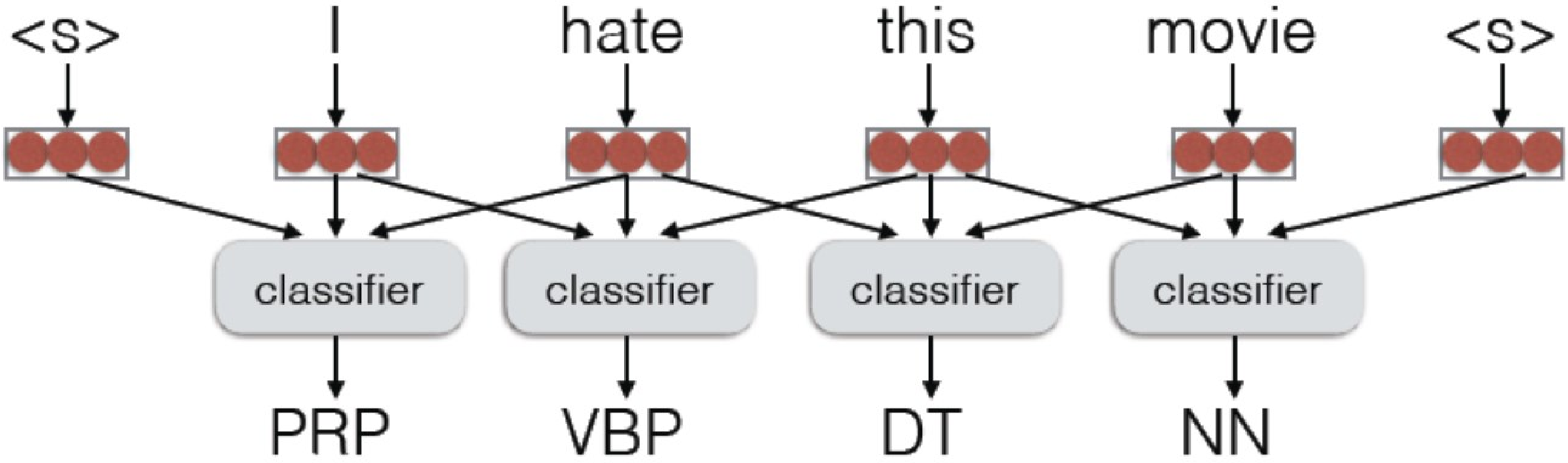
BiLSTM+CRF and Other Neural Models for Sequence Labeling

Mausam

(Many slides by Graham Neubig)

