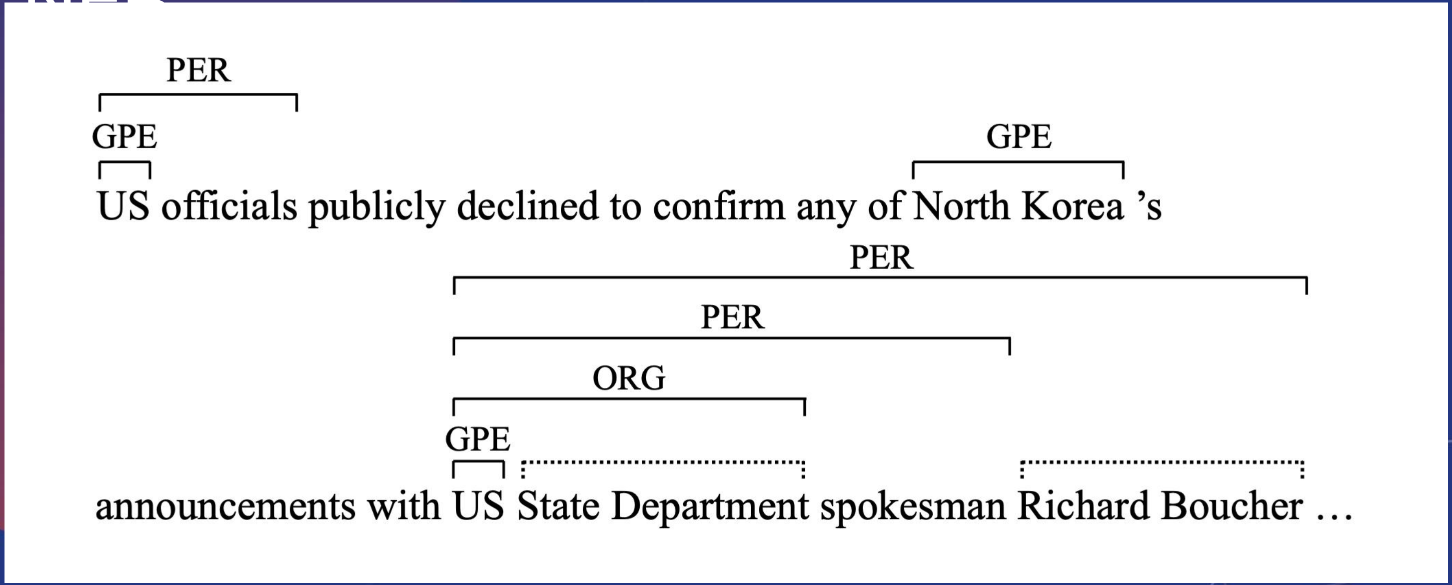
Nested Named Entity Recognition with Partially Observed TreeCRFs

Yao Fu[1*], Chuanqi Tan[2*] Mosha Chen[2], Songfang Huang[2], Fei Huang[2]

[*]Equal Contribution [1]University of Edinburgh [2]Alibaba Group

Task: Nested NER



Formulation: constituency parsing with partially-observed trees

Goal

A model that properly tackles nested NER
- Jointly model the observed and the latent
- Good performance

A simple and efficient model
- Avoid re-inventing neural architectures
- Batchified and tensorized computation

Approach

TreeCRFs with partially marginalization

- But inefficient, a major drawback in previous literature

No clear batchification:
Difference sentences, different tree structures

O(n^3) complexity for each sentence

This work focus on these two problems

Our contribution

No clear batchification: Difference sentences, different tree structures

↓

Batchfied likelihood evaluation

O(n^3) complexity for each sentence

↓

O(n log n) complexity

We propose efficient partial marginalization with MASKED INSIDE algorithm

Model

$$e_1, \ldots, e_n = \mathrm{FF}(\mathrm{Enc}(x))$$
$$s_{ijk} = e_i^\intercal U_k^{(1)} e_j + (e_i + e_j)^\intercal U_k^{(2)} + b_k$$

Standard Biaffine Scoring (Dozat and Manning 16)

$$\log p(T|x) = s(T) - \log Z$$
$$s(T) = \log \sum_{\tilde{T} \in \tilde{\mathcal{T}}} \exp(s(\tilde{T}))$$

