



Learning with Sparse Latent Structure

Vlad Niculae University of Amsterdam (PhD opening!)

Work with: Wilker Aziz, Mathieu Blondel, Claire Cardie, Gonalo M. Correia, Andr  Martins.

Rich Underlying Structure



A disastrous show of pompous and inconsequential gibberish, garish visuals and tedious storytelling

[themadmoviemane](#) 21 December 2019

I've got nothing against movie musicals, director Tom Hooper, or even anybody who's a part of making this film. But goodness me, *Cats* is an absolute monstrosity. Garish, non-sensical, boring and everything in between, it's a pompous and pointless musical that plays out with barely a redeeming feature, proving one of the most unbearable cinema experiences I've had in a very long time.

While I haven't been a big fan of Hooper's work in the past, particularly *Les Misérables*, *Cats* pales in comparison to anything the director has made before, failing on all levels in its pathetic attempts to provide even a semblance of fun, magical theatre, and instead staggering along through its repetitive and frankly tedious story on its way to a terrible ending that can't come soon enough.

Rich Underlying Structure



title

A disastrous show of pompous and inconsequential gibberish, garish visuals and tedious storytelling

[themadmoviemane](#) 21 December 2019

author

I've got nothing against movie musicals, director Tom Hooper, or even anybody who's a part of making this film. But goodness me, *Cats* is an absolute monstrosity. Garish, non-sensical, boring and everything in between, it's a pompous and pointless musical that plays out with barely a redeeming feature, proving one of the most unbearable cinema experiences I've had in a very long time.

date

While I haven't been a big fan of Hooper's work in the past, particularly *Les Misérables*, *Cats* pales in comparison to anything the director has made before, failing on all levels in its pathetic attempts to provide even a semblance of fun, magical theatre, and instead staggering along through its repetitive and frankly tedious story on its way to a terrible ending that can't come soon enough.

body

Rich Underlying Structure



segmentation:
sentences,
words,
and so on

A disastrous show of pompous and inconsequential gibberish, garish visuals and tedious storytelling

[themadmoviemane](#) 21 December 2019

I've got nothing against movie musicals, director Tom Hooper, or even anybody who's a part of making this film. But goodness me, *Cats* is an absolute monstrosity. Garish, non-sensical, boring and everything in between, it's a pompous and pointless musical that plays out with barely a redeeming feature, proving one of the most unbearable cinema experiences I've had in a very long time.

While I haven't been a big fan of Hooper's work in the past, particularly *Les Misérables*, *Cats* pales in comparison to anything the director has made before, failing on all levels in its pathetic attempts to provide even a semblance of fun, magical theatre, and instead staggering along through its repetitive and frankly tedious story on its way to a terrible ending that can't come soon enough.

Rich Underlying Structure



segmentation:
sentences,
words,
and so on

entities

A disastrous show of pompous and inconsequential gibberish, garish visuals and tedious storytelling

[themadmoviemane](#) 21 December 2019

I've got nothing against movie musicals, director [Tom Hooper](#) or even anybody who's a part of making this film. But goodness me, [Cats](#) is an absolute monstrosity. Garish, non-sensical, boring and everything in between, it's a pompous and pointless musical that plays out with barely a redeeming feature, proving one of the most unbearable cinema experiences I've had in a very long time.

While I haven't been a big fan of [Hooper](#)'s work in the past, particularly [Les Misérables](#), [Cats](#) pales in comparison to anything the director has made before, failing on all levels in its pathetic attempts to provide even a semblance of fun, magical theatre, and instead staggering along through its repetitive and frankly tedious story on its way to a terrible ending that can't come soon enough.

Rich Underlying Structure



relationships
e.g., dependency

A disastrous show of pompous and inconsequential gibberish, garish visuals and tedious storytelling

themadmoviemane 21 December 2019

I've got nothing against movie musicals, director Tom Hooper, or even anybody who's a part of making this film. But goodness me, *Cats* is an absolute monstrosity. Garish, non-sensical, boring and everything in between, it's a pompous and pointless musical that plays out with barely a redeeming feature, proving one of the most unbearable cinema experiences I've had in a very long time.

While I haven't been a big fan of Hooper's work in the past, particularly *Les Misérables*, *Cats* pales in comparison to anything the director has made before, failing on all levels in its pathetic attempts to provide even a semblance of fun, magical theatre, and instead staggering along through its repetitive and frankly tedious story on its way to a terrible ending that can't come soon enough.

Rich Underlying Structure



A disastrous show of pompous and inconsequential gibberish, garish visuals and tedious storytelling

[themadmoviemani](#) 21 December 2019

I've got nothing against movie musicals, director Tom Hooper, or even anybody who's a part of making this film. But goodness me, *Cats* is an absolute monstrosity. Garish, non-sensical, boring and everything in between, it's a pompous and pointless musical that plays out with barely a redeeming feature, proving one of the most unbearable cinema experiences I've had in a very long time.

While I haven't been a big fan of Hooper's work in the past, particularly *Les Misérables*, *Cats* pales in comparison to anything the director has made before, failing on all levels in its pathetic attempts to provide even a semblance of fun, magical theatre, and instead staggering along through its repetitive and frankly tedious story on its way to a terrible ending that can't come soon enough.

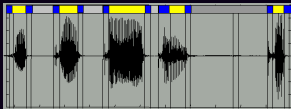
Most of this structure is **hidden**.

Rich Underlying Structure

Widely occurring pattern!

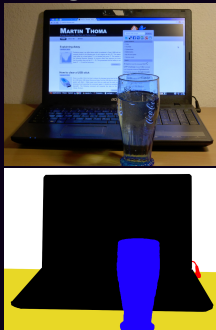
speech

(Andre-Obrecht, 1988)



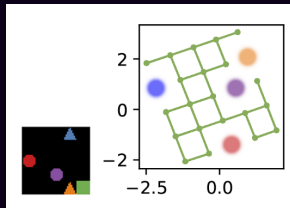
objects

(Long et al., 2015)



transition graphs

(Kipf, Pol, et al., 2020)

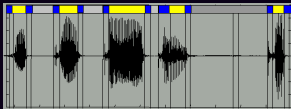


Rich Underlying Structure

Widely occurring pattern!

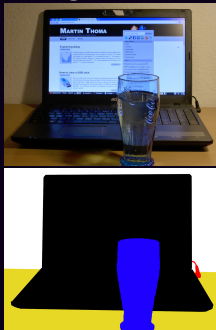
speech

(Andre-Obrecht, 1988)



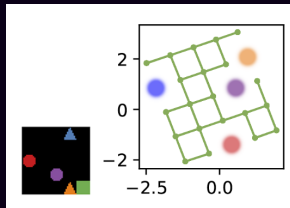
objects

(Long et al., 2015)



transition graphs

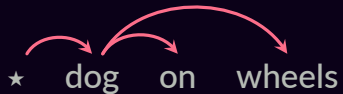
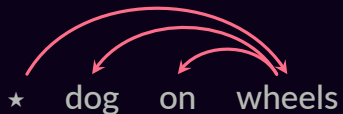
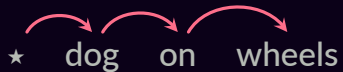
(Kipf, Pol, et al., 2020)



But we'll focus on NLP.

Structured Prediction

...

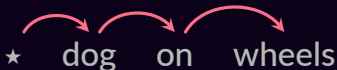


...

Structured Prediction

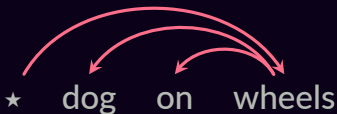
...

VERB PREP NOUN
dog on wheels



dog hond
on op
wheels wielen

NOUN PREP NOUN
dog on wheels



dog hond
on op
wheels wielen

NOUN DET NOUN
dog on wheels



dog hond
on op
wheels wielen

...

Structured Prediction



Traditional Pipeline Approach

input



output



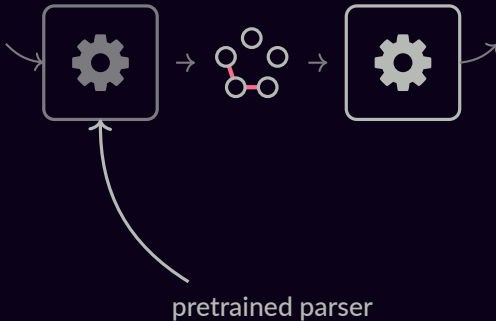
positive

neutral

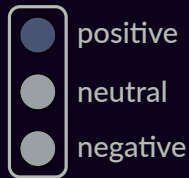
negative

Traditional Pipeline Approach

input

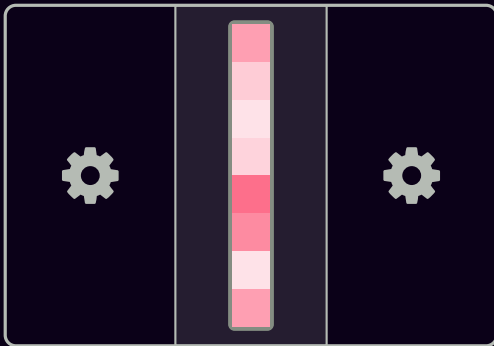


output

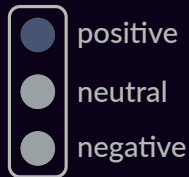


Deep Learning & Hidden Representations

input

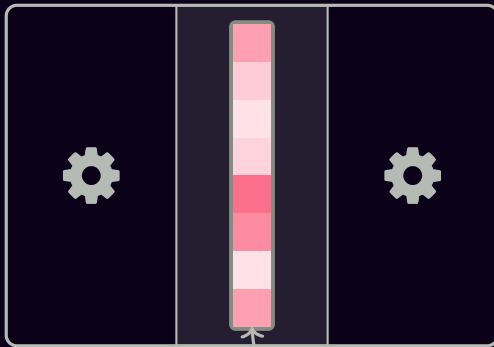


output



Deep Learning & Hidden Representations

input



output



positive

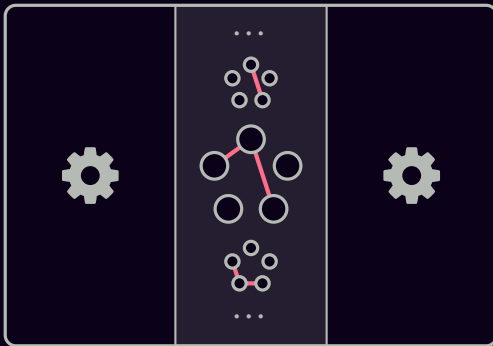
neutral

negative

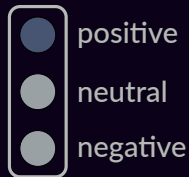
dense vector

Latent Structure Models

input



output



record scratch

freeze frame

**How to select an item
from a set?**

How to select an item from a set?



...



How to select an item from a set?

c_1

c_2

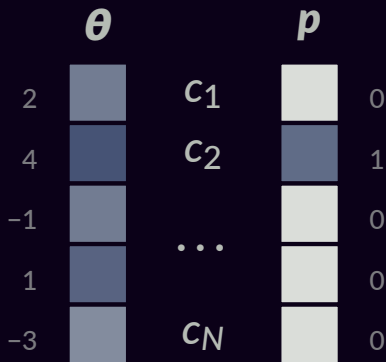
...

c_N

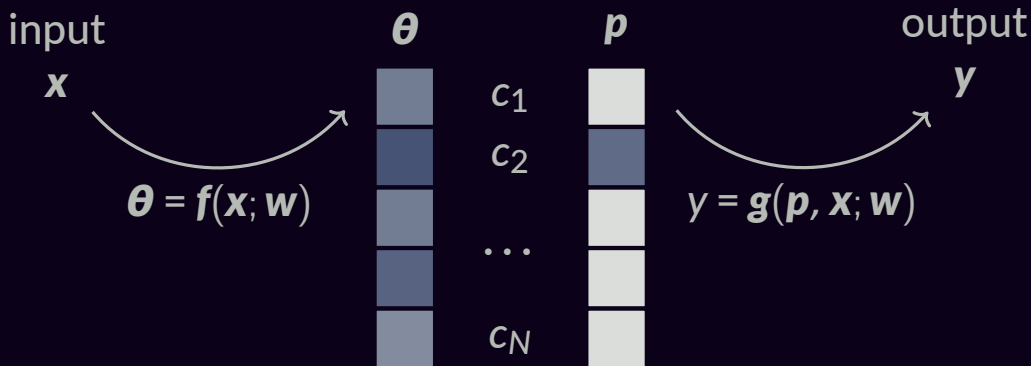
How to select an item from a set?



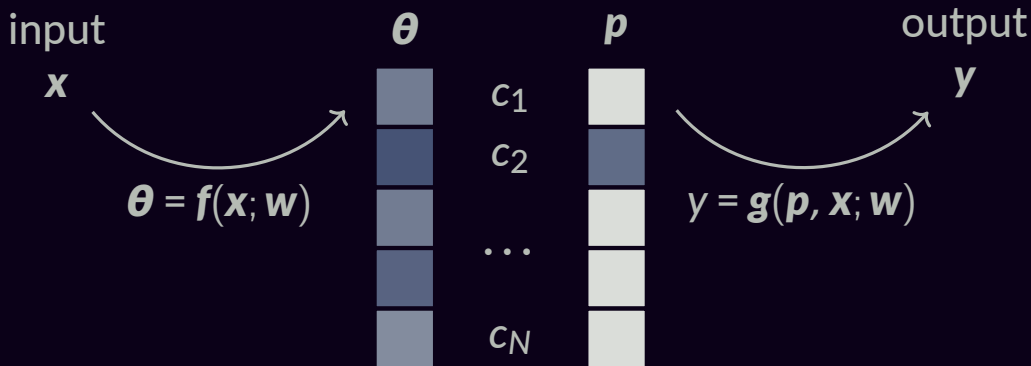
How to select an item from a set?



How to select an item from a set?

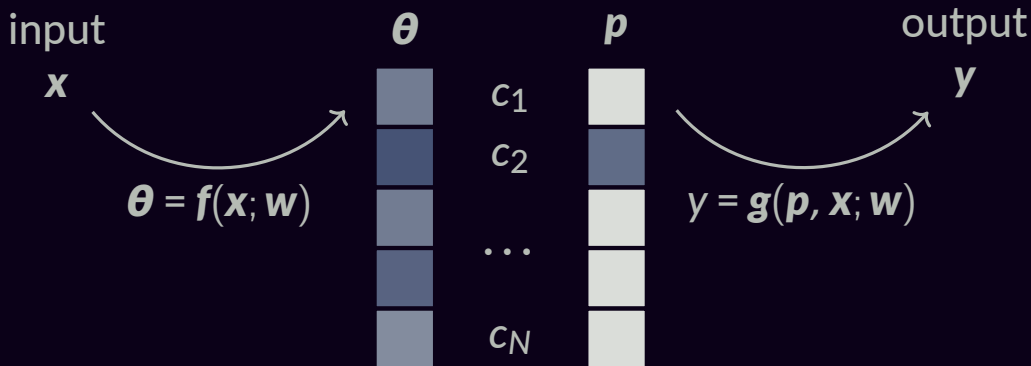


How to select an item from a set?



$$\frac{\partial y}{\partial w} = ?$$

How to select an item from a set?












$$\frac{\partial y}{\partial w} = ?$$

or, essentially,









$$\frac{\partial p}{\partial \theta} = ?$$

Argmax

θ		p
	c_1	
	c_2	
	\dots	
		
	c_N	









$$\frac{\partial p}{\partial \theta} = ?$$

Argmax

θ		p
	c_1	
	c_2	
	\dots	
		
	c_N	








$$\frac{\partial p}{\partial \theta} = ?$$

Argmax

θ		p
	c_1	
	c_2	
	\dots	
		
	c_N	








$$\frac{\partial p}{\partial \theta} = ?$$

Argmax

θ		p
	c_1	
	c_2	
	\dots	
		
	c_N	

$$\frac{\partial p}{\partial \theta} = ?$$

Argmax

θ		p
	c_1	
	c_2	
	\dots	
		
	c_N	

$$\frac{\partial p}{\partial \theta} = ?$$

Argmax



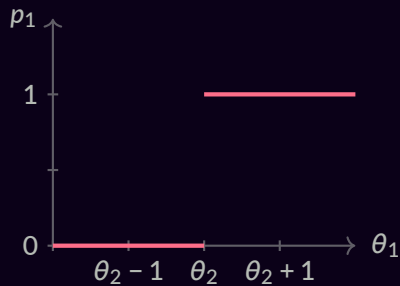
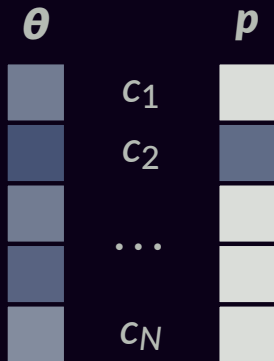
$$\frac{\partial p}{\partial \theta} = ?$$