Sequence Labeling as Independent Classification

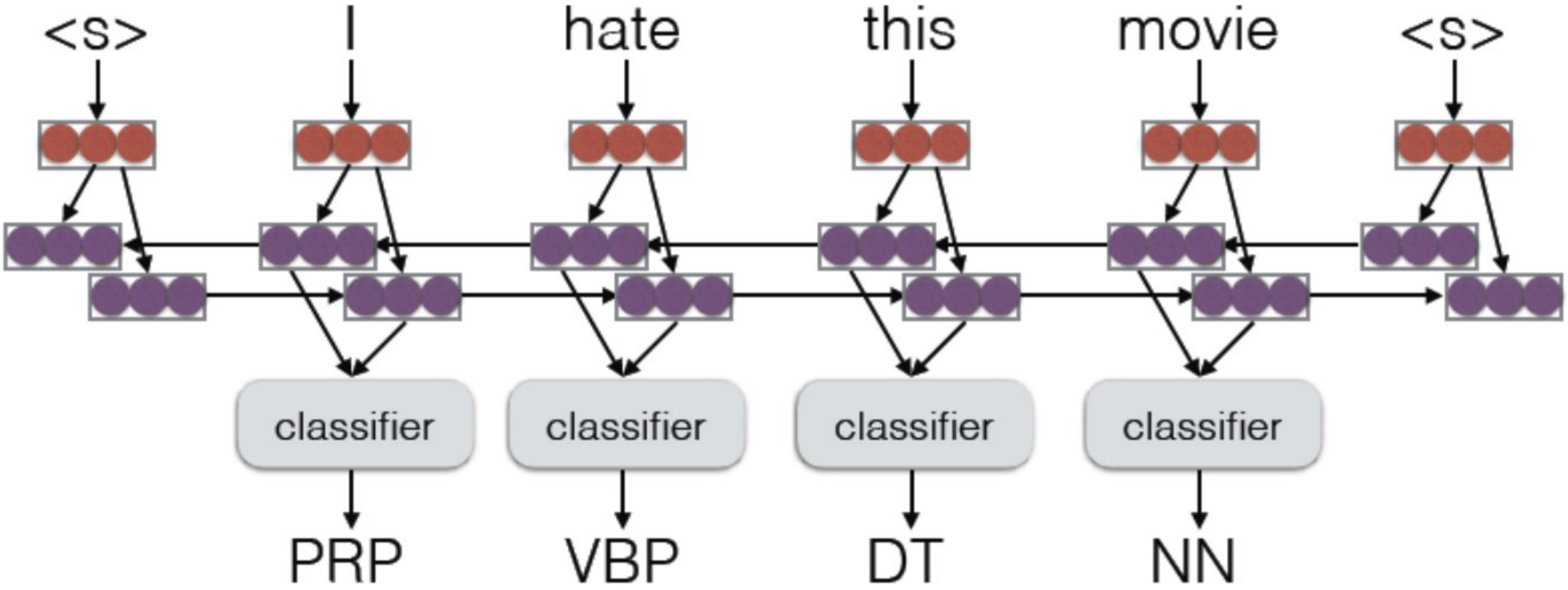


Structured Prediction task

But not a Structured Prediction Model

Instead: independent multi-class classification

Sequence Labeling with (Transducer) BiLSTM



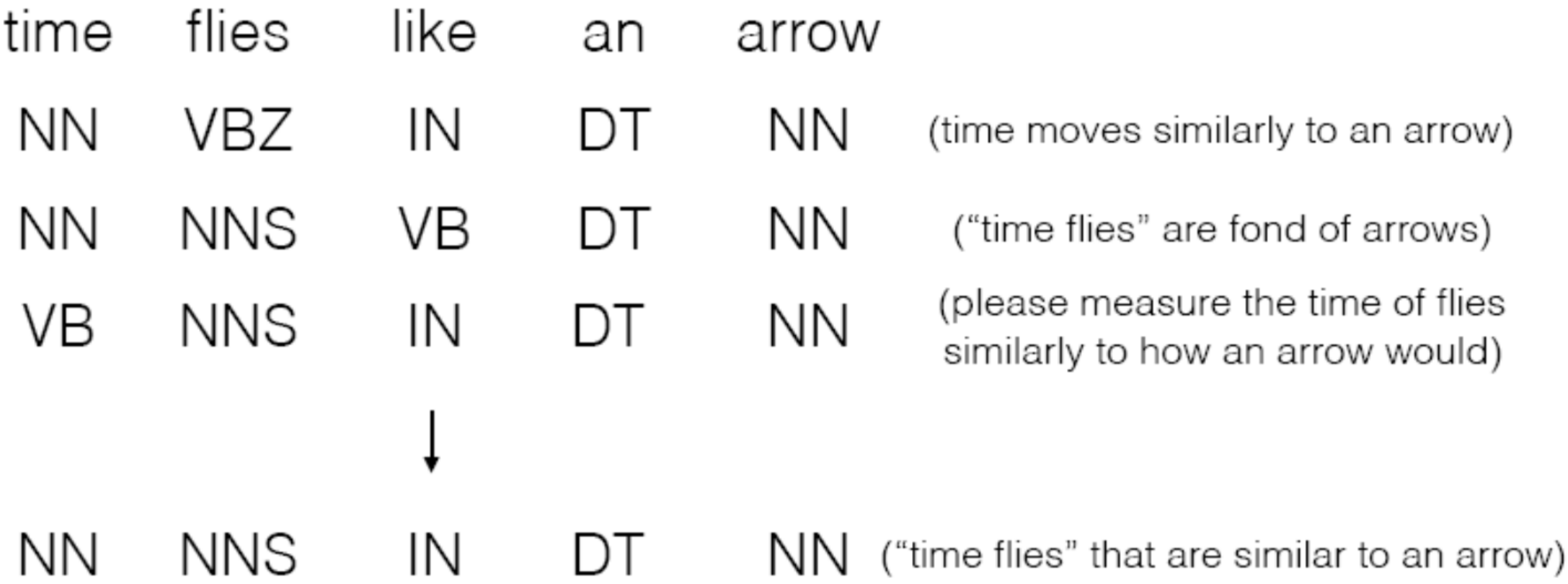
What is missing?

Still not modeling output structure!

Outputs are independent (of each other)

Why Model Interactions in Output?

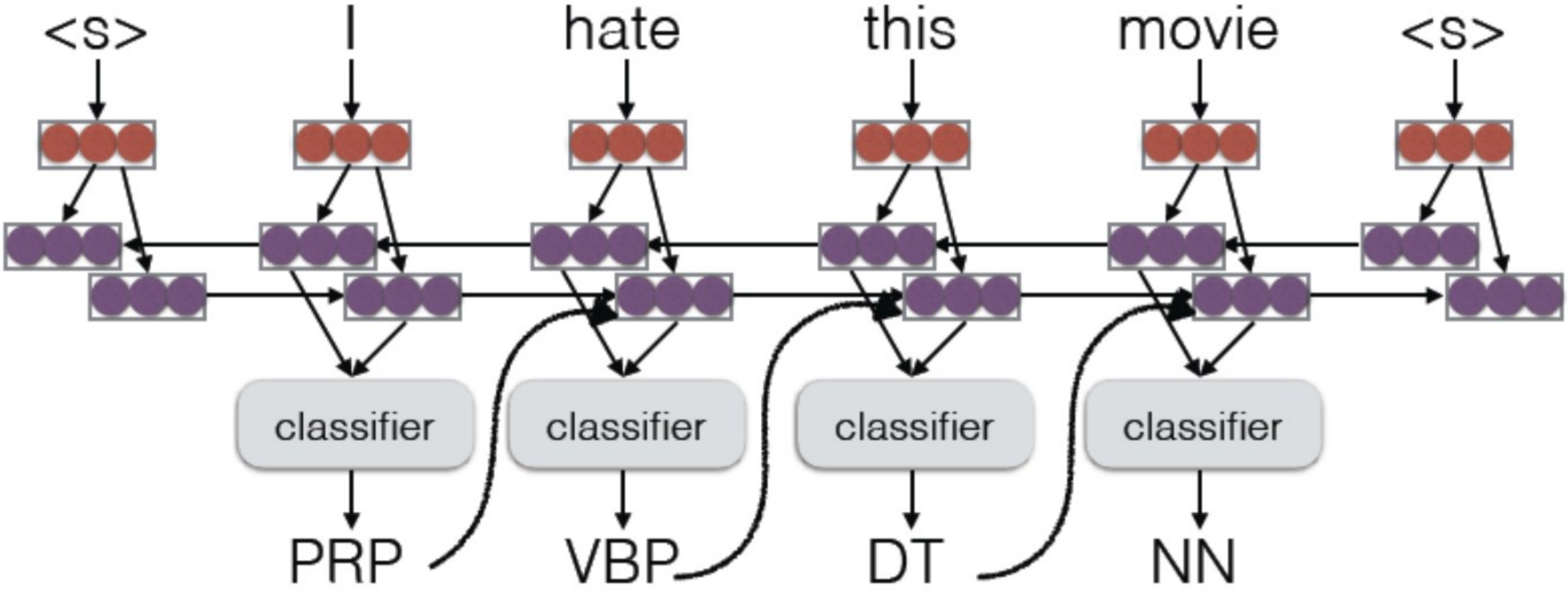
- Consistency is important!



