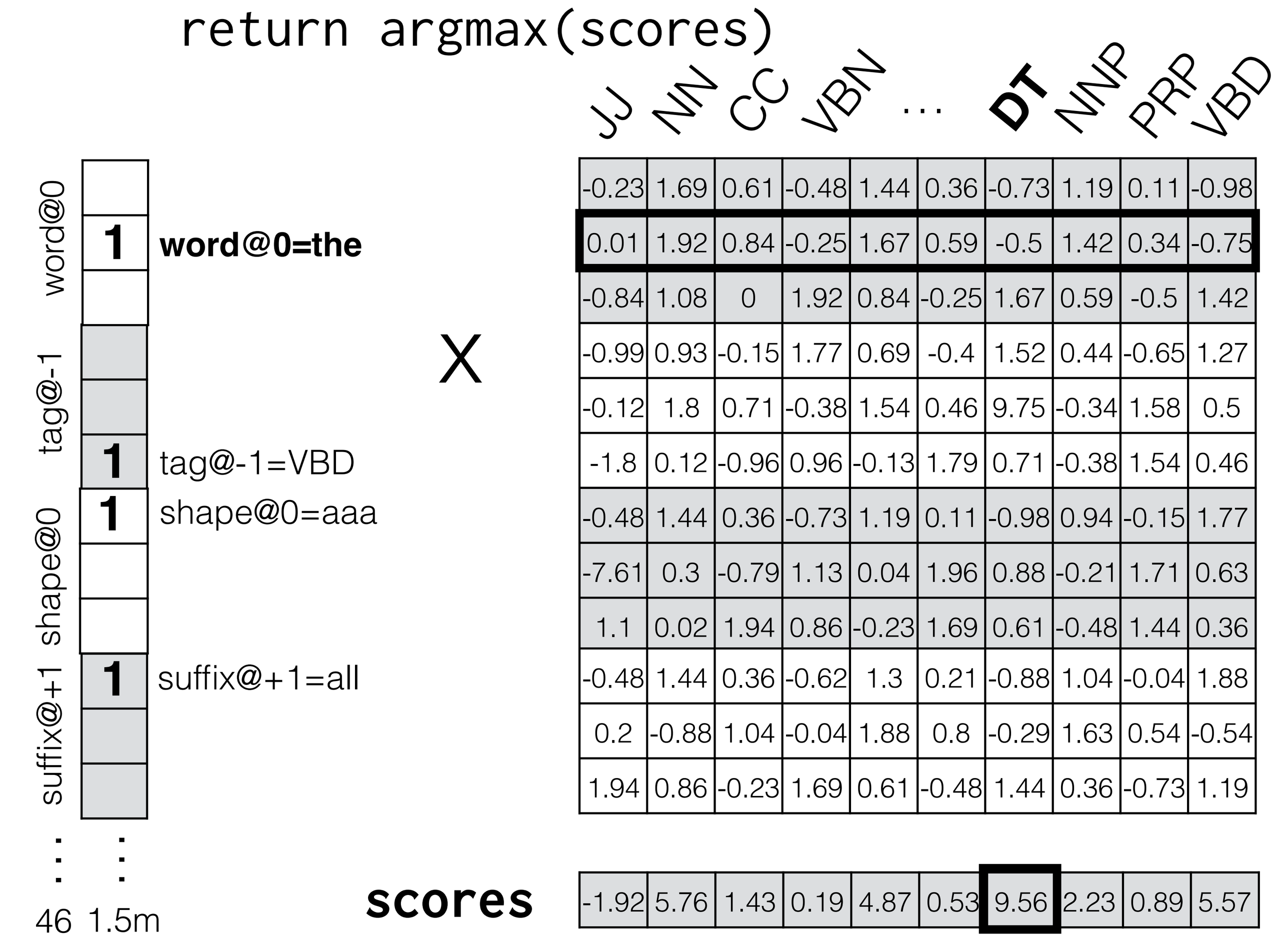
Approximate (1) using as few terms as possible from (2).

Dependency parsing: accuracy vs. speed



Dynamic feature selection

```
def predict(token) :=
   for(template <- feature_templates)
     feature_index := compute_feature(token, template)
     scores += weights[feature_index]
     if(confident)
       return argmax(scores)
```