Argmax



$$\frac{\partial p}{\partial \theta} = ?$$

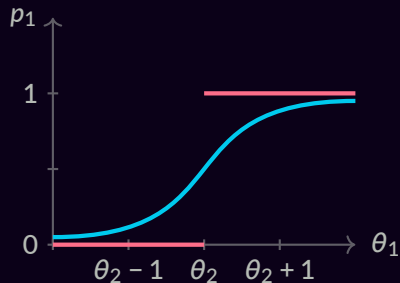
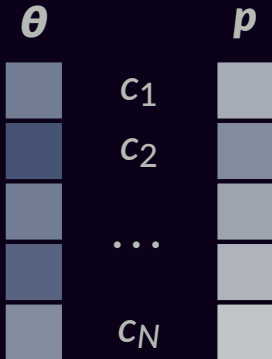
Argmax



$$\frac{\partial p}{\partial \theta} = \mathbf{0}$$

Argmax vs. Softmax

$$p_j = \exp(\theta_j)/Z$$



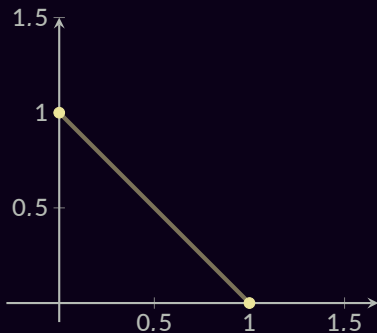
$$\frac{\partial \mathbf{p}}{\partial \boldsymbol{\theta}} = \text{diag}(\mathbf{p}) - \mathbf{p}\mathbf{p}^T$$

A Softmax Origin Story

$$\Delta = \{\mathbf{p} \in \mathbb{R}^N : \mathbf{p} \geq \mathbf{0}, \mathbf{1}^\top \mathbf{p} = 1\}$$

A Softmax Origin Story

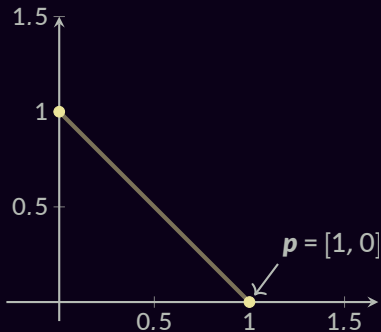
$$\Delta = \{p \in \mathbb{R}^N : p \geq 0, \mathbf{1}^\top p = 1\}$$



$N = 2$

A Softmax Origin Story

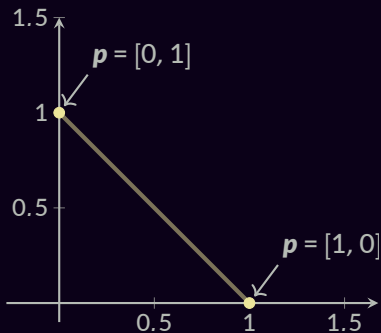
$$\Delta = \{p \in \mathbb{R}^N : p \geq 0, \mathbf{1}^\top p = 1\}$$



$N = 2$

A Softmax Origin Story

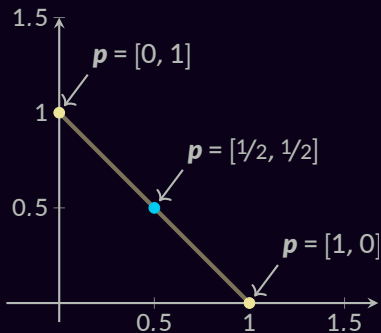
$$\Delta = \{ \mathbf{p} \in \mathbb{R}^N : \mathbf{p} \geq \mathbf{0}, \mathbf{1}^\top \mathbf{p} = 1 \}$$



$N = 2$

A Softmax Origin Story

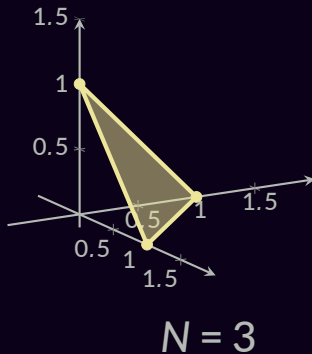
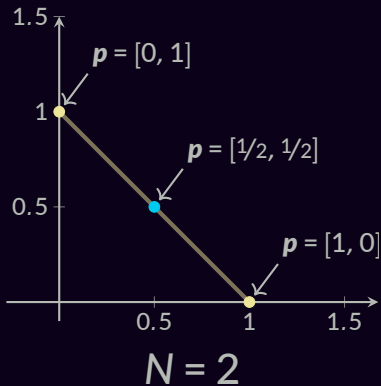
$$\Delta = \{\mathbf{p} \in \mathbb{R}^N : \mathbf{p} \geq \mathbf{0}, \mathbf{1}^\top \mathbf{p} = 1\}$$



$N = 2$

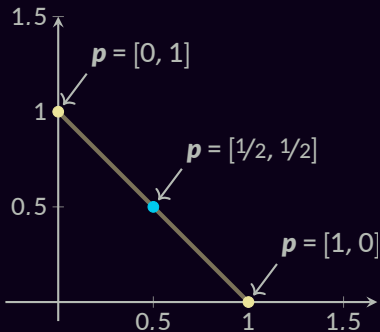
A Softmax Origin Story

$$\Delta = \{p \in \mathbb{R}^N : p \geq 0, \mathbf{1}^\top p = 1\}$$

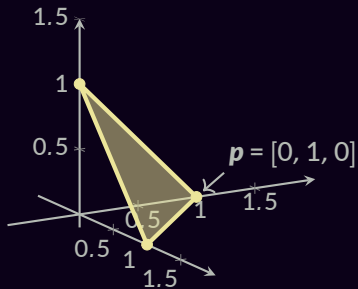


A Softmax Origin Story

$$\Delta = \{p \in \mathbb{R}^N : p \geq 0, \mathbf{1}^\top p = 1\}$$



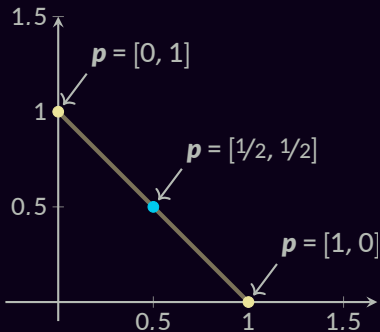
$N = 2$



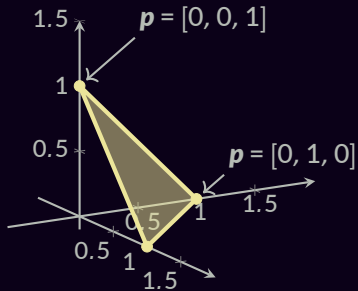
$N = 3$

A Softmax Origin Story

$$\Delta = \{p \in \mathbb{R}^N : p \geq 0, \mathbf{1}^\top p = 1\}$$



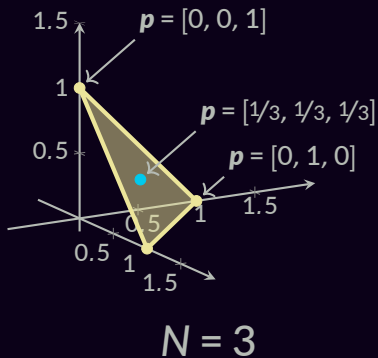
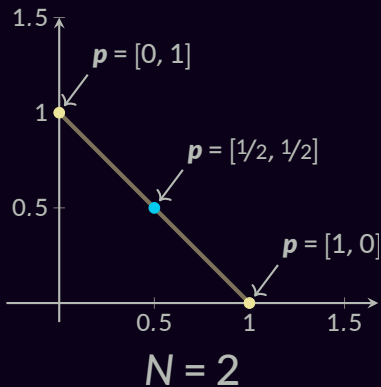
$N = 2$



$N = 3$

A Softmax Origin Story

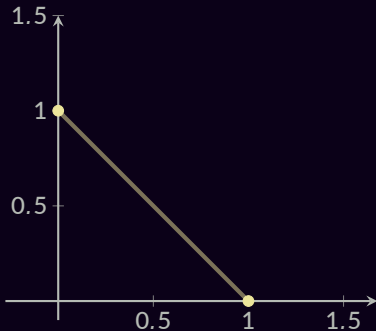
$$\Delta = \{ \mathbf{p} \in \mathbb{R}^N : \mathbf{p} \geq \mathbf{0}, \mathbf{1}^\top \mathbf{p} = 1 \}$$



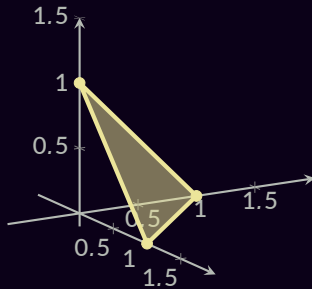
A Softmax Origin Story

$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



$N = 2$

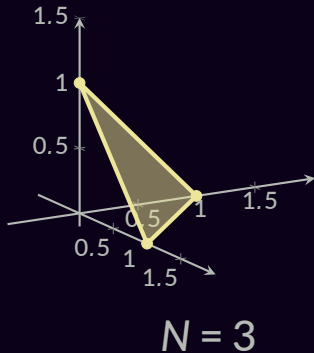
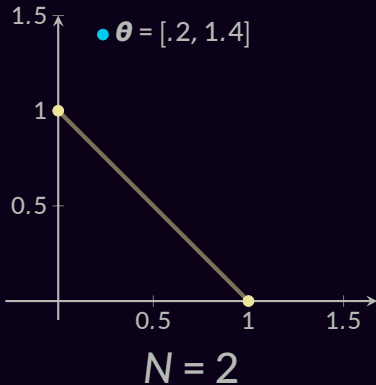


$N = 3$

A Softmax Origin Story

$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

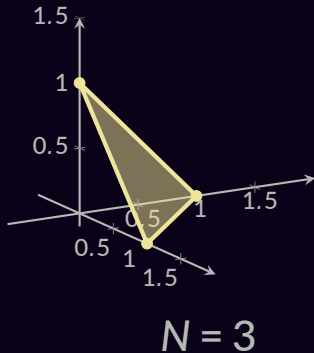
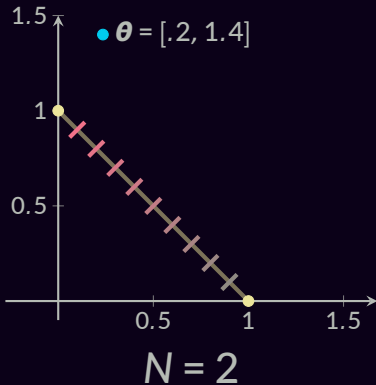
Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



A Softmax Origin Story

$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

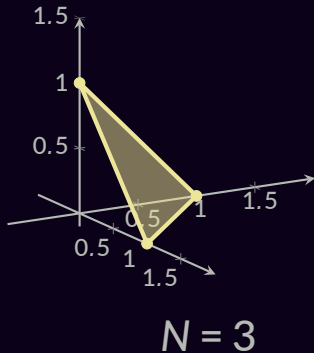
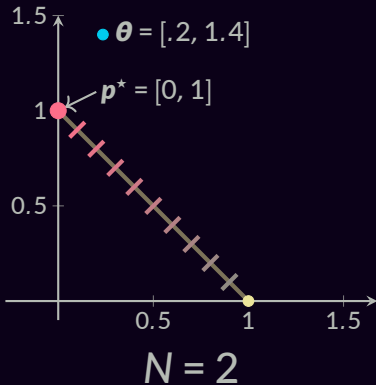
Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



A Softmax Origin Story

$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

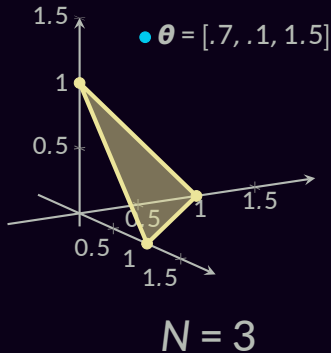
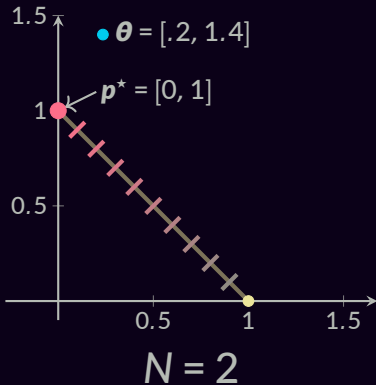
Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



A Softmax Origin Story

$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

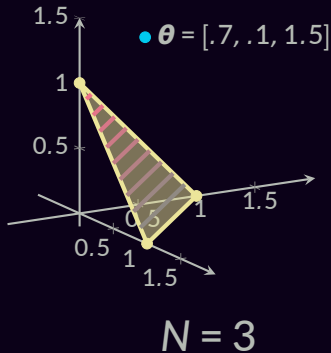
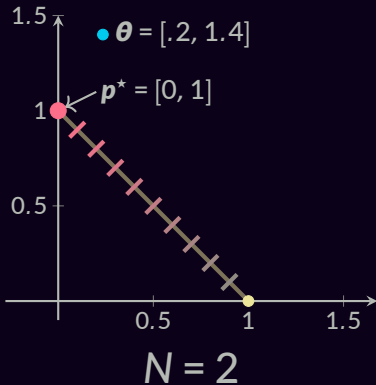
Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



A Softmax Origin Story

$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

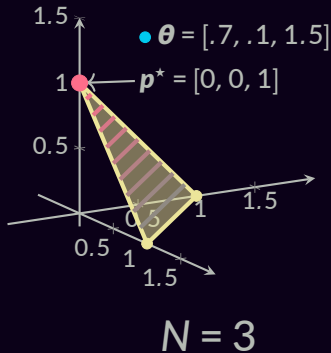
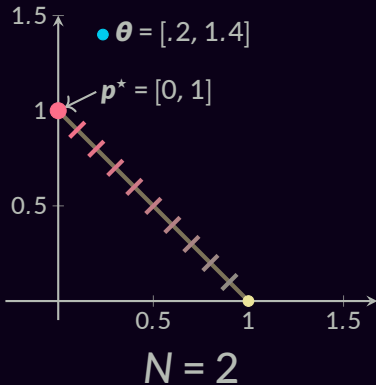
Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



A Softmax Origin Story

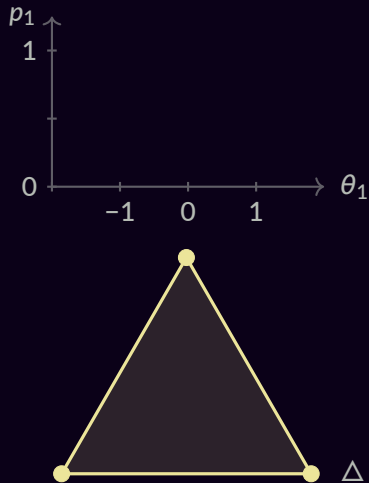
$$\max_j \theta_j = \max_{p \in \Delta} p^T \theta$$

Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



Smoothed Max Operators

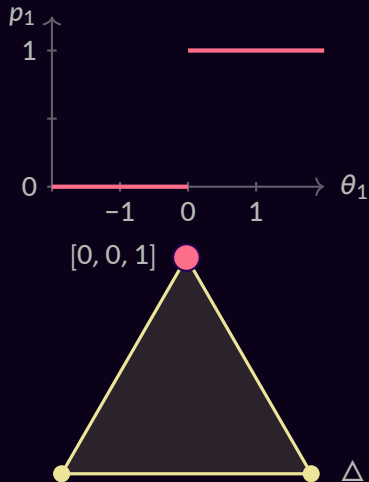
$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^{\top} \boldsymbol{\theta} - \Omega(\boldsymbol{p})$$



Smoothed Max Operators

$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^{\top} \boldsymbol{\theta} - \Omega(\mathbf{p})$$

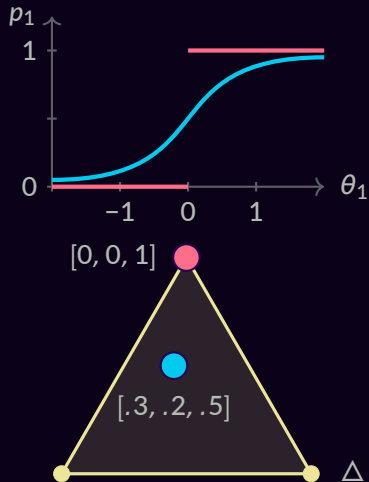
- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)



Smoothed Max Operators

$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^{\top} \boldsymbol{\theta} - \Omega(\mathbf{p})$$

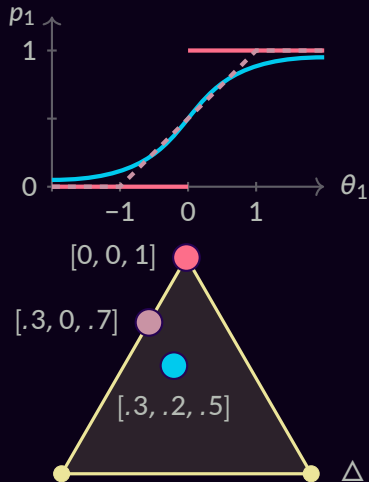
- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)
- softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$



Smoothed Max Operators

$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^{\top} \boldsymbol{\theta} - \Omega(\mathbf{p})$$

- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)
- softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$
- sparsemax: $\Omega(\mathbf{p}) = 1/2 \|\mathbf{p}\|_2^2$



Smoothed Max Operators

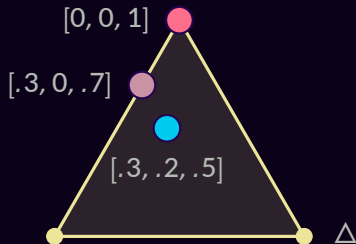
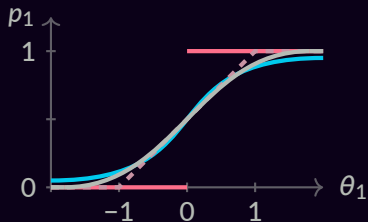
$$\pi_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^{\top} \boldsymbol{\theta} - \Omega(\mathbf{p})$$

● argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)

● softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$

● sparsemax: $\Omega(\mathbf{p}) = 1/2 \|\mathbf{p}\|_2^2$

α -entmax: $\Omega(\mathbf{p}) = 1/\alpha(\alpha-1) \sum_i p_i^{\alpha}$



Tsallis (1988); a generalized entropy (Grünwald and Dawid, 2004)
 (Blondel, Martins, and Niculae 2019a;
 Peters, Niculae, and Martins 2019;
 Correia, Niculae, and Martins 2019)

Sparsemax

$$\begin{aligned}\text{sparsemax}(\boldsymbol{\theta}) &= \arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^\top \boldsymbol{\theta} - 1/2 \|\boldsymbol{p}\|_2^2 \\ &= \arg \min_{\boldsymbol{p} \in \Delta} \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2\end{aligned}$$

Sparsemax

$$\begin{aligned}\text{sparsemax}(\boldsymbol{\theta}) &= \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta} - 1/2 \|\mathbf{p}\|_2^2 \\ &= \arg \min_{\mathbf{p} \in \Delta} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2\end{aligned}$$

Computation:

$$\mathbf{p}^\star = [\boldsymbol{\theta} - \tau \mathbf{1}]_+$$

$$\theta_i > \theta_j \Rightarrow p_i \geq p_j$$

$O(d)$ via partial sort

(Held et al., 1974; Brucker, 1984; Condat, 2016)

Sparsemax

$$\begin{aligned}\text{sparsemax}(\boldsymbol{\theta}) &= \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta} - 1/2 \|\mathbf{p}\|_2^2 \\ &= \arg \min_{\mathbf{p} \in \Delta} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2\end{aligned}$$

Computation:

$$\mathbf{p}^\star = [\boldsymbol{\theta} - \tau \mathbf{1}]_+$$

$$\theta_i > \theta_j \Rightarrow p_i \geq p_j$$

$O(d)$ via partial sort

(Held et al., 1974; Brucker, 1984; Condat, 2016)

Backward pass:

$$\mathbf{J}_{\text{sparsemax}} = \text{diag}(\mathbf{s}) - \frac{1}{|\mathcal{S}|} \mathbf{s} \mathbf{s}^\top$$

$$\text{where } \mathcal{S} = \{j : p_j^\star > 0\},$$

$$s_j = \mathbb{I}[j \in \mathcal{S}]$$

(Martins and Astudillo, 2016)

Sparsemax

$$\begin{aligned}\text{sparsemax}(\boldsymbol{\theta}) &= \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta} - 1/2 \|\mathbf{p}\|_2^2 \\ &= \arg \min_{\mathbf{p} \in \Delta} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2\end{aligned}$$

Computation:

$\mathbf{p}^* = [\theta_i]$
 $\theta_i > \theta_j$
 $O(d)$ via

argmin differentiation

(Gould et al., 2016; Amos and Kolter, 2017)

Backward pass:

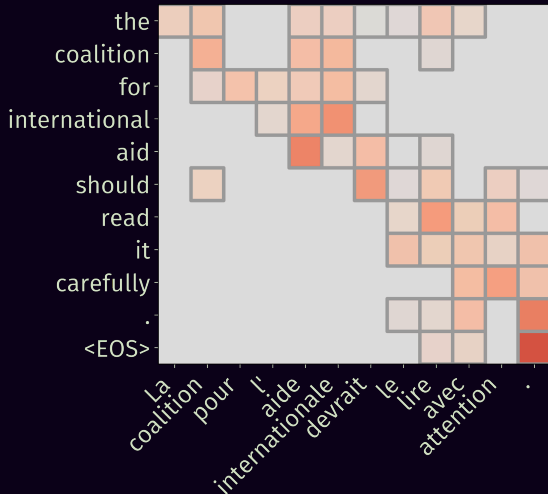
$\frac{\partial \text{sparsemax}(\mathbf{s})}{\partial \mathbf{s}} = \mathbf{I} - \frac{1}{|\mathcal{S}|} \mathbf{s} \mathbf{s}^\top$
 $\mathbf{p}_j^* > 0\}$,
 $\mathbf{p}_j^* \in \mathcal{S}]$

(Held et al., 1974; Brucker, 1984; Condat, 2016)

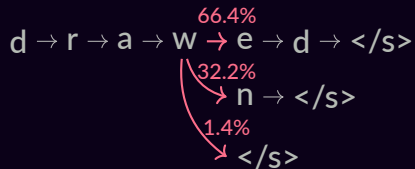
(Martins and Astudillo, 2016)

First applications:

sparse attention



sparse losses (& seq2seq)



Latent variable models!

$$\begin{aligned} p(y \mid x) &= \sum_{h \in \mathcal{H}} p_{\boldsymbol{\phi}}(y \mid h, x) p_{\boldsymbol{\pi}}(h \mid x) \\ &= \mathbb{E}_{h \sim p_{\boldsymbol{\pi}}(h \mid x)} p_{\boldsymbol{\phi}}(y \mid h, x) \end{aligned}$$

Latent variable models!

$$\begin{aligned} p(y | x) &= \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x) \\ &= \mathbb{E}_{h \sim p_{\pi}(h|x)} p_{\phi}(y | h, x) \end{aligned}$$

receiver

sender

- Emergent communication: h is a word from a big vocabulary. $p_{\phi}(y | h)$ is expensive.

Latent variable models!

sum over
all possible messages

$$p(y | x) \Rightarrow \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

receiver

$$= \mathbb{E}_{h \sim p_{\pi}(h|x)} p_{\phi}(y | h, x)$$

sender

- Emergent communication: h is a word from a big vocabulary. $p_{\phi}(y | h)$ is expensive.

Latent variable models!

sum over
all possible messages

$$p(y | x) \Rightarrow \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

receiver

$$= \mathbb{E}_{h \sim p_{\pi}(h|x)} p_{\phi}(y | h, x)$$

sender

- Emergent communication: h is a word from a big vocabulary. $p_{\phi}(y | h)$ is expensive.
- Standard: Monte Carlo gradient estimators (e.g. SFE, Gumbel)

Latent variable models!

sum over
all possible messages

$$p(y | x) \Rightarrow \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

receiver

$$= \mathbb{E}_{h \sim p_{\pi}(h|x)} p_{\phi}(y | h, x)$$

sender

- Emergent communication: h is a word from a big vocabulary. $p_{\phi}(y | h)$ is expensive.
- Standard: Monte Carlo gradient estimators (e.g. SFE, Gumbel)
- **Idea: parametrize $p_{\pi}(h | x)$ using sparsemax!** Sum only over $|\bar{\mathcal{H}}| \ll |\mathcal{H}|$.
No bias AND no variance by **changing the question**

Emergent Communication



... but make it harder: $|\mathcal{H}| = 256$



Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		

Emergent Communication



... but make it harder: $|\mathcal{H}| = 256$



Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		
Gibbs	93.37 ± 0.42	256

Emergent Communication



... but make it harder: $|\mathcal{H}| = 256$

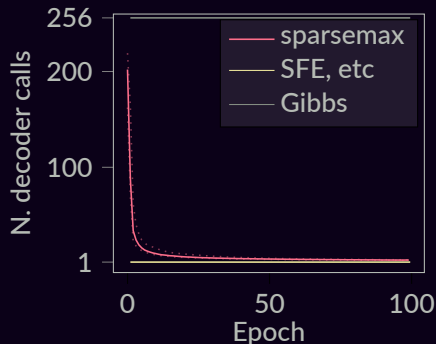


Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		
Gibbs	93.37 ± 0.42	256
Sparse	93.35 ± 0.50	3.13 ± 0.48

Emergent Communication

🎃 ... but make it harder: $|\mathcal{H}| = 256$ 🎃

Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		
Gibbs	93.37 ± 0.42	256
Sparse	93.35 ± 0.50	3.13 ± 0.48



Limitations

- Mostly (and eventually) very sparse.
But $\text{sparsemax}(\mathbf{0}) = 1/d \mathbf{1}$: fully dense worst case.
- For the same reason, sparsemax cannot handle structured h .

Limitations

- Mostly (and eventually) very sparse.
But $\text{sparsemax}(\mathbf{0}) = 1/d \mathbf{1}$: fully dense worst case.
- For the same reason, sparsemax cannot handle structured h .

One solution: **top-k sparsemax**

$$k\text{-sparsemax}(\boldsymbol{\theta}) = \arg \min_{\mathbf{p} \in \Delta, \|\mathbf{p}\|_0 \leq k} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2$$

Limitations

- Mostly (and eventually) very sparse.
But $\text{sparsemax}(\mathbf{0}) = 1/d \mathbf{1}$: fully dense worst case.
- For the same reason, sparsemax cannot handle structured h .

One solution: **top-k sparsemax**

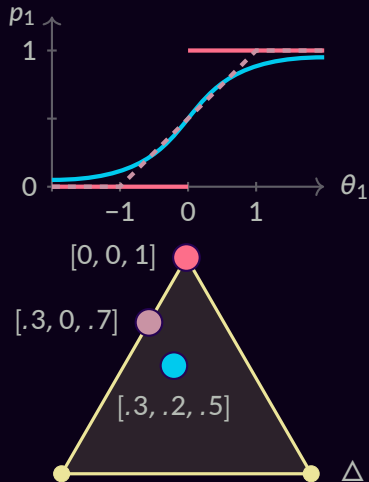
$$k\text{-sparsemax}(\boldsymbol{\theta}) = \arg \min_{\mathbf{p} \in \Delta, \|\mathbf{p}\|_0 \leq k} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2$$

- Non-convex but easy: sparsemax over the k highest scores (Kyrillidis et al., 2013).
- Top-k oracle available for some structured problems.
- Certificate: if at least one of the top-k h gets $p(h) = 0$, **k-sparsemax = sparsemax!**
thus, for latent variables: biased early on, but it goes away.

Smoothed Max Operators

$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^{\top} \boldsymbol{\theta} - \Omega(\mathbf{p})$$

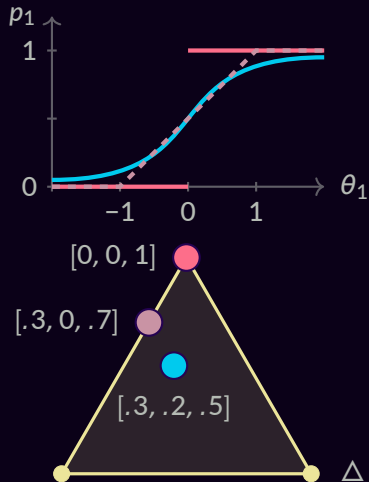
- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)
- softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$
- sparsemax: $\Omega(\mathbf{p}) = 1/2 \|\mathbf{p}\|_2^2$
- α -entmax: $\Omega(\mathbf{p}) = 1/\alpha(\alpha-1) \sum_j p_j^{\alpha}$



Smoothed Max Operators

$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{\theta}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^{\top} \boldsymbol{\theta} - \Omega(\mathbf{p})$$

- argmax: $\Omega(\mathbf{p}) = 0$ (no smoothing)
- softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$
- sparsemax: $\Omega(\mathbf{p}) = 1/2 \|\mathbf{p}\|_2^2$
- α -entmax: $\Omega(\mathbf{p}) = 1/\alpha(\alpha-1) \sum_j p_j^{\alpha}$
- fusedmax: $\Omega(\mathbf{p}) = 1/2 \|\mathbf{p}\|_2^2 + \sum_j |p_j - p_{j-1}|$

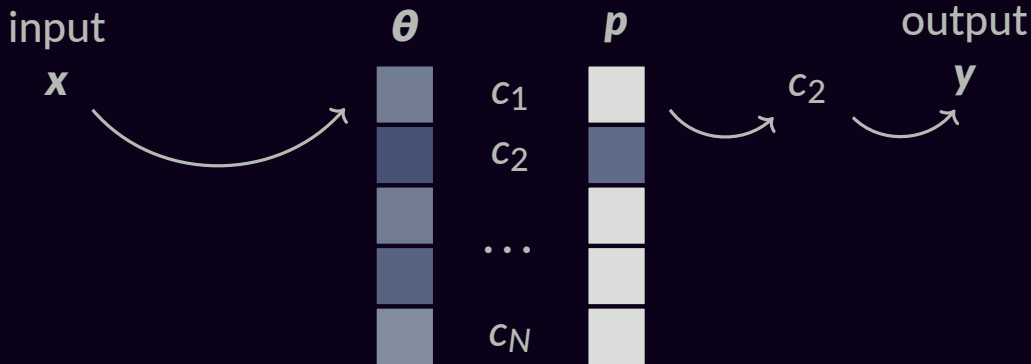


Structured Prediction

finally

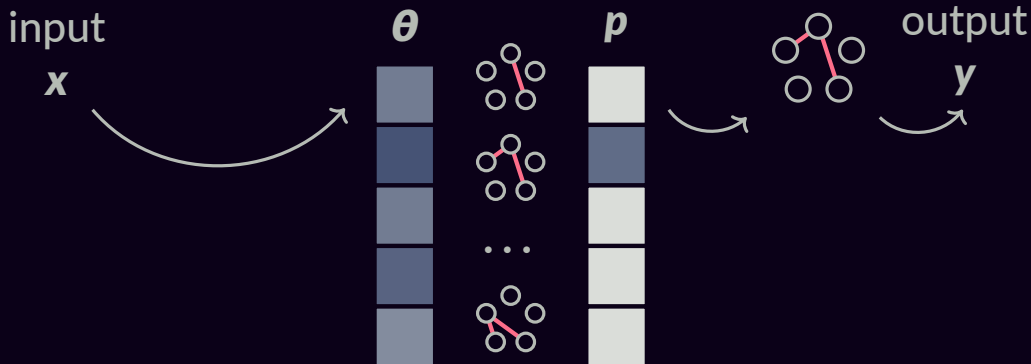
Structured Prediction

is essentially a (very high-dimensional) argmax



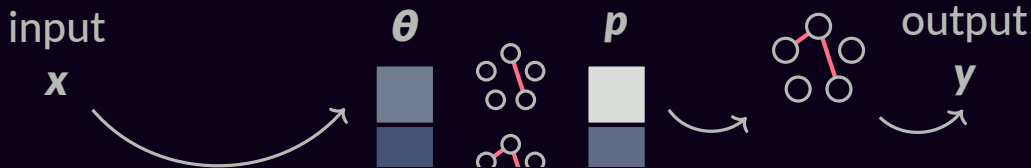
Structured Prediction

is essentially a (very high-dimensional) argmax



Structured Prediction

is essentially a (very high-dimensional) argmax



There are exponentially
many structures

(θ cannot fit in memory!)

Factorization Into Parts

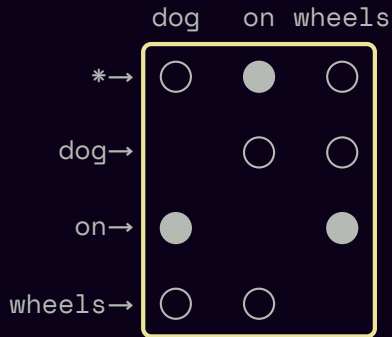


Factorization Into Parts



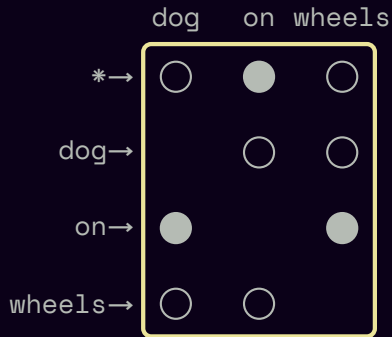
	dog	on	wheels
*→	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>
dog→		<input type="radio"/>	<input type="radio"/>
on→	<input checked="" type="radio"/>		<input checked="" type="radio"/>
wheels→	<input type="radio"/>	<input type="radio"/>	

Factorization Into Parts



TREE

Factorization Into Parts



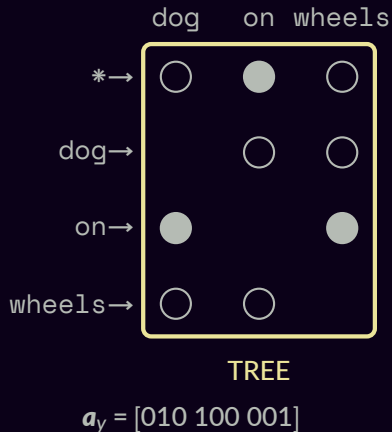
TREE

$$\mathbf{a}_y = [010 \ 100 \ 001]$$

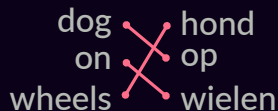
Factorization Into Parts



$$\mathbf{A} = \begin{array}{l} \star \rightarrow \text{dog} \\ \text{on} \rightarrow \text{dog} \\ \text{wheels} \rightarrow \text{dog} \\ \star \rightarrow \text{on} \\ \text{dog} \rightarrow \text{on} \\ \text{wheels} \rightarrow \text{on} \\ \star \rightarrow \text{wheels} \\ \text{dog} \rightarrow \text{wheels} \\ \text{on} \rightarrow \text{wheels} \end{array} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & \dots & 0 & 0 & \dots \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix} \quad \boldsymbol{\eta} = \begin{bmatrix} .1 \\ .2 \\ -.1 \\ .3 \\ .8 \\ .1 \\ -.3 \\ .2 \\ -.1 \end{bmatrix}$$

