Still working on POS-tagging?

Convex Approximation for Parallel Marginal Inference in CRF

Joseph Le Roux

09/12/25

Outline

- Intro
- Sequence Tagging & Structured Prediction
- Bregman Projection for POS-tagging
- Conclusion

Topic

- Intro
- Sequence Tagging & Structured Prediction
- Bregman Projection for POS-tagging
- Conclusion

Structured Prediction

Many tasks in NLP can be framed as Structured Prediction

Umbrella Definition

- Mostly supervised learning methods (enough problems already)
- Predicted labels are *structured*: graphs (trees, arborescences, chains. . .)
- Amounts to:
  - Predicting/Scoring parts of the structure (edge, arcs, subtrees of depth d . . .)
  - Enforcing (global) well-formedness constraints

NLP Applications

- Tagging
- Segmentation
- Parsing
- Relation Extraction, Translation. . .

Structured Prediction

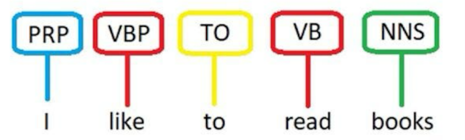
Many tasks in NLP can be framed as Structured Prediction

Umbrella Definition

- Mostly supervised learning methods (enough problems already)
- Predicted labels are *structured*: graphs (trees, arborescences, chains. . .)
- Amounts to:
  - Predicting/Scoring parts of the structure (edge, arcs, subtrees of depth d . . .)
  - Enforcing (global) well-formedness constraints

NLP Applications

- Tagging



Structured Prediction

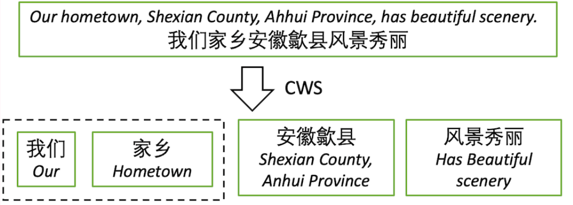
Many tasks in NLP can be framed as Structured Prediction

Umbrella Definition

- Mostly supervised learning methods (enough problems already)
- Predicted labels are *structured*: graphs (trees, arborescences, chains. . .)
- Amounts to:
  - Predicting/Scoring parts of the structure (edge, arcs, subtrees of depth d . . .)
  - Enforcing (global) well-formedness constraints

NLP Applications

- Segmentation



Structured Prediction

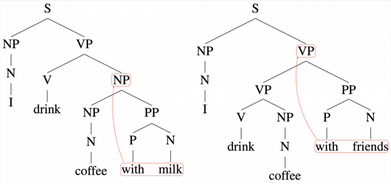
Many tasks in NLP can be framed as Structured Prediction

Umbrella Definition

- Mostly supervised learning methods (enough problems already)
- Predicted labels are *structured*: graphs (trees, arborescences, chains...)
- Amounts to:
  - Predicting/Scoring parts of the structure (edge, arcs, subtrees of depth d ...)
  - Enforcing (global) well-formedness constraints

NLP Applications

- Parsing



Why Sequence Tagging?

Essential NLP Task

- Part-Of-Speech tagging *[Chu88]*
- Named Entity Recognition *[RM95]*
- Token Segmentation *[Xue03]*
- Syntactic and Semantic Parsing *[FX09; MFT17; AC22]*

Machine Learning

- probabilistic graphical models (MRF/CRF) *[WJ08]*
- generalization to Perceptron/SVM *[Col02; Tas04]*

Questions

Can we have structured models that can be implemented efficiently on GPUs (parallelization)?

- joint work with Caio Corro, Mathieu Lacroix

Topic

- Intro
- Sequence Tagging & Structured Prediction
- Bregman Projection for POS-tagging
- Conclusion

Problem Definition

Definition

- Given a sentence $x_i \ldots x_n$ where $x_i \in \mathcal{V}$ is a word
- Find a tag for each word, that is find $t_1 \ldots t_n$ with $t_i \in \mathcal{T} = \{1 \ldots, T\}$
- Represent each $t_i \in \mathcal{T}$ by a one-hot vector $y_i \in \{0, 1\}^T$ and concatenate all y_i in a vector $y \in \{0, 1\}^{nT}$

Probabilistic Model

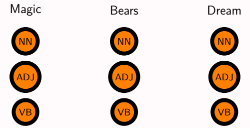
Define a parametrized conditional model
$p(\mathbf{y}|\mathbf{x}) = p_\theta(y_1 \ldots y_n | x_1 \ldots x_n)$

Tagging amounts to finding the most probable tag sequence

Learning θ can be implemented as MLE: Maximum Likelihood Estimation

Can we find such a model which is both accurate and efficient?