time	flies	like	an	arrow	
NN	VBZ	IN	DT	NN	(time moves similarly to an arrow)
NN	NNS	VB	DT	NN	(“time flies” are fond of arrows)
VB	NNS	IN	DT	NN	(please measure the time of flies similarly to how an arrow would)
		↓			
NN	NNS	IN	DT	NN	(“time flies” that are similar to an arrow)

- Example 2: Paris Hilton

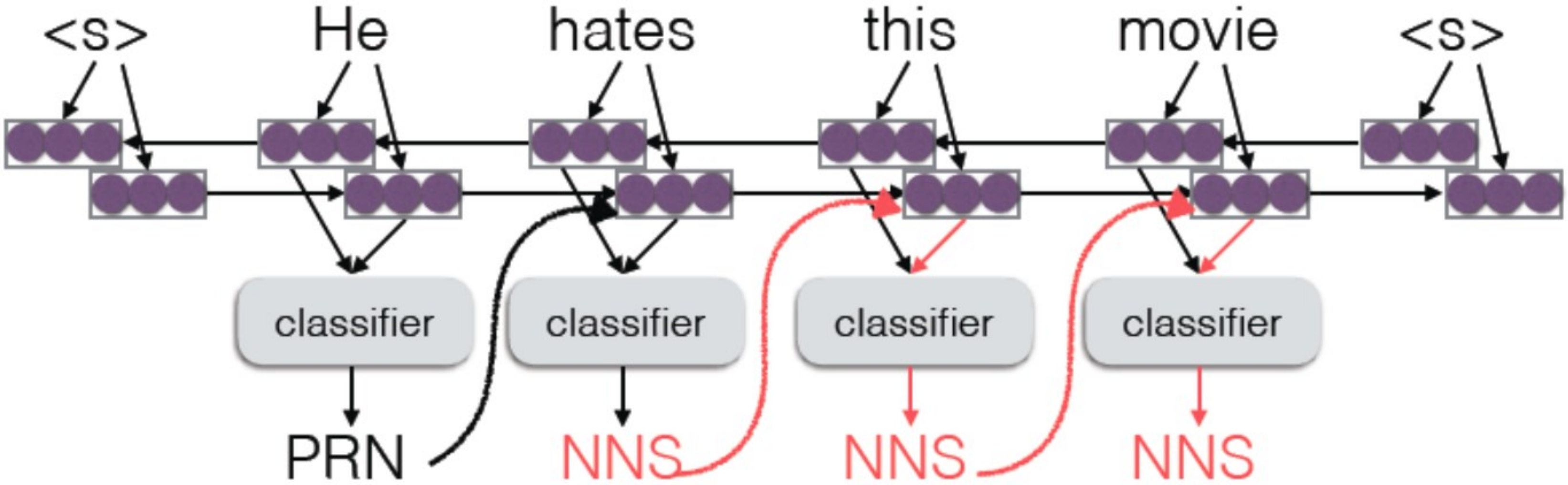
A Tagger Considering Output Structure



Tags are inter-dependent (no joint optimization)

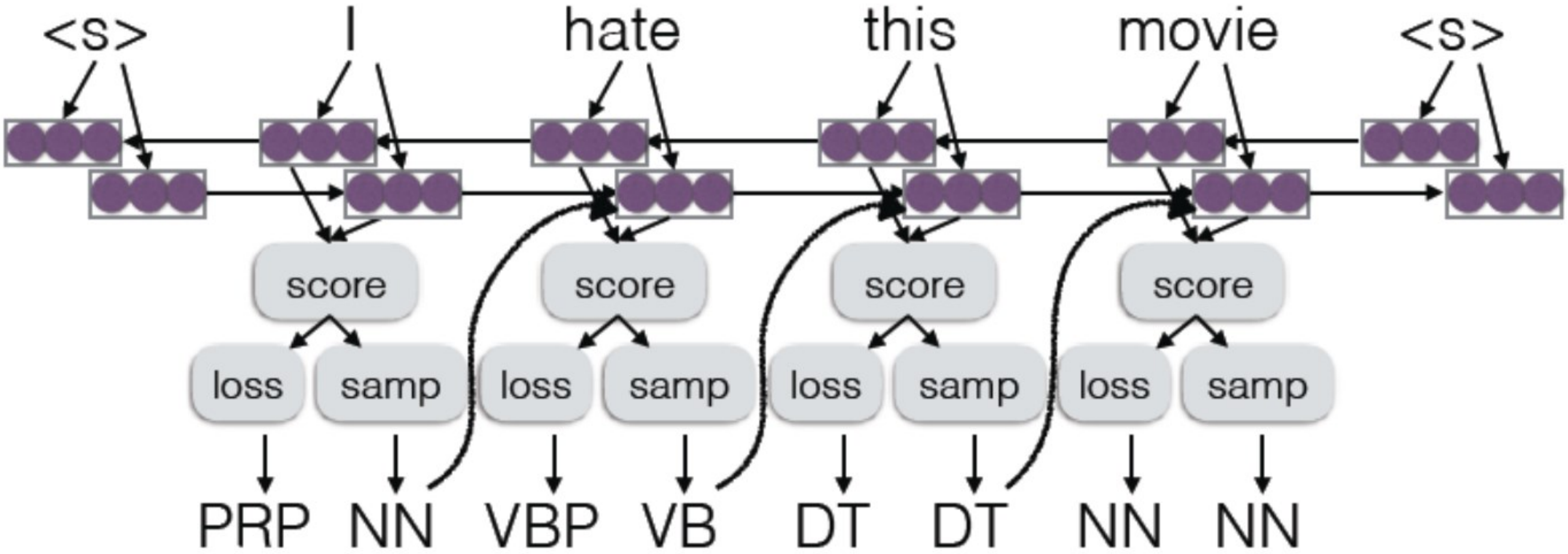
How to Train this Model?

