# Learning with Sparse Latent Structure

**Vlad Niculae**

Instituto de Telecomunicações; Lisbon → University of Amsterdam (PhD opening!)

Work with: André Martins, Claire Cardie, Mathieu Blondel

github.com/deep-spin/lp-sparsemap @vnfrombucharest https://vene.ro

# Rich Underlying Structure

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

author

date

body

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

themadmovieman   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

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

themadmovieman  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**
themadmovieman 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.

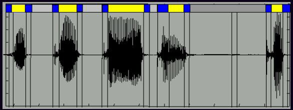
Most of this structure is **hidden**.

# Rich Underlying Structure

## Widely occuring pattern!

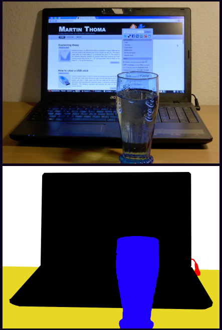
*speech*

(Andre-Obrecht, 1988)



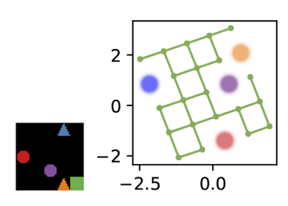
*objects*

(Long et al., 2015)




*transition graphs*

(Kipf, Pol, et al., 2020)

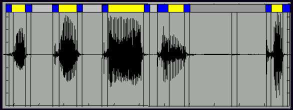


# Rich Underlying Structure

## Widely occuring pattern!

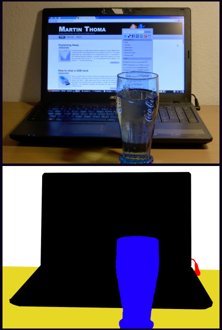
*speech*

(Andre-Obrecht, 1988)



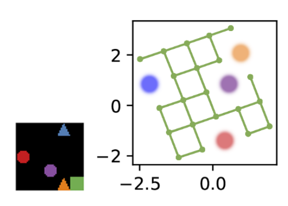
*objects*

(Long et al., 2015)



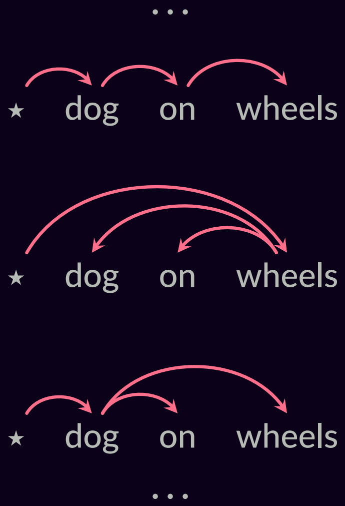

*transition graphs*

(Kipf, Pol, et al., 2020)

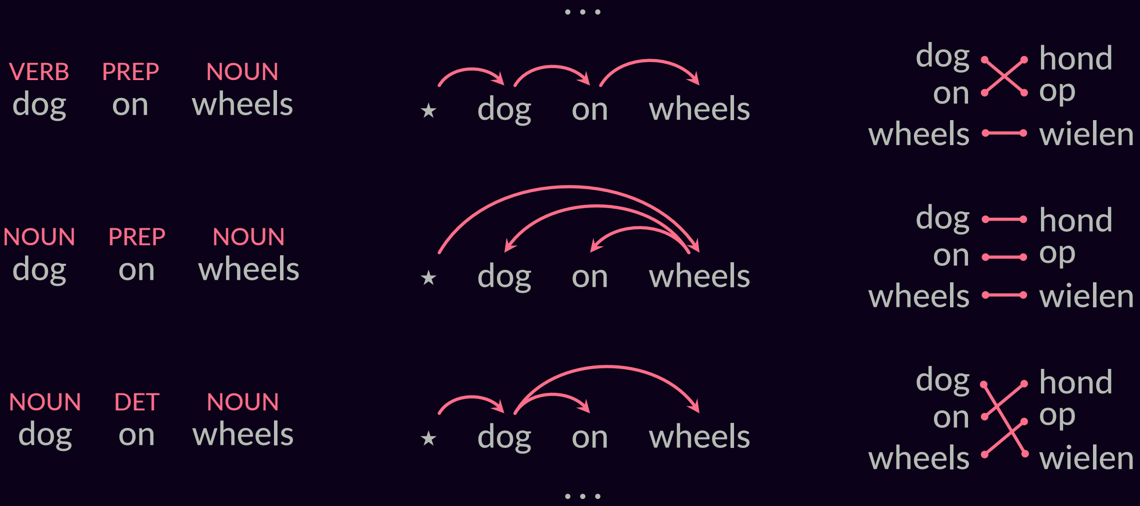



But we'll focus on NLP.

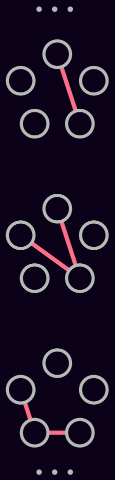
# Structured Prediction



# Structured Prediction



# Structured Prediction



# Traditional Pipeline Approach

input

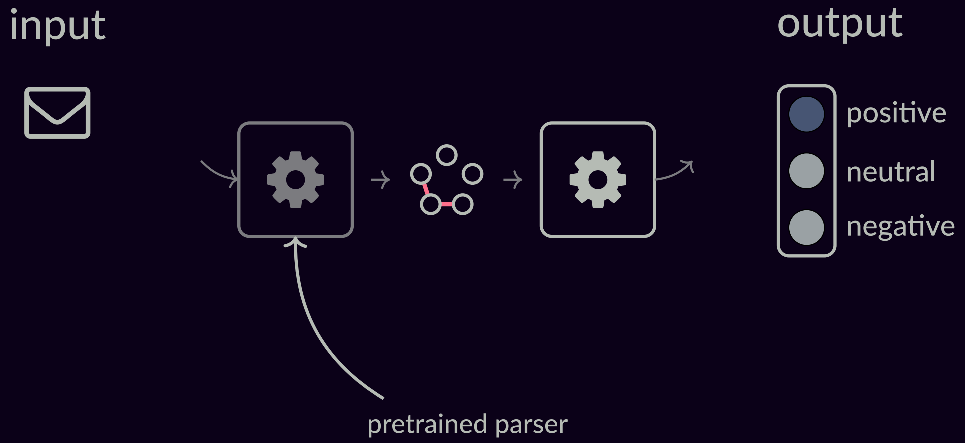
output

positive

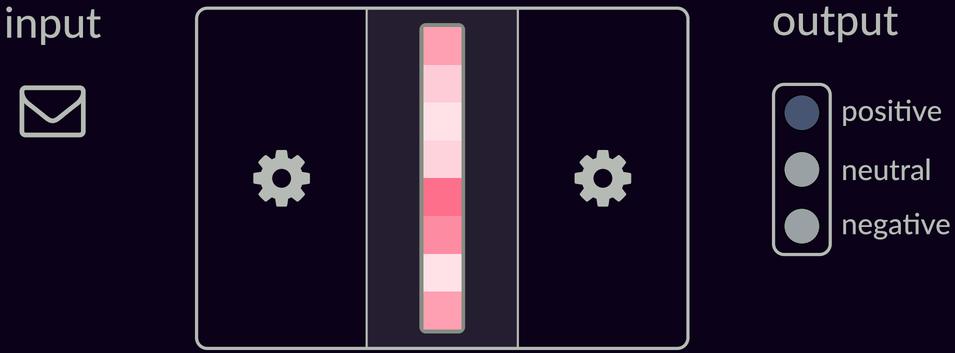
neutral

negative

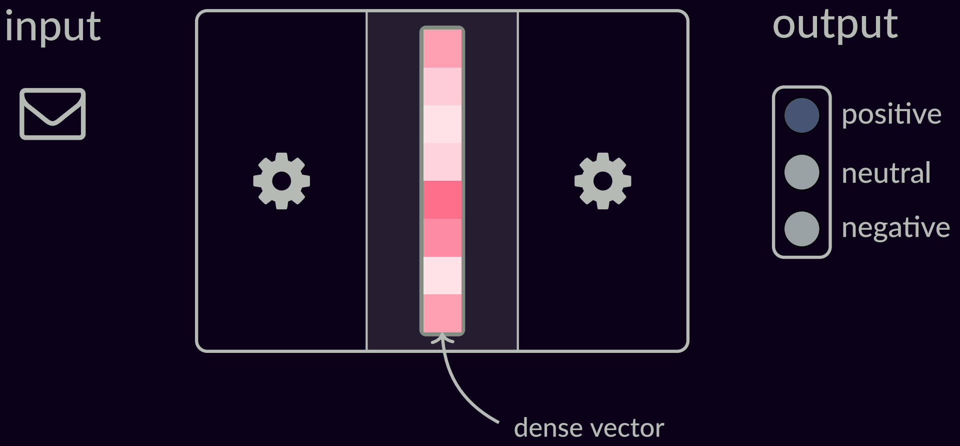
# Traditional Pipeline Approach



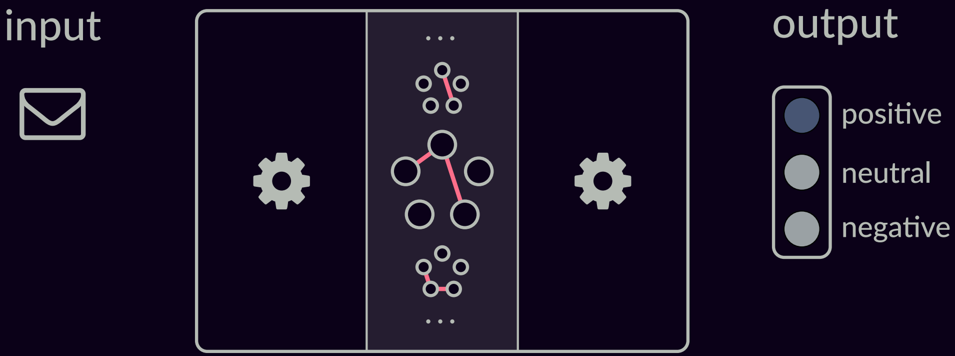
# Deep Learning & Hidden Representations



# Deep Learning & Hidden Representations



# Latent Structure Models

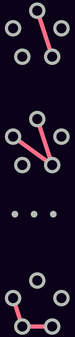


*record scratch*

*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?