Training maximize the likelihood for a partial tree …

… by summing over all compatible full trees

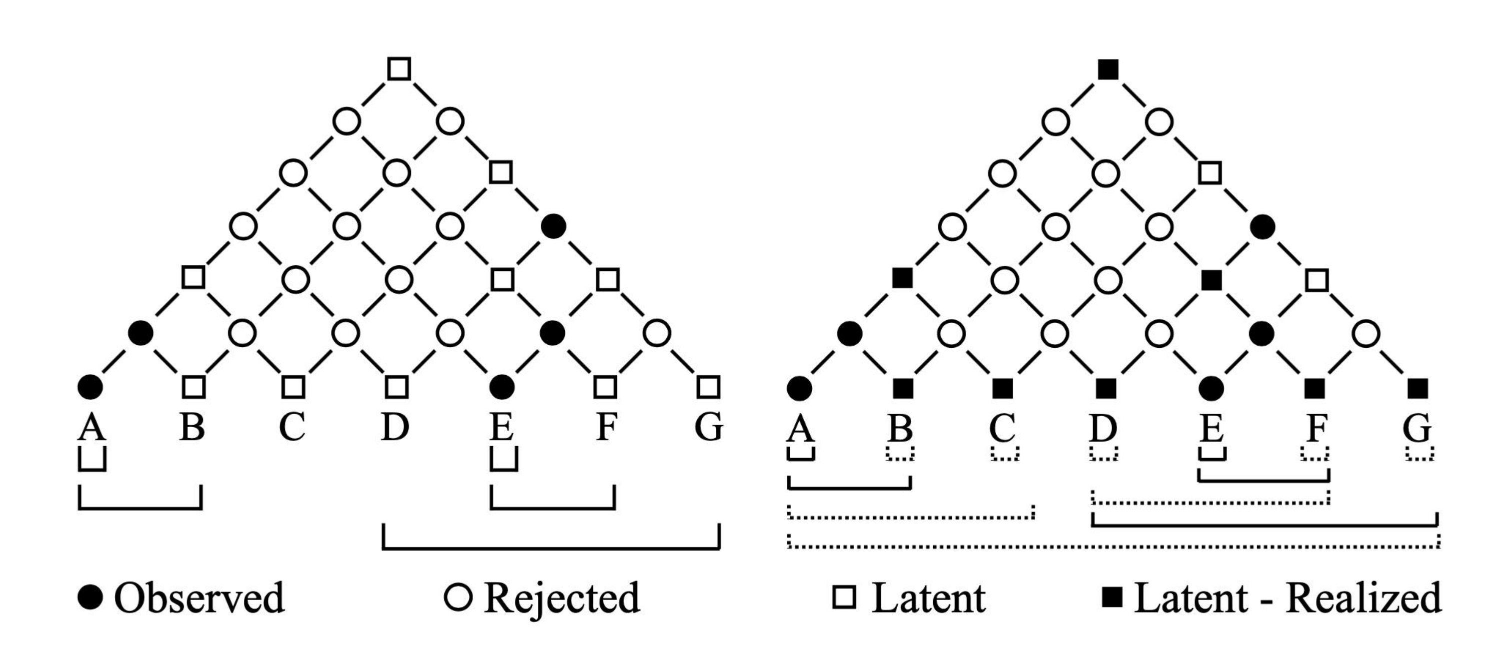
Model

$$s(T) = \log \sum_{\tilde{T} \in \tilde{\mathcal{T}}} \exp(s(\tilde{T}))$$

This summation can be down with an Inside-styled DP

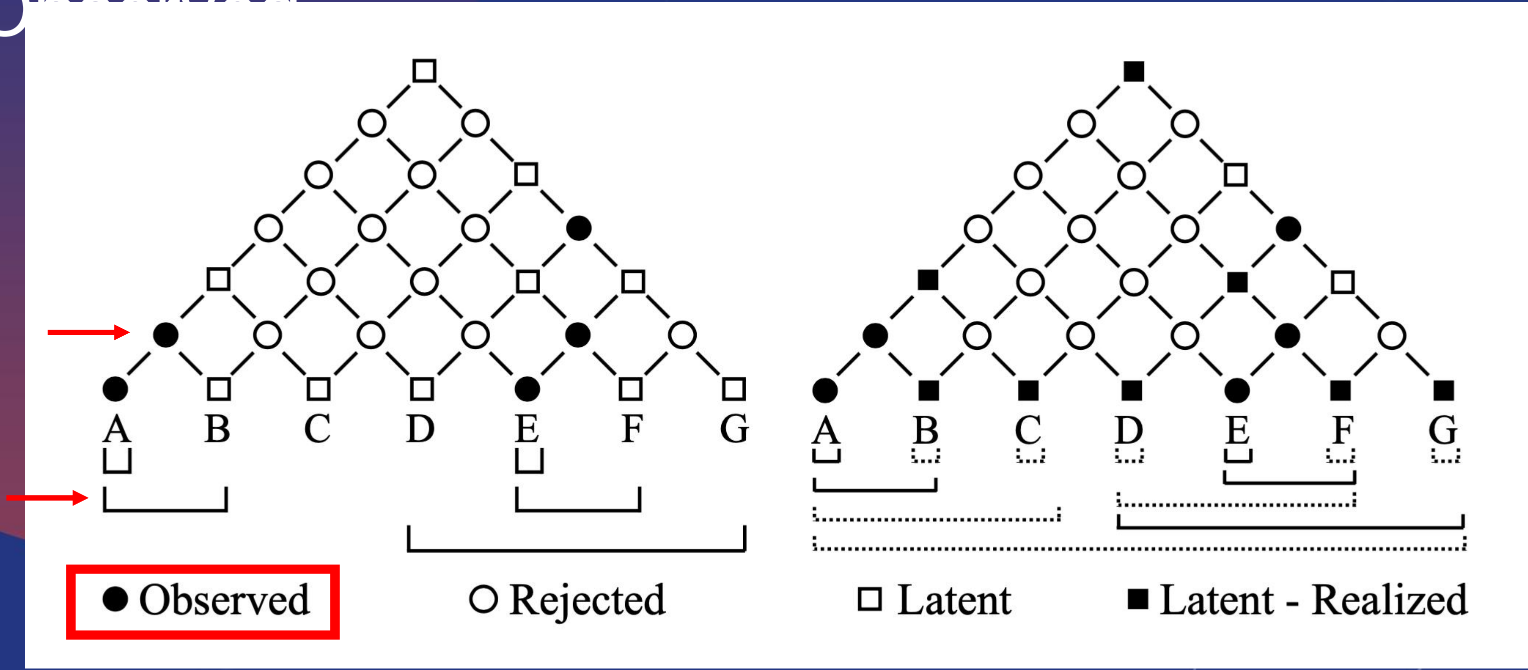
But no clear batchification for sentences with different partial trees because DP graph is different