- Issues with vanilla training
  - Slow convergence. Model instability. Poor skill.
- Simple idea: **Teacher Forcing**
  - Just feed in the *correct* previous tag during training
- Drawback: **Exposure bias**
  - Not exposed to mistakes during training



Solution to Exposure Bias #1

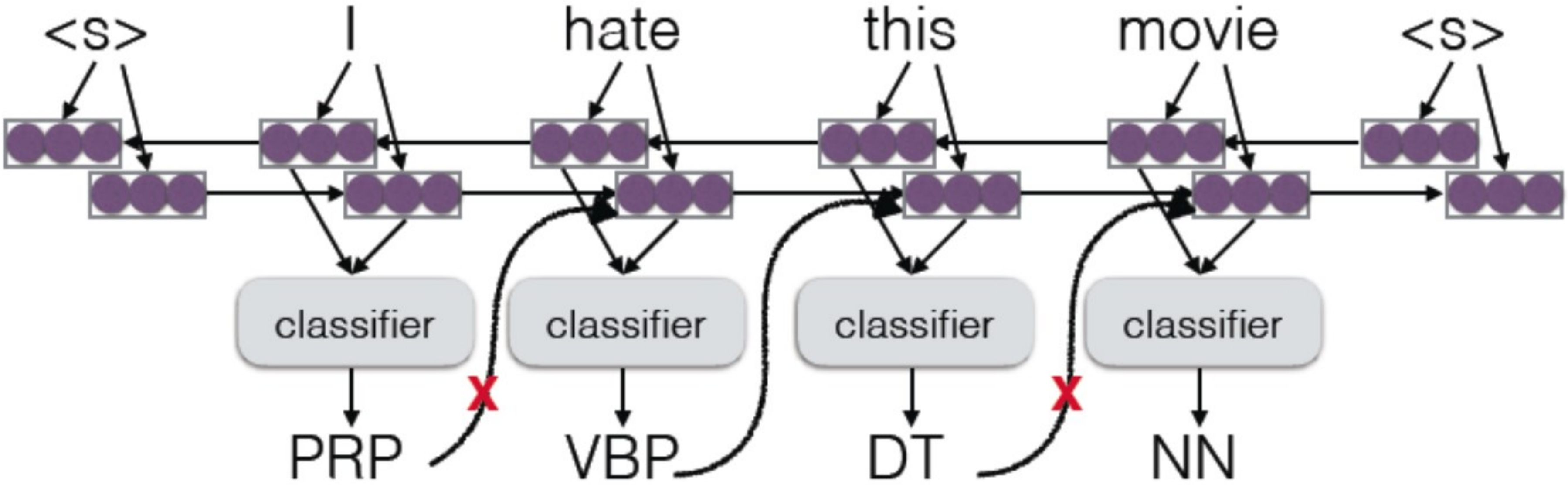
- DAgger (Ross et al. 2010) ~ "scheduled sampling"



- Start with no mistakes, and then
  - gradually introduce them using annealing
- How to choose the next tag?
  - Use gold standard/create "dynamic oracle" (Goldberg & Nivre 13)

Solution to Exposure Bias #2

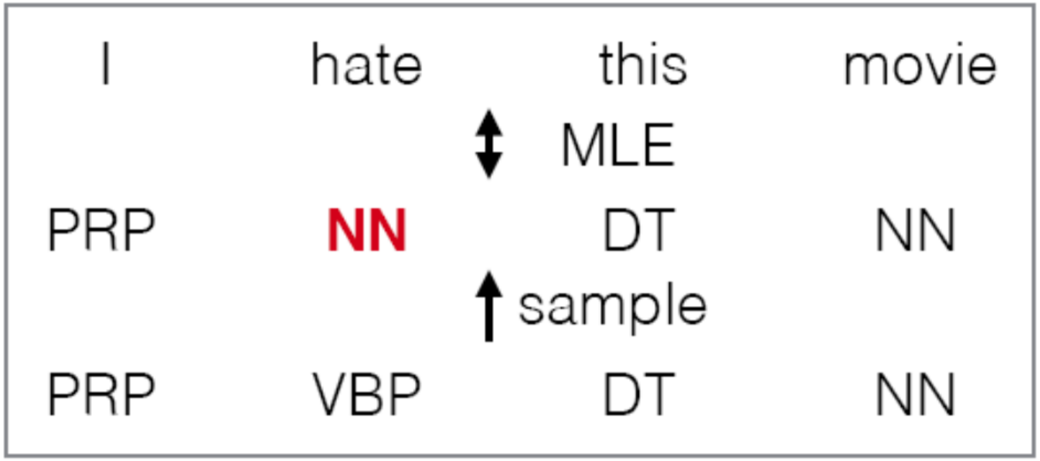
- Dropout inputs



- Helps ensure that the model doesn't rely too heavily on predictions, while still using them