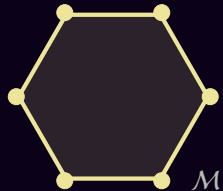
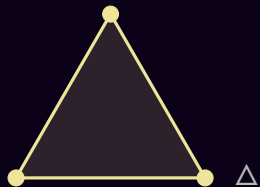


Factorization Into Parts

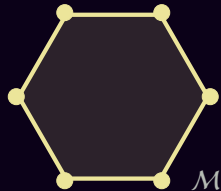
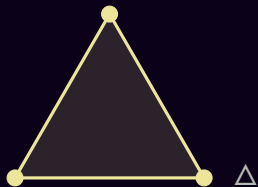


$$\mathbf{A} = \begin{bmatrix} \star \rightarrow \text{dog} & 1 & 0 & 0 \\ \text{on} \rightarrow \text{dog} & 0 & 1 & 1 \\ \text{wheels} \rightarrow \text{dog} & 0 & 0 & 0 \\ \star \rightarrow \text{on} & 0 & 1 & 1 \\ \text{dog} \rightarrow \text{on} & 1 & \dots & 0 & 0 & \dots \\ \text{wheels} \rightarrow \text{on} & 0 & 0 & 0 \\ \star \rightarrow \text{wheels} & 0 & 0 & 0 \\ \text{dog} \rightarrow \text{wheels} & 0 & 1 & 0 \\ \text{on} \rightarrow \text{wheels} & 1 & 0 & 1 \end{bmatrix} \quad \boldsymbol{\eta} = \begin{bmatrix} .1 \\ .2 \\ -.1 \\ .3 \\ .8 \\ .1 \\ -.3 \\ .2 \\ -.1 \end{bmatrix}$$

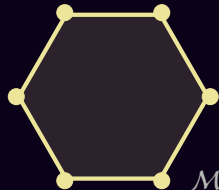
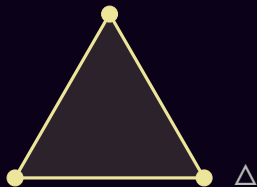
$$\mathbf{A} = \begin{bmatrix} \text{dog} - \text{hond} & 1 & 0 & 0 \\ \text{dog} - \text{op} & 0 & 1 & 1 \\ \text{dog} - \text{wielen} & 0 & 0 & 0 \\ \text{on} - \text{hond} & 0 & 0 & 0 \\ \text{on} - \text{op} & 1 & \dots & 0 & 0 & \dots \\ \text{on} - \text{wielen} & 0 & 1 & 1 \\ \text{wheels} - \text{hond} & 0 & 1 & 0 \\ \text{wheels} - \text{op} & 0 & 0 & 0 \\ \text{wheels} - \text{wielen} & 1 & 0 & 1 \end{bmatrix} \quad \boldsymbol{\eta} = \begin{bmatrix} .1 \\ .2 \\ -.1 \\ .3 \\ .8 \\ .1 \\ -.3 \\ .2 \\ -.1 \end{bmatrix}$$



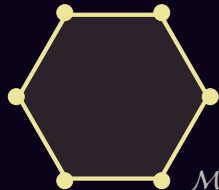
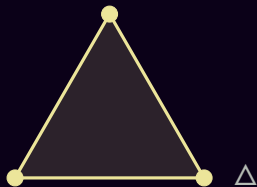
$$\mathcal{M} := \text{conv} \{ \mathbf{a}_h : h \in \mathcal{H} \}$$



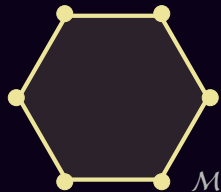
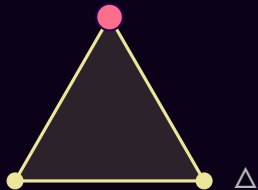
$$\begin{aligned}\mathcal{M} &:= \text{conv} \{ \mathbf{a}_h : h \in \mathcal{H} \} \\ &= \{ \mathbf{A}\mathbf{p} : \mathbf{p} \in \Delta \}\end{aligned}$$



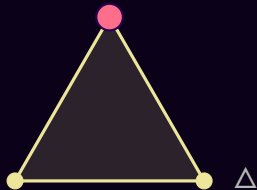
$$\begin{aligned}\mathcal{M} &:= \text{conv} \{ \mathbf{a}_h : h \in \mathcal{H} \} \\ &= \{ \mathbf{A} \mathbf{p} : \mathbf{p} \in \Delta \} \\ &= \{ \mathbb{E}_{H \sim \mathbf{p}} \mathbf{a}_H : \mathbf{p} \in \Delta \}\end{aligned}$$



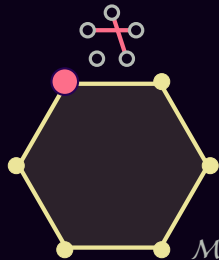
● $\mathbf{argmax} \arg \max_{p \in \Delta} p^T \theta$



• **argmax** $\arg \max_{p \in \Delta} p^\top \theta$



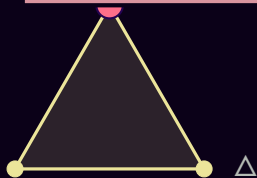
• **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta$



• $\text{argmax}_{p \in \Delta} p^T \theta$

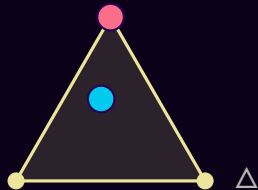
• $\text{MAP}_{\mu \in \mathcal{M}} \mu^T \eta$

e.g. dependency parsing → **Chu-Liu/Edmonds**
matching → **Kuhn-Munkres**

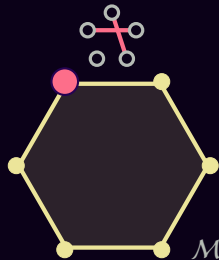


- **argmax** $\arg \max_{p \in \Delta} p^\top \theta$

- **softmax** $\arg \max_{p \in \Delta} p^\top \theta + H(p)$

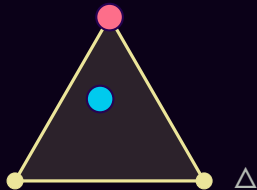


- **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta$



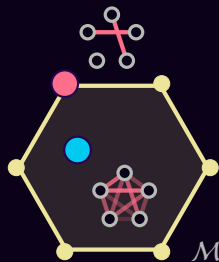
● **argmax** $\arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta}$

● **softmax** $\arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta} + H(\mathbf{p})$



● **MAP** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta}$

● **marginals** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} + \tilde{H}(\boldsymbol{\mu})$



● **argmax** $\arg \max_{p \in \Delta} p^\top \theta$

● **softmax** $\arg \max_{p \in \Delta} p^\top \theta + H(p)$

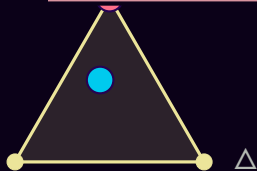
● **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta$

● **marginals** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta + \tilde{H}(\mu)$

e.g. sequence labeling \rightarrow forward-backward

(Rabiner, 1989)

As attention: (Kim et al., 2017)



- **argmax** $\arg \max_{p \in \Delta} p^T \theta$

- **softmax** $\arg \max_{p \in \Delta} p^T \theta + H(p)$

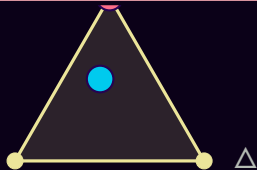
- **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^T \eta$

- **marginals** $\arg \max_{\mu \in \mathcal{M}} \mu^T \eta + \tilde{H}(\mu)$

e.g. dependency parsing → **the Matrix-Tree theorem**

(Koo et al., 2007; D. A. Smith and N. A. Smith, 2007; McDonald and Satta, 2007)

As attention: (Liu and Lapata, 2018)



● **argmax** $\arg \max_{p \in \Delta} p^\top \theta$

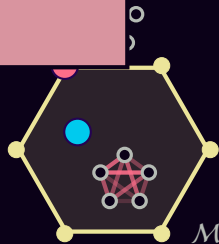
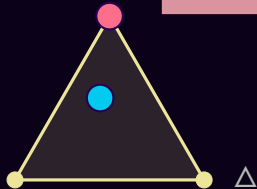
● **softmax** $\arg \max_{p \in \Delta} p^\top \theta + H(p)$

● **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta$

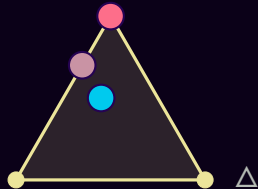
● **marginals** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta + \tilde{H}(\mu)$

e.g. matchings \rightarrow **#P-complete!**

(Taskar, 2004; Valiant, 1979)

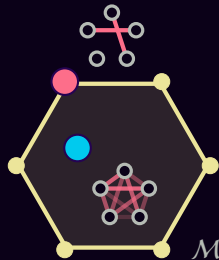


- **argmax** $\arg \max_{p \in \Delta} p^\top \theta$
- **softmax** $\arg \max_{p \in \Delta} p^\top \theta + H(p)$
- **sparsemax** $\arg \max_{p \in \Delta} p^\top \theta - 1/2 \|p\|^2$



● **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta$

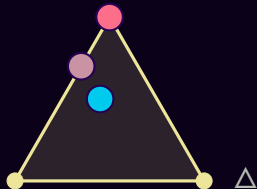
● **marginals** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta + \tilde{H}(\mu)$



● **argmax** $\arg \max_{p \in \Delta} p^\top \theta$

● **softmax** $\arg \max_{p \in \Delta} p^\top \theta + H(p)$

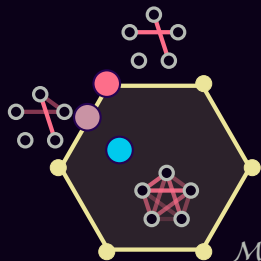
● **sparsemax** $\arg \max_{p \in \Delta} p^\top \theta - 1/2 \|p\|^2$



● **MAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta$

● **marginals** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta + \tilde{H}(\mu)$

● **SparseMAP** $\arg \max_{\mu \in \mathcal{M}} \mu^\top \eta - 1/2 \|\mu\|^2$



Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner of \mathcal{M}

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner of \mathcal{M}

$$\mathbf{a}_{y^*} = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \underbrace{(\boldsymbol{\eta} - \boldsymbol{\mu}^{(t-1)})}_{\tilde{\boldsymbol{\eta}}}$$

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner of \mathcal{M}
- update the (sparse) coefficients of \boldsymbol{p}
 - Update rules: vanilla, away-step, pairwise

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner of \mathcal{M}
- update the (sparse) coefficients of \mathbf{p}
 - Update rules: vanilla, away-step, pairwise
 - Quadratic objective: **Active Set**
(Nocedal and Wright, 1999, Ch. 16.4 & 16.5)
(Wolfe, 1976; Vinyes and Obozinski, 2017)

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner
- update the (sparse) **Active Set achieves finite & linear convergence!**

- Update rules: van
- Quadratic objective: **Active Set**

(Nocedal and Wright, 1999, Ch. 16.4 & 16.5)

(Wolfe, 1976; Vinyes and Obozinski, 2017)

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner of \mathcal{M}
- update the (sparse) coefficients of \mathbf{p}
 - Update rules: vanilla, away-step, pairwise
 - Quadratic objective: **Active Set**
(Nocedal and Wright, 1999, Ch. 16.4 & 16.5)
(Wolfe, 1976; Vinyes and Obozinski, 2017)

Backward pass

$\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}$ is sparse

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Conditional Gradient

(Frank and Wolfe, 1956; Lacoste-Julien and Jaggi, 2015)

- select a new corner of \mathcal{M}
- update the (sparse) coefficients of \mathbf{p}
 - Update rules: vanilla, away-step, pairwise
 - Quadratic objective: **Active Set**
(Nocedal and Wright, 1999, Ch. 16.4 & 16.5)
(Wolfe, 1976; Vinyes and Obozinski, 2017)

Backward pass

$\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}$ is sparse

computing $\left(\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}\right)^\top d\mathbf{y}$
takes $O(\dim(\boldsymbol{\mu}) \text{nnz}(\mathbf{p}^*))$

Algorithms for SparseMAP

$$\boldsymbol{\mu}^* = \arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2 \|\boldsymbol{\mu}\|^2$$

linear constraints
(*alas, exponentially many!*)

quadratic objective

Condition Completely modular: just add MAP pass

(Frank and Wolfe, 1956)

- select a new corner of \mathcal{M}
- update the (sparse) coefficients of \mathbf{p}
 - Update rules: vanilla, away-step, pairwise
 - Quadratic objective: **Active Set**
(Nocedal and Wright, 1999, Ch. 16.4 & 16.5)
(Wolfe, 1976; Vinyes and Obozinski, 2017)

$\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}$ is sparse

computing $\left(\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}\right)^\top d\boldsymbol{\eta}$
takes $O(\dim(\boldsymbol{\mu}) \text{nnz}(\mathbf{p}^*))$

SparseMAP Applications

- **Sparse alignment attention**
(Nicolae, Martins, Blondel, and Cardie, 2018)
- **Latent TreeLSTM**
(Nicolae, Martins, and Cardie, 2018)
- **As loss: supervised dependency parsing**
(Nicolae, Martins, Blondel, and Cardie 2018;
Blondel, Martins, and Nicolae 2019b)

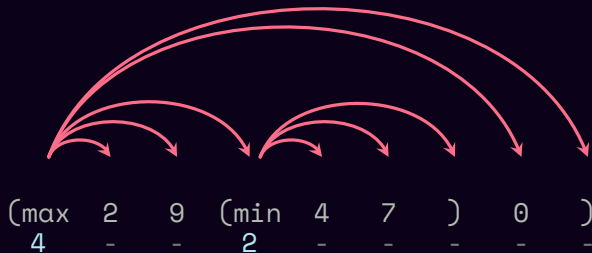
Latent Dependency Trees

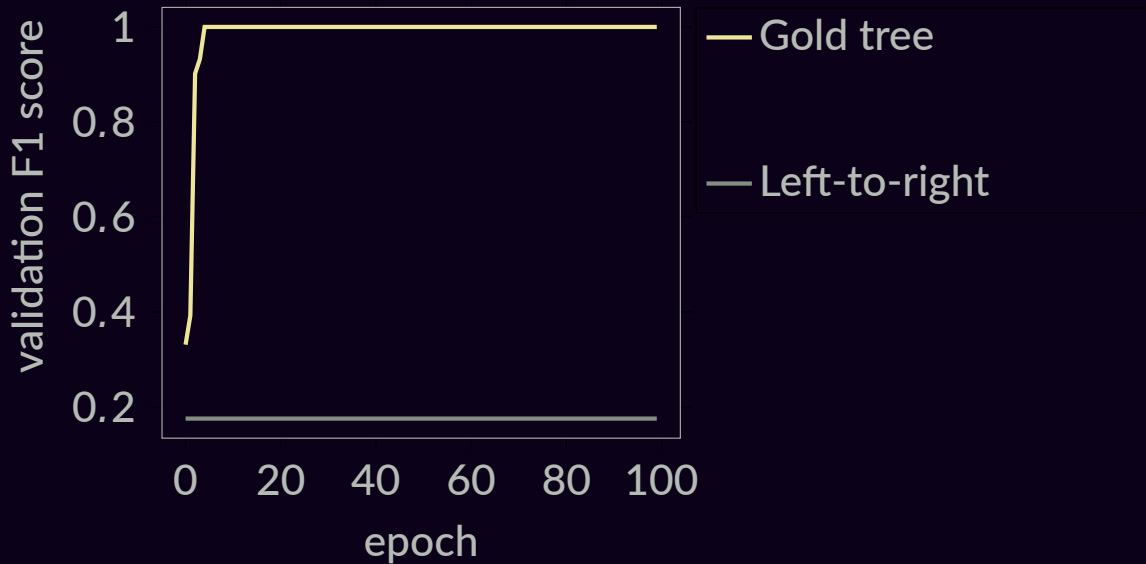
Arity tagging with latent GCN (Corro and Titov, 2019; Kipf and Welling, 2017)

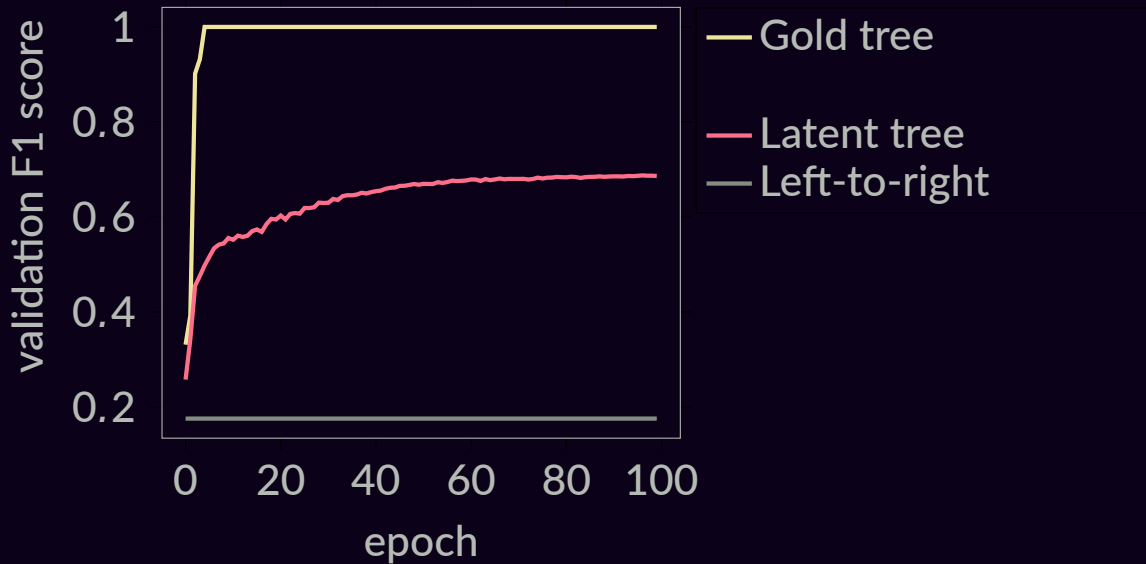
(max 2 9 (min 4 7) 0)