To perform partial marginalization

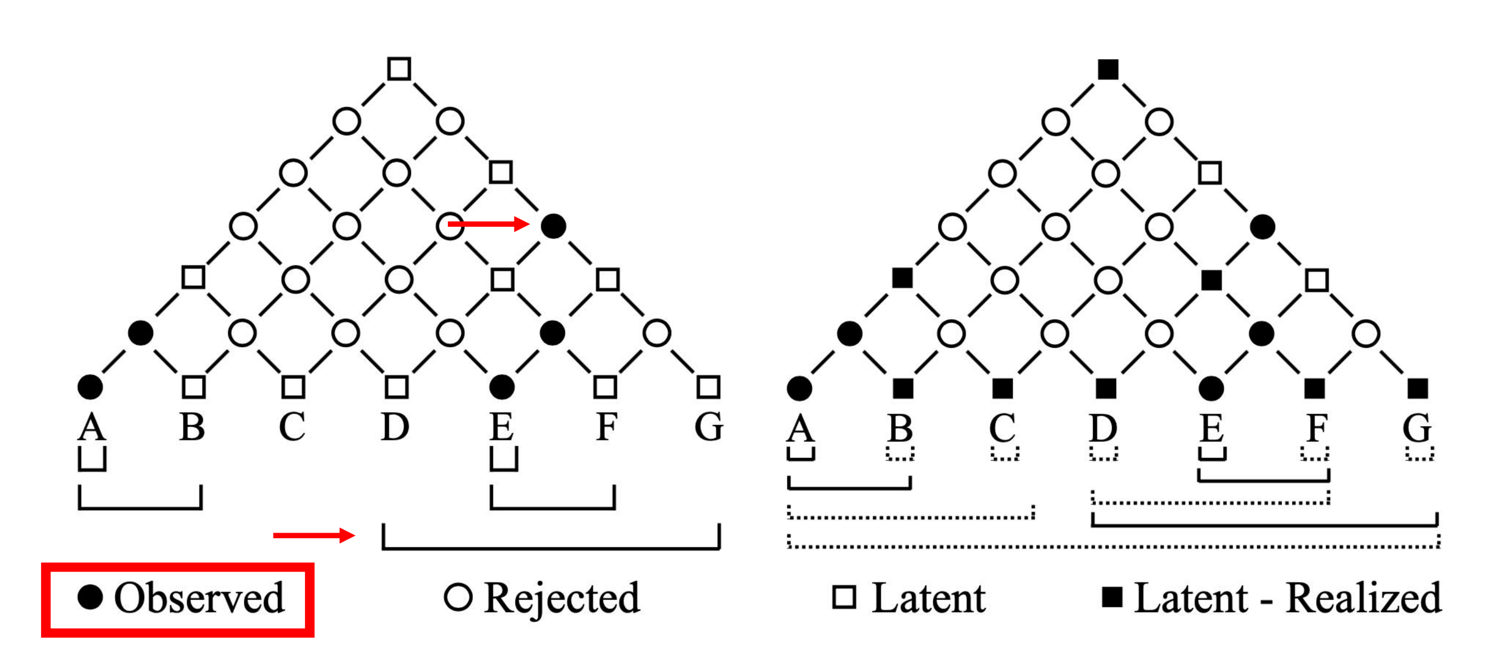


Left: Observed partial tree
Right: An example full tree realized from left (other possible full trees exist)