Solution to Exposure Bias #3

- Corrupt training data (Nourozi et al 16)
- Sample incorrect training data; train with ML



I	hate	this	movie
PRP	**NN**	DT	NN
PRP	VBP	DT	NN

Sampling probability proportional to goodness of output

Coupling Tag-Tag Dependencies

- S(Y|X) = $\sum_{i=1}^{n} t_{i\,[\hat{y}_i]} + \sum_{i=1}^{n+1} A_{[\hat{y}_{i-1},\hat{y}_i]},$

- For a tagset of K possible tags,
  - introduce a scoring matrix A ϵ R^{KxK} in which
  - A[g,h]= compatibility score of the tag sequence g h.

- Global inference
  - Viterbi algorithm

Solution #4: Global loss Functions

- Instead of optimizing each label independently, use global objectives
- Structured Perceptron

$$\ell_{\text{percept}}(X, Y) = \max(0, S(\hat{Y} \mid X; \theta) - S(Y \mid X; \theta))$$

- Structured Hinge Loss

$$\ell_{\text{hinge}}(x, y; \theta) = \max(0, m + S(\hat{y} \mid x; \theta) - S(y \mid x; \theta))$$

- CRF Loss

$$\ell_{\text{CRF}}(X, Y; \theta) = -\log \frac{e^{S(Y|X)}}{\sum_{\tilde{Y}} e^{S(\tilde{Y}|X)}}$$

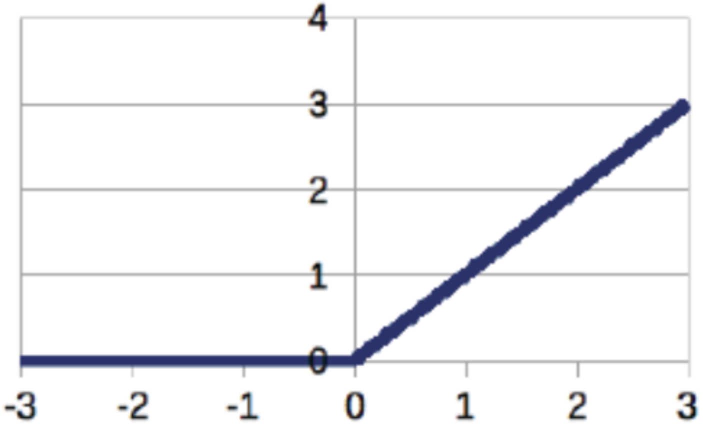
Structured Perceptron Loss

- An extremely simple way of training (non probabilistic) global models
- Find the one-best, and if it's score is better than the correct answer, adjust parameters to fix this

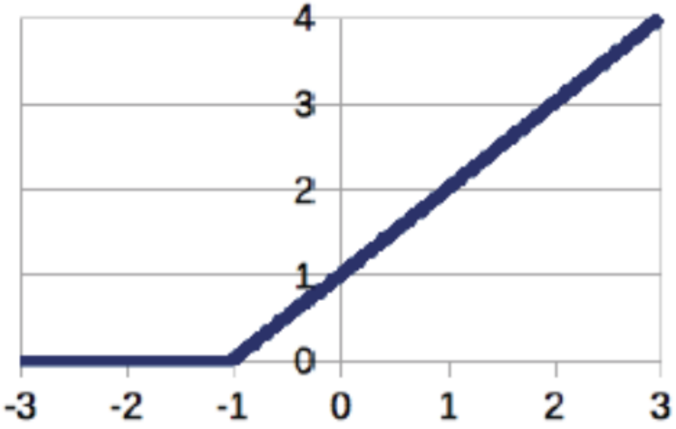
```
$\hat{Y} = \text{argmax}_{\tilde{Y} \neq Y} S(\tilde{Y} \mid X; \theta)$   ← Find one best
if $S(\hat{Y} \mid X; \theta) \geq S(Y \mid X; \theta)$ then   ← If score better than reference
    $\theta \leftarrow \theta + \alpha(\frac{\partial S(Y|X;\theta)}{\partial \theta} - \frac{\partial S(\hat{Y}|X;\theta)}{\partial \theta})$   ← Increase score of ref, decrease score of one-best (here, SGD update)
end if
```

Structured Hinge Loss (Margin)

Penalize when incorrect answer is within margin *m*



Perceptron



Hinge

Cost-augmented Hinge

- Sometimes some decisions are worse than others
  - e.g. VB -> VBP mistake not so bad, VB -> NN mistake much worse for downstream apps
- Also: good to find structures that score well in model, but are relatively wrong structurally
- Cost-augmented hinge defines a cost for each incorrect decision, and sets margin equal to this

$$\hat{Y} = \mathrm{argmax}_{\tilde{Y} \neq Y} \mathrm{cost}(\tilde{Y}, Y) + S(\tilde{Y} \mid X; \theta)$$

$$\ell_{\text{ca-hinge}}(X, Y; \theta) = \max(0, \mathrm{cost}(\hat{Y}, Y) + S(\hat{Y} \mid X; \theta) - S(Y \mid X; \theta))$$

Cost functions (for augmentation)

Zero-one loss: 1 if sentences differ, zero otherwise

$$\text{cost}_{\text{zero-one}}(\hat{Y}, Y) = \delta(\hat{Y} \neq Y)$$

Hamming loss: 1 for every different element (lengths are identical)

$$\text{cost}_{\text{hamming}}(\hat{Y}, Y) = \sum_{j=1}^{|Y|} \delta(\hat{y}_j \neq y_j)$$

Other losses: edit distance, 1-BLEU, etc.