Latent Dependency Trees

Arity tagging with latent GCN (Corro and Titov, 2019; Kipf and Welling, 2017)

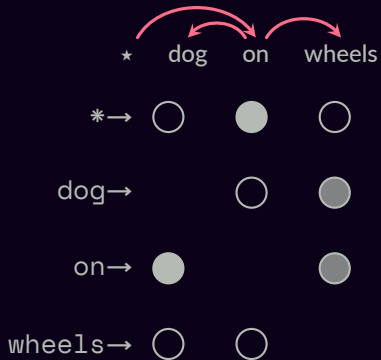




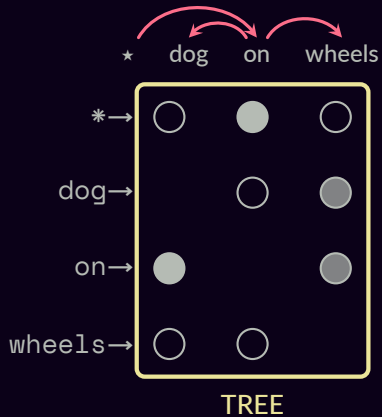


**What if MAP is not
available?**

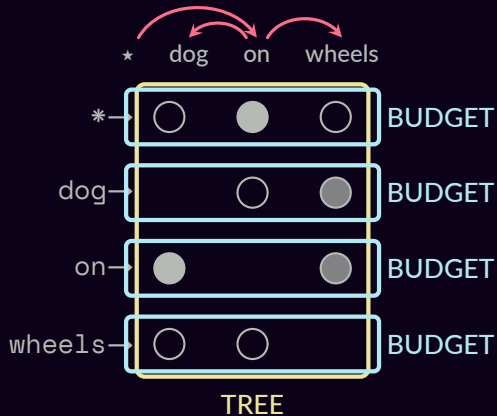
Multiple, Overlapping Factors



Multiple, Overlapping Factors

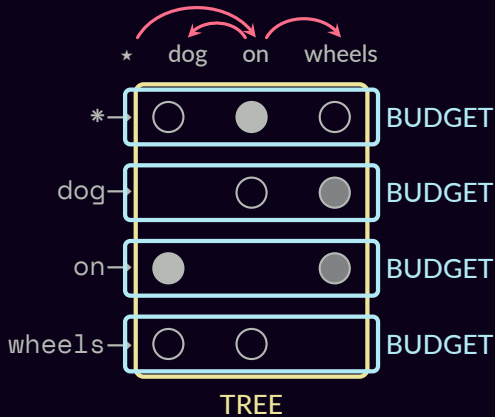


Multiple, Overlapping Factors

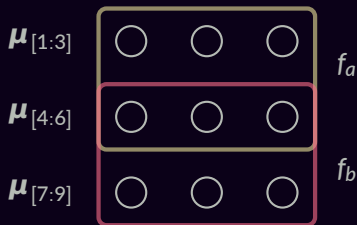


Multiple, Overlapping Factors

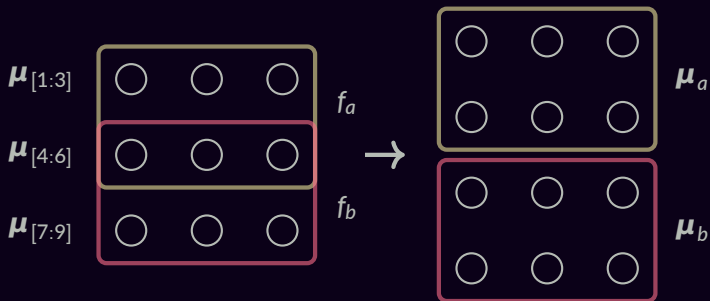
Maximization in factor graphs: NP-hard, even when each factor is tractable.



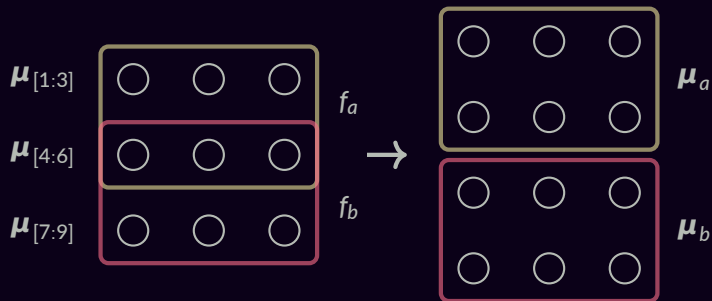
Optimization as Consensus-Seeking



Optimization as Consensus-Seeking



Optimization as Consensus-Seeking

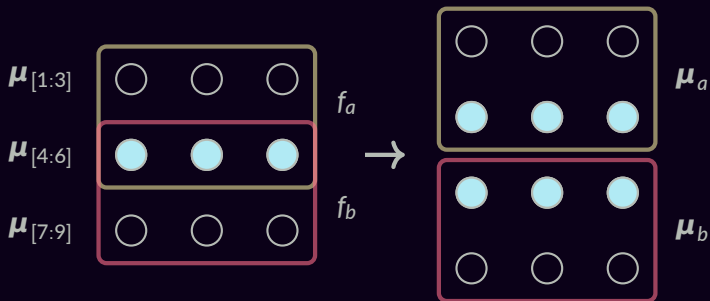


$$\max_{\mu_f} \sum_{f \in \mathcal{F}} \eta_f^\top \mu_f$$

s.t.

$$\mu_f \in \mathcal{M}_f \text{ for } f \in \mathcal{F}$$

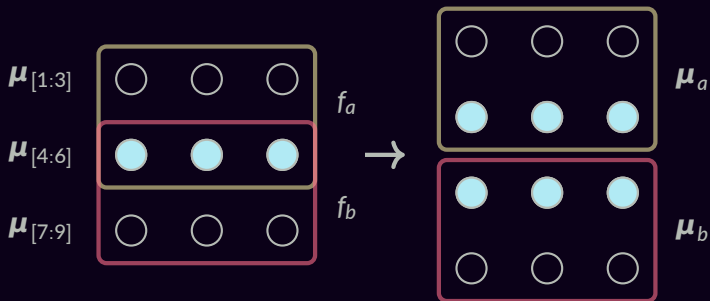
Optimization as Consensus-Seeking



Agreement on overlap: $\boldsymbol{\mu}_{a,[4:6]} = \boldsymbol{\mu}_{b,[4:6]} = \boldsymbol{\mu}_{[4:6]}$

$$\max_{\boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f \quad \text{s.t.} \quad \boldsymbol{\mu}_f \in \mathcal{M}_f \text{ for } f \in \mathcal{F}$$

Optimization as Consensus-Seeking



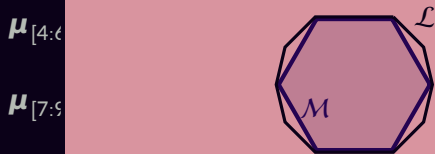
Agreement on overlap: $\mu_{a,[4:6]} = \mu_{b,[4:6]} = \mu_{[4:6]}$

$$\max_{\mu, \mu_f} \sum_{f \in \mathcal{F}} \eta_f^\top \mu_f$$

$$\text{s.t. } \mathbf{C}_f \mu = \mu_f, \mu_f \in \mathcal{M}_f \text{ for } f \in \mathcal{F}$$

the local polytope:

$$\mathcal{L} := \{\boldsymbol{\mu} : \mathbf{C}_f \boldsymbol{\mu} \in \mathcal{M}_f, f \in \mathcal{F}\} \supseteq \mathcal{M}$$

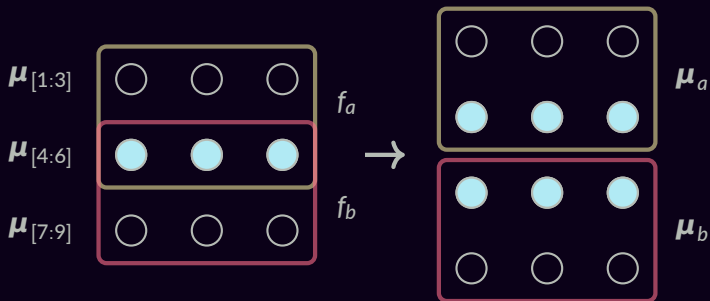


Agreement on overlap: $\boldsymbol{\mu}_{a,[4:6]} = \boldsymbol{\mu}_{b,[4:6]} = \boldsymbol{\mu}_{[4:6]}$

$$\max_{\boldsymbol{\mu}, \boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^T \boldsymbol{\mu}_f$$

$$\text{s.t. } \mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f, \boldsymbol{\mu}_f \in \mathcal{M}_f \text{ for } f \in \mathcal{F}$$

Optimization as Consensus-Seeking

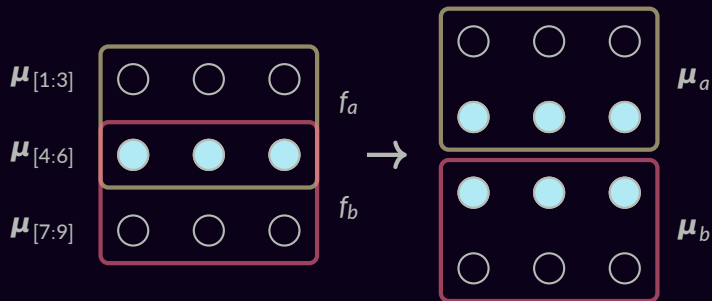


Agreement on overlap: $\boldsymbol{\mu}_{a,[4:6]} = \boldsymbol{\mu}_{b,[4:6]} = \boldsymbol{\mu}_{[4:6]}$

$$\max_{\boldsymbol{\mu}, \boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f$$

$$\text{s.t. } \mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f, \boldsymbol{\mu}_f \in \mathcal{M}_f \text{ for } f \in \mathcal{F}$$

Optimization as Consensus-Seeking



Agreement on overlap: $\boldsymbol{\mu}_{a,[4:6]} = \boldsymbol{\mu}_{b,[4:6]} = \boldsymbol{\mu}_{[4:6]}$

$$\max_{\boldsymbol{\mu}, \boldsymbol{\mu}_f} \left(\sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f \right) - \frac{1}{2} \|\boldsymbol{\mu}\|^2 \quad \text{s.t.} \quad \mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f, \boldsymbol{\mu}_f \in \mathcal{M}_f \text{ for } f \in \mathcal{F}$$

Algorithms for LP-SparseMAP

Forward pass

$$\begin{aligned} & \arg \max_{\mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f} \left(\sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f \right) - 1/2 \|\boldsymbol{\mu}\|^2 \\ &= \arg \max_{\mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \left(\boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f - 1/2 \|\mathbf{D}_f \boldsymbol{\mu}_f\|^2 \right) \end{aligned}$$

- Separable objective,
agreement constraints
ADMM in consensus form
- SparseMAP subproblem for each f

Algorithms for LP-SparseMAP

Forward pass

$$\begin{aligned} & \arg \max_{\mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f} \left(\sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f \right) - 1/2 \|\boldsymbol{\mu}\|^2 \\ &= \arg \max_{\mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \left(\boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f - 1/2 \|\mathbf{D}_f \boldsymbol{\mu}_f\|^2 \right) \end{aligned}$$

- Separable objective, agreement constraints
ADMM in consensus form
- SparseMAP subproblem for each f

Backward pass

- Jacobian fixed-point characterization

$$\mathbf{J} = \begin{bmatrix} \mathbf{C}_{f_a} \\ \mathbf{C}_{f_b} \\ \vdots \end{bmatrix}^\top \begin{bmatrix} \mathbf{J}_{f_a} & \cdots & 0 \\ \vdots & \mathbf{J}_{f_b} & \vdots \\ 0 & \cdots & \ddots \end{bmatrix} \begin{bmatrix} \mathbf{C}_{f_a} \\ \mathbf{C}_{f_b} \\ \vdots \end{bmatrix} \mathbf{J}$$

- Efficient iteration for vjp
- Combines the SparseMAP Jacobians of each factor

(use specialized impl. when available: many commonly used factors derived in paper.)

Differentiable Sparse Structured Prediction



```
fg = FactorGraph()
var = [fg.variable() for i in range(n)] # handwave

fg.add(Tree(var))

for i in range(n):
    fg.add(Budget(var[i, :], budget=5))

mu = fg.lp_sparsemap(eta)
```

Factor graphs as a hidden-layer DSL!

Differentiable Sparse Structured Prediction



```
fg = FactorGraph()
var = [fg.variable() for i in range(n)] # handwave

fg.add(Tree(var))

for i in range(n):
    fg.add(Budget(var[i, :], budget=5))

mu = fg.lp_sparsemap(eta)
```

Factor graphs as a hidden-layer DSL!

If $|\mathcal{F}| = 1$, recovers SparseMAP.

Differentiable Sparse Structured Prediction



Factor graphs as a hidden-layer DSL!

If $|\mathcal{F}| = 1$, recovers SparseMAP.

Modular library.

Built-in specialized factors:

- OR, XOR, AND
- OR-with-output
- Budget, Knapsack
- Pairwise

```
class Factor:
    def map( $\eta_f$ ): # abstract, private
        raise NotImplemented

    def sparsemap( $\eta_f$ ):
        # active set algo, uses self.map

    def backward( $d\mu_f$ ):
        # analytic, uses active set result
```

```
class Budget(Factor):
    def sparsemap( $\eta_f$ ):
        # specialized

    def backward( $d\mu_f$ ):
        # specialized
```

Differentiable Sparse Structured Prediction



Factor graphs as a hidden-layer DSL!

If $|\mathcal{F}| = 1$, recovers SparseMAP.

Modular library.

Built-in specialized factors:

- OR, XOR, AND
- OR-with-output
- Budget, Knapsack
- Pairwise

New factors only require MAP.

```
class Factor:
    def map( $\eta_f$ ): # abstract, private
        raise NotImplemented

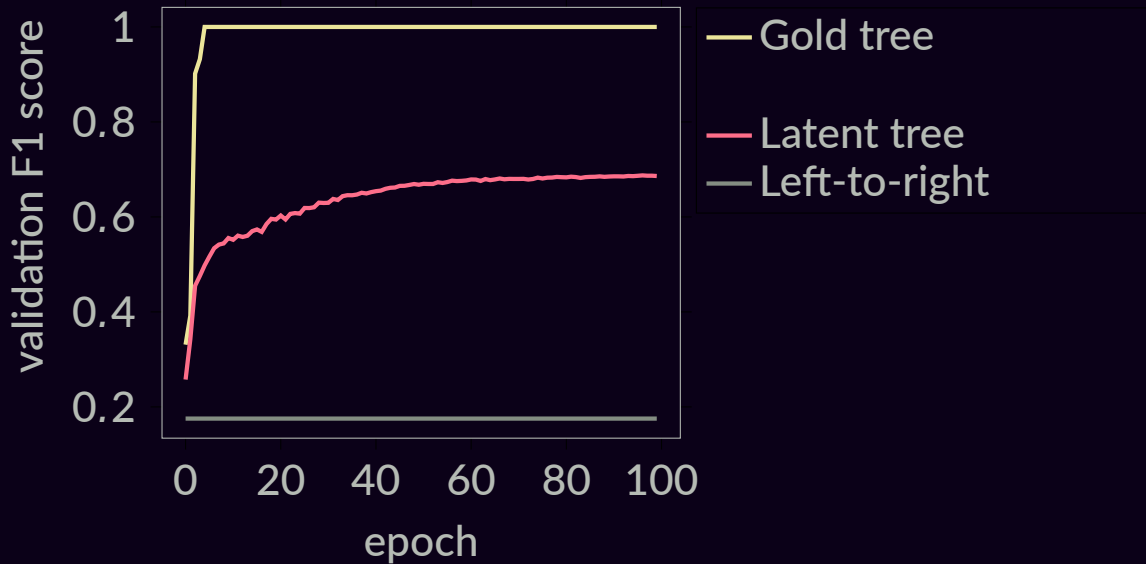
    def sparsemap( $\eta_f$ ):
        # active set algo, uses self.map

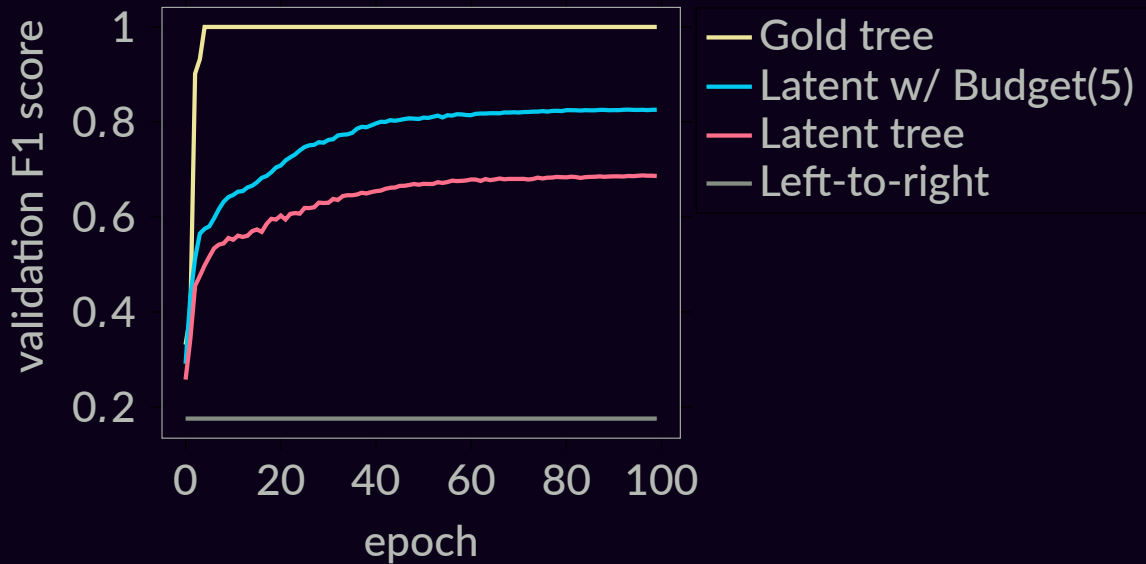
    def backward( $d\mu_f$ ):
        # analytic, uses active set result
```

```
class Budget(Factor):
    def sparsemap( $\eta_f$ ):
        # specialized

    def backward( $d\mu_f$ ):
        # specialized
```

```
class Tree(Factor):
    def map( $\eta$ ):
        # Chu-Liu/Edmonds algo
```