Example



Factorized (Unigram) Models

Definition

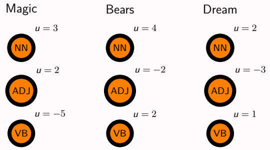
$$p_\theta(y_1 \dots y_n|\boldsymbol{x}) = \prod_i p_\theta(y_i|\boldsymbol{x}) = \prod_i \frac{\exp\langle u(\boldsymbol{x}, i; \theta), y_i\rangle}{\sum_{y_i'} \exp\langle u(\boldsymbol{x}, i; \theta), y_i'\rangle}$$

Assume independance between tags (ignore interactions/correlation) not really structured prediction!

Typical implementation

- Transformer to compute feature representation h_i from $\boldsymbol{x}$
- Then MLP+softmax to compute u scores from h_i

Example



Tagging and Learning with Factorized Models

Tagging

$$\begin{aligned}\operatorname*{argmax}_{y_1,\dots,y_n} p(y_1 \dots y_n|\boldsymbol{x}) &= \operatorname*{argmax}_{y_1,\dots,y_n} \log p(y_1 \dots y_n|\boldsymbol{x}) \\ &= \operatorname*{argmax}_{y_1,\dots,y_n} \log \prod_i \frac{\exp\langle u(\boldsymbol{x}, i; \theta), y_i\rangle}{Z_i} \\ &= \operatorname*{argmax}_{y_1,\dots,y_n} \sum_i \langle u(\boldsymbol{x}, i; \theta), y_i\rangle \\ &= (\operatorname*{argmax}_{y_1} \langle u(\boldsymbol{x}, 1; \theta), y_1\rangle, \dots, \operatorname*{argmax}_{y_n} \langle u(\boldsymbol{x}, n; \theta), y_n\rangle)\end{aligned}$$

Argmax at each position simple and parallelizable

Learning

$$\min_\theta -L(\theta; \boldsymbol{x}, \boldsymbol{y}) = \min_\theta -\log p_\theta(y_1 \dots y_n|\boldsymbol{x}) = \min_\theta \sum_i -\log p(y_i|\boldsymbol{x}; \theta)$$

Negative Log-Likelihood Loss at each position parallelization

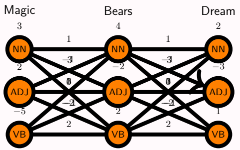
Linear-Chain CRF (1)

Definition

$$p_\theta(y_1 \dots y_n|\boldsymbol{x}) = \frac{\exp\left(\sum_i \left(\langle u(\boldsymbol{x}, i; \theta), y_i\rangle + y_i^\top b(\boldsymbol{x}, i; \theta) y_{i+1}\right)\right)}{\sum_{y'_1 \dots y'_n} \exp\left(\sum_i \left(\langle u(\boldsymbol{x}, i; \theta), y'_i\rangle + y_i'^\top b(\boldsymbol{x}, i; \theta) y'_{i+1}\right)\right)}$$

- u implements a unigram model
- b *links* adjacent labels together (hence *chain*)
- normalized to output a valid probability distribution

Example



Remarks

- can be extended to take into account triplets of labels. . .
- u can be integrated into b
- if b is zero then we have a factorized model

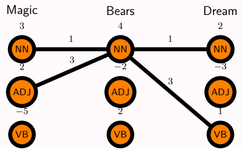
Linear-Chain CRF (1)

Definition

$$p_\theta(y_1 \dots y_n | \mathbf{x}) = \frac{\exp\left(\sum_i \left(\langle u(\mathbf{x}, i; \theta), y_i \rangle + y_i^\top b(\mathbf{x}, i; \theta) y_{i+1}\right)\right)}{\sum_{y'_1 \dots y'_n} \exp\left(\sum_i \left(\langle u(\mathbf{x}, i; \theta), y'_i \rangle + y_i'^\top b(\mathbf{x}, i; \theta) y'_{i+1}\right)\right)}$$

- u implements a unigram model
- b *links* adjacent labels together (hence *chain*)
- normalized to output a valid probability distribution

Example



Remarks

- can be extended to take into account triplets of labels...
- u can be integrated into b
- if b is zero then we have a factorized model

Linear-Chain CRF (2)

From Quadratic to Linear

- so far a quadratic cost function

$$\sum_i \langle u(\boldsymbol{x}, i; \theta), y_i \rangle + {y_i}^\top b(\boldsymbol{x}, i; \theta) y_{i+1} = \sum_i {y_i}^\top w(\boldsymbol{x}, i; \theta) y_{i+1}$$

- quadratic since w returns scores for pairs of tags.