CRF Loss (BiLSTM+CRF)

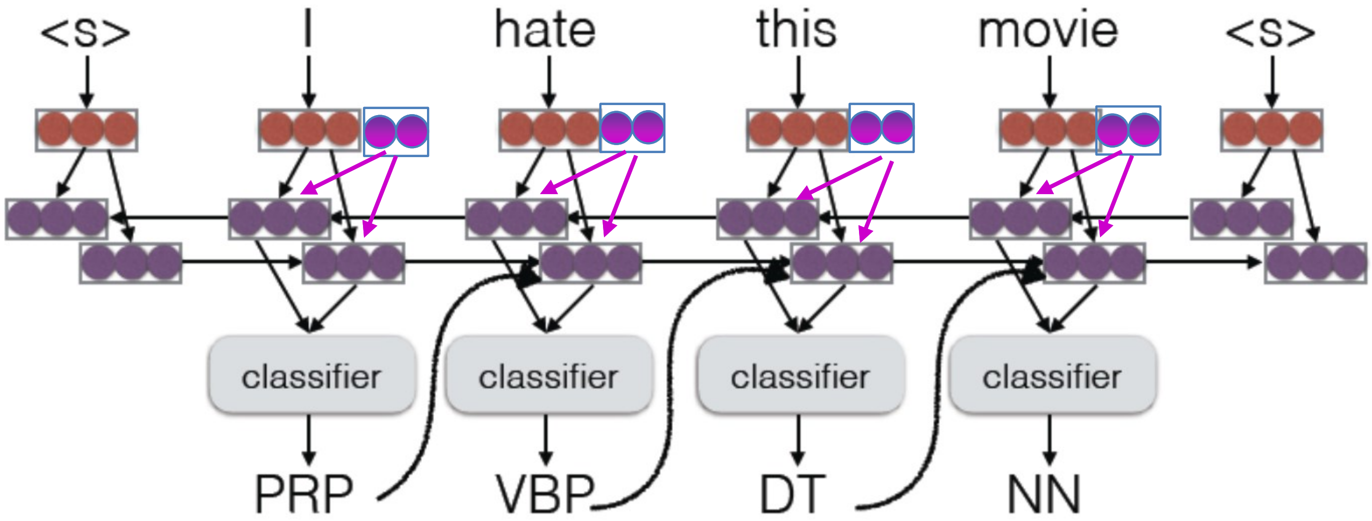
$$\text{score}_{\text{CRF}}(s, \boldsymbol{y}) = P(\boldsymbol{y} \mid s) = \frac{e^{\text{score}(s,\boldsymbol{y})}}{\sum_{y' \in \mathcal{Y}(s)} e^{\text{score}(s,\boldsymbol{y}')}}$$

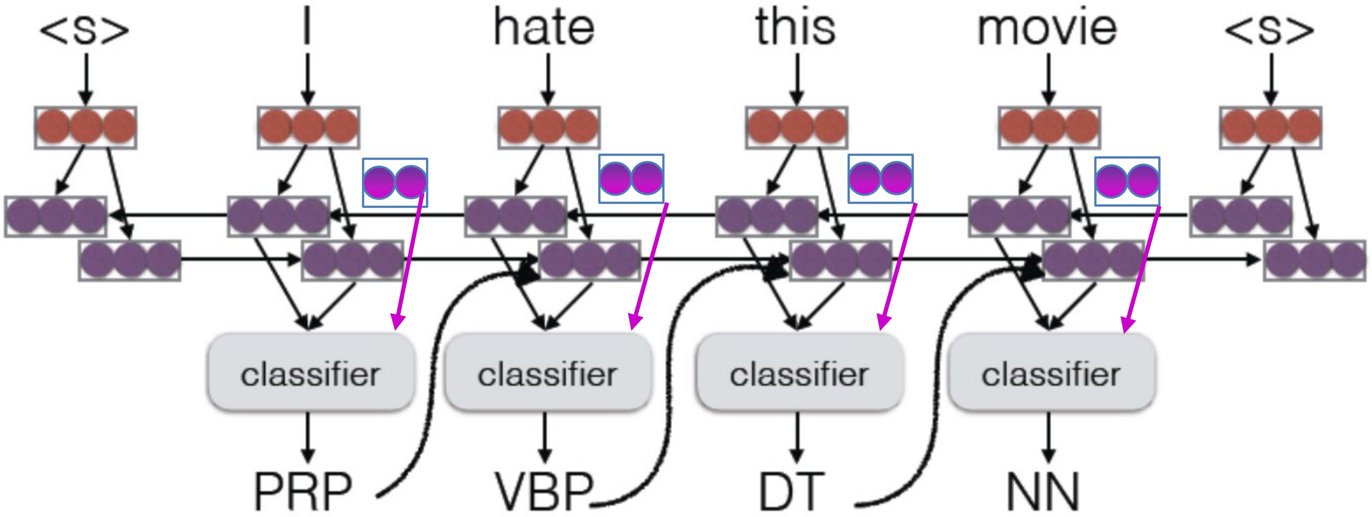
$$= \frac{\exp(\sum_{i=1}^{n} \boldsymbol{t}_{\boldsymbol{i}\,[y_i]} + \sum_{i=1}^{n} \boldsymbol{A}_{[y_i, y_{i+1}]})}{\sum_{\boldsymbol{y}' \in \mathcal{Y}(s)} \exp(\sum_{i=1}^{n} \boldsymbol{t}_{\boldsymbol{i}\,[y'_i]} + \sum_{i=1}^{n} \boldsymbol{A}_{[y'_i, y'_{i+1}]})}.$$