Structured Attention for Alignments

NLI

premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)

⚙	A	A	⚙
	gentleman	police	
	overlooking	officer	
	
	situation	closely	

output



entails

contradicts

neutral

(Model: decomposable attention (Parikh et al., 2016))

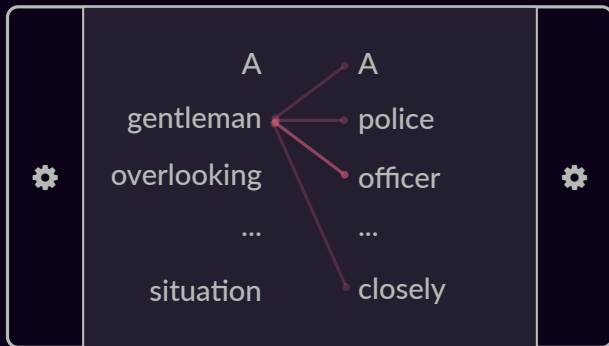
Structured Attention for Alignments

NLI

premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)



output



entails

contradicts

neutral

(Model: decomposable attention (Parikh et al., 2016))

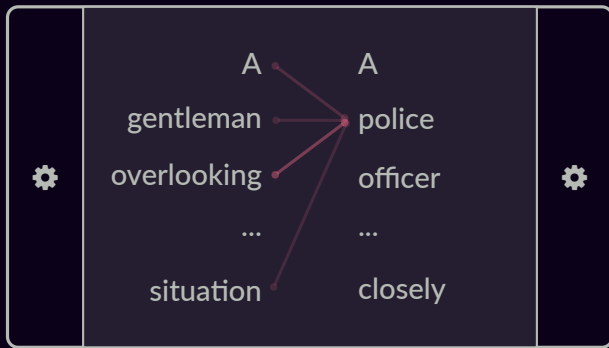
Structured Attention for Alignments

NLI

premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)



output



entails

contradicts

neutral

(Model: decomposable attention (Parikh et al., 2016))

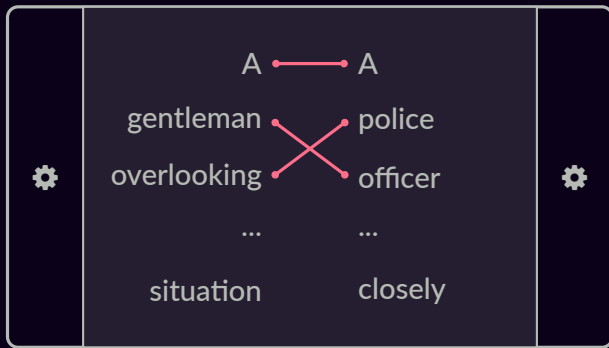
Structured Attention for Alignments

NLI

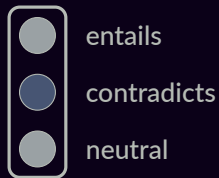
premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)



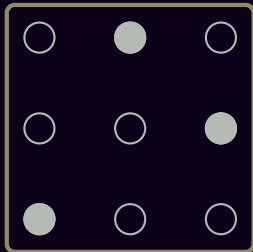
output



(Proposed model: global structured alignment.)

Structured Alignment Models

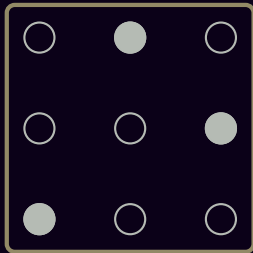
matching



SparseMAP w/ Kuhn-Munkres
(Kuhn, 1955)

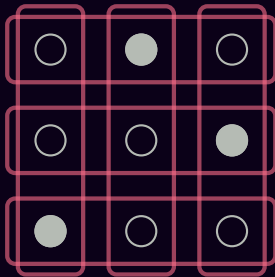
Structured Alignment Models

matching



SparseMAP w/ Kuhn-Munkres
(Kuhn, 1955)

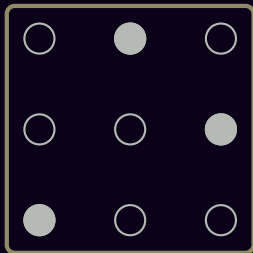
LP-matching



LP-SparseMAP w/ XORs
(equivalent; different solver)

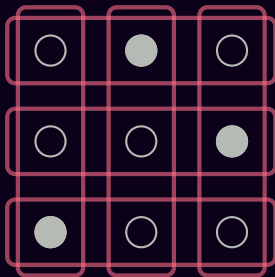
Structured Alignment Models

matching



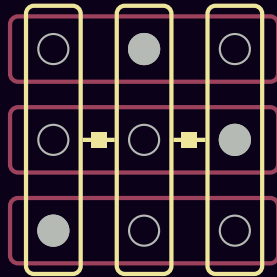
SparseMAP w/ Kuhn-Munkres
(Kuhn, 1955)

LP-matching



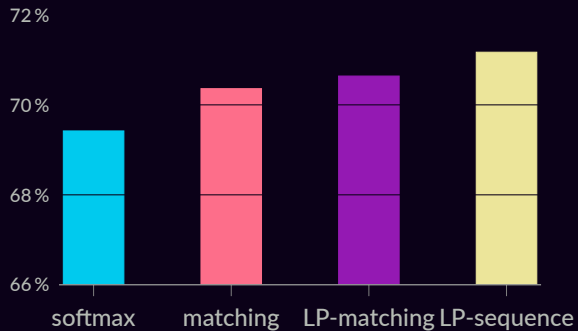
LP-SparseMAP w/ XORs
(equivalent; different solver)

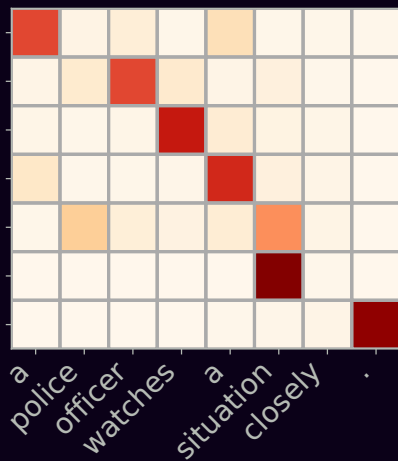
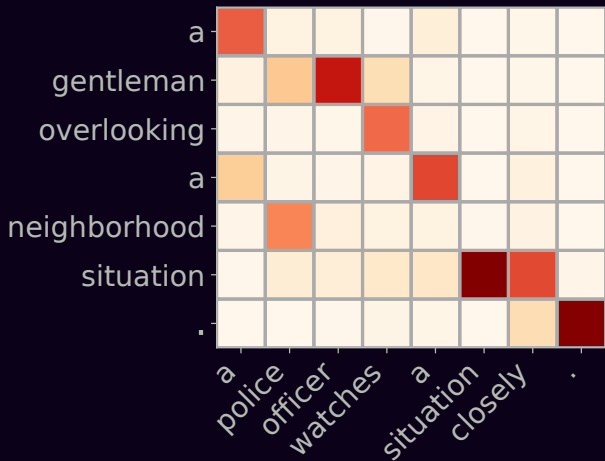
LP-sequence

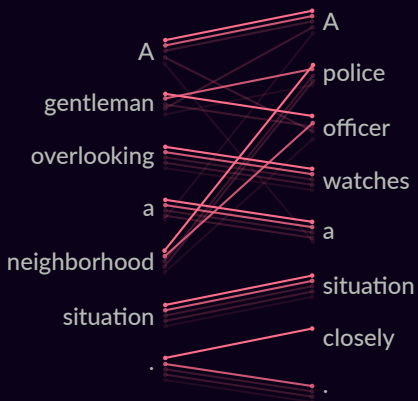
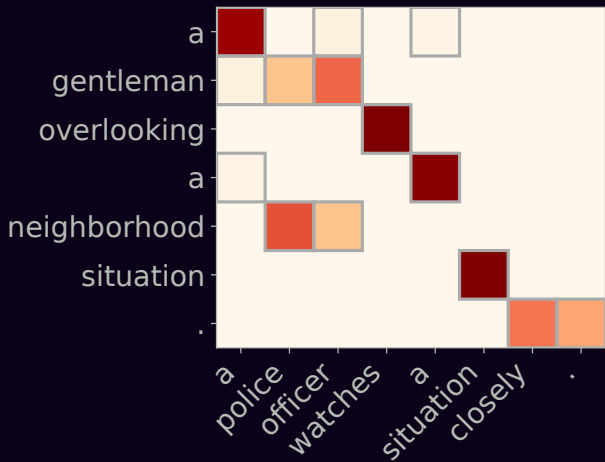


additional score
for *contiguous alignments*
 $(i, j) - (i + 1, j \pm 1)$

MultiNLI (Williams et al., 2017)





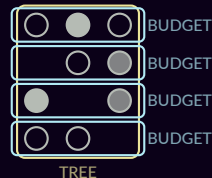
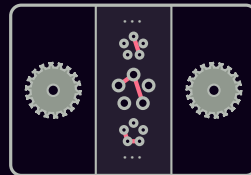


Conclusions

Differentiable & sparse
structured inference

Generic, extensible, efficient algorithms

Interpretable **structured attention**



Conclusions

Differentiable & sparse
structured inference

Generic, extensible, efficient algorithms

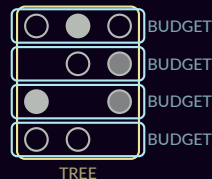
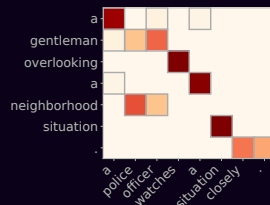
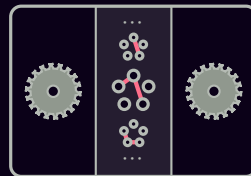
Interpretable **structured attention**

Future work

Structure beyond NLP

Weak & semi-supervision

Generative latent structure models



Extra slides

Acknowledgements



This work was supported by the European Research Council (ERC StG DeepSPIN 758969) and by the Fundação para a Ciência e Tecnologia through contract UID/EEA/50008/2013.

Some icons by Dave Gandy and Freepik via flaticon.com.

Fusedmax

$$\begin{aligned}\text{fusedmax}(\boldsymbol{\theta}) &= \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^T \boldsymbol{\theta} - 1/2 \|\mathbf{p}\|_2^2 - \sum_{2 \leq j \leq d} |p_j - p_{j-1}| \\ &= \arg \min_{\mathbf{p} \in \Delta} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2 + \sum_{2 \leq j \leq d} |p_j - p_{j-1}| \\ \text{prox}_{\text{fused}}(\boldsymbol{\theta}) &= \arg \min_{\mathbf{p} \in \mathbb{R}^d} \|\mathbf{p} - \boldsymbol{\theta}\|_2^2 + \sum_{2 \leq j \leq d} |p_j - p_{j-1}|\end{aligned}$$

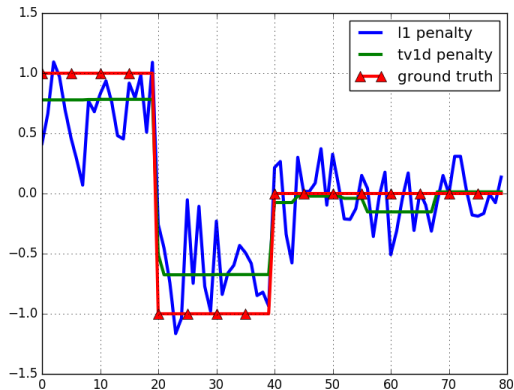
Proposition: $\text{fusedmax}(\boldsymbol{\theta}) = \text{sparsemax}(\text{prox}_{\text{fused}}(\boldsymbol{\theta}))$

(Niculae and Blondel, 2017)

fusedmax(

prox_{fused}(

Proposi



“Fused Lasso” a.k.a. 1-d Total Variation

(Tibshirani et al., 2005)

(Nicolae and Blondel, 2017)

$|p_j - p_{j-1}|$

$|p_{j-1}|$

$|p_{j-1}|$

fused(θ)

Danskin's Theorem

Let $\phi : \mathbb{R}^d \times \mathcal{Z} \rightarrow \mathbb{R}$, $\mathcal{Z} \subset \mathbb{R}^d$ compact.

$$\partial \max_{\mathbf{z} \in \mathcal{Z}} \phi(\mathbf{x}, \mathbf{z}) = \text{conv} \{ \nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{z}^*) \mid \mathbf{z}^* \in \arg \max_{\mathbf{z} \in \mathcal{Z}} \phi(\mathbf{x}, \mathbf{z}) \}.$$

Example: maximum of a vector

Danskin's Theorem

Let $\phi : \mathbb{R}^d \times \mathcal{Z} \rightarrow \mathbb{R}$, $\mathcal{Z} \subset \mathbb{R}^d$ compact.

$$\partial \max_{\mathbf{z} \in \mathcal{Z}} \phi(\mathbf{x}, \mathbf{z}) = \text{conv} \{ \nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{z}^*) \mid \mathbf{z}^* \in \arg \max_{\mathbf{z} \in \mathcal{Z}} \phi(\mathbf{x}, \mathbf{z}) \}.$$

Example: maximum of a vector

$$\begin{aligned} \partial \max_{j \in [d]} \theta_j &= \partial \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta} \\ &= \partial \max_{\mathbf{p} \in \Delta} \phi(\mathbf{p}, \boldsymbol{\theta}) \\ &= \text{conv} \{ \nabla_{\boldsymbol{\theta}} \phi(\mathbf{p}^*, \boldsymbol{\theta}) \} \\ &= \text{conv} \{ \mathbf{p}^* \} \end{aligned}$$

Danskin's Theorem

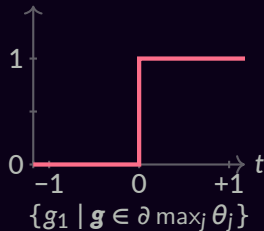
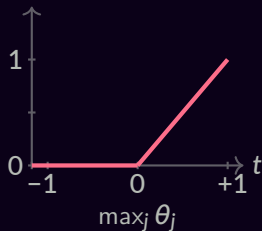
Let $\phi : \mathbb{R}^d \times \mathcal{Z} \rightarrow \mathbb{R}$, $\mathcal{Z} \subset \mathbb{R}^d$ compact.

$$\partial \max_{\mathbf{z} \in \mathcal{Z}} \phi(\mathbf{x}, \mathbf{z}) = \text{conv} \{ \nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{z}^*) \mid \mathbf{z}^* \in \arg \max_{\mathbf{z} \in \mathcal{Z}} \phi(\mathbf{x}, \mathbf{z}) \}.$$

Example: maximum of a vector

$$\begin{aligned} \partial \max_{j \in [d]} \theta_j &= \partial \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \boldsymbol{\theta} \\ &= \partial \max_{\mathbf{p} \in \Delta} \phi(\mathbf{p}, \boldsymbol{\theta}) \\ &= \text{conv} \{ \nabla_{\boldsymbol{\theta}} \phi(\mathbf{p}^*, \boldsymbol{\theta}) \} \\ &= \text{conv} \{ \mathbf{p}^* \} \end{aligned}$$

$$\boldsymbol{\theta} = [t, 0]$$



**Dynamically inferring
the computation graph**

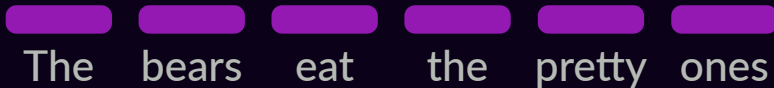
So far: a structured hidden layer

$$\mathbb{E}_H[\mathbf{a}_H]$$

Network must handle “soft” combinations of structures.
Fine for attention, but can be limiting.

