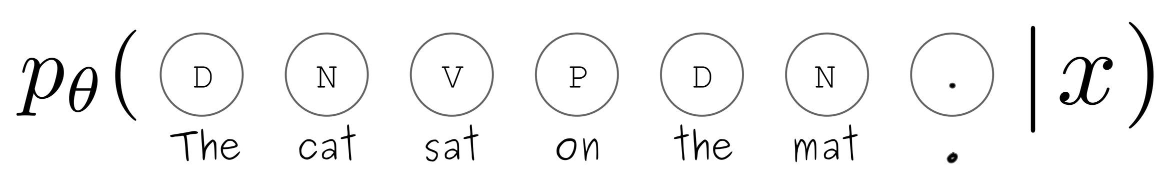
# Speed-Accuracy Tradeoffs in Tagging with Variable-Order CRFs and Structured Sparsity

Tim Vieira,* Ryan Cotterell* and Jason Eisner

Johns Hopkins University

## Variable-Order CRFs

**Goal**: Define a good conditional distribution over tag sequences.

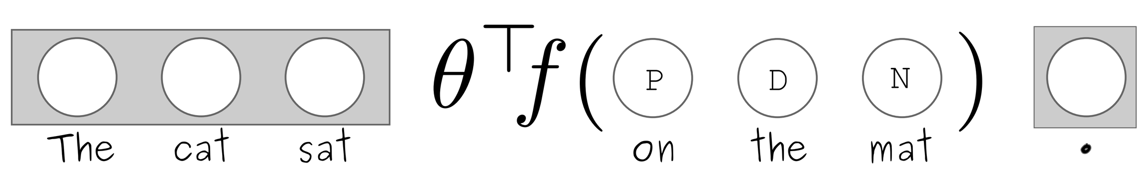



Certain combinations go well together, some don't.

label-word: D-the ✓, V-the ✗

label-label: D-N ✓, D-V ✗

Sometimes, it's useful to look at larger label combinations.




**The problem**: For features to look at output contexts of size $k$, we need $\mathcal{O}(n \cdot |Y|^k)$ time for inference even if most combinations don't improve the model, e.g., combinations that are easily ruled out by local features.

$$p_\theta(y \,|\, x) = \frac{1}{Z_\theta(x)} \exp\left( \sum_{t=1}^{n+1} \theta^\top f(x, t, y_{t-k-1} \dots y_t) \right)$$

**The VoCRF idea**: Remove output contexts that aren't necessary!

Turn $\mathcal{O}(n \cdot |Y|^k)$ into $\mathcal{O}(n \cdot |\overline{\mathcal{W}}|)$

all contexts — *important* contexts

**Technical details**:
Need closure under prefixes
& last-character substitution

$\mathcal{W} \longrightarrow \overline{\mathcal{W}}$

Even if you don't use those features

Can lift assumption with $\phi$-arcs.