- Use linearization: instead of tags, bigrams:

$$y_{i,t,t'} = 1 \text{ iff } y_{i,t} = 1 \text{ and } y_{i+1,t'} = 1$$

- Cost function becomes linear:

$$\sum_i \langle w(\boldsymbol{x}, i; \theta), y_i \rangle = \langle w(\boldsymbol{x}; \theta), \boldsymbol{y} \rangle = \langle \boldsymbol{w}, \boldsymbol{y} \rangle$$

Well-formedness constraints

Valid sequence $y_1 \ldots y_n$ must obey:

1/ one bigram at each position (one-hot)

$$\forall i, \quad \sum_{t,t'} y_{i,t,t'} = 1$$

2/ flow constraints so that bigrams are connected

$$\forall i, t \quad \sum_s y_{i,s,t} = \sum_u y_{i+1,t,u}$$

For sentence $\boldsymbol{x}$, the set of well-formed label sequences denoted $\mathcal{Y}_x$

Finding the best tagging

Best tagging: longest path problem

$$\operatorname*{argmax}_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x};\theta) = \operatorname*{argmax}_{y_1\dots y_n \in \mathcal{Y}_{\boldsymbol{x}}} \langle \boldsymbol{w}, \boldsymbol{y} \rangle$$

Each sequence in $\mathcal{Y}_x$ is a path from a tag in position 1 to a tag in position n.

- the highest cost corresponds to a longest path in a DAG.
- Viterbi algorithm [Vit67]
  - computes iteratively best path from position 1 to position $1 \leq i \leq n$
  - Time complexity $O(|T|^2 n)$ (n steps of evaluating T choices for T nodes)
- Not parallelizable

Dynamic Programming

$$\begin{aligned}
c_{1,t}(\boldsymbol{w}) &\triangleq 0, \\
c_{i,t}(\boldsymbol{w}) &\triangleq \max\left(\left[c_{i-1,t'}(\boldsymbol{w}) + \boldsymbol{w}_{i,t',t}\right]_{t' \in T} \right), \\
c &\triangleq \max\left(\left[c_{n,t}(\boldsymbol{w})\right]_{t \in T} \right).
\end{aligned}$$

Learning a tagger

Maximum Likelihood Estimation

$$\max_\theta \log p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \max_\theta \log \frac{\exp\langle \boldsymbol{w}, \boldsymbol{y}\rangle}{\sum_{\boldsymbol{y}'\in\mathcal{Y}_{\boldsymbol{x}}} \exp\langle \boldsymbol{w}, \boldsymbol{y}'\rangle} = \max_\theta \log \frac{\exp\langle \boldsymbol{w}, \boldsymbol{y}\rangle}{Z(\boldsymbol{x};\theta)} \quad = \max_\theta \langle \boldsymbol{w}, \boldsymbol{y}\rangle - \log \sum_{\boldsymbol{y}'\in\mathcal{Y}_{\boldsymbol{x}}} \exp\langle \boldsymbol{w}, \boldsymbol{y}'\rangle$$

- First term easy to compute: a dot product
- Second term difficult:
  - sum over exponential number of paths
  - remark that *logsumexp* is a soft version of *max* (with entropy regularization) [MB18]
  - adapt the viterbi algorithm: replace *max* with *logsumexp*

$$\begin{aligned} c_{1,t}(\boldsymbol{w}) &\triangleq 0, \\ c_{i,t}(\boldsymbol{w}) &\triangleq \text{logsumexp}\left(\left[c_{i-1,t'}(\boldsymbol{w}) + \boldsymbol{w}_{i,t',t}\right]_{t'\in T} \right), \\ c &\triangleq \text{logsumexp}\left(\left[c_{n,t}(\boldsymbol{w})\right]_{t\in T} \right). \end{aligned}$$

- Not parallelizable

Parallel CRF via Batchify Training/Decoding

Inspiration from RNNs

Parallelize at the batch level

- each input is computed sequentially
- process position i of all sentences in the batch in parallel