Dependency TreeLSTM

(Tai et al., 2015)

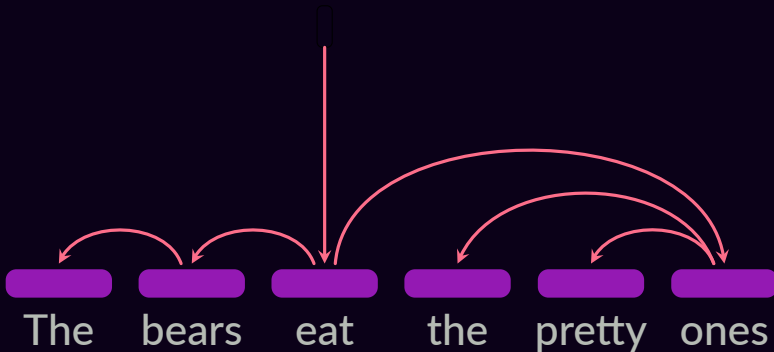


The diagram illustrates word embeddings for the sentence "The bears eat the pretty ones". Each word is represented by a blue rounded rectangle above its text label. The words are arranged horizontally, with "The" and "ones" being red, "bears" and "eat" being blue, and "the" and "pretty" being green.

The bears eat the pretty ones

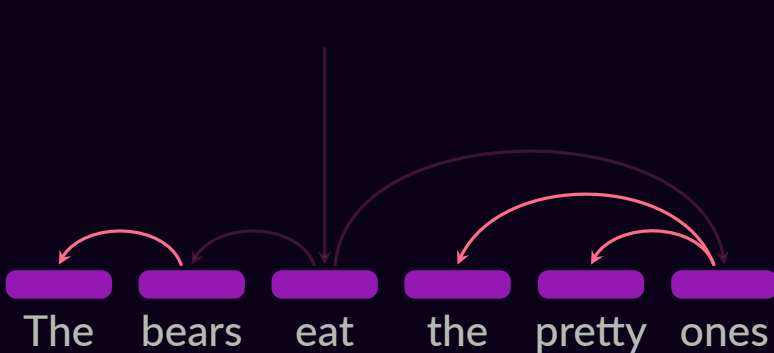
Dependency TreeLSTM

(Tai et al., 2015)



Dependency TreeLSTM

(Tai et al., 2015)



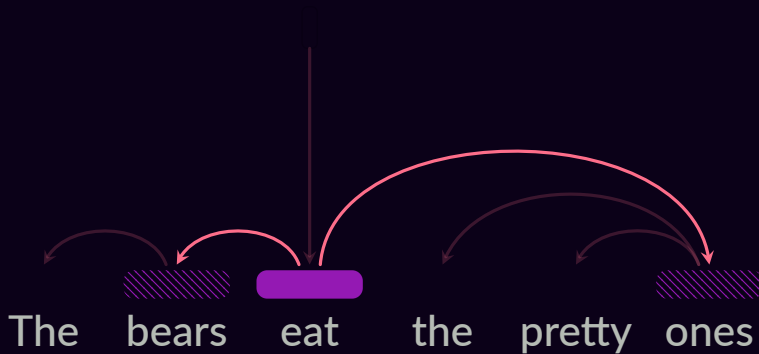
Dependency TreeLSTM

(Tai et al., 2015)



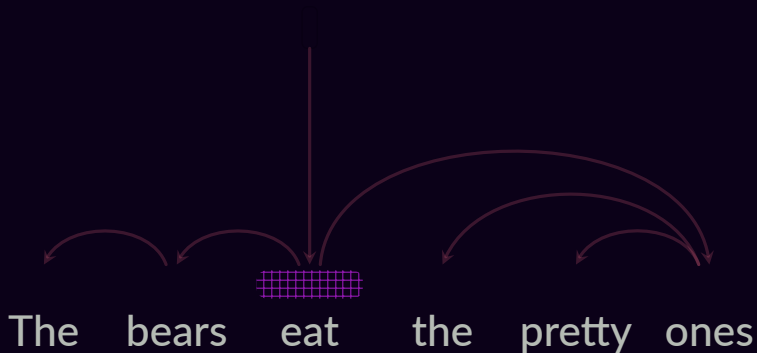
Dependency TreeLSTM

(Tai et al., 2015)



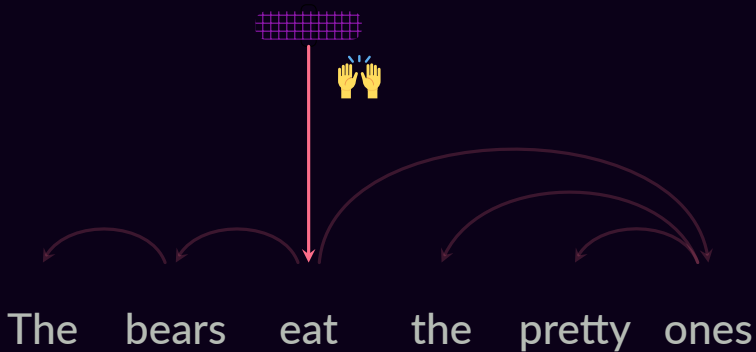
Dependency TreeLSTM

(Tai et al., 2015)



Dependency TreeLSTM

(Tai et al., 2015)

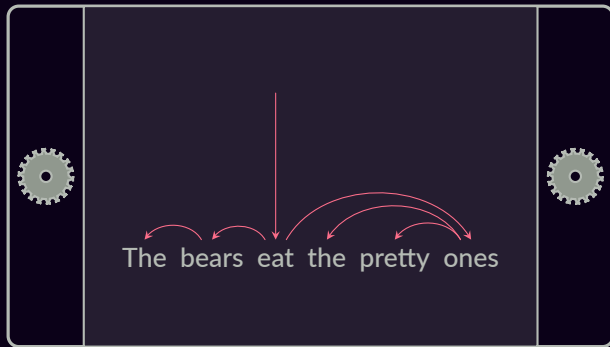


Latent Dependency TreeLSTM

(Niculae, Martins, and Cardie, 2018)

input

x



output

y

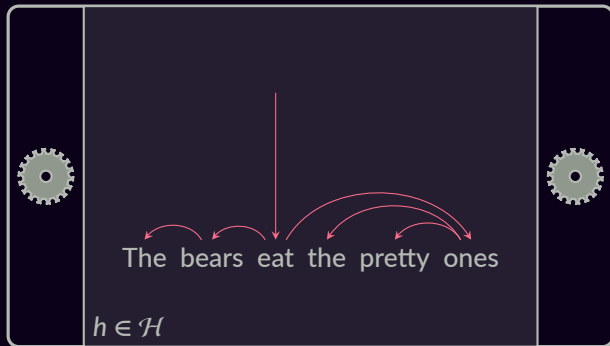
Latent Dependency TreeLSTM

(Niculae, Martins, and Cardie, 2018)

$$p(y|x) = \sum_{h \in \mathcal{H}} p(y | h, x) p(h | x)$$

input

x



output

y

Structured Latent Variable Models

$$p(y | x) = \sum_{h \in \mathcal{H}} p(y | h, x) p(h | x)$$


Structured Latent Variable Models

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

Structured Latent Variable Models

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

e.g., a TreeLSTM defined by h



Structured Latent Variable Models

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

e.g., a TreeLSTM defined by h

latent classifier

The diagram illustrates the components of the equation. An arrow points from the text 'e.g., a TreeLSTM defined by h ' to the parameter ϕ in the term $p_{\phi}(y | h, x)$. Another arrow points from the text 'latent classifier' to the parameter π in the term $p_{\pi}(h | x)$.

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

Exponentially large sum!

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1

idea 2

idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

$$\sum_{h \in \mathcal{H}}$$

idea 1

idea 2

idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$

idea 1

idea 2

idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

idea 2

idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$



idea 2

idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$



idea 2

idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

idea 2 $p_{\pi}(h | x) \propto \exp(\text{score}_{\pi}(h; x))$

softmax

idea 3

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$



Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$



idea 2 $p_{\pi}(h | x) \propto \exp(\text{score}_{\pi}(h; x))$

softmax



idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$



idea 2 $p_{\pi}(h | x) \propto \exp(\text{score}_{\pi}(h; x))$

softmax



idea 3

Structured Latent Variable Models

sum over
all possible trees

e.g., a TreeLSTM defined by h

$$p(y | x) = \sum_{h \in \mathcal{H}} p_{\phi}(y | h, x) p_{\pi}(h | x)$$

latent classifier

How to define p_{π} ?

idea 1 $p_{\pi}(h | x) = 1$ if $h = h^*$ else 0

argmax

$$\sum_{h \in \mathcal{H}} \frac{\partial p(y | x)}{\partial \pi}$$



idea 2 $p_{\pi}(h | x) \propto \exp(\text{score}_{\pi}(h; x))$

softmax




idea 3

SparseMAP




SparseMAP

 $= .7$


 $+ .3$



SparseMAP

 $= .7$

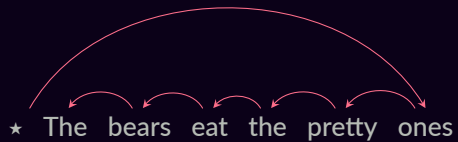
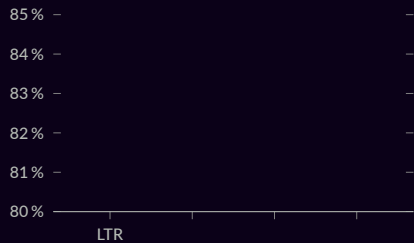
 $+ .3$

 $+ 0$  $+ \dots$

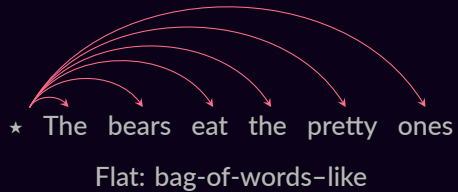
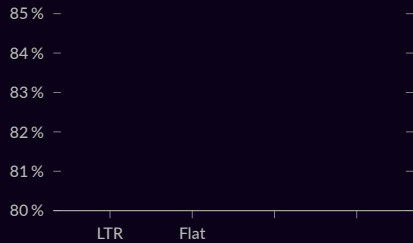
SparseMAP

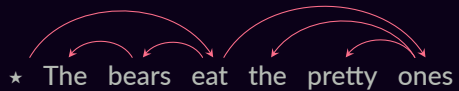
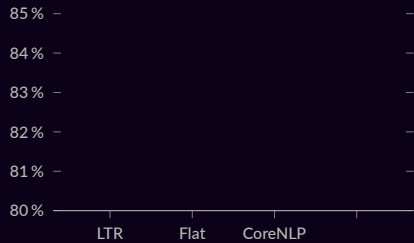
$$\begin{aligned}
 & \text{Diagram 1} = .7 \quad \text{Diagram 2} + .3 \quad \text{Diagram 3} + 0 \text{Diagram 4} + \dots \\
 p(y \mid x) = & .7 p_{\phi}(y \mid \text{Diagram 1}) + .3 p_{\phi}(y \mid \text{Diagram 2})
 \end{aligned}$$

The diagrams consist of three red dots in a horizontal row. In the first and second diagrams, a red curved arrow points from the first dot to the second, and another red curved arrow points from the second dot to the third. In the third diagram, a red curved arrow points from the second dot to the first, and another red curved arrow points from the third dot to the second. The fourth diagram is identical to the third.



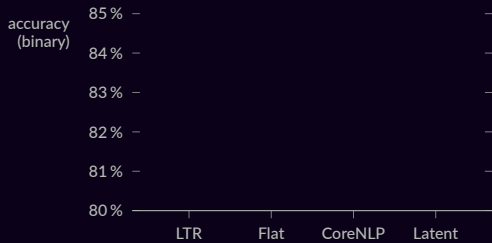
Left-to-right: regular LSTM



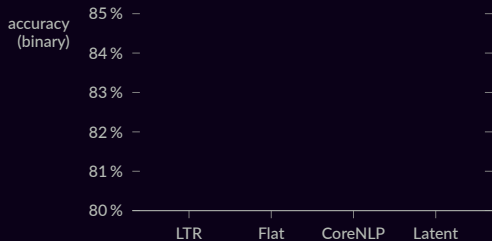


CoreNLP: off-line parser

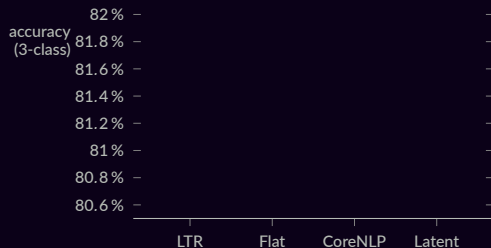
Sentiment classification (SST)



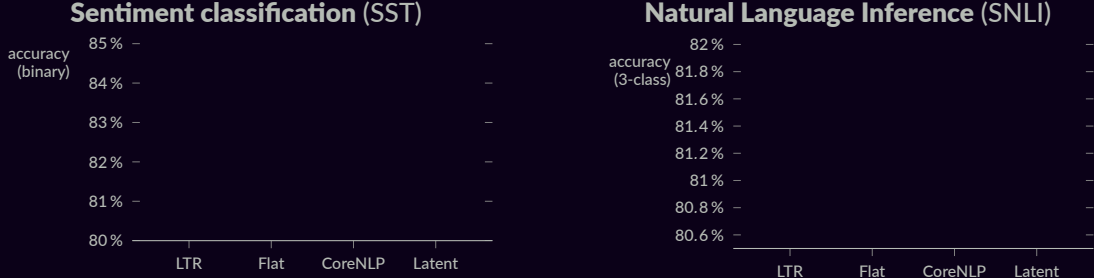
Sentiment classification (SST)



Natural Language Inference (SNLI)

Sentence pair classification (P, H)

$$p(y \mid P, H) = \sum_{h_P \in \mathcal{H}(P)} \sum_{h_H \in \mathcal{H}(H)} p_{\phi}(y \mid h_P, h_H) p_{\pi}(h_P \mid P) p_{\pi}(h_H \mid H)$$

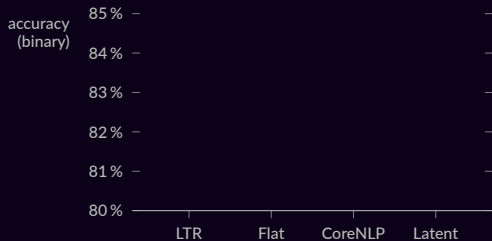


Reverse dictionary lookup

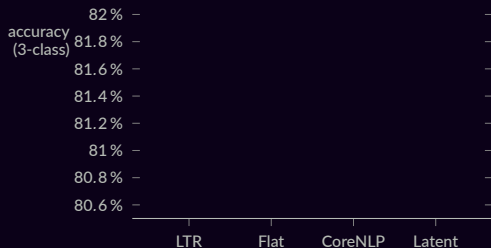
given word description, predict word embedding (Hill et al., 2016)

instead of $p(y | x)$, we model $\mathbb{E}_{p_{\pi}} \mathbf{g}(x) = \sum_{h \in \mathcal{H}} \mathbf{g}(x; h) p_{\pi}(h | x)$

Sentiment classification (SST)

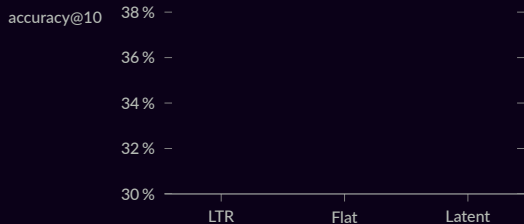


Natural Language Inference (SNLI)

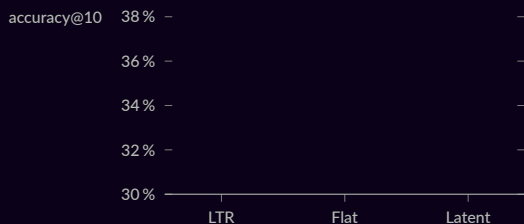


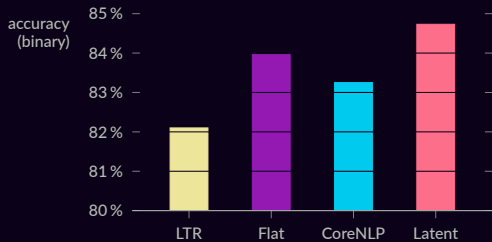
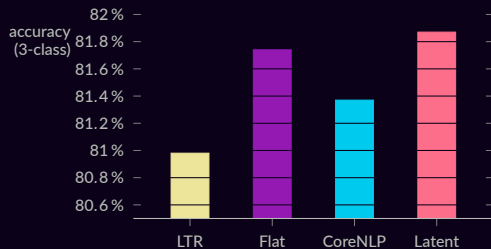
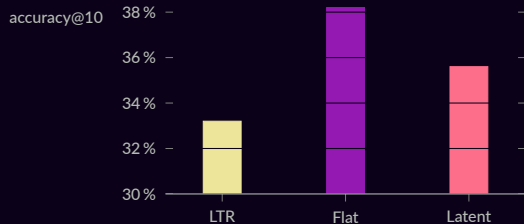
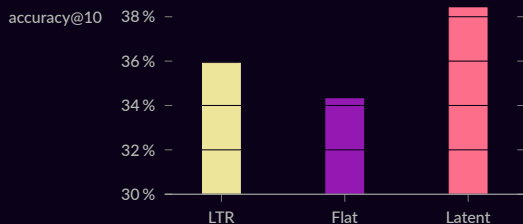
Reverse dictionary lookup

(definitions)



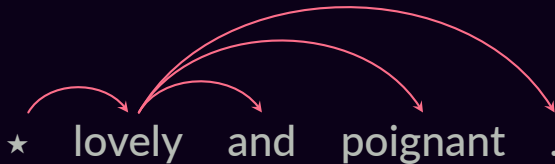
(concepts)



Sentiment classification (SST)**Natural Language Inference (SNLI)****Reverse dictionary lookup****(definitions)****(concepts)**

Syntax vs. Composition Order

CoreNLP parse, $p = 21.4\%$

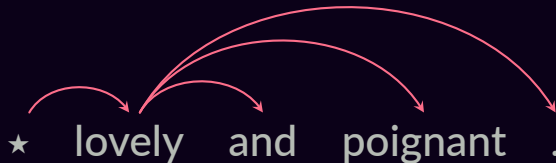


Syntax vs. Composition Order

$p = 22.6\%$

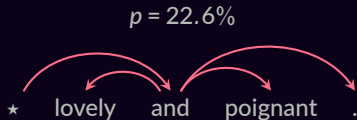


CoreNLP parse, $p = 21.4\%$

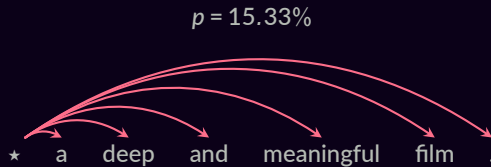


...