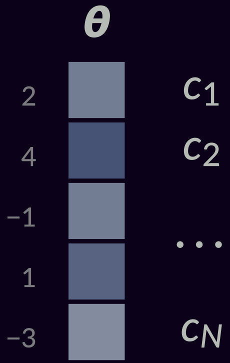
$$c_1$$

$$c_2$$

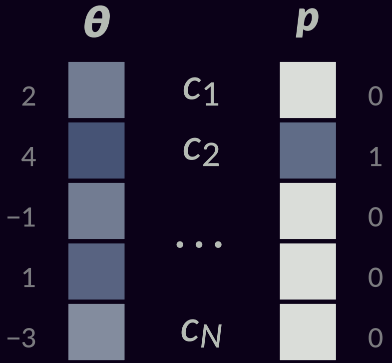
$$\dots$$

$$c_N$$

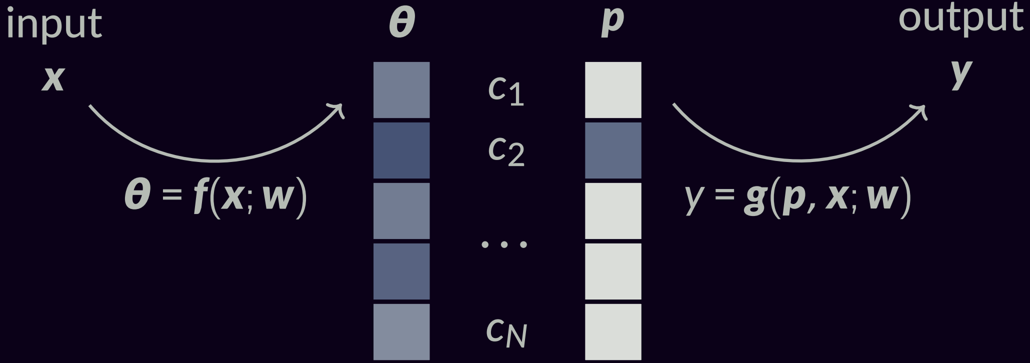
# How to select an item from a set?



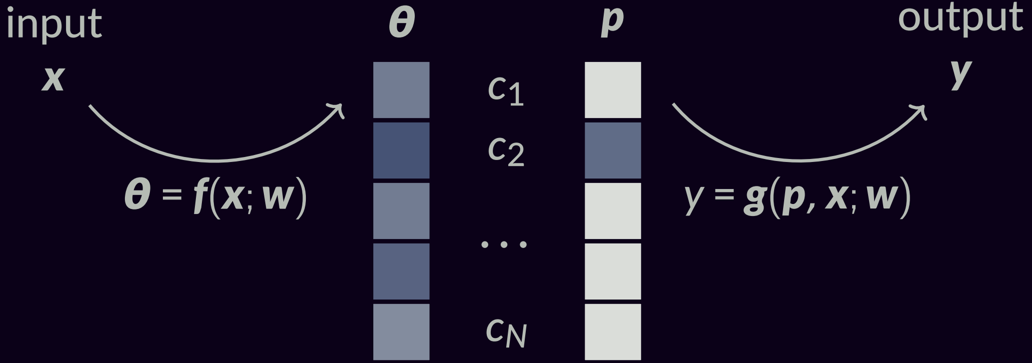
# How to select an item from a set?



# How to select an item from a set?



# How to select an item from a set?

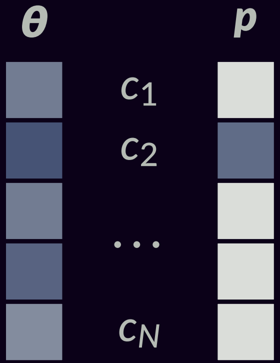



$$\frac{\partial y}{\partial \boldsymbol{w}} = ?$$

# How to select an item from a set?

input $\boldsymbol{x}$

$\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{w})$

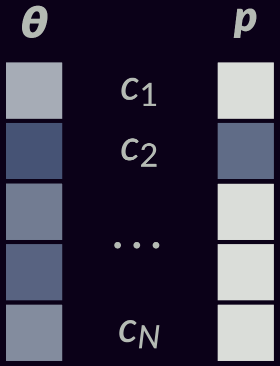



output $\boldsymbol{y}$

$y = \boldsymbol{g}(\boldsymbol{p}, \boldsymbol{x}; \boldsymbol{w})$

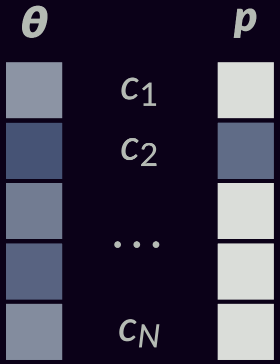
$\frac{\partial y}{\partial \boldsymbol{w}} = ?$ or, essentially, $\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$

# Argmax



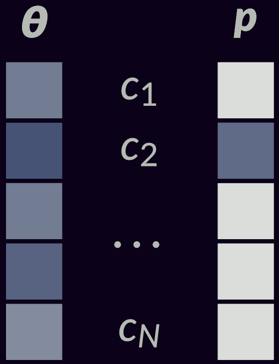

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$$

# Argmax



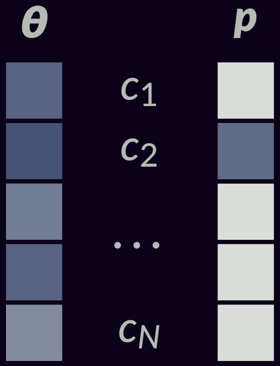

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$$

# Argmax



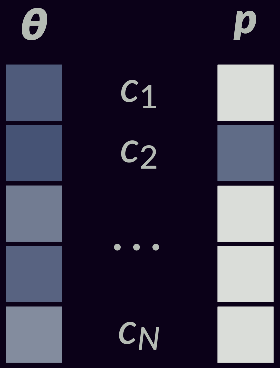

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$$

# Argmax



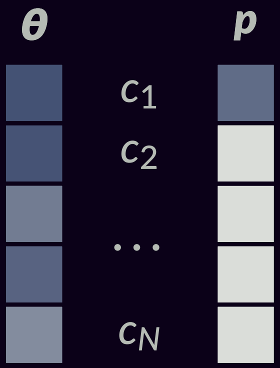

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$$

# Argmax



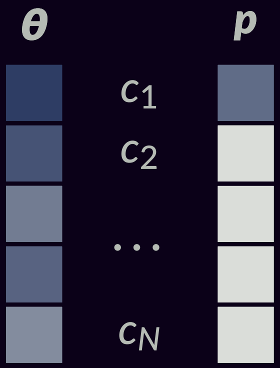

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$$

# Argmax



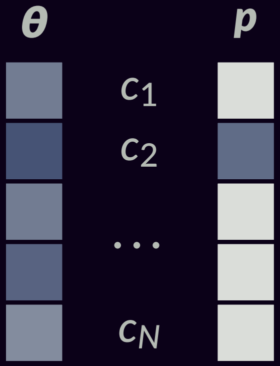

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = ?$$

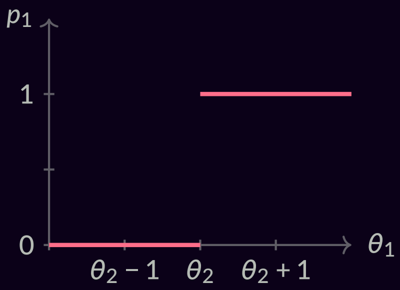
# Argmax




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

# Argmax

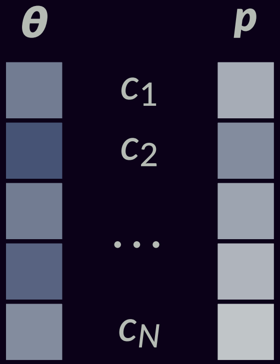


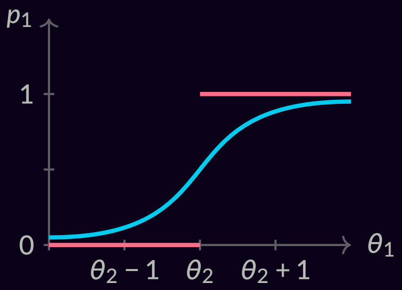




$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = \mathbf{0}$$

# Argmax vs. Softmax

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



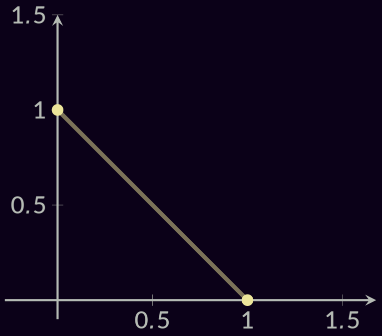

$$\frac{\partial \boldsymbol{p}}{\partial \boldsymbol{\theta}} = \operatorname{diag}(\boldsymbol{p}) - \boldsymbol{p}\boldsymbol{p}^\top$$

# A Softmax Origin Story 🦸🏽‍♀️

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

# A Softmax Origin Story 🦸🏽‍♀️

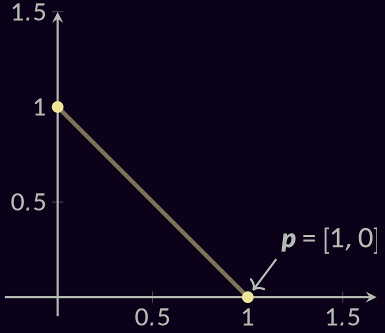
$$\Delta = \{\boldsymbol{p} \in \mathbb{R}^N : \boldsymbol{p} \geq \boldsymbol{0},\ \boldsymbol{1}^\top \boldsymbol{p} = 1\}$$




$N = 2$

# A Softmax Origin Story 🦸🏽‍♀️

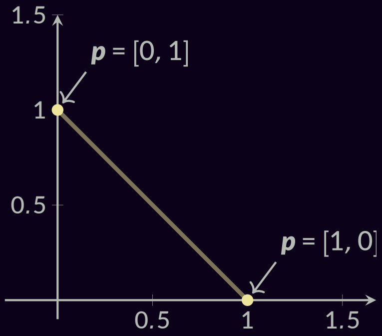
$$\Delta = \{\boldsymbol{p} \in \mathbb{R}^N : \ \boldsymbol{p} \geq \mathbf{0}, \ \mathbf{1}^\top \boldsymbol{p} = 1\}$$




$N = 2$

# A Softmax Origin Story 🦸🏽‍♀️

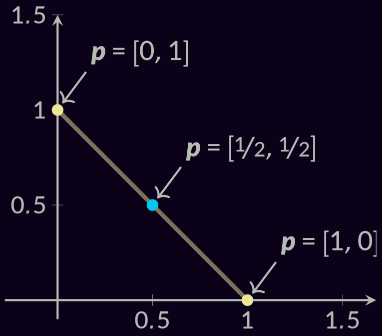
$$\Delta = \{\boldsymbol{p} \in \mathbb{R}^N : \boldsymbol{p} \geq \mathbf{0},\ \mathbf{1}^\top \boldsymbol{p} = 1\}$$