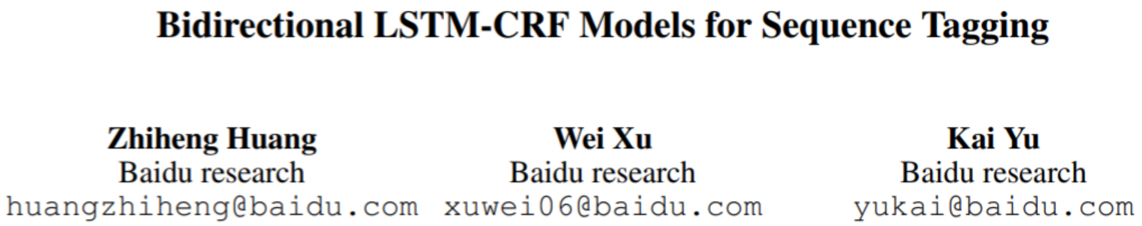
- Loss

$$-\log P(\boldsymbol{y}|s) = \underbrace{-\left(\sum_{i=1}^{n+1} \boldsymbol{t}_{\boldsymbol{i}\,[y_i]} + \sum_{i=1}^{n+1} \boldsymbol{A}_{[y_{i-1}, y_i]}\right)}_{\text{score of gold}} + \underbrace{\log \sum_{\boldsymbol{y}' \in \mathcal{Y}(s)} \exp\left(\sum_{i=1}^{n+1} \boldsymbol{t}_{\boldsymbol{i}\,[y'_i]} + \sum_{i=1}^{n+1} \boldsymbol{A}_{[y'_{i-1}, y'_i]}\right)}_{\text{using dynamic program}}$$