Limited Parallelization

- Sentences in batch must have the same length to be efficient
- Still sequential in the end (ie RNNs vs. Transformers)
  - the asymptotic complexity is the same (but faster in practice)

Parallel CRF via Mean-Field Approximation (1)

Approximate the linear-chain model with a factorized Model

$$\widehat{r}(\boldsymbol{y}|\boldsymbol{x}) = \operatorname*{argmin}_r D_{KL}[r|p]$$

where

- p is the linear chain distribution
- r is a factorized model $r(\boldsymbol{y}|\boldsymbol{x}) = \prod_i (y_i|\boldsymbol{x})$
- D_{KL} is the Kullback-Liebler divergence: $D_{KL}[r|p] = \sum_{\boldsymbol{y}} r(\boldsymbol{y}|x) \log \frac{r(\boldsymbol{y}|x)}{p(\boldsymbol{y}|x)}$

Algorithms for MF

- computing $\widehat{r}$ is a non-convex optimization problem (with simplex constraints)
- coordinate ascent gives an iterative method (local optimum) based on iterative message passing
- [Wan+20] show that this method can be performed for all positions in parallel instead of sequentially
  - convergence guarantees are lost (can diverge!), ok in practice

Parallel CRF via Mean-Field Approximation (2)

Learn through MF [Dom08], for tagging [Wan+20]

- Message Passing algorithm for MF = built from differentiable operations
- $\rightarrow$ can be interpreted as a *layer* of the neural network [Dia+17; MLE19]
- the model can be learned end-to-end

Summary

- can learn and predict efficiently (in parallel at each position)
- at the expense of factorized approximation
- $\rightarrow$ cannot take into account interactions (eg, forbid a transition)

Neural Architecture

In our experiments

Feature extractions for each input position

- word embeddings (768)
- N Transformer Encoders ($N = 2$)
- for each position i a vector h_i

Unigram logits

- MLP to compute a vector of T scores from h_1

Bigram Logits

- MLP to compute a vector of T^2 scores from h_1
- (could be a $T \times T$ matrix shared at all positions)

Some Results

UD Corpus Performance

	Dutch	English	French	German
CRF	94.7	91.9	96.2	94.3
MF10	94.5	91.0	95.8	94.2
Unigram	93.4	90.8	96.0	94.0

- CRF improves performance
- but unigram and mean-field are close behind

UD Tagging Speed-up

	Dutch	English	French	German
CRF	$\times 1.0$	$\times 1.0$	$\times 1.0$	$\times 1.0$
MF10	$\times 6.6$	$\times 8.0$	$\times 10.4$	$\times 8.5$
Unigram	$\times 8.8$	$\times 9.8$	$\times 11.9$	$\times 9.9$

- MF and Unigram 1 order of magnitude faster than CRF

Topic

- Intro
- Sequence Tagging & Structured Prediction
- **Bregman Projection for POS-tagging**
- Conclusion

From Linear to Convex Programming

What is difficult for MLE with LC-CRF?

- Computing the log-partition expressed as sum of an exponential number of chains:

$$\log Z(\boldsymbol{x};\theta) = \log \sum_{\boldsymbol{y}\in\mathcal{Y}_{\boldsymbol{x}}} \exp\langle \boldsymbol{w}, \boldsymbol{y}\rangle$$

Log-partition as an optimization problem: Marginal Inference

Use connection between *logsumexp* and entropy-regularized *max* expectation (proof on blackboard?)

$$\log Z(\boldsymbol{x};\theta) = \max_{p\in\triangle(\mathcal{Y}_{\boldsymbol{x}})} \mathbb{E}_{\boldsymbol{y}\sim p}\left[\langle \boldsymbol{w}, \boldsymbol{y}\rangle\right] + H(p)$$

- where H is the Shannon Entropy $\quad -\sum_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x}) \log p(\boldsymbol{y}|\boldsymbol{x})$
- Intractable in general (MRF) $\rightarrow$ requires approximations
- here tractable, but approximation to *parallelize*

Marginal Inference (1)

$$
\begin{aligned}
F(p) &= \mathbb{E}_{\boldsymbol{y}\sim p}\left[\langle \boldsymbol{w}, \boldsymbol{y}\rangle\right] + H(p) \\
&= \sum_{\boldsymbol{y}} p(\boldsymbol{y})\langle \boldsymbol{w}, \boldsymbol{y}\rangle - \sum_{\boldsymbol{y}} p(\boldsymbol{y}) \log p(\boldsymbol{y}) \text{ s.c. } \sum_{y} p(\boldsymbol{y}) = 1
\end{aligned}
$$