$N = 2$

# A Softmax Origin Story 🦸🏽‍♀️

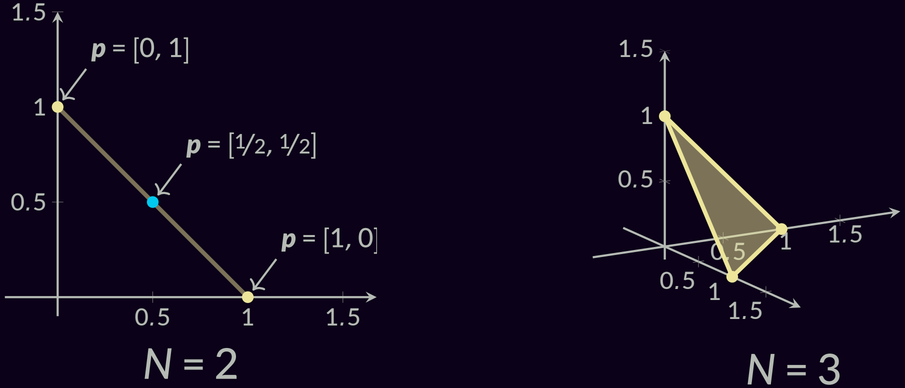
$$\Delta = \{\boldsymbol{p} \in \mathbb{R}^N : \boldsymbol{p} \geq 0,\ \mathbf{1}^\top \boldsymbol{p} = 1\}$$




$N = 2$

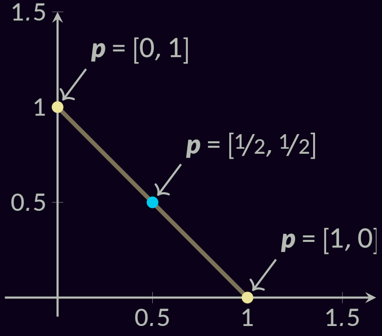
# A Softmax Origin Story 🦸🏽‍♀️

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

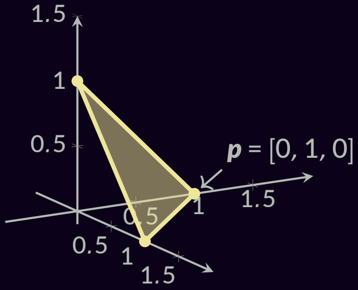


# A Softmax Origin Story 🦸🏽‍♀️

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



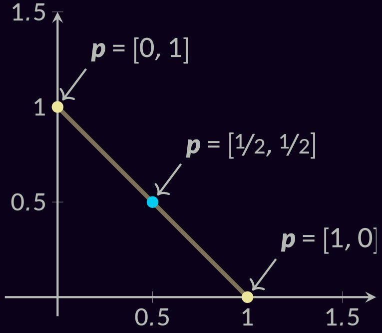

$N = 2$



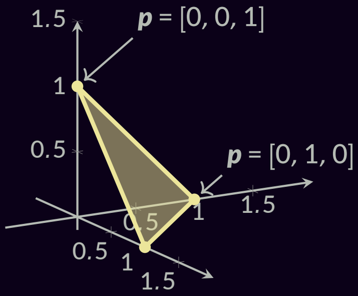

$N = 3$

# A Softmax Origin Story 🦸‍♀️

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



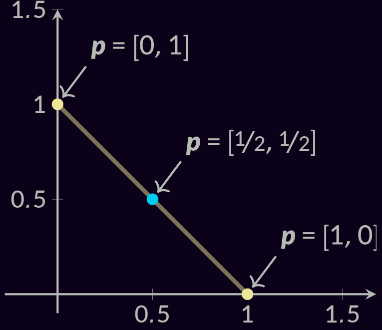

$N = 2$



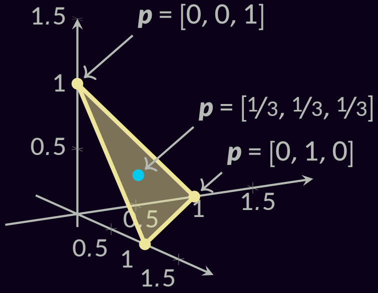

$N = 3$

# A Softmax Origin Story 🦸🏽‍♀️

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




$N = 2$

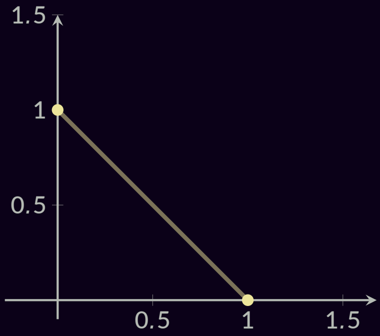



$N = 3$

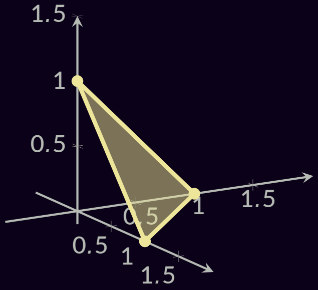
# A Softmax Origin Story 🦸‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

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




$N = 2$

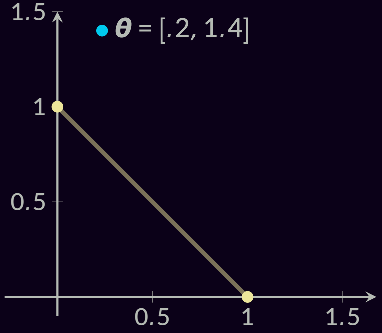



$N = 3$

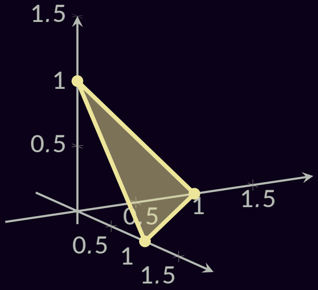
# A Softmax Origin Story 🦸🏽‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

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




$N = 2$

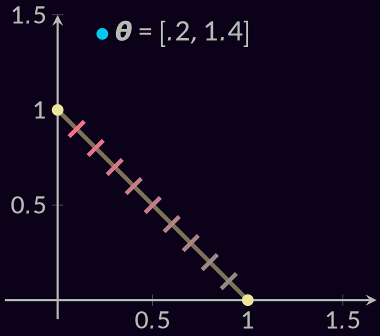



$N = 3$

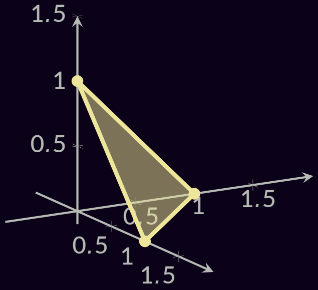
# A Softmax Origin Story 🦸🏽‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

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




$N = 2$

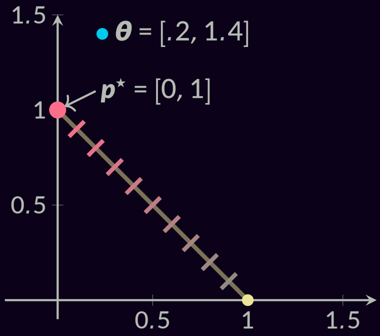



$N = 3$

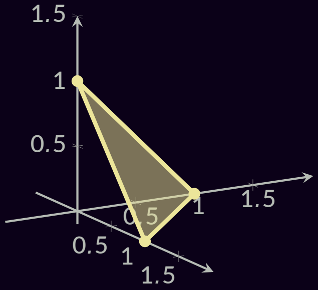
# A Softmax Origin Story 🦸🏽‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

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




$N = 2$

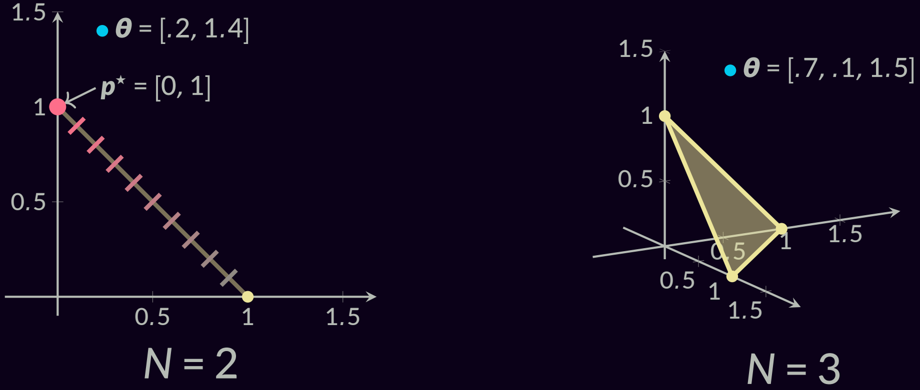



$N = 3$

# A Softmax Origin Story 🦸🏽‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

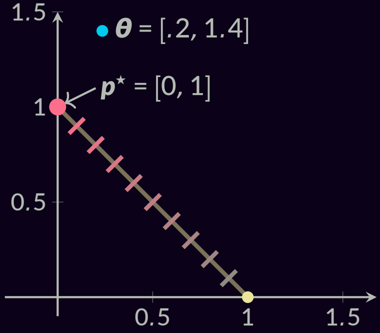
Fundamental Thm. Lin. Prog.
(Dantzig et al., 1955)



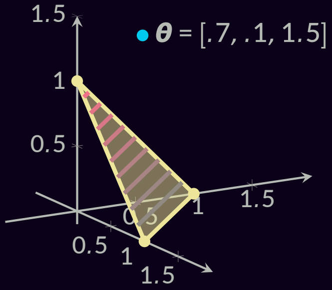
# A Softmax Origin Story 🦸🏽‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

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




$N = 2$

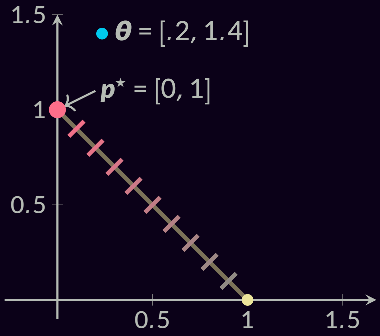



$N = 3$

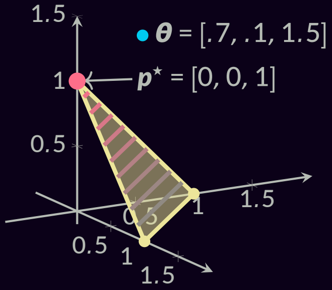
# A Softmax Origin Story 🦸🏽‍♀️

$$\max_j \theta_j = \max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$$

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




$N = 2$

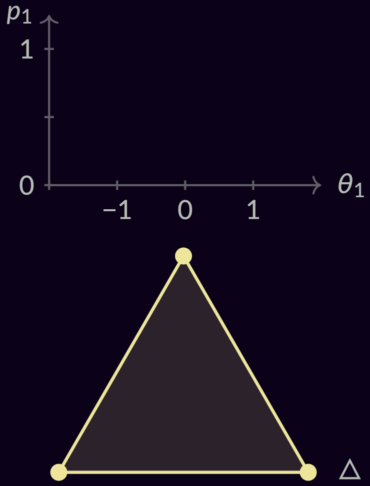



$N = 3$

(Niculae and Blondel, 2017)