Adding Features into BiLSTM-CRF







Bidirectional LSTM-CRF Models for Sequence Tagging

Zhiheng Huang
Baidu research
huangzhiheng@baidu.com

Wei Xu
Baidu research
xuwei06@baidu.com

Kai Yu
Baidu research
yukai@baidu.com

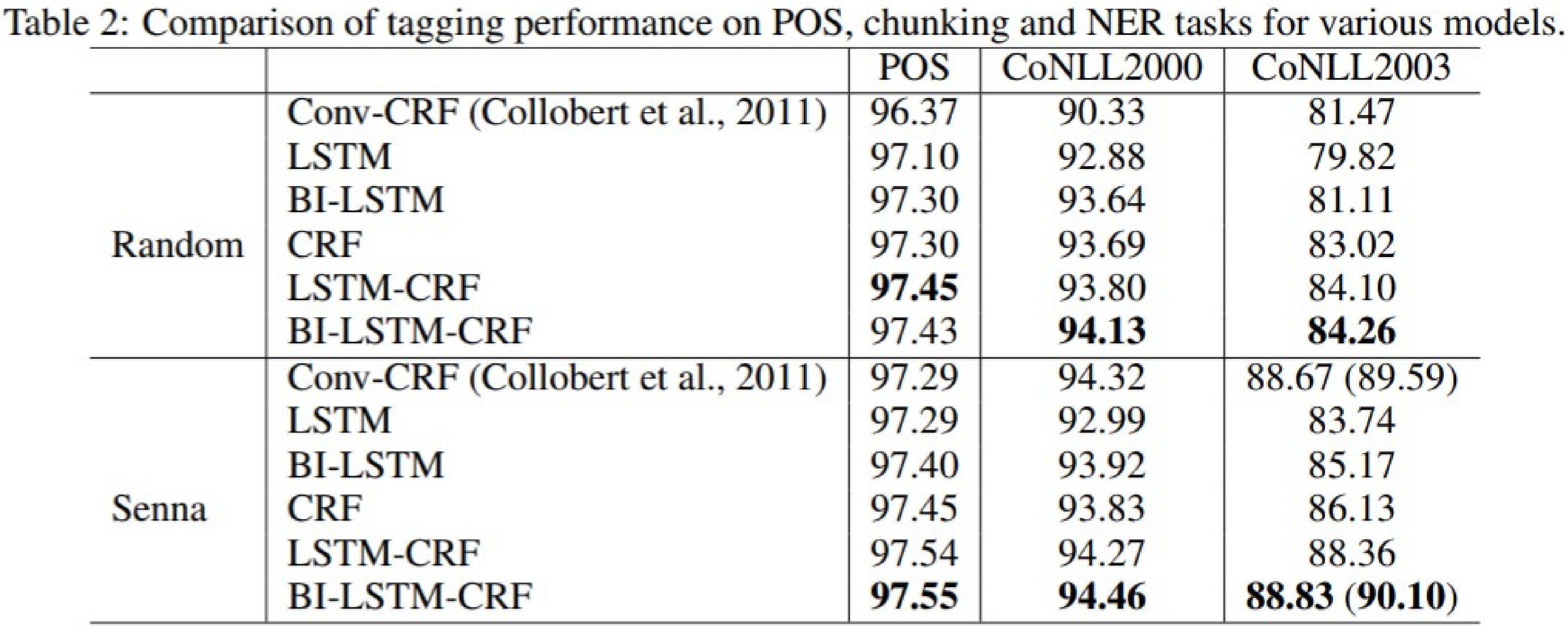
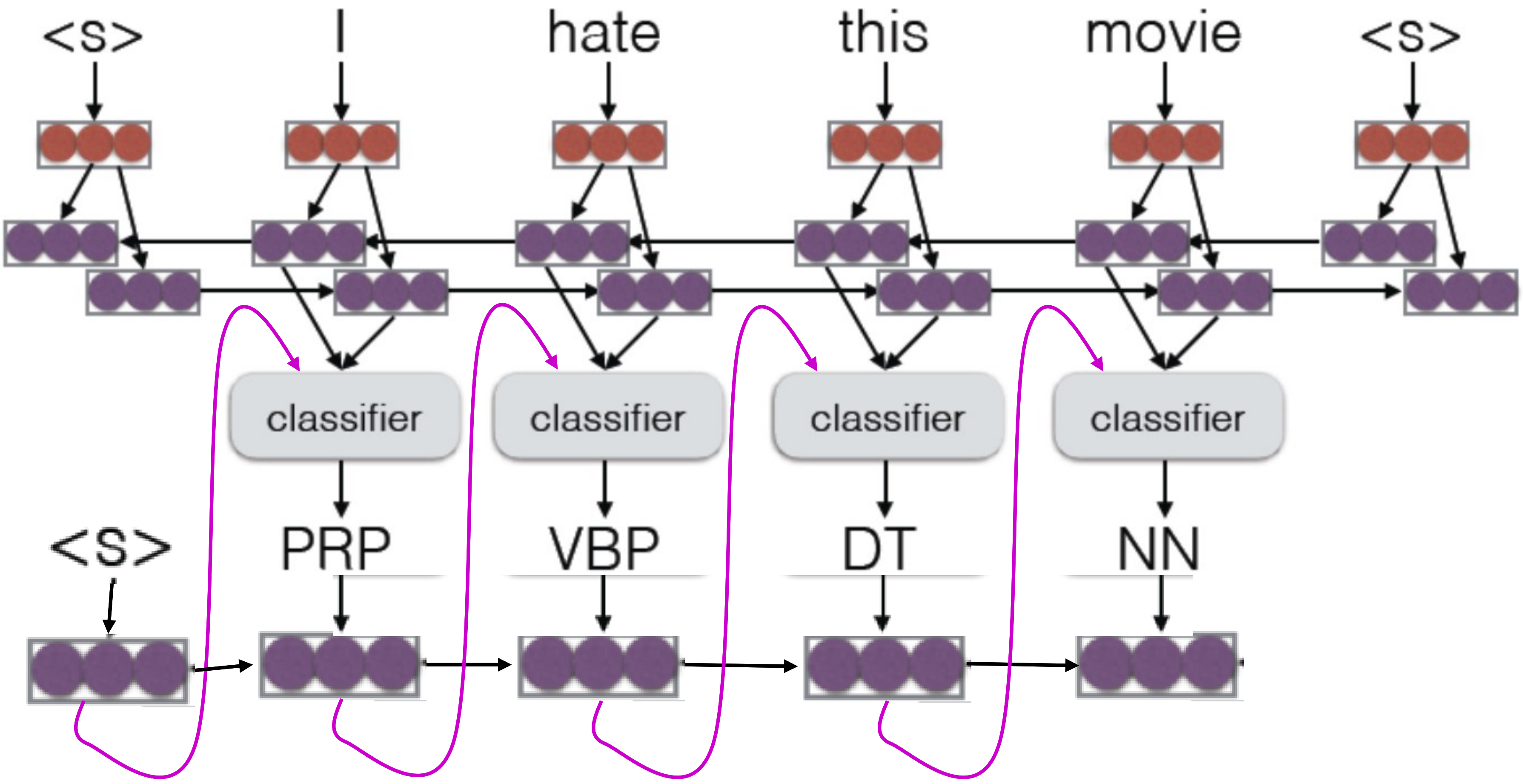


Table 2: Comparison of tagging performance on POS, chunking and NER tasks for various models.

		POS	CoNLL2000	CoNLL2003
Random	Conv-CRF (Collobert et al., 2011)	96.37	90.33	81.47
	LSTM	97.10	92.88	79.82
	BI-LSTM	97.30	93.64	81.11
	CRF	97.30	93.69	83.02
	LSTM-CRF	**97.45**	93.80	84.10
	BI-LSTM-CRF	97.43	**94.13**	**84.26**
Senna	Conv-CRF (Collobert et al., 2011)	97.29	94.32	88.67 (89.59)
	LSTM	97.29	92.99	83.74
	BI-LSTM	97.40	93.92	85.17
	CRF	97.45	93.83	86.13
	LSTM-CRF	97.54	94.27	88.36
	BI-LSTM-CRF	**97.55**	**94.46**	**88.83 (90.10)**

Non-Markovian Tag Dependency



Approx. Loss with Beam Search

- Inference: approximate by Beam search
- Learning objective: approximate CRF

$$\text{score}_{\textsc{ApproxCrf}}(s, y) = \tilde{P}(y|s) = \frac{e^{\text{score}(s,\mathbf{y})}}{\sum_{\mathbf{y}' \in \tilde{\mathcal{Y}}(s,r)} e^{\text{score}(s,\mathbf{y}')}}$$

$$\tilde{\mathcal{Y}}(s,r) = \{y^{\mathbf{1}}, \ldots, y^{r}\} \cup \{y\}.$$

Summary

- BiLSTM+CRF
  - combines feature engineering of LSTMs
  - global reasoning of CRFs

- When are CRFs helpful?
  - Joint inference
  - Low data setting