We dualize the sum-to-1 constraint (positivity is still implicit)

$$
\begin{aligned}
L(p, \lambda) &= \sum_{\boldsymbol{y}} p(\boldsymbol{y})\langle \boldsymbol{w}, \boldsymbol{y}\rangle - \sum_{\boldsymbol{y}} p(\boldsymbol{y}) \log p(\boldsymbol{y}) + \lambda(1 - \sum_{y} p(\boldsymbol{y})) \\
&= \sum_{\boldsymbol{y}} p(\boldsymbol{y})\big(\langle \boldsymbol{w}, \boldsymbol{y}\rangle - \log p(\boldsymbol{y}) - \lambda\big) + \lambda \text{ unconstrained.}
\end{aligned}
$$

Marginal Inference (2)

We compute derivatives and solve for zero

$$\frac{\partial L}{\partial p(y)} = \langle \boldsymbol{w}, \boldsymbol{y} \rangle - \log p(\boldsymbol{y}) - \lambda - \frac{p(\boldsymbol{y})}{p(\boldsymbol{y})} = \langle \boldsymbol{w}, \boldsymbol{y} \rangle - \log p(\boldsymbol{y}) - \lambda - 1$$

$$\Rightarrow p^*(\boldsymbol{y}) = \frac{\exp\langle \boldsymbol{w}, \boldsymbol{y} \rangle - 1}{\exp \lambda}$$

$$\frac{\partial L}{\partial \lambda} = 1 - \sum_{y} p(\boldsymbol{y})$$

$$\Rightarrow \sum_{y} \frac{\exp\langle \boldsymbol{w}, \boldsymbol{y} \rangle - 1}{\exp \lambda} = 1$$

$$\Rightarrow p^*(\boldsymbol{y}) = \frac{\exp\langle \boldsymbol{w}, \boldsymbol{y} \rangle - 1}{\sum_{\boldsymbol{y}'} \exp\langle \boldsymbol{w}, \boldsymbol{y}' \rangle - 1}$$

$$\Rightarrow p^*(\boldsymbol{y}) = \frac{\exp\langle \boldsymbol{w}, \boldsymbol{y} \rangle}{\sum_{\boldsymbol{y}'} \exp\langle \boldsymbol{w}, \boldsymbol{y}' \rangle}$$

The optimal distribution is the softmax distribution!

Marginal Inference (3)

Now we compute $F(p)$ for $p = p^*$:

$$
\begin{aligned}
F(p^*) &= \mathbb{E}_{\boldsymbol{y}\sim p^*}\left[\langle \boldsymbol{w}, \boldsymbol{y}\rangle\right] + H(p^*) \\
&= (\sum_{\boldsymbol{y}} p^*(\boldsymbol{y})\langle \boldsymbol{w}, \boldsymbol{y}\rangle) - (\sum_{y} p^*(\boldsymbol{y}) \log p^*(\boldsymbol{y})) \\
&= \sum_{\boldsymbol{y}} p^*(\boldsymbol{y})(\langle \boldsymbol{w}, \boldsymbol{y}\rangle - \langle \boldsymbol{w}, \boldsymbol{y}\rangle - \log Z) \\
&= \sum_{\boldsymbol{y}} p^*(\boldsymbol{y}) \log Z \\
&= \log Z \sum_{\boldsymbol{y}} p^*(\boldsymbol{y}) \\
&= \log Z
\end{aligned}
$$

This concludes the proof of equivalence

Mean Regularization: Definition

Approximate the search space of Marginal Inference

$$\max_{p \in \triangle(\mathcal{Y}_{\boldsymbol{x}})} \mathbb{E}_{\boldsymbol{y} \sim p}\left[\langle \boldsymbol{w}, \boldsymbol{y} \rangle\right] + H(p) = \max_{p \in \triangle(\mathcal{Y}_{\boldsymbol{x}})} \langle p, Y_{\boldsymbol{x}} \boldsymbol{w} \rangle + H(p) = \max_{p \in \triangle(\mathcal{Y}_{\boldsymbol{x}})} \langle p Y_{\boldsymbol{x}}^{\top}, \boldsymbol{w} \rangle + H(p)$$

$$= \max_{q \in \triangle(conv(\mathcal{Y}_{\boldsymbol{x}}))} \langle q, \boldsymbol{w} \rangle + R(q)$$

$$\approx \max_{q \in \triangle(conv(\mathcal{Y}_{\boldsymbol{x}}))} \langle q, \boldsymbol{w} \rangle + H(q)$$