# Smoothed Max Operators

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

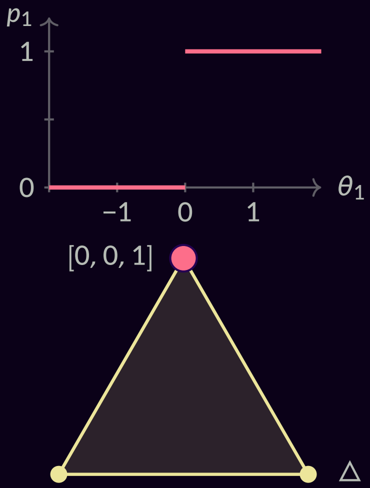


(Niculae and Blondel, 2017)

# Smoothed Max Operators

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

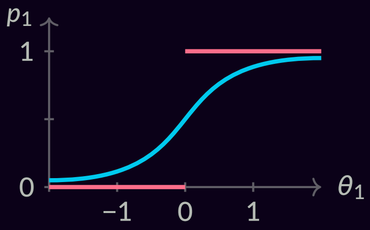
- argmax: $\Omega(\boldsymbol{p}) = 0$ *(no smoothing)*

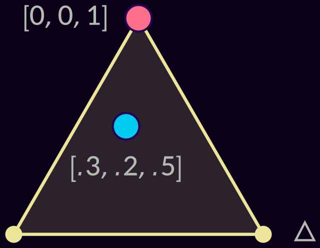


# Smoothed Max Operators

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

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



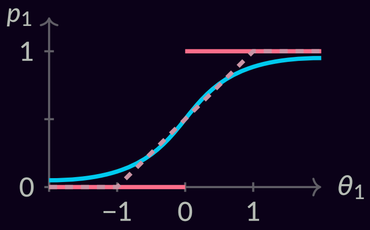



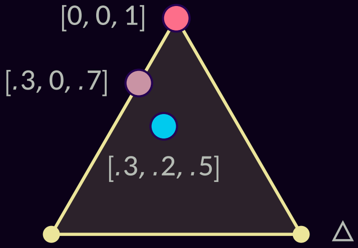
(Niculae and Blondel, 2017)

# Smoothed Max Operators

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

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



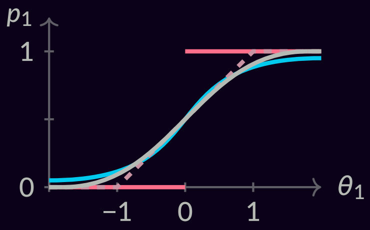

(Martins and Astudillo, 2016)

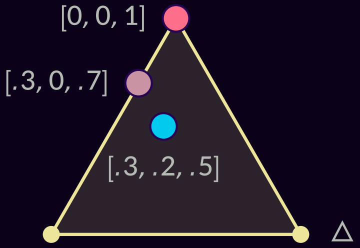
(Niculae and Blondel, 2017)

# Smoothed Max Operators

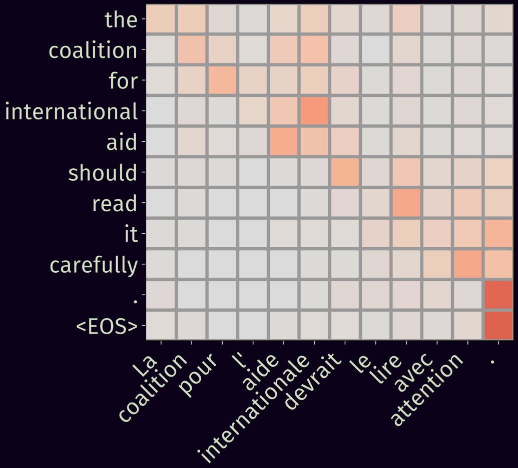
$$\boldsymbol{\pi}_\Omega(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} - \Omega(\boldsymbol{p})$$

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

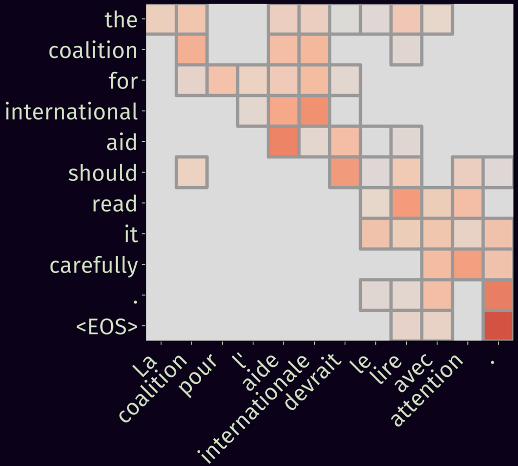


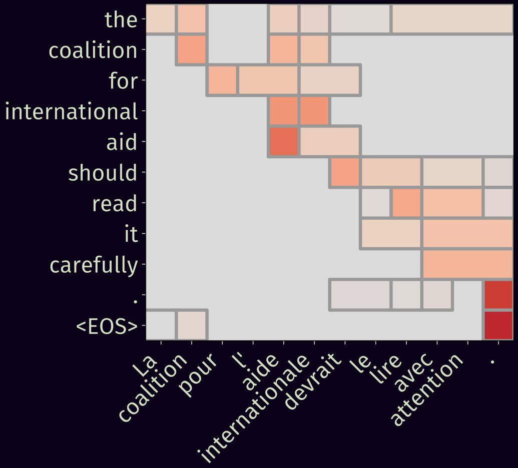

Tsallis (1988); a generalized entropy (Grünwald and Dawid, 2004)
(Blondel, Martins, and Niculae 2019a;
Peters, Niculae, and Martins 2019;
Correia, Niculae, and Martins 2019)


the
coalition
for
international
aid
should
read
it
carefully
.
<EOS>
La
coalition
pour
l'
aide
internationale
devrait
le
lire
avec
attention
.

softmax


the
coalition
for
international
aid
should
read
it
carefully
.
<EOS>
La
coalition
pour
l'
aide
internationale
devrait
le
lire
avec
attention
.

sparsemax

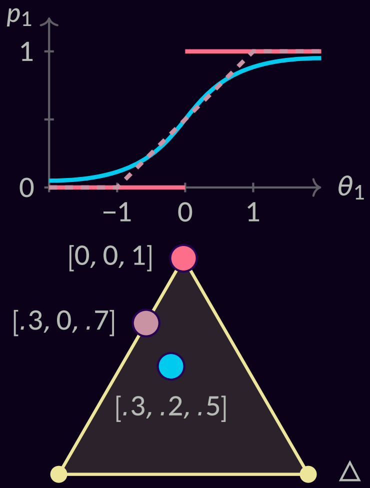

the
coalition
for
international
aid
should
read
it
carefully
.
<EOS>
La
coalition
pour
l'
aide
internationale
devrait
le
lire
avec
attention
.

fusedmax ?!

# Smoothed Max Operators

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

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

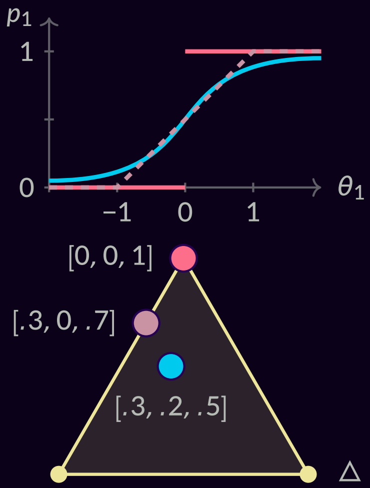


(Niculae and Blondel, 2017)

# Smoothed Max Operators

$$\pi_\Omega(\boldsymbol{\theta}) = \underset{\boldsymbol{p}\in\triangle}{\arg\max}\, \boldsymbol{p}^\top \boldsymbol{\theta} - \Omega(\boldsymbol{p})$$

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

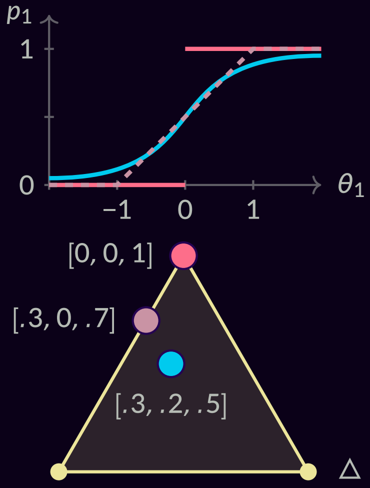


(Niculae and Blondel, 2017)

# Smoothed Max Operators

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

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

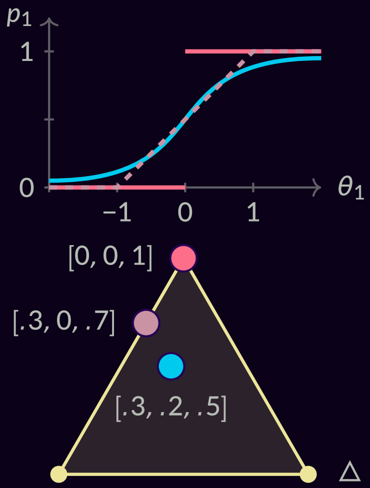


(Niculae and Blondel, 2017)