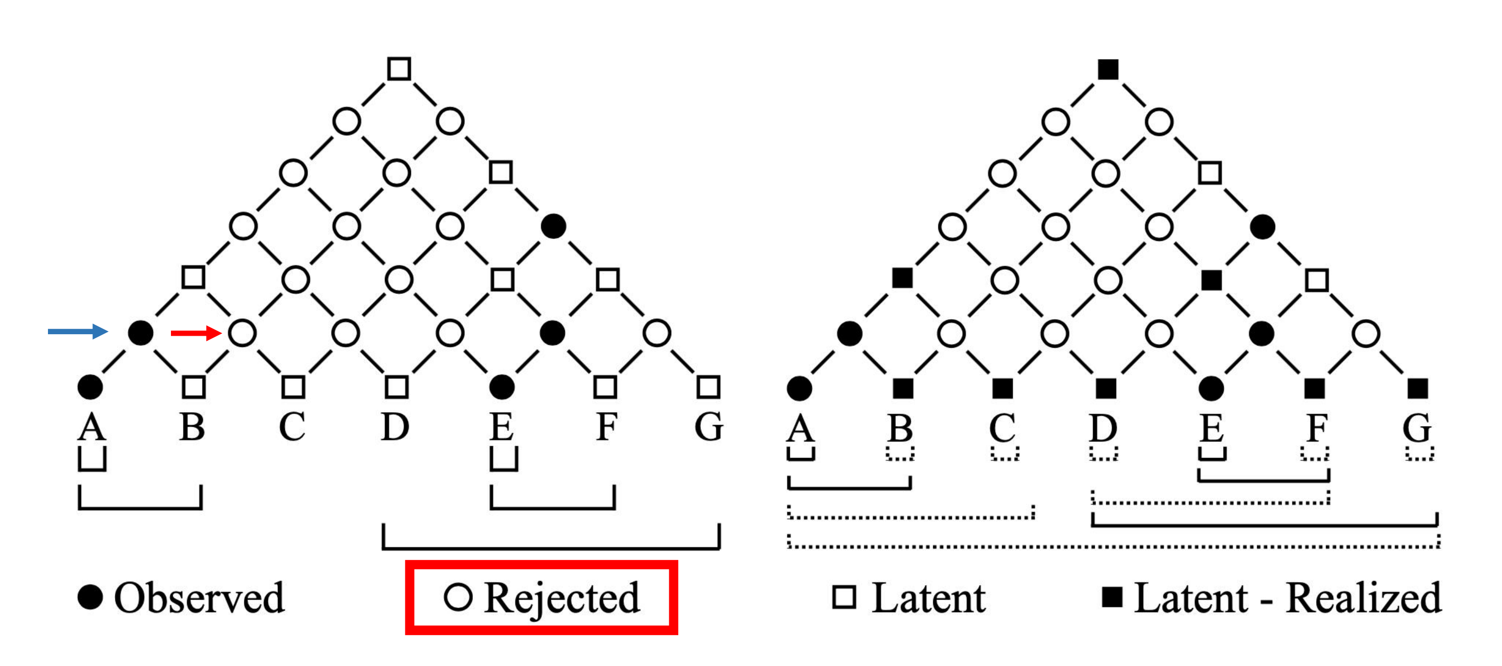
Example Observed



Example
Observed

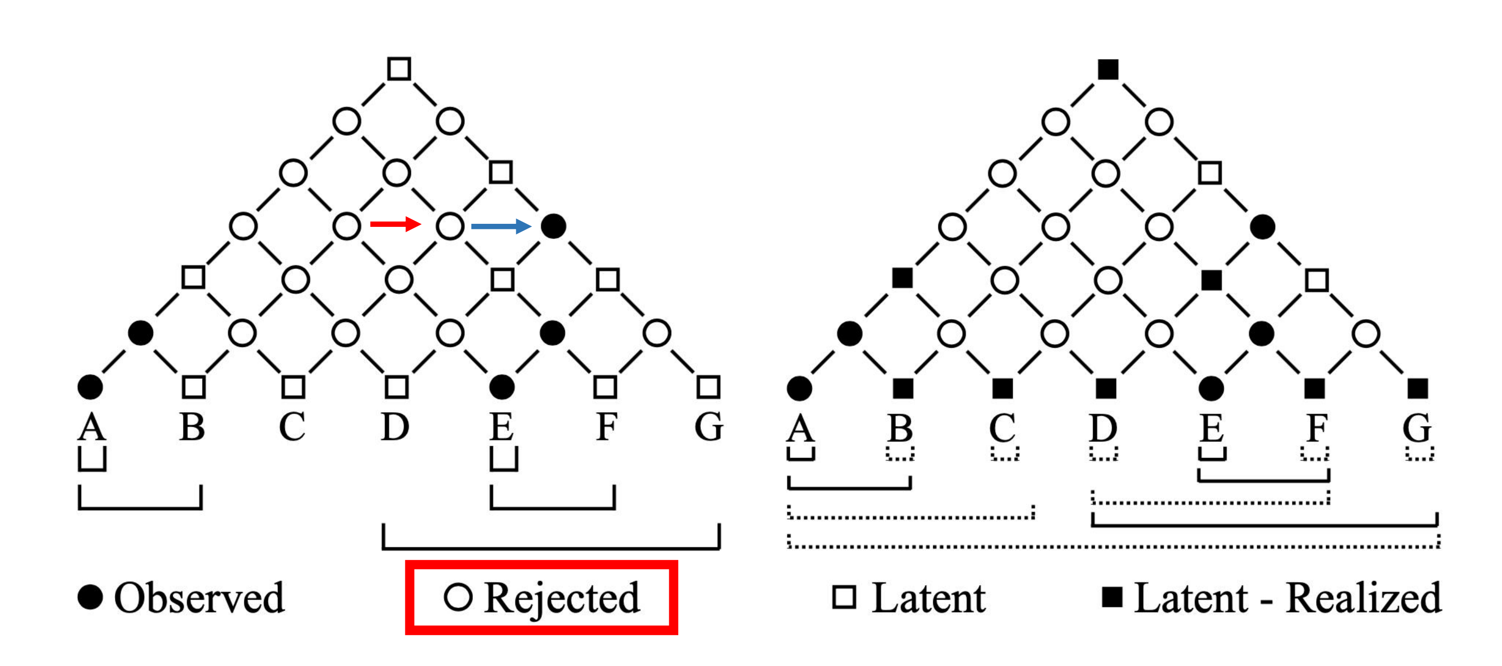


Example
Rejected



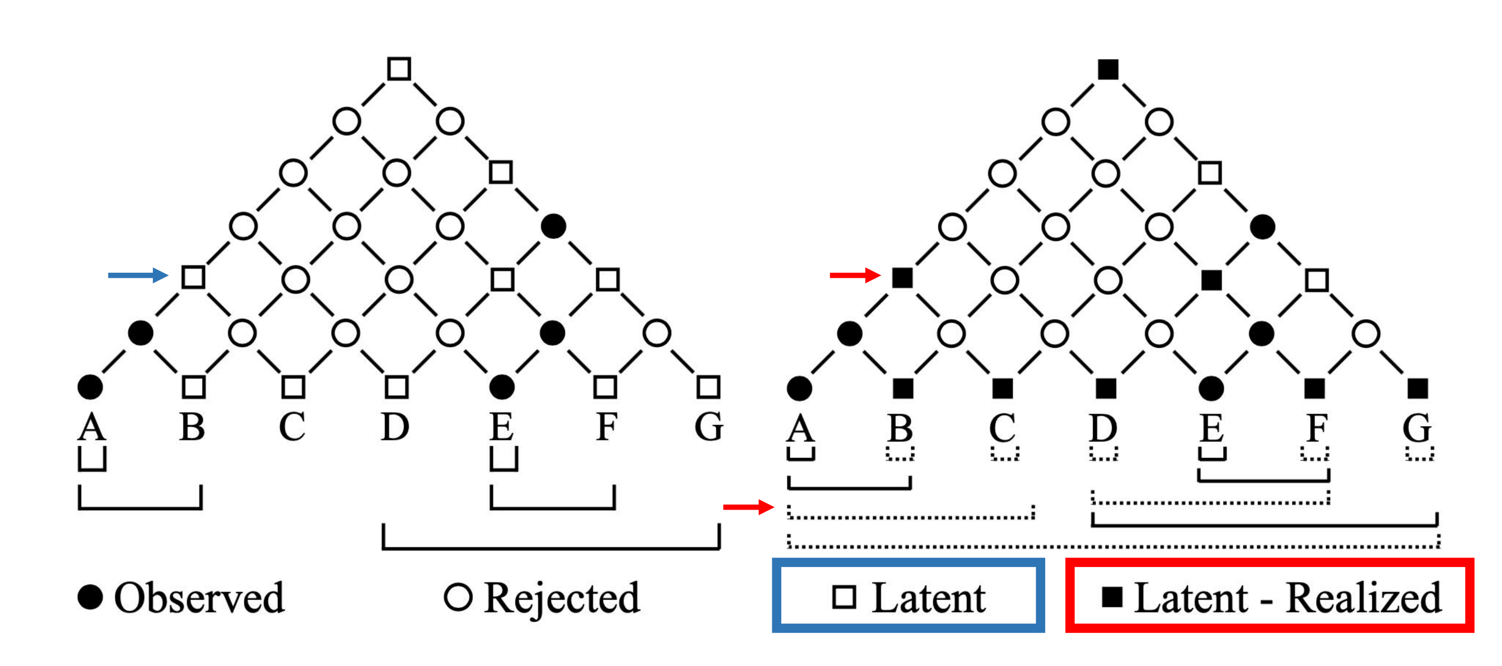
Rejected because of overlapped spans

Example
Rejected



Rejected because of overlapped spans

Example Latent



Latent nodes can be realized in a full tree

Tagging for Latent