Intuition

- Compute scores of all y and then compute mean $\rightarrow$ Compute mean of all y and then compute score

Pros and Cons:

$+$ work on a simpler search space (from $|\mathcal{Y}_{\boldsymbol{x}}|$ to $|y|$ dimensions)

$-$ Regularization term becomes difficult to express

Mean Regularization

Replace complicated regularization with a simpler one: entropy!

Mean Regularization: Prediction and Learning

How can we use Marginal Inference for MAP/Decoding?

Since searching over the convex hull is the same (integer polytope, no interior point) as the original MAP

$$\underset{y \in \mathcal{Y}_x}{\operatorname{argmax}} \langle \boldsymbol{w}, y \rangle = \underset{y \in conv(\mathcal{Y}_x)}{\operatorname{argmax}} \langle \boldsymbol{w}, y \rangle$$

$\Rightarrow$ only difference between MAP and marginal inference is entropy

Marginal Inference and Decoding are the same in the limit

$$\lim_{\tau \to 0} \max_{\boldsymbol{y} \in conv(\mathcal{Y}_x)} \langle \boldsymbol{w}, \boldsymbol{y} \rangle + \tau H(\boldsymbol{y}) = \max_{\boldsymbol{y} \in conv(\mathcal{Y}_x)} \langle \boldsymbol{w}, \boldsymbol{y} \rangle$$

#1. Could set τ to zero a) run Viterbi but non-parallelizable or b) use our methods (cf. next slide) but unstable. Set τ to small value (ie 10^{-3}), run marginal inference (cf. next) to get q and bigram marginals.

- At each position compute tag marginal $q(i,t) = \sum_{t'} q(i,t,t')$ and pick highest one
- Similar to Minimum Bayes Risk decoding [GB00; Goo99]

Learning

Simply replace $\log Z$ by mean regularization approximation in MLE.

$$\min_{\theta} -\log p(\boldsymbol{y}|\boldsymbol{x};\theta) \approx \min_{\theta} \left(\max_{\boldsymbol{q} \in \triangle(conv(\mathcal{Y}_{\boldsymbol{x}}))} \langle \boldsymbol{q}, \boldsymbol{w} \rangle + H(\boldsymbol{q}) - \langle \boldsymbol{w}, \boldsymbol{y} \rangle \right)$$

Mean Regularization: KL Projections

From Marginal Inference to KL Projections

$$\begin{aligned}
\underset{\boldsymbol{y}\in conv(\mathcal{Y}_x)}{\operatorname{argmax}} \langle \boldsymbol{w}, \boldsymbol{y}\rangle + \tau H(\boldsymbol{y}) &= \underset{\boldsymbol{y}\in conv(\mathcal{Y}_x)}{\operatorname{argmax}} \langle \tau^{-1}\boldsymbol{w}, \boldsymbol{y}\rangle + H(\boldsymbol{y}) \\
&= \underset{\boldsymbol{y}\in conv(\mathcal{Y}_x)}{\operatorname{argmin}} -\langle \tau^{-1}\boldsymbol{w}, \boldsymbol{y}\rangle - H(\boldsymbol{y}) \\
&= \underset{\boldsymbol{y}\in conv(\mathcal{Y}_x)}{\operatorname{argmin}} \langle \boldsymbol{y}, \log \boldsymbol{y}\rangle - \langle \boldsymbol{y}, \log \exp \tau^{-1}\boldsymbol{w}\rangle \\
&= \underset{\boldsymbol{y}\in conv(\mathcal{Y}_x)}{\operatorname{argmin}} D_{KL}[\boldsymbol{y} | \exp \tau^{-1}\boldsymbol{w}]
\end{aligned}$$

- problem *similar* to MF approximation (KL projection) but on different objects: do not assume factorized distribution!

Mean Regularization: Efficient KL Projections (1)

KL Projection seems hard

Convex optimization over a highly structured search space (convex hull of chains), but:

1. if we show that the search space can be expressed as an intersection of convex sets...
2. ... and that D_{KL} projections on each intersected set can be solved efficiently

$\Rightarrow$ We can use *Bregman Iterative Projection* to solve our problem!