# Smoothed Max Operators

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

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

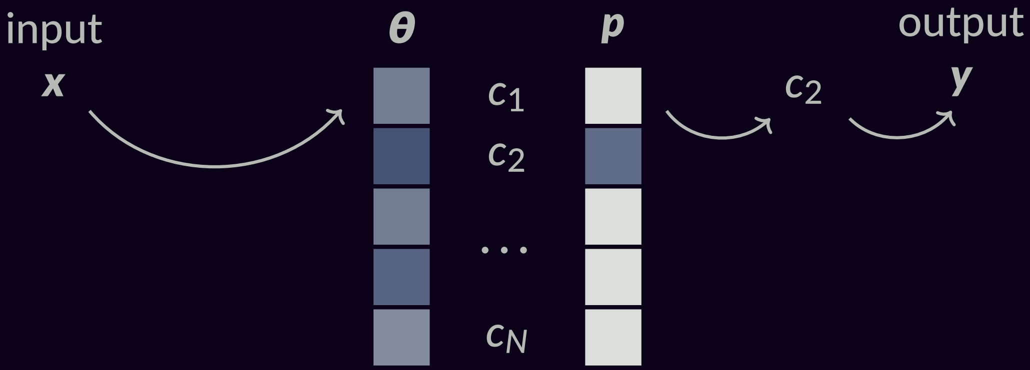


# Structured Prediction

**finally**

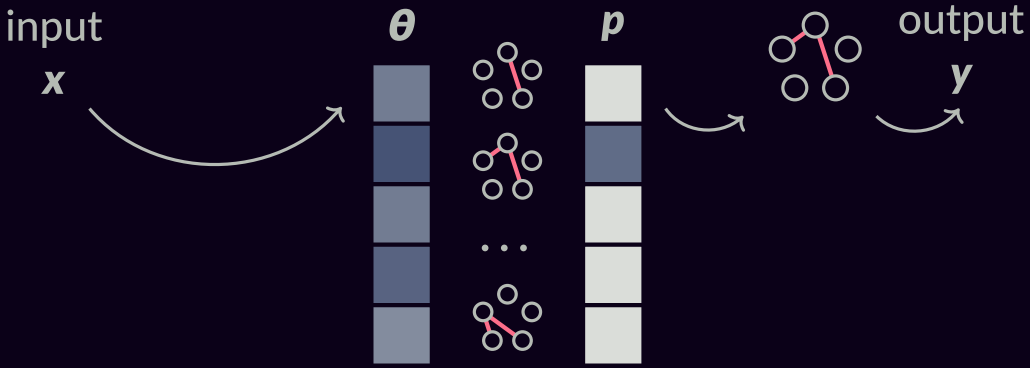
# Structured Prediction

is essentially a (very high-dimensional) argmax



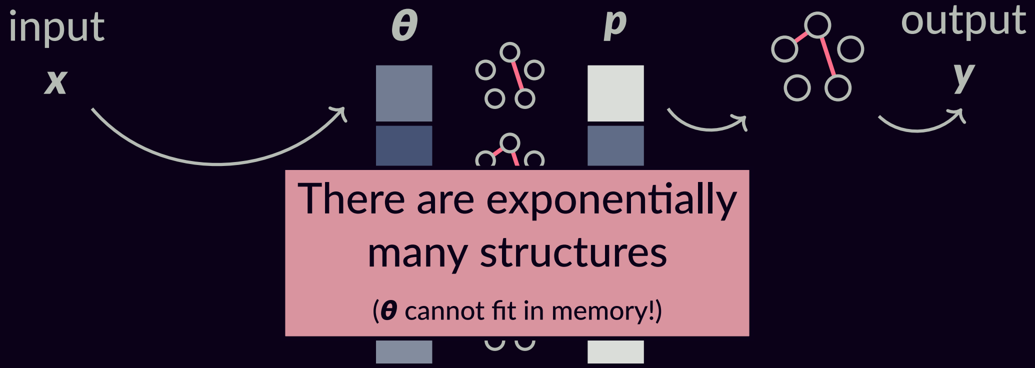
# Structured Prediction

is essentially a (very high-dimensional) argmax



# Structured Prediction

is essentially a (very high-dimensional) argmax




There are exponentially many structures

($\theta$ cannot fit in memory!)

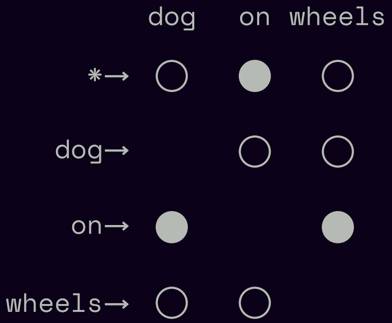
# Factorization Into Parts

$$\boldsymbol{\theta} = A^\top \boldsymbol{\eta}$$

$\star$ dog on wheels

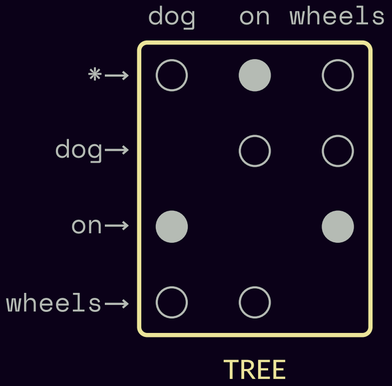
# Factorization Into Parts

$$\boldsymbol{\theta} = A^{\top} \boldsymbol{\eta}$$

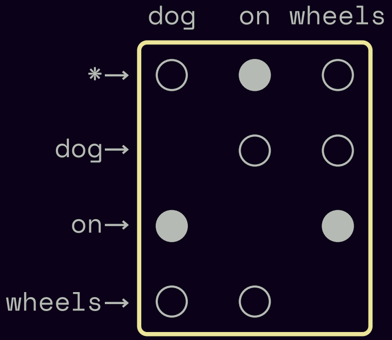
# Factorization Into Parts

$$\boldsymbol{\theta} = A^\top \boldsymbol{\eta}$$

# Factorization Into Parts

$$\boldsymbol{\theta} = A^\top \boldsymbol{\eta}$$

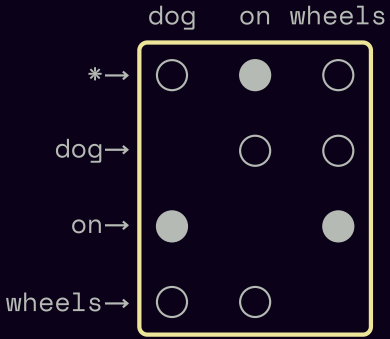

TREE

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

# Factorization Into Parts

$$\boldsymbol{\theta} = A^\top \boldsymbol{\eta}$$


$$\boldsymbol{A} = \begin{array}{r} \star\to\text{dog} \\ \text{on}\to\text{dog} \\ \text{wheels}\to\text{dog} \\ \star\to\text{on} \\ \text{dog}\to\text{on} \\ \text{wheels}\to\text{on} \\ \star\to\text{wheels} \\ \text{dog}\to\text{wheels} \\ \text{on}\to\text{wheels} \end{array} \left[\begin{array}{ccccc} 1 & & 0 & \boxed{0} & \\ 0 & & 1 & 1 & \\ 0 & & 0 & 0 & \\ \hline 0 & & 1 & 1 & \\ 1 & \dots & 0 & 0 & \dots \\ 0 & & 0 & 0 & \\ \hline 0 & & 0 & 0 & \\ 0 & & 1 & 0 & \\ 1 & & 0 & 1 & \end{array}\right] \quad \boldsymbol{\eta} = \left[\begin{array}{r} .1 \\ .2 \\ -.1 \\ \hline .3 \\ .8 \\ .1 \\ \hline -.3 \\ .2 \\ -.1 \end{array}\right]$$




TREE

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

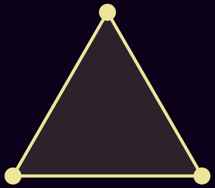
# Factorization Into Parts

$$\boldsymbol{\theta} = A^\top \boldsymbol{\eta}$$

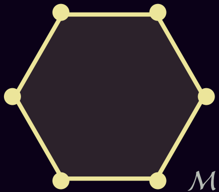


$$A = \begin{array}{r} \star\to\text{dog} \\ \text{on}\to\text{dog} \\ \text{wheels}\to\text{dog} \\ \star\to\text{on} \\ \text{dog}\to\text{on} \\ \text{wheels}\to\text{on} \\ \star\to\text{wheels} \\ \text{dog}\to\text{wheels} \\ \text{on}\to\text{wheels} \end{array} \left[\begin{array}{ccccc} 1 & & 0 & 0 & \\ 0 & & 1 & 1 & \\ 0 & & 0 & 0 & \\ \hline 0 & & 1 & 1 & \\ 1 & \dots & 0 & 0 & \dots \\ 0 & & 0 & 0 & \\ \hline 0 & & 0 & 0 & \\ 0 & & 1 & 0 & \\ 1 & & 0 & 1 & \end{array}\right] \quad \boldsymbol{\eta} = \left[\begin{array}{r} .1 \\ .2 \\ -.1 \\ \hline .3 \\ .8 \\ .1 \\ \hline -.3 \\ .2 \\ -.1 \end{array}\right]$$

$$A = \begin{array}{r} \text{dog—hond} \\ \text{dog—op} \\ \text{dog—wielen} \\ \text{on—hond} \\ \text{on—op} \\ \text{on—wielen} \\ \text{wheels—hond} \\ \text{wheels—op} \\ \text{wheels—wielen} \end{array} \left[\begin{array}{ccccc} 1 & & 0 & 0 & \\ 0 & & 1 & 1 & \\ 0 & & 0 & 0 & \\ \hline 0 & & 0 & 0 & \\ 1 & \dots & 0 & 0 & \dots \\ 0 & & 1 & 1 & \\ \hline 0 & & 1 & 0 & \\ 0 & & 0 & 0 & \\ 1 & & 0 & 1 & \end{array}\right] \quad \boldsymbol{\eta} = \left[\begin{array}{r} .1 \\ .2 \\ -.1 \\ \hline .3 \\ .8 \\ .1 \\ \hline -.3 \\ .2 \\ -.1 \end{array}\right]$$

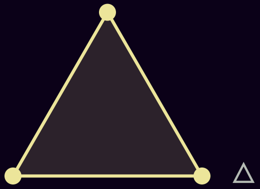


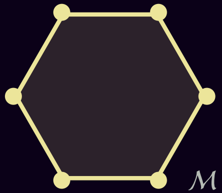
$\triangle$



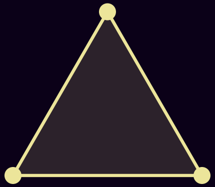
$\mathcal{M}$

$$\mathcal{M} := \operatorname{conv}\left\{\boldsymbol{a}_h : h \in \mathcal{H}\right\}$$

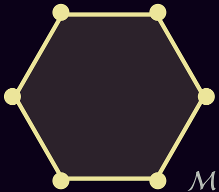


$$\mathcal{M} := \operatorname{conv}\left\{\boldsymbol{a}_h : h \in \mathcal{H}\right\}$$
$$= \left\{\boldsymbol{A}\boldsymbol{p} : \boldsymbol{p} \in \Delta\right\}$$

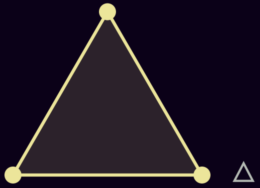


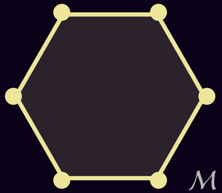
$\Delta$



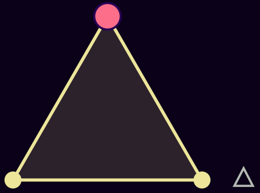
$\mathcal{M}$

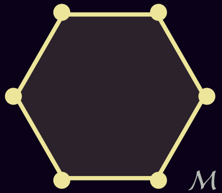
$$
\begin{aligned}
\mathcal{M} &:= \operatorname{conv}\left\{\boldsymbol{a}_h : h \in \mathcal{H}\right\} \\
&= \left\{\boldsymbol{A}\boldsymbol{p} : \boldsymbol{p} \in \Delta\right\} \\
&= \left\{\mathbb{E}_{H \sim \boldsymbol{p}}\, \boldsymbol{a}_H : \boldsymbol{p} \in \Delta\right\}
\end{aligned}
$$