LATENT_1

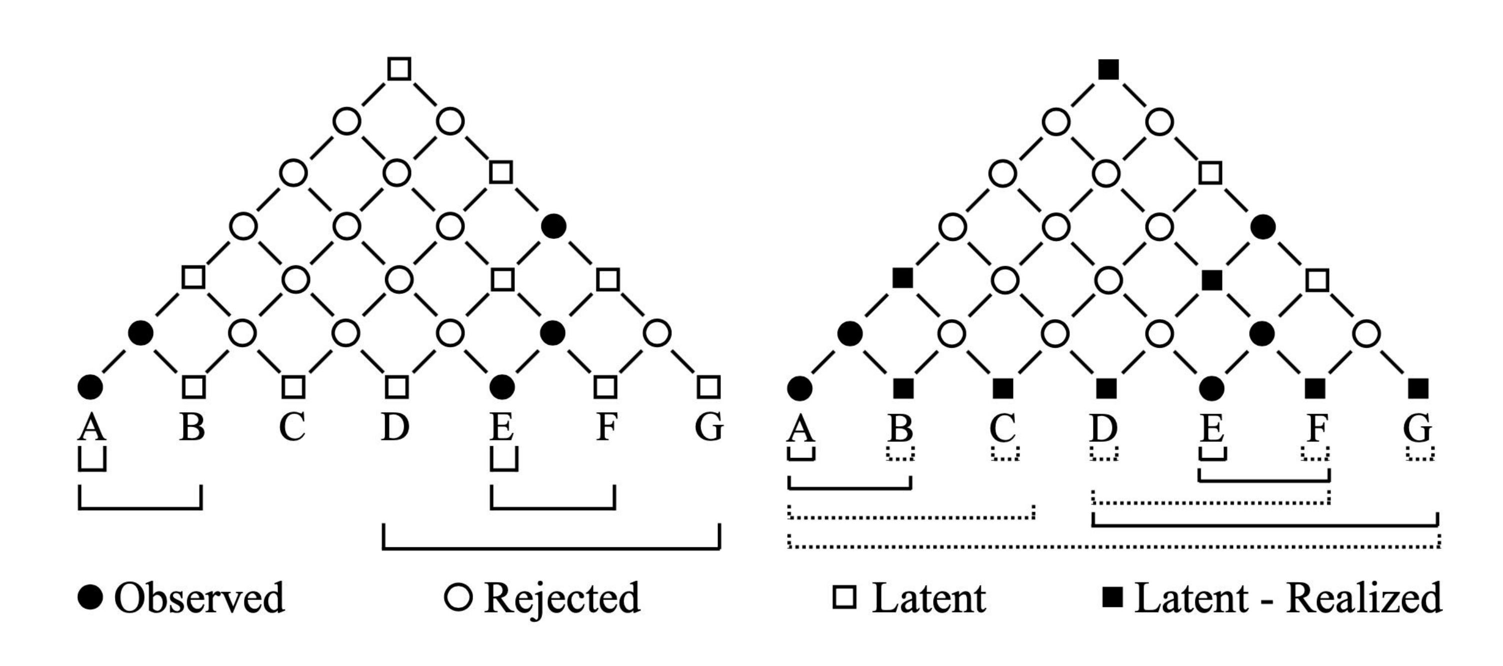
PERSON

… State Department Spokesman Richard Boucher …

Latent nodes cannot have observed tags: PER, LOC, ORG …
Latent nodes can only be labeled as: LATENT_1, LATENT_2 …

During training we marginalize all possible latent tags
During Inference we drop entities with latent tags to get partial trees