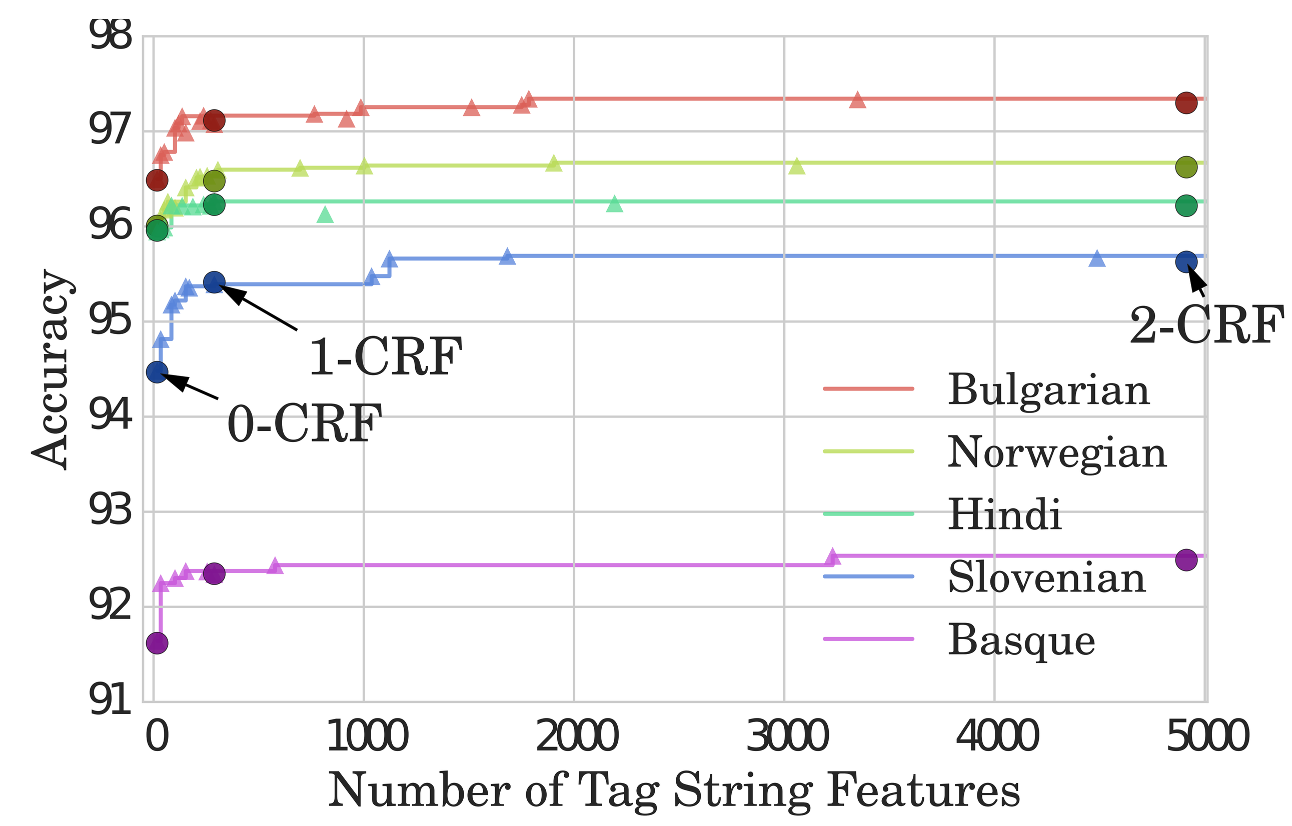
**Very flexible**
- No need to specify a fixed size.
- Covers semi-Markov & higher-order
- (Ye et al., 2009, Cuong et al., 2014)

- Can use any subset of Y*.
- Easy to implement!
  One alg. many models.

**"Correcting" prior work**

Original algorithm for computing gradients and expectations was unnecessarily slow and complicated. Our revised algorithm is $\mathcal{O}(|\overline{\mathcal{W}}|)$ times faster $\mathcal{O}$(a few pages) simpler!

- Just run autodiff on their forward algorithm!
- Protip: Evaluating the gradient should be as fast as the function!
- Check out Jason's paper at the structured prediction workshop for more on the connection between autodiff and inference.




## Structured Sparsity

**Goal**: Select higher-order features $\mathcal{W}$, which gives us the best possible accuracy under a budget for runtime.

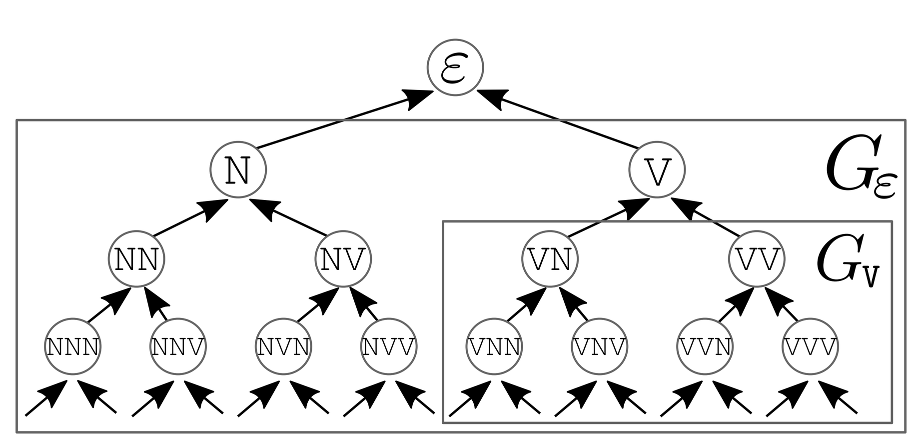
**How**: Augmenting the training objective with a penaltiy for runtime!