$conv(\mathcal{Y})$ as intersection

$$\forall 2 \leq i \leq n-1 \qquad \mathcal{C}_i = \{\boldsymbol{y} \in \mathbb{R}_{\geq 0}^{(n-1)\times T \times T} | \sum_{t,t'} y_{i,t,t} = 1; \forall t \sum_{s} y_{i,s,t} = \sum_{u} y_{i+1,t,u}\}$$

- We can then write $conv(\mathcal{Y}) = \bigcap_i \mathcal{C}_i$
- Remark that we can also define intermediate sets:
  - $\mathcal{C}_{even} = \bigcap_i \mathcal{C}_{2i}, \quad \mathcal{C}_{odd} = \bigcap_i \mathcal{C}_{2i+1}$
  - And we can write $conv(\mathcal{Y}) = \mathcal{C}_{even} \cap \mathcal{C}_{odd}$

Mean Regularization: Efficient KL Projections (2)

Decomposition over simple sets or even/odd

Since projection $\min_{y \in conv(\mathcal{Y})} D_{KL}$ decomposes over arcs i, t, t', if a union of C_i have independent variables, we can process them in parallel. For instance we have:

$$\min_{y \in \mathcal{C}_{even}} D_{KL}[y|\exp \tau^{-1} w] = \sum \min_{y \in \mathcal{C}_i} D_{KL}[y|\exp \tau^{-1} w]$$

(each time we consider restriction of variables/scores relevant to the subspace)

Closed forms for projections on $\mathcal{C}_i$

For each position we solve:

- a restricted version of marginal inference
- with sum-to-one constraints
- with flow constraints

We dualize constraints and find closed-form solutions through KKT conditions:
$\Rightarrow$ Solving KL projections for $\mathcal{C}_i$, $\mathcal{C}_{odd}$, $\mathcal{C}_{even}$ is really fast!

Mean Regularization: Bregman Iterative Projection

Following [Ben+15] for optimal transport, we derive an algorithm to solve KL projections efficiently:

- Initialize: $y^{(0)} = \exp \tau^{-1} b$
- For $i = 0$ to I:
  - $y^{(n+\frac{1}{2})} \min_{y \in C_{even}} D_{KL}[y | y^{(n)}]$
  - $y^{(n+1)} \min_{y \in C_{odd}} D_{KL}[y | y^{(n+\frac{1}{2})}]$
- Iteratively project on $\mathcal{C}_{even}$ then $\mathcal{C}_{odd}$
- Converge to $\min_y D_{KL}[y | \exp \tau^{-1} b]$ as I goes to infinity (=10 in practice)

Some Results

UD Corpus Performance (CRF Training)

	Dutch	English	French	German
CRF	94.7	91.9	96.2	94.3
Bregman5/10	94.7	91.9	96.2	94.3
MF10	94.5	91.0	95.8	94.2
Unigram	93.4	90.8	96.0	94.0

- Bregman-CRF bridges the performance gap

UD Tagging Speed-up

	Dutch	English	French	German
CRF	$\times 1.0$	$\times 1.0$	$\times 1.0$	$\times 1.0$
Bregman10	$\times 4.7$	$\times 4.9$	$\times 7.0$	$\times 5.2$
MF10	$\times 6.6$	$\times 8.0$	$\times 10.4$	$\times 8.5$
Unigram	$\times 8.8$	$\times 9.8$	$\times 11.9$	$\times 9.9$

- slower than MF:

Topic

- Intro
- Sequence Tagging & Structured Prediction
- Bregman Projection for POS-tagging
- Conclusion

Conclusion (1)

Bregman Projection for Mean-Regularized CRF

Approximation of CRF

like MF not to be tractable, but to be parallelized

not like MF based on mean regularization

- Model converge to exact decoding when $\tau \to 0$
- Able to forbid specific transitions
- Algorithmic convergence (Bregman Iterative Projection)

Conclusion (2)

DL pipelines are a real challenge for structured prediction

- *bitter lesson* (simple models with lots of data are better than clever models)
- Use Approximations of exact decoding competitive in practice with unigrams
  - Mean-Field
  - Bregman CRF
- Designed with parallelization in mind

Bibliography I

[AC22] Afra Amini and Ryan Cotterell. "On Parsing as Tagging". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 8884–8900. DOI: 10.18653/v1/2022.emnlp-main.607. URL: https://aclanthology.org/2022.emnlp-main.607/.