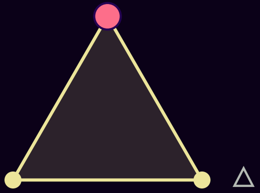



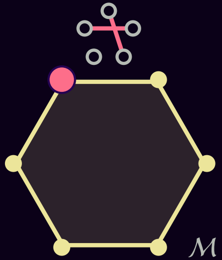
- **argmax** $\arg\max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$



- **argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$
- **MAP** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta}$

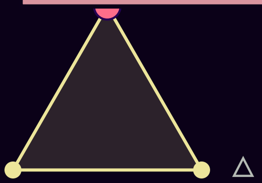


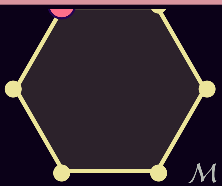



**argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$

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

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

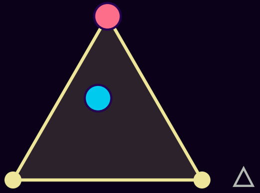


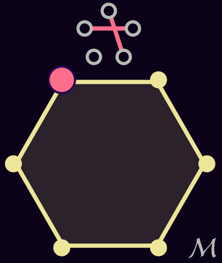



**argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta}$

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

**softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$



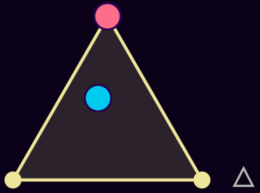



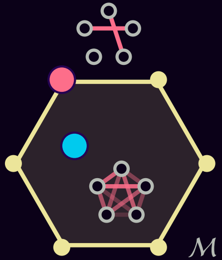
**argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta}$

**softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$

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

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



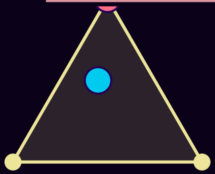
- **argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta}$
- **softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$

- **MAP** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top\boldsymbol{\eta}$
- **marginals** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top\boldsymbol{\eta} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$

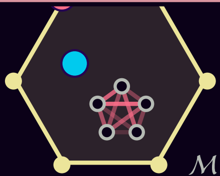
*e.g.* sequence labeling → forward-backward

(Rabiner, 1989)

As attention: (Kim et al., 2017)



$\triangle$



$\mathcal{M}$

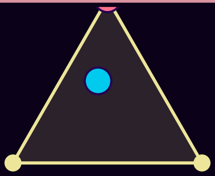
- **argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$
- **softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$

- **MAP** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta}$
- **marginals** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$

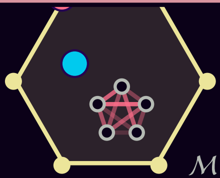
*e.g.* dependency parsing → **the Matrix-Tree theorem**

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

As attention: (Liu and Lapata, 2018)



$\triangle$



$\mathcal{M}$

**argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta}$

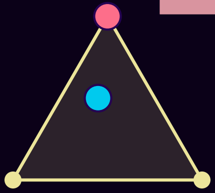
**softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$

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

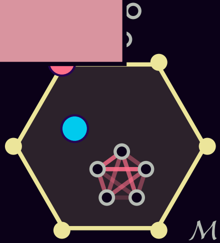
**marginals** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$

*e.g.* matchings → **#P-complete!**

(Taskar, 2004; Valiant, 1979)



$\triangle$



$\mathcal{M}$

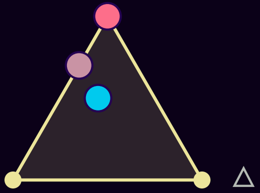
**argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta}$

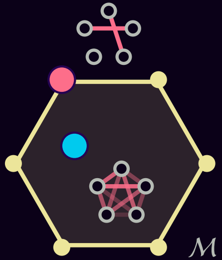
**softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$

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

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

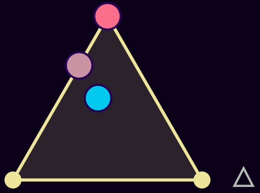
**marginals** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top\boldsymbol{\eta} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$

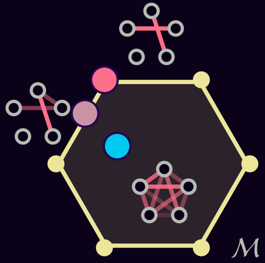


(Niculae, Martins, Blondel, and Cardie, 2018)

| | |
|---|---|
| **argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta}$ | **MAP** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top\boldsymbol{\eta}$ |
| **softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} + \mathrm{H}(\boldsymbol{p})$ | **marginals** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top\boldsymbol{\eta} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$ |
| **sparsemax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} - 1/2\|\boldsymbol{p}\|^2$ | **SparseMAP** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top\boldsymbol{\eta} - 1/2\|\boldsymbol{\mu}\|^2$ |



# Algorithms for SparseMAP

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

# Algorithms for SparseMAP

$$\boldsymbol{\mu}^\star = \underset{\boldsymbol{\mu}\in\mathcal{M}}{\arg\max}\, \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2\|\boldsymbol{\mu}\|^2$$

linear constraints
*(alas, exponentially many!)*

quadratic objective

# Algorithms for SparseMAP

$$\boldsymbol{\mu}^\star = \underset{\boldsymbol{\mu}\in\mathcal{M}}{\arg\max}\, \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2\|\boldsymbol{\mu}\|^2$$

linear constraints
*(alas, exponentially many!)*

quadratic objective

**Conditional Gradient**

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

# Algorithms for SparseMAP

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

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

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

linear constraints
*(alas, exponentially many!)*

quadratic objective

**Conditional Gradient**

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

- select a new corner of $\mathcal{M}$

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

# Algorithms for SparseMAP

$$\boldsymbol{\mu}^\star = \underset{\boldsymbol{\mu}\in\mathcal{M}}{\arg\max}\, \boldsymbol{\mu}^\top\boldsymbol{\eta} - \tfrac{1}{2}\|\boldsymbol{\mu}\|^2$$

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

$$\boldsymbol{\mu}^\star = \underset{\boldsymbol{\mu} \in \mathcal{M}}{\arg\max}\, \boldsymbol{\mu}^\top \boldsymbol{\eta} - 1/2\|\boldsymbol{\mu}\|^2$$

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
  - Quadratic objective: **Active Set**
    (Nocedal and Wright, 1999, Ch. 16.4 & 16.5)
    (Wolfe, 1976; Vinyes and Obozinski, 2017)

# Algorithms for SparseMAP

$$\boldsymbol{\mu}^\star = \arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{\eta} - {}^1\!/\!_2\|\boldsymbol{\mu}\|^2$$

linear constraints
*(alas, exponentially many!)*

quadratic objective

**Conditional Gradient**

(Frank and Wolfe, 1956; Lacost

- select a new corne
- update the (sparse)
  - Update rules: van
  - Quadratic objective: **Active Set**
    (Nocedal and Wright, 1999, Ch. 16.4 & 16.5)
    (Wolfe, 1976; Vinyes and Obozinski, 2017)

Active Set achieves
**finite** & **linear** convergence!

# Algorithms for SparseMAP

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

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
  - 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

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

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
  - 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 $\left(\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}\right)^\top \boldsymbol{dy}$

takes $\mathcal{O}(\dim(\boldsymbol{\mu})\,\mathrm{nnz}(\boldsymbol{p}^\star))$

# Algorithms for SparseMAP

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

linear constraints
*(alas, exponentially many!)*

quadratic objective

**Conditi**

(Frank and Wolfe, 1956

- select a new co

- update the (sparse) coefficients of $\boldsymbol{p}$
  - 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)

Completely modular: just add MAP

**pass**

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

computing $\left(\frac{\partial\boldsymbol{\mu}}{\partial\boldsymbol{\eta}}\right)^\top \boldsymbol{dy}$
takes $\mathcal{O}(\dim(\boldsymbol{\mu})\,\mathrm{nnz}(\boldsymbol{p}^\star))$

# SparseMAP Applications

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

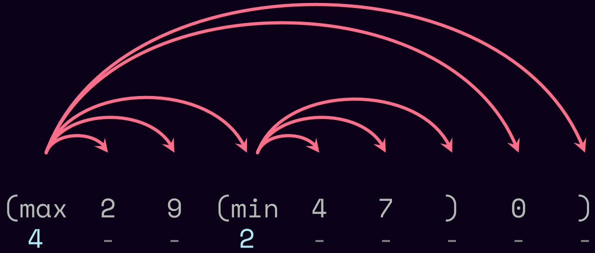
# Latent Dependency Trees

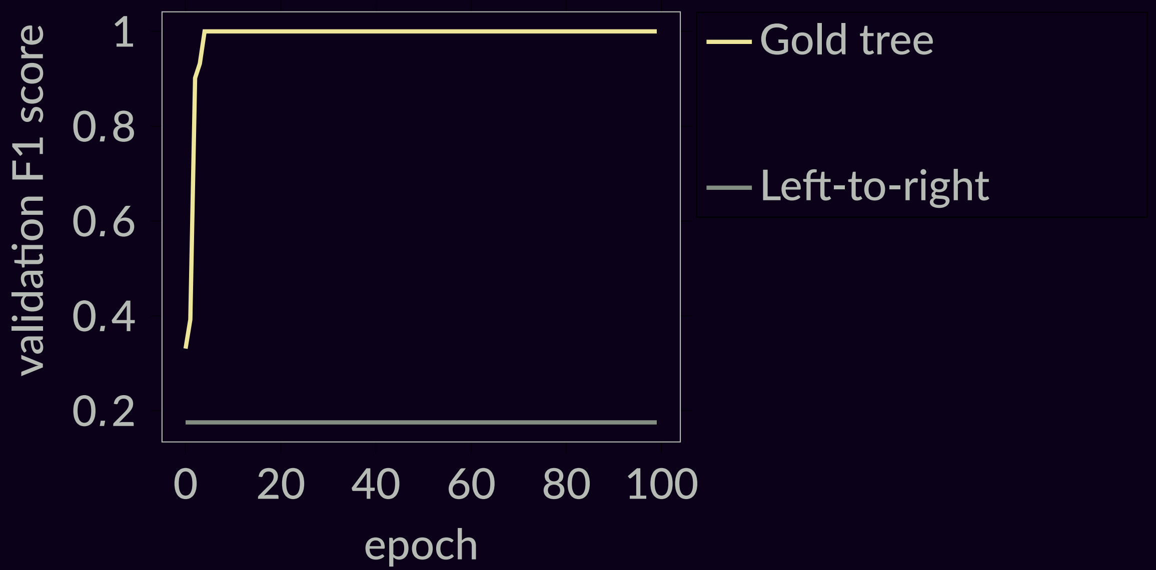
Arity tagging with latent GCN (Corro and Titov, 2019; Kipf and Welling, 2017)