$$\underbrace{\sum_{i=1}^{m} -\log p_\theta(y^{(i)} \,|\, x^{(i)})}_{\text{loss}} + \underbrace{\lambda ||\theta||_2^2}_{\text{regularizer}} + \underbrace{\gamma \, \mathcal{R}(\theta)}_{\text{runtime}}$$

$\theta$ implicitly encodes $\mathcal{W}$ in its nonzero entries.

Sparsity -> Speed

Dependencies among features
- prefix closure
  NNV -> NN -> N -> $\varepsilon$
- last tag subst. closure
  NN<u>V</u> -> NN<u>N</u>
  N<u>N</u> -> N<u>V</u>

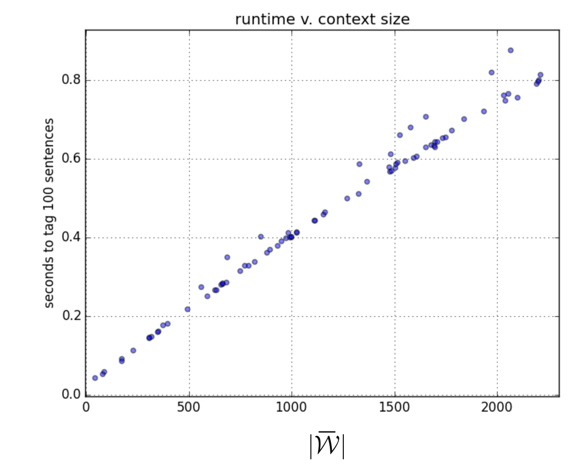



*not all groups shown

**Ideal runtime**

$$\mathcal{R}^*(\theta) = |\overline{\mathcal{W}}| = \sum_{w \in Y^*} ||\theta_{G_w}||_0$$

too hard to optimize!