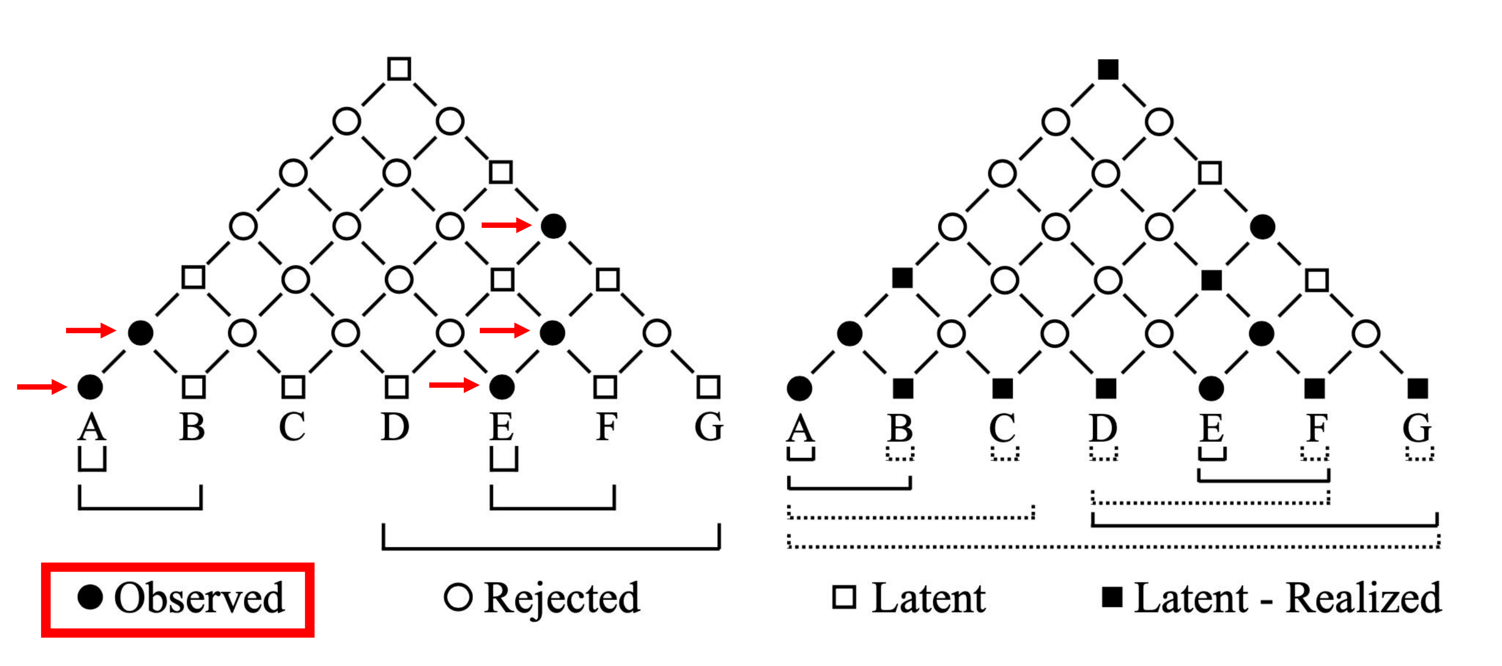
Why we care about nodes with difference



$$s(T) = \log \sum_{\tilde{T} \in \tilde{\mathcal{T}}} \exp(s(\tilde{T}))$$

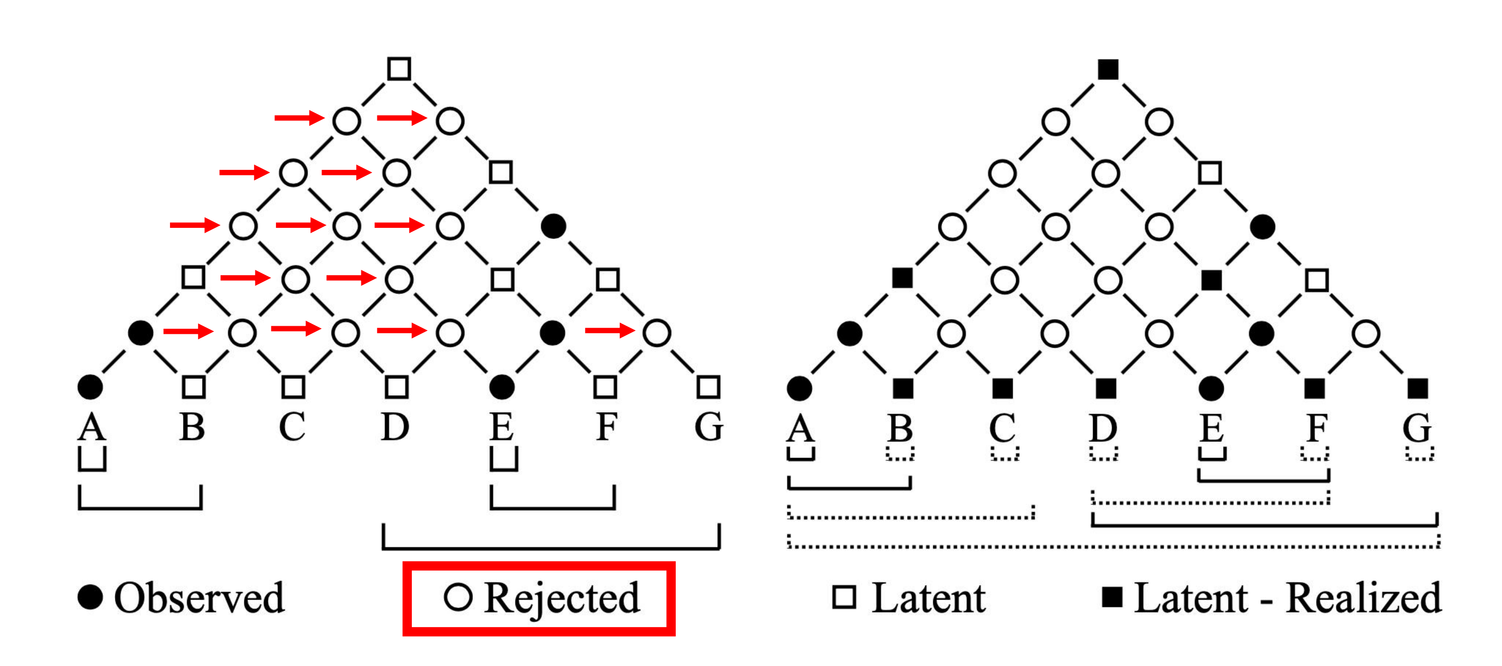
Different operations for different nodes in this DP summation