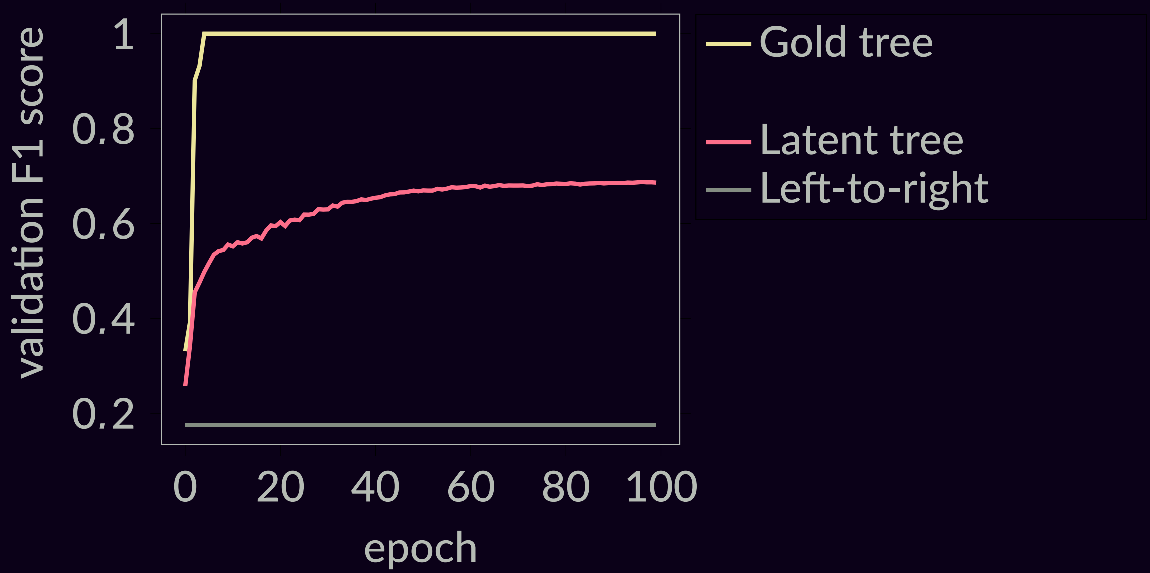
```
(max  2   9  (min  4   7   )   0   )
```

# Latent Dependency Trees

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

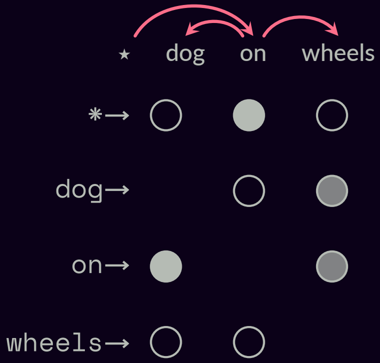



validation F1 score
1
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch
Gold tree
Left-to-right

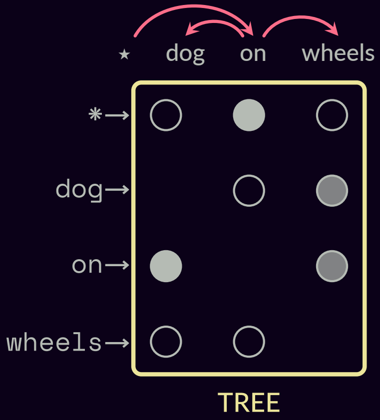

validation F1 score
1
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch
Gold tree
Latent tree
Left-to-right

# What if MAP is not available?

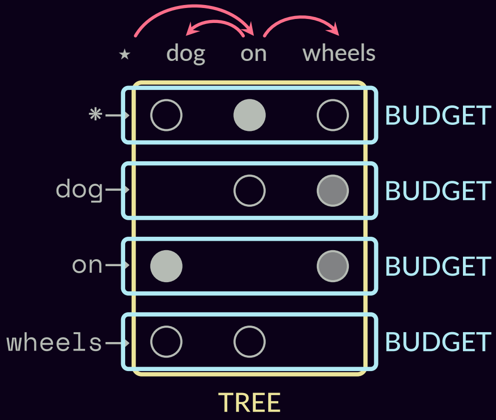
# Multiple, Overlapping Factors



# Multiple, Overlapping Factors

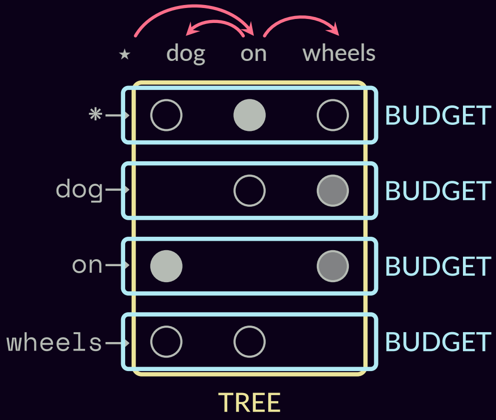


# Multiple, Overlapping Factors

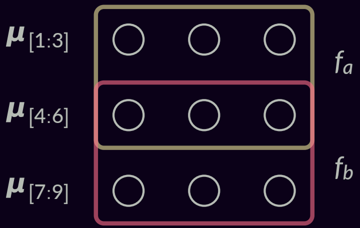


# Multiple, Overlapping Factors

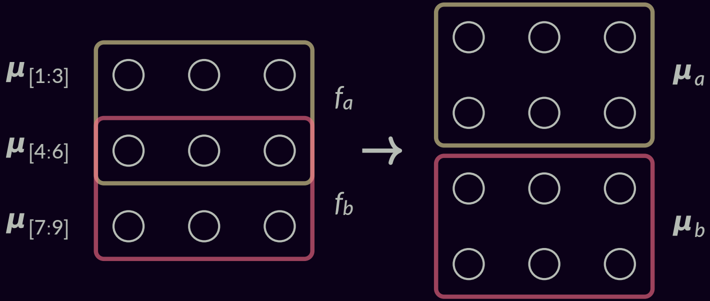
Maximization in factor graphs: NP-hard, even when each factor is tractable.



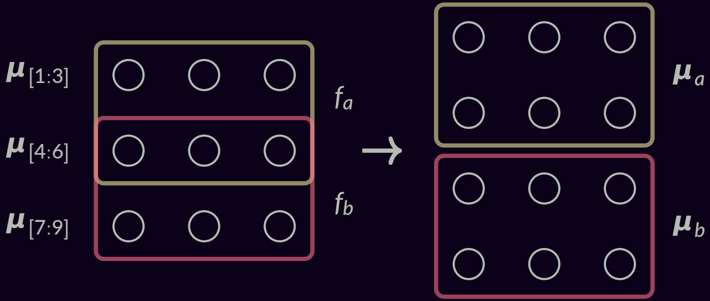
# Optimization as Consensus-Seeking



# Optimization as Consensus-Seeking

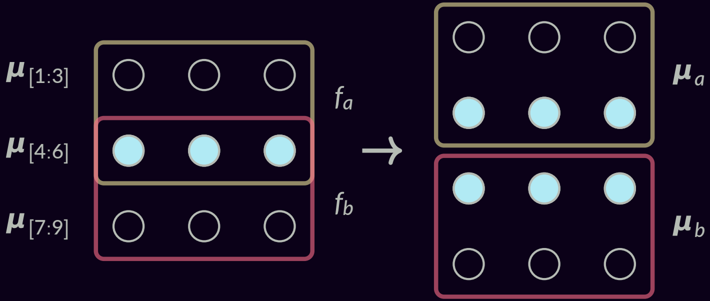


# Optimization as Consensus-Seeking




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

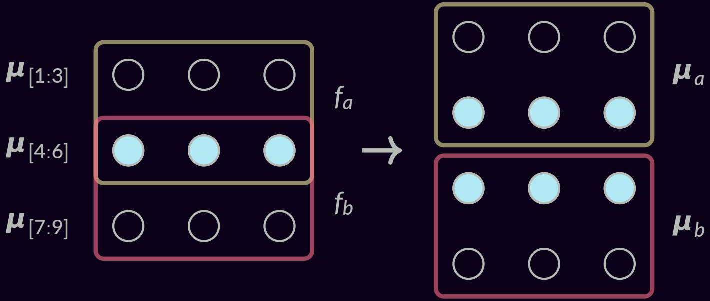
# Optimization as Consensus-Seeking




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

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

# Optimization as Consensus-Seeking




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

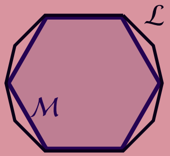
$$\max_{\boldsymbol{\mu},\boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f \quad \text{s.t.} \quad \mathbf{C}_f \boldsymbol{\mu} = \boldsymbol{\mu}_f,\ \boldsymbol{\mu}_f \in \mathcal{M}_f \ \text{ for } f \in \mathcal{F}$$

# Optim... ...eking

LP relaxation (Wainwright and Jordan, 2008)

the local polytope:

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




$\boldsymbol{\mu}_{[1:3}$ $\boldsymbol{\mu}_{[4:6}$ $\boldsymbol{\mu}_{[7:9}$

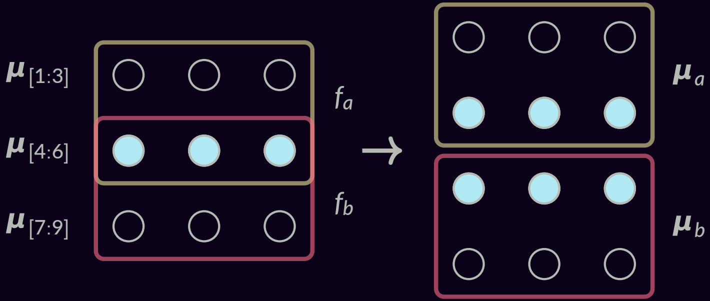
$\boldsymbol{\mu}_a$ $\boldsymbol{\mu}_b$

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

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

(Niculae and Martins, 2020)

# Optimization as Consensus-Seeking

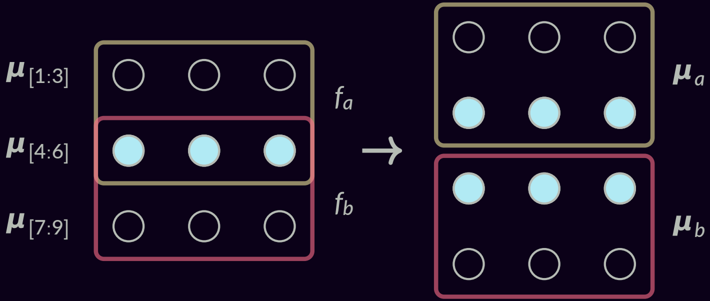



Agreement on overlap: $\boldsymbol{\mu}_{a,[4:6]} = \boldsymbol{\mu}_{b,[4:6]} = \boldsymbol{\mu}_{[4:6]}$