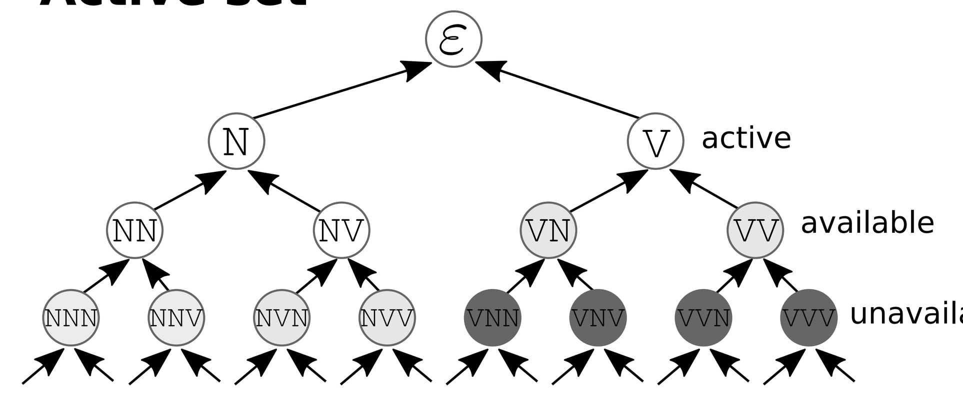


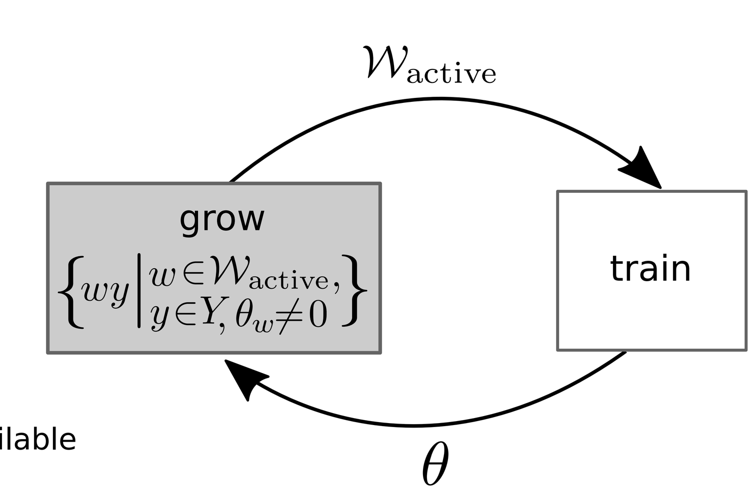

**Convex surrogate**

$$\mathcal{R}(\theta) = \sum_{w \in Y^*} ||\theta_{G_w}||_2$$

group lasso

**Active set**




## Experiments

- Part of speech tagging with Universal Tags in 5 languages.
- Best system in **bold**.
- Superscript k indicates a significant difference from the k-CRF's accuracy (paired-permutation p < 0.5), color indicates better or worse.
- Underlined system is the fastest "statistically indistinguishable" model compared to the 2-CRF.

| | k-CRF ($\lvert\overline{W}\rvert = 17^{k+1}$) | | | VoCRF at different model sizes $\lvert\overline{W}\rvert$ (which is proportional to runtime) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 0 (17) | 1 (289) | 2 (4913) | ≤ 34 | ≤ 85 | ≤ 170 | ≤ 340 | ≤ 850 | ≤ 1700 | ≤ 2550 | ≤ 3400 | ≤ 4250 | ≤ 5100 |
| Ba | $91.61^{1,2}$ | $92.35^{0}$ | $92.49^{0}$ | $92.25^{0,2}$ | $92.25^{0,2}$ | $92.38^{0}$ | $92.34^{0}$ | $92.44^{0}$ | $92.44^{0}$ | $92.44^{0}$ | **92.54**$^{0}$ | $92.54^{0}$ | $92.54^{0}$ |
| Bu | $96.48^{1,2}$ | $97.11^{0,2}$ | $97.29^{0,1}$ | $96.75^{0,1,2}$ | $96.78^{0,1,2}$ | $96.99^{0,1,2}$ | $97.08^{0,2}$ | $97.18^{0,1}$ | $97.25^{0,1}$ | **97.34**$^{0,1}$ | $97.34^{0,1}$ | $97.34^{0,1}$ | $97.34^{0,1}$ |
| Hi | $95.96^{1,2}$ | $96.22^{0}$ | $96.21^{0}$ | $95.97^{1,2}$ | $96.22^{0}$ | $96.22^{0}$ | $96.26^{0}$ | $96.13^{0}$ | $96.13^{0}$ | **96.24**$^{0}$ | $96.24^{0}$ | $96.24^{0}$ | $96.24^{0}$ |
| No | $96.00^{1,2}$ | $96.64^{0}$ | $96.66^{0}$ | $96.07^{1,2}$ | $96.26^{0,1,2}$ | $96.41^{0}$ | $96.60^{0}$ | $96.62^{0}$ | $96.64^{0}$ | **96.67**$^{0}$ | $96.64^{0}$ | $96.64^{0}$ | $96.64^{0}$ |
| Sl | $94.46^{1,2}$ | $95.41^{0,2}$ | $95.62^{0,1}$ | $94.82^{1,2}$ | $95.18^{0,2}$ | $95.36^{0,2}$ | $95.39^{0,2}$ | $95.39^{0,2}$ | **95.69**$^{0,1}$ | $95.69^{0,1}$ | $95.69^{0,1}$ | $95.69^{0,1}$ | $95.67^{0,1}$ |