Likelihood Evaluation for Observed



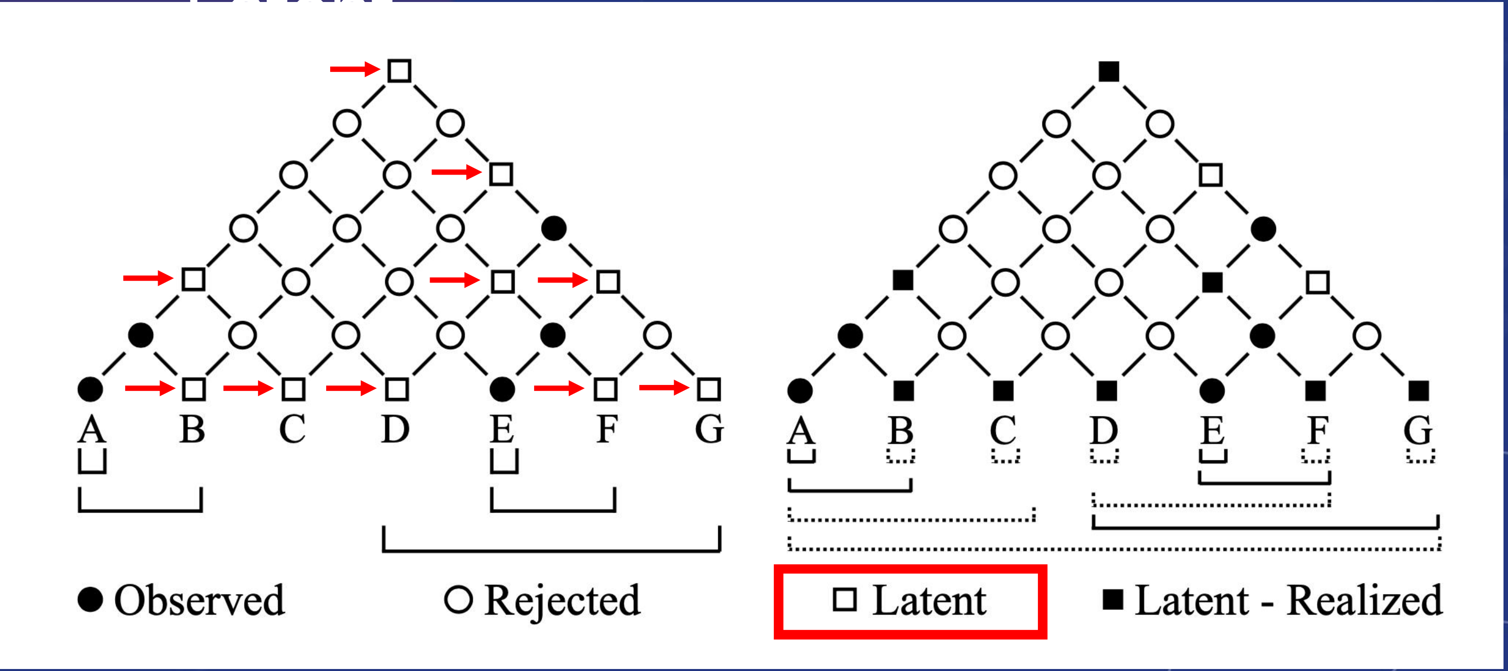
$$\beta_{ijk} = \exp(s_{ijk})\cdots$$

Rejection for Rejected



$$\beta_{ijk} = 0$$

Marginalization for the Latent



$$\beta_{ijk} = \sum_{k \in \mathscr{L}_l} \exp(s_{ijk}) \cdots$$

Unify Operations with Mask

$$\beta_{ijk} = \exp(s_{ijk})\cdots$$

Likelihood Evaluation

$$\beta_{ijk} = 0$$

Rejectio
n

$$\beta_{ijk} = \sum_{k \in \mathscr{L}_l} \exp(s_{ijk})\cdots$$

Partial Marginalization

$$\beta_{ijk} = \sum_{k} m_{ijk} \cdot \exp(s_{ijk})\cdots$$

A uniform masked summation

$$m_{ijk} = 1, m_{ijk'} = 0$$

k observed tag, k' all other tags

Likelihood Evaluation

$$\forall k, m_{ijk} = 0$$

k all tags

Rejectio
n

$$\forall k_1, m_{ijk_1} = 1, \forall k_2, m_{ijk_2} = 0$$

k_1 latent tags, k_2 observed tags

Partial Marginalization

Compared with the original Inside

$$\beta_{ijk} = \sum_k \exp(s_{ijk}) \cdots$$

Sum over all possible trees

$$\beta_{ijk} = \sum_k m_{ijk} \cdot \exp(s_{ijk}) \cdots$$

Sum over all possible full trees compatible with a partial tree

Only difference is the mask term

Masked Inside

for $d \leftarrow 1$ to $n-1$ **do**

Parallelization on i, tensor operation on l, k, k_1, k_2

$$1 \leq i \leq n-d; \quad j = i + d; \quad k, k_1, k_2 \in \{1, ..., |\mathcal{L}|\}$$

$\beta[i, j, k] = (\underline{M[i, j, k]} \exp(s_{ijk})) \cdot$ ▷ Masked scores

$$\sum_{l=i}^{j-1} \sum_{k_1, k_2 \in \mathcal{L}} \beta[i, l, k_1] \beta[l+1, j, k_2]$$

Return: $s(T) = \log(\sum_{k \in \mathcal{L}} \beta[1, n, k])$

One single line change of the original Inside algorithm
Unify the DP graph for sentences with different partial trees

Reuse recent efficient bachification and tensorization works for the original Inside. We use Torch-Struct