Syntax vs. Composition Order



CoreNLP parse, $p = 21.4\%$



$p = 15.27\%$



...

CoreNLP parse, $p = 0\%$



Structured Output Prediction

SparseMAP

$$L_A(\boldsymbol{\eta}, \bar{\boldsymbol{\mu}}) = \max_{\boldsymbol{\mu} \in \mathcal{M}} \left\{ \boldsymbol{\eta}^\top \boldsymbol{\mu} - 1/2 \|\boldsymbol{\mu}\|^2 \right\} \\ - \boldsymbol{\eta}^\top \bar{\boldsymbol{\mu}} + 1/2 \|\bar{\boldsymbol{\mu}}\|^2$$

Instance of a structured Fenchel-Young loss, like CRF, SVM, etc. (Blondel, Martins, and Niculae, 2019b)

Structured Output Prediction

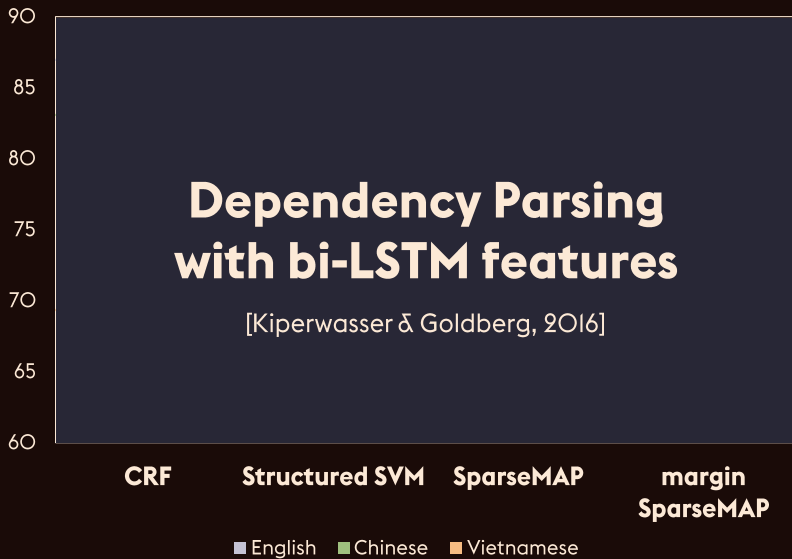
SparseMAP

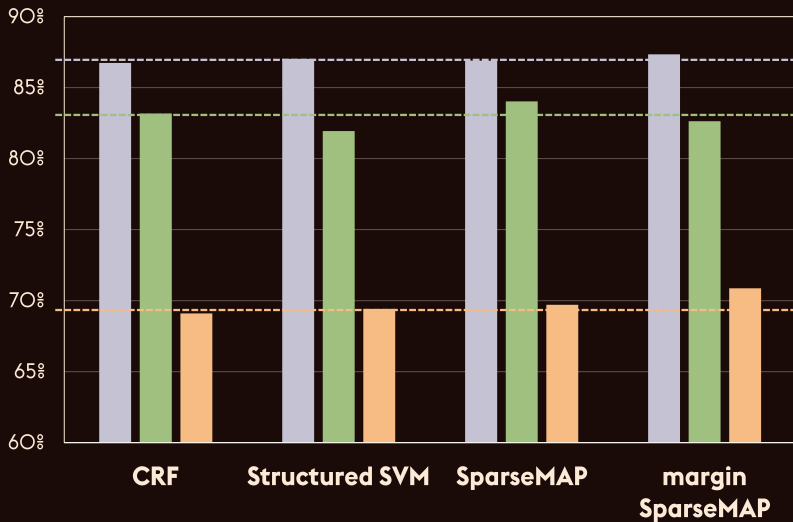
$$L_A(\boldsymbol{\eta}, \bar{\boldsymbol{\mu}}) = \max_{\boldsymbol{\mu} \in \mathcal{M}} \left\{ \boldsymbol{\eta}^\top \boldsymbol{\mu} - 1/2 \|\boldsymbol{\mu}\|^2 \right. \\ \left. - \boldsymbol{\eta}^\top \bar{\boldsymbol{\mu}} + 1/2 \|\bar{\boldsymbol{\mu}}\|^2 \right\}$$

cost-SparseMAP

$$L_A^\rho(\boldsymbol{\eta}, \bar{\boldsymbol{\mu}}) = \max_{\boldsymbol{\mu} \in \mathcal{M}} \left\{ \boldsymbol{\eta}^\top \boldsymbol{\mu} - 1/2 \|\boldsymbol{\mu}\|^2 + \rho(\boldsymbol{\mu}, \bar{\boldsymbol{\mu}}) \right\} \\ - \boldsymbol{\eta}^\top \bar{\boldsymbol{\mu}} + 1/2 \|\bar{\boldsymbol{\mu}}\|^2$$

Instance of a structured Fenchel-Young loss, like CRF, SVM, etc. (Blondel, Martins, and Niculae, 2019b)



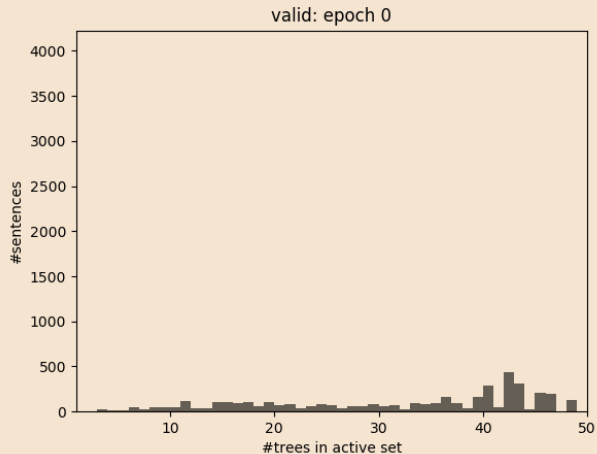
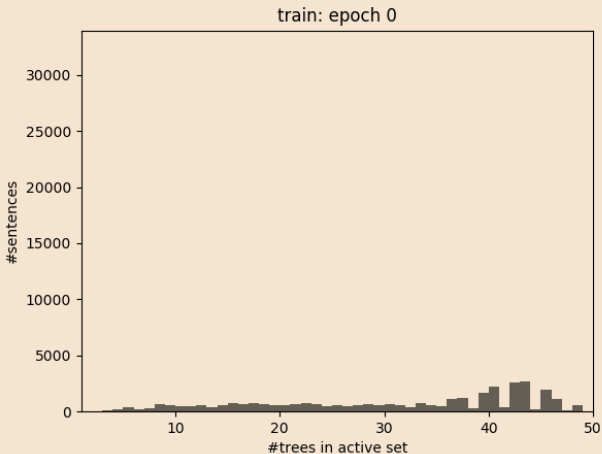


Unlabeled Accuracy (UAS)
Universal Dependencies dataset

■ English ■ Chinese ■ Vietnamese

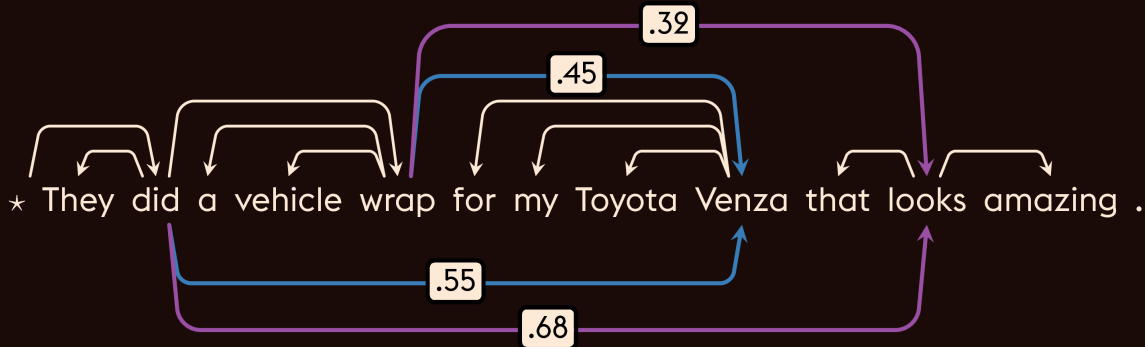
Sparse Structured Output Prediction

As models train, inference gets sparser!



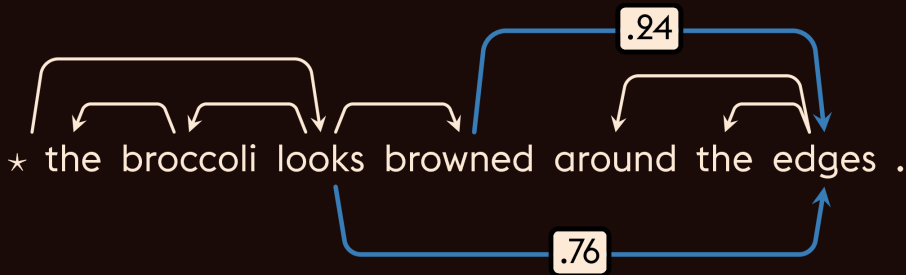
Sparse Structured Output Prediction

Inference captures linguistic ambiguity!



Sparse Structured Output Prediction

Inference captures linguistic ambiguity!



References I

- Amos, Brandon and J. Zico Kolter (2017). “OptNet: Differentiable optimization as a layer in neural networks”. In: *Proc. of ICML*.
- Andre-Obrecht, Regine (1988). “A new statistical approach for the automatic segmentation of continuous speech signals”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 36.1, pp. 29–40.
- Bertsekas, Dimitri P (1999). *Nonlinear Programming*. Athena Scientific Belmont.
- Blondel, Mathieu, André FT Martins, and Vlad Niculae (2019a). “Learning classifiers with Fenchel-Young losses: Generalized entropies, margins, and algorithms”. In: *Proc. of AISTATS*.
- (2019b). “Learning with Fenchel-Young Losses”. In: *preprint arXiv:1901.02324*.
- Brucker, Peter (1984). “An $O(n)$ algorithm for quadratic knapsack problems”. In: *Operations Research Letters* 3.3, pp. 163–166.
- Condat, Laurent (2016). “Fast projection onto the simplex and the ℓ_1 ball”. In: *Mathematical Programming* 158.1-2, pp. 575–585.
- Correia, Gonçalo M., Vlad Niculae, Wilker Aziz, et al. (2020). “Efficient marginalization of discrete and structured latent variables via sparsity”. In: *Proc. NeurIPS*.
- Correia, Gonçalo M., Vlad Niculae, and André FT Martins (2019). “Adaptively Sparse Transformers”. In: *Proc. EMNLP*.
- Corro, Caio and Ivan Titov (2019). “Learning latent trees with stochastic perturbations and differentiable dynamic programming”. In: *Proc. of ACL*.

References II

- Danskin, John M (1966). “The theory of max-min, with applications”. In: *SIAM Journal on Applied Mathematics* 14.4, pp. 641–664.
- Dantzig, George B, Alex Orden, and Philip Wolfe (1955). “The generalized simplex method for minimizing a linear form under linear inequality restraints”. In: *Pacific Journal of Mathematics* 5.2, pp. 183–195.
- Frank, Marguerite and Philip Wolfe (1956). “An algorithm for quadratic programming”. In: *Nav. Res. Log.* 3.1-2, pp. 95–110.
- Gould, Stephen et al. (2016). “On differentiating parameterized argmin and argmax problems with application to bi-level optimization”. In: *preprint arXiv:1607.05447*.
- Grünwald, Peter D and A Philip Dawid (2004). “Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory”. In: *Annals of Statistics*, pp. 1367–1433.
- Held, Michael, Philip Wolfe, and Harlan P Crowder (1974). “Validation of subgradient optimization”. In: *Mathematical Programming* 6.1, pp. 62–88.
- Hill, Felix et al. (2016). “Learning to understand phrases by embedding the dictionary”. In: *TACL* 4.1, pp. 17–30.
- Kim, Yoon et al. (2017). “Structured attention networks”. In: *Proc. of ICLR*.
- Kipf, Thomas, Elise van der Pol, and Max Welling (2020). “Contrastive Learning of Structured World Models”. In: *Proc. of ICLR*.

References III

- Kipf, Thomas and Max Welling (2017). “Semi-supervised classification with graph convolutional networks”. In: *Proc. of ICLR*.
- Koo, Terry et al. (2007). “Structured prediction models via the matrix-tree theorem”. In: *Proc. of EMNLP*.
- Kuhn, Harold W (1955). “The Hungarian method for the assignment problem”. In: *Nav. Res. Log.* 2.1-2, pp. 83–97.
- Kyrillidis, Anastasios et al. (2013). “Sparse projections onto the simplex”. In: *Proc. ICML*.
- Lacoste-Julien, Simon and Martin Jaggi (2015). “On the global linear convergence of Frank-Wolfe optimization variants”. In: *Proc. of NeurIPS*.
- Lazaridou, Angeliki, Alexander Peysakhovich, and Marco Baroni (2017). “Multi-agent cooperation and the emergence of (natural) language”. In: *Proc. ICLR*.
- Liu, Yang and Mirella Lapata (2018). “Learning structured text representations”. In: *TACL* 6, pp. 63–75.
- Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). “Fully convolutional networks for semantic segmentation”. In: *Proc. of CVPR*.
- Martins, André FT and Ramón Fernandez Astudillo (2016). “From softmax to sparsemax: A sparse model of attention and multi-label classification”. In: *Proc. of ICML*.

References IV

- McDonald, Ryan T and Giorgio Satta (2007). “On the complexity of non-projective data-driven dependency parsing”. In: *Proc. of ICPT*.
- Nangia, Nikita and Samuel Bowman (2018). “ListOps: A diagnostic dataset for latent tree learning”. In: *Proc. of NAACL SRW*.
- Niculae, Vlad and Mathieu Blondel (2017). “A regularized framework for sparse and structured neural attention”. In: *Proc. of NeurIPS*.
- Niculae, Vlad and André FT Martins (2020). “LP-SparseMAP: Differentiable relaxed optimization for sparse structured prediction”. In: *Proc. of ICML*.
- Niculae, Vlad, André FT Martins, Mathieu Blondel, et al. (2018). “SparseMAP: Differentiable sparse structured inference”. In: *Proc. of ICML*.
- Niculae, Vlad, André FT Martins, and Claire Cardie (2018). “Towards dynamic computation graphs via sparse latent structure”. In: *Proc. of EMNLP*.
- Nocedal, Jorge and Stephen Wright (1999). *Numerical Optimization*. Springer New York.
- Parikh, Ankur et al. (2016). “A decomposable attention model for natural language inference”. In: *Proc. of EMNLP*.
- Peters, Ben, Vlad Niculae, and André FT Martins (2019). “Sparse sequence-to-sequence models”. In: *Proc. ACL*.

References V

- Rabiner, Lawrence R. (1989). "A tutorial on Hidden Markov Models and selected applications in speech recognition". In: *P. IEEE* 77.2, pp. 257–286.
- Smith, David A and Noah A Smith (2007). "Probabilistic models of nonprojective dependency trees". In: *Proc. of EMNLP*.
- Tai, Kai Sheng, Richard Socher, and Christopher D Manning (2015). "Improved semantic representations from tree-structured Long Short-Term Memory networks". In: *Proc. of ACL-IJCNLP*.
- Taskar, Ben (2004). "Learning structured prediction models: A large margin approach". PhD thesis. Stanford University.
- Tibshirani, Robert et al. (2005). "Sparsity and smoothness via the fused lasso". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67.1, pp. 91–108.
- Tsallis, Constantino (1988). "Possible generalization of Boltzmann-Gibbs statistics". In: *Journal of Statistical Physics* 52, pp. 479–487.
- Valiant, Leslie G (1979). "The complexity of computing the permanent". In: *Theor. Comput. Sci.* 8.2, pp. 189–201.
- Vinyes, Marina and Guillaume Obozinski (2017). "Fast column generation for atomic norm regularization". In: *Proc. of AISTATS*.
- Wainwright, Martin J and Michael I Jordan (2008). *Graphical models, exponential families, and variational inference*. Vol. 1. 1–2. Now Publishers, Inc., pp. 1–305.

References VI

- Williams, Adina, Nikita Nangia, and Samuel R Bowman (2017). “A broad-coverage challenge corpus for sentence understanding through inference”. In: *preprint arXiv:1704.05426*.
- Wolfe, Philip (1976). “Finding the nearest point in a polytope”. In: *Mathematical Programming* 11.1, pp. 128–149.