[Ben+15] Jean-David Benamou et al. "Iterative Bregman Projections for Regularized Transportation Problems". In: *SIAM J. Sci. Comput.* 37.2 (2015). URL: https://doi.org/10.1137/141000439.

[Chu88] Kenneth Ward Church. "A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text". In: *Second Conference on Applied Natural Language Processing*. Austin, Texas, USA: Association for Computational Linguistics, Feb. 1988, pp. 136–143. DOI: 10.3115/974235.974260. URL: https://aclanthology.org/A88-1019/.

[Col02] Michael Collins. "Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms". In: *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*. Association for Computational Linguistics, July 2002, pp. 1–8. DOI: 10.3115/1118693.1118694. URL: https://aclanthology.org/W02-1001/.

Bibliography II

[Dia+17] Steven Diamond et al. "Unrolled Optimization with Deep Priors". In: *CoRR* abs/1705.08041 (2017). arXiv: 1705.08041. URL: http://arxiv.org/abs/1705.08041.

[Dom08] Justin Domke. "Learning Convex Inference of Marginals". In: *UAI 2008, Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, Helsinki, Finland, July 9-12, 2008*. Ed. by David A. McAllester and Petri Myllymäki. AUAI Press, 2008, pp. 137–144.

[FX09] Ji Feng and Qiu Xipeng. "A new Chinese Dependency Analysis Method Based On Sequence Labeling Model". In: *Computer Applications and Software*. Oct. 2009.

[GB00] Vaibhava Goel and William J. Byrne. "Minimum Bayes-risk automatic speech recognition". In: *Comput. Speech Lang.* 14.2 (2000), pp. 115–135. DOI: 10.1006/csla.2000.0138. URL: https://doi.org/10.1006/csla.2000.0138.

[Goo99] Joshua Goodman. "Semiring parsing". In: *Computational Linguistics* 25.4 (1999), pp. 573–606.

[MB18] Arthur Mensch and Mathieu Blondel. "Differentiable Dynamic Programming for Structured Prediction and Attention". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 3462–3471. URL: https://proceedings.mlr.press/v80/mensch18a.html.

Bibliography III

[MFT17] Diego Marcheggiani, Anton Frolov, and Ivan Titov. "A Simple and Accurate Syntax-Agnostic Neural Model for Dependency-based Semantic Role Labeling". In: *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*. Ed. by Roger Levy and Lucia Specia. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 411–420. DOI: 10.18653/v1/K17-1041. URL: https://aclanthology.org/K17-1041/.

[MLE19] Vishal Monga, Yuelong Li, and Yonina C. Eldar. "Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing". In: *CoRR* (2019). arXiv: 1912.10557 [eess.IV]. URL: http://arxiv.org/abs/1912.10557v3.

[RM95] Lance Ramshaw and Mitch Marcus. "Text Chunking using Transformation-Based Learning". In: *Third Workshop on Very Large Corpora*. 1995. URL: https://aclanthology.org/W95-0107/.

[Tas04] Ben Taskar. "Learning Structured Prediction Models: A Large Margin Approach". PhD thesis. Stanford University, 2004.

[Vit67] Andrew Viterbi. "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". In: *IEEE Transactions on Information Theory* 13.2 (Apr. 1967), pp. 260–269. ISSN: 1557-9654. DOI: 10.1109/tit.1967.1054010. URL: http://dx.doi.org/10.1109/TIT.1967.1054010.

Bibliography IV

[Wan+20] Xinyu Wang et al. “AIN: Fast and Accurate Sequence Labeling with Approximate Inference Network”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Bonnie Webber et al. Online: Association for Computational Linguistics, Nov. 2020, pp. 6019–6026. DOI: 10.18653/v1/2020.emnlp-main.485. URL: https://aclanthology.org/2020.emnlp-main.485.

[WJ08] Martin J Wainwright and Michael I Jordan. “Graphical models, exponential families, and variational inference”. In: *Foundations and Trendső in Machine Learning* 1.1–2 (2008), pp. 1–305.

[Xue03] Nianwen Xue. “Chinese Word Segmentation as Character Tagging”. In: *International Journal of Computational Linguistics & Chinese Language Processing, Volume 8, Number 1, February 2003: Special Issue on Word Formation and Chinese Language Processing*. Feb. 2003, pp. 29–48. URL: https://aclanthology.org/O03-4002/.