At the end of the day

Algorithm 2 INSIDE FOR PARTIAL MARGINALIZATION

Input: Scores s, partial tree T and its corresponding $\bar{T}$
for $i \leftarrow 1$ to n **do**
    if $\bar{T}[i,i] = \bullet$ **then** $\triangleright$ Observed leaf
        $\exists k \in \mathcal{L}_o, T_{iik} = 1, \beta[i,i,k] = \exp(s_{iik})$
        $\forall m \neq k, \beta[i,i,m] = 0$
    else if $\bar{T}[i,i] = \square$ **then** $\triangleright$ Latent leaf
        $\forall k \in \mathcal{L}_o, \beta[i,i,k] = 0$
        $\forall k \in \mathcal{L}_l, \beta[i,i,k] = \exp(s_{iik})$
for $d \leftarrow 1$ to $n-1$ **do**
    for $i \leftarrow 1$ to $n-d$ **do**
        $j = i + d$
        if $\bar{T}[i,j] = \bullet$ **then** $\triangleright$ Observed
            $\exists k \in \mathcal{L}_o, T_{ijk} = 1$
            $\beta[i,j,k] = \exp(s_{ijk}) \cdot \sum_{l=i}^{j-1} \sum_{k_1,k_2 \in \mathcal{L}} \beta[i,l,k_1]\beta[l+1,j,k_2]$
            $\forall m \neq k, \beta[i,j,m] = 0$
        else if $\bar{T}[i,j] = \square$ **then** $\triangleright$ Latent
            $\forall k \in \mathcal{L}_l, \beta[i,j,k] = \exp(s_{ijk}) \cdot \sum_{l=i}^{j-1} \sum_{k_1,k_2 \in \mathcal{L}} \beta[i,l,k_1]\beta[l+1,j,k_2]$
            $\forall k \in \mathcal{L}_o, \beta[i,j,k] = 0$
        else if $\bar{T}[i,j] = \circ$ **then** $\triangleright$ Rejected
            $\forall k \in \mathcal{L}, \beta[i,j,k] = 0$
if $\bar{T}[1,n] = \bullet$ **then** $\triangleright$ Observed root
    $\exists k \in \mathcal{L}_o, T_{1nk} = 1$. Return $s(T) = \beta[1,n,k]$
else if $\bar{T}[1,n] = \square$ **then** $\triangleright$ Latent root
    Return $s(T) = \log(\sum_{k \in \mathcal{L}_l} \beta[1,n,k])$

$$
\begin{aligned}
s(T) &= \textsc{MaskedInside}(s, M) \\
&= \textsc{Inside}(\log M + s)
\end{aligned}
$$

Turn a conceptually complicated, practically inefficient partial marginalization algorithm into a simple and efficient Masked Inside

Performance

Model	ACE2004 P	ACE2004 R	ACE2004 F1	ACE2005 P	ACE2005 R	ACE2005 F1	GENIA P	GENIA R	GENIA F1
LSTM-CRF (Lample et al. 2016)	71.3	50.5	58.3	70.3	55.7	62.2	75.2	64.6	69.5
FOFE(c=6) (Xu et al. 2017)	68.2	54.3	60.5	76.5	66.3	71.0	75.4	67.8	71.4
Transition (Wang et al. 2018)	74.9	71.8	73.3	74.5	71.5	73.0	78.0	70.2	73.9
Cascaded-CRF (Ju et al. 2018)	-	-	-	74.2	70.3	72.2	78.5	71.3	74.7
SH(c=n) (Wang and Lu 2018)	77.7	72.1	74.5	76.8	72.3	74.5	77.0	73.3	75.1
ML (Fisher and Vlachos 2019)	-	-	-	75.1	74.1	74.6	-	-	-
BENSC (Tan et al. 2020)	78.1	72.8	75.3	77.1	74.2	75.6	78.9	72.7	75.7
Pyramid (Jue et al. 2020)	81.1	79.4	80.3	80.0	78.9	79.4	78.6	77.0	77.8
with Pretrained LM									
MGNER (ELMo) (Xia et al. 2019)	81.7	77.4	79.5	79.0	77.3	78.2	-	-	-
ML (ELMo) (Fisher and Vlachos 2019)	-	-	-	79.7	78.0	78.9	-	-	-
ML (BERT) (Fisher and Vlachos 2019)	-	-	-	82.7	82.1	82.4	-	-	-
Seq2seq (Straková, Straka, and Hajic 2019)	-	-	84.3	-	-	83.4	-	-	78.2
BENSC (BERT) (Tan et al. 2020)	85.8	84.8	85.3	83.8	83.9	83.9	79.2	77.4	78.3
Pyramid (BERT) (Jue et al. 2020)	86.1	86.5	86.3	84.0	85.4	84.7	79.5	78.9	79.2
with Additional Supervision									
DYGIE (Luan et al. 2019)	-	-	84.7	-	-	82.9	-	-	76.2
Yu, Bohnet, and Poesio (2020)	87.3	86.0	86.7	85.2	85.6	85.4	81.8	79.3	80.5
BERT-MRC (Li et al. 2020)	85.0	86.3	86.0	87.2	86.6	86.9	85.2	81.1	83.8
PO-TreeCRFs (ours)	86.7 ±0.4	86.5 ±0.4	86.6 ±0.3	84.5 ±0.4	86.4 ±0.2	85.4 ±0.1	78.2 ±0.7	78.2 ±0.8	78.2 ±0.1
PO-TreeCRFs Ablation Study									
Change Biaffine to Bilinear	86.0	86.7	86.4	83.0	86.5	84.7	79.9	75.5	77.6
W/o. Structure Smoothing	86.1	86.4	86.2	83.5	85.8	84.6	78.7	76.5	77.6
W/o. Potential Normalization and Structure Smoothing	86.0	85.3	85.7	82.7	86.2	84.4	76.5	78.1	77.3
W/o. TreeCRFs	84.4	85.4	84.9	82.0	86.4	84.1	80.5	74.5	77.4

Table 2: Main results and ablation studies on three datasets. We report the average scores of 5 runs for main results.

Time Complexity

Method	Inside (Vanilla)	MASKED INSIDE	Biaffine
GPU Time	14m58s	3m20s	2m27s
CPU Time	2h5m	24m	22m10s
Complexity	$O(n^3)$	$O(n \log n)$	$O(1)$

Conclusion

A method using partially-observed TreeCRFs for nested NER

Key contribution is about efficient inference

- Construct masks to unify different inference operations
- Replace original partial marginalization algorithm with Masked Inside algorithm

Conclusion

Code: https://github.com/FranxYao/Partially-Observed-TreeCRFs

Any questions, please contact:
chuanqi.tcq@alibaba-inc.com
chenmosha.cms@alibaba-inc.com

Thanks!