Learning

Consider model predictions $P_{i,y}$ for each template prefix (Eq. 2).
Hinge loss with margin m on prefix score i:

$$h(P_i,y) = \max\{0, \max_{y'\neq y} P_{i,y'} - P_{i,y} + m\}$$

Per-example gradient:

$$\frac{\partial \ell}{\partial \mathbf{w}_j} = \sum_{k=j}^{i_y^*} \Phi_j(x, y_{\text{loss}}(P_k,y)) - \Phi_j(x,y)$$

where $i_y^* = \min_{i\in\{1..k\}} i$ s.t. $h(P_i,y)=0$

and $y_{\text{loss}}(P_i,y) = \arg\max_{y'} P_{i,y'} - m\cdot\mathbb{1}(y'=y)$

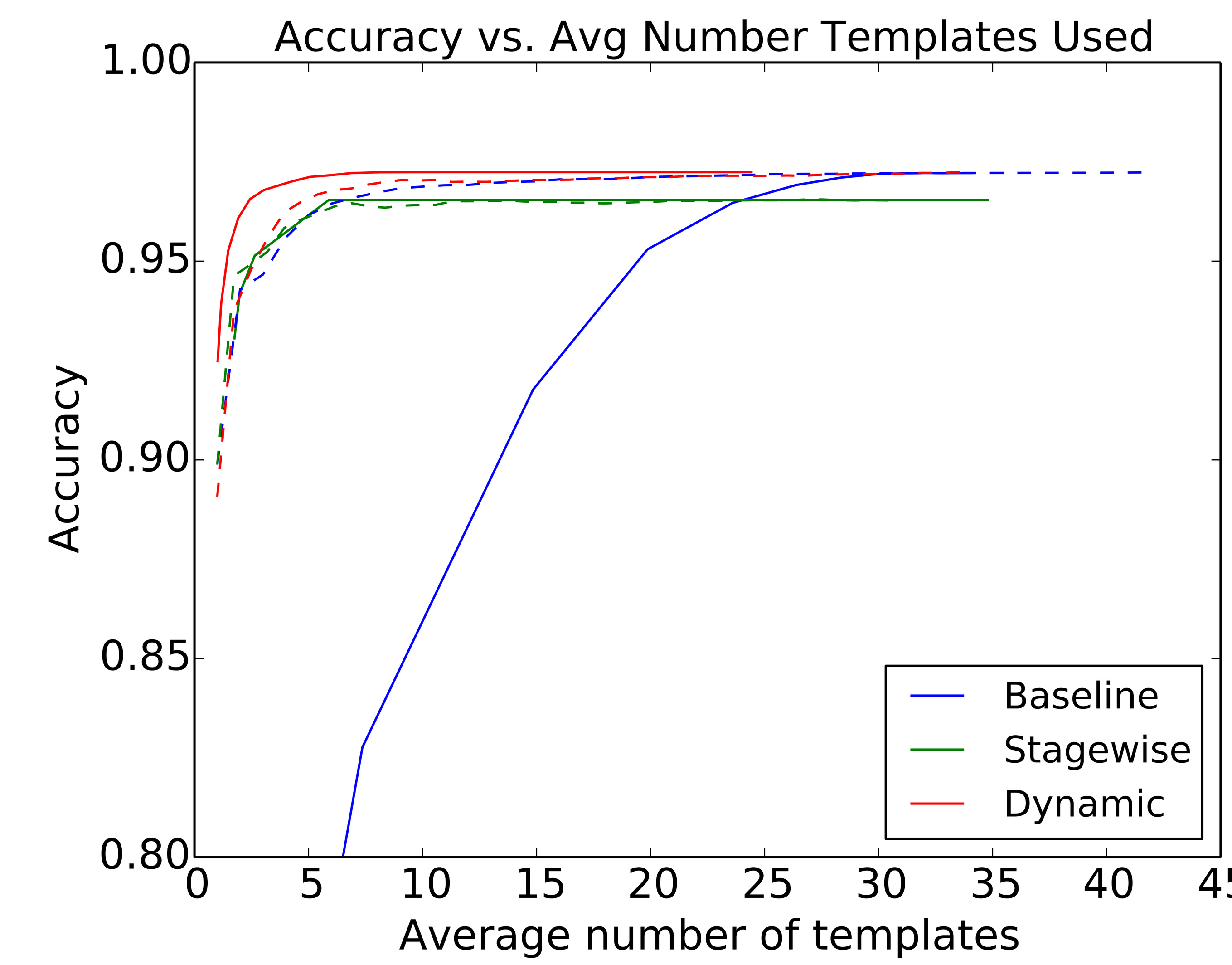
Inference

- Test time, compute prefix scores until any label has margin m.
- Train time, compute prefix scores until correct with margin m.

Experimental results

Part-of-speech tagging:

Model	Accuracy	Templates	Speedup
baseline	97.22	46	1x
dynamic *conservative*	97.21	6.89	3.41x
dynamic *aggressive*	97.02	4.33	5.22x
dynamic *v. aggressive*	96.09	1.92	10.36x



Transition-based dependency parsing:

Model	LAS	UAS	Templates	Speedup
baseline	90.31	91.83	60	1x
dynamic *conservative*	90.27	91.80	15.83	2.71x
dynamic *aggressive*	89.07	90.73	8.57	4.66x

UMASSCS