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

(Niculae and Martins, 2020)

# Optimization as Consensus-Seeking




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

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

# Algorithms for LP-SparseMAP

**Forward pass**

$$\operatorname*{arg\,max}_{\boldsymbol{C}_f\boldsymbol{\mu}=\boldsymbol{\mu}_f}\Big(\sum_{f\in\mathcal{F}}\boldsymbol{\eta}_f^\top\boldsymbol{\mu}_f\Big)-{}^1\!/_2\|\boldsymbol{\mu}\|^2$$

$$=\operatorname*{arg\,max}_{\boldsymbol{C}_f\boldsymbol{\mu}=\boldsymbol{\mu}_f}\sum_{f\in\mathcal{F}}\Big(\boldsymbol{\eta}_f^\top\boldsymbol{\mu}_f-{}^1\!/_2\|\boldsymbol{D}_f\boldsymbol{\mu}_f\|^2\Big)$$

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

(Niculae and Martins, 2020)

# Algorithms for LP-SparseMAP

### Forward pass

$$\arg\max_{C_f \boldsymbol{\mu} = \boldsymbol{\mu}_f} \Big( \sum_{f \in \mathcal{F}} \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f \Big) - 1/2 \|\boldsymbol{\mu}\|^2$$

$$= \arg\max_{C_f \boldsymbol{\mu} = \boldsymbol{\mu}_f} \sum_{f \in \mathcal{F}} \Big( \boldsymbol{\eta}_f^\top \boldsymbol{\mu}_f - 1/2 \|\boldsymbol{D}_f \boldsymbol{\mu}_f\|^2 \Big)$$

- Separable objective, agreement constraints

  ADMM in consensus form

- SparseMAP subproblem for each $f$

### Backward pass

- Jacobian fixed-point characterization

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

- 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 ≠ j]  # handwave

fg.add(Tree(var))

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

μ = fg.lp_sparsemap(η)
```

Factor graphs as a hidden-layer DSL!

(Niculae and Martins, 2020)

# Differentiable Sparse Structured Prediction


```
fg = FactorGraph()
var = [fg.variable() for i ≠ j]   # handwave

fg.add(Tree(var))

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

μ = fg.lp_sparsemap(η)
```

Factor graphs as a hidden-layer DSL!

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

(Niculae and Martins, 2020)

# 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(ηf):   # abstract, private
        raise NotImplemented

    def sparsemap(ηf):
        # active set algo, uses self.map

    def backward(dμf):
        # analytic, uses active set result


class Budget(Factor):
    def sparsemap(ηf):
        # specialized

    def backward(dμf):
        # specialized
```

(Niculae and Martins, 2020)

# Differentiable Sparse Structured Prediction


Factor graphs as a hidden-layer DSL!

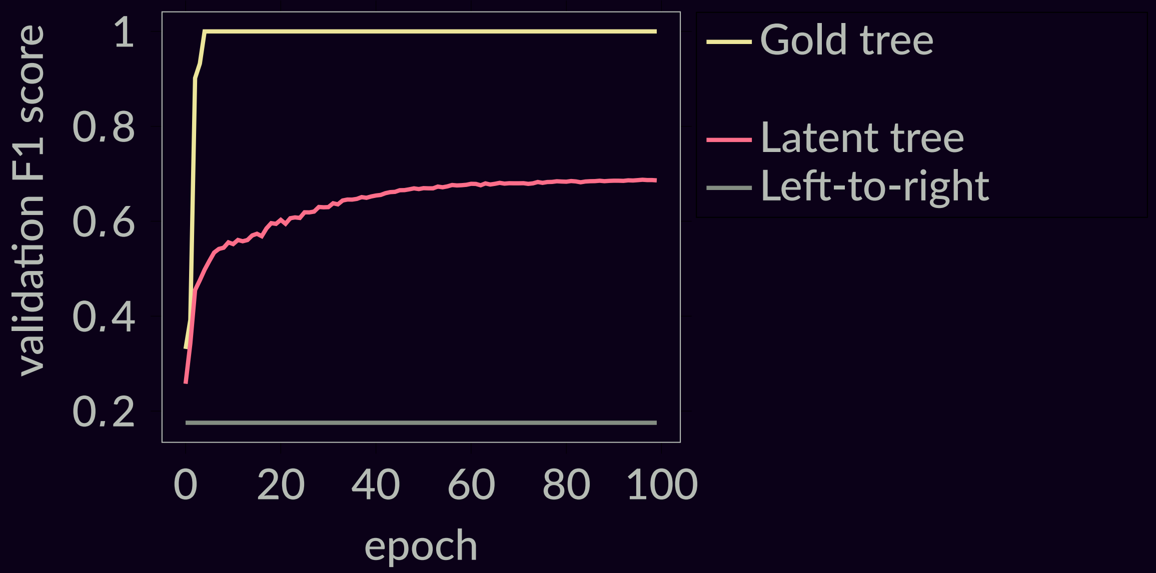
If $|\mathcal{F}| = 1$, recovers SparseMAP.

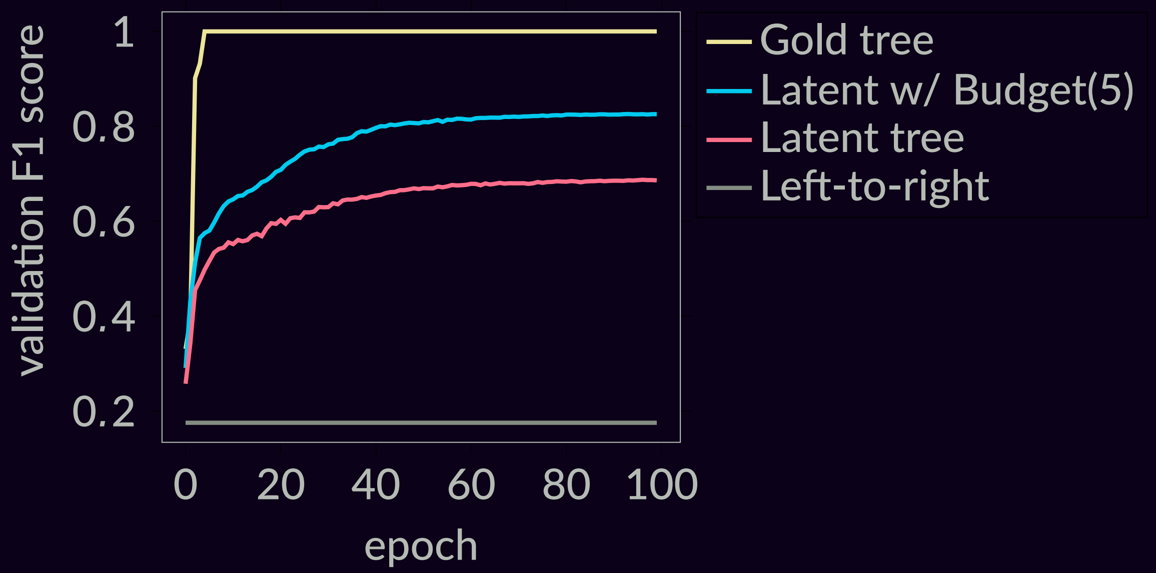
Modular library.
Built-in specialized factors:

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

New factors only require MAP.

```
class Factor:
    def map(ηf):   # abstract, private
        raise NotImplemented

    def sparsemap(ηf):
        # active set algo, uses self.map

    def backward(dμf):
        # analytic, uses active set result


class Budget(Factor):
    def sparsemap(ηf):
        # specialized

    def backward(dμf):
        # specialized


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


validation F1 score
1
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch
Gold tree
Latent tree
Left-to-right


validation F1 score
1
0.8
0.6
0.4
0.2
0
20
40
60
80
100
epoch
Gold tree
Latent w/ Budget(5)
Latent tree
Left-to-right

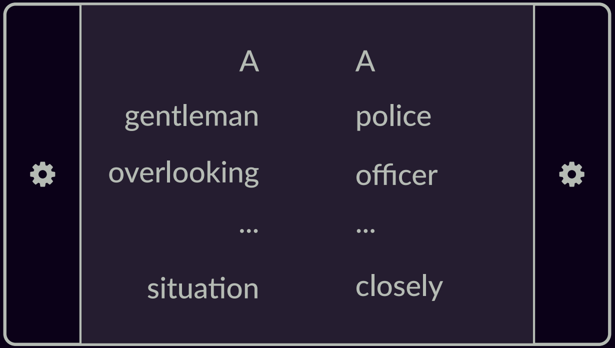
# Structured Attention for Alignments

NLI

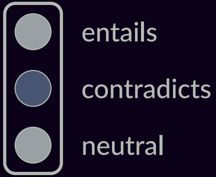
premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)




output




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

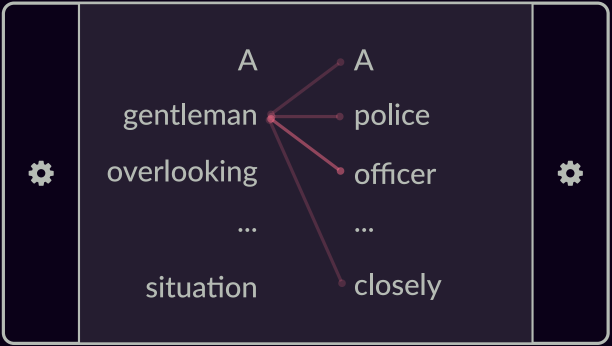
# Structured Attention for Alignments

NLI

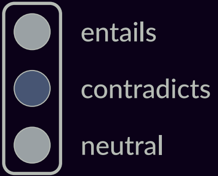
premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)




output




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

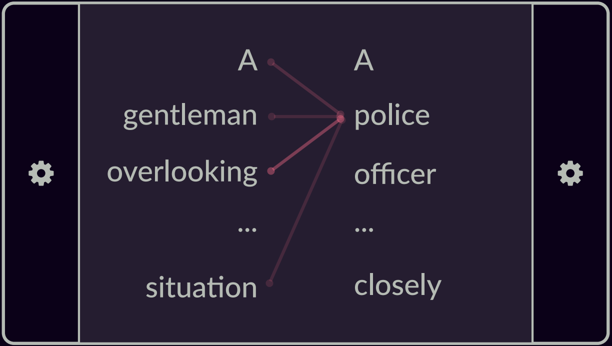
# Structured Attention for Alignments

NLI

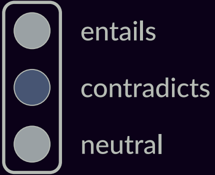
premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.

input

(P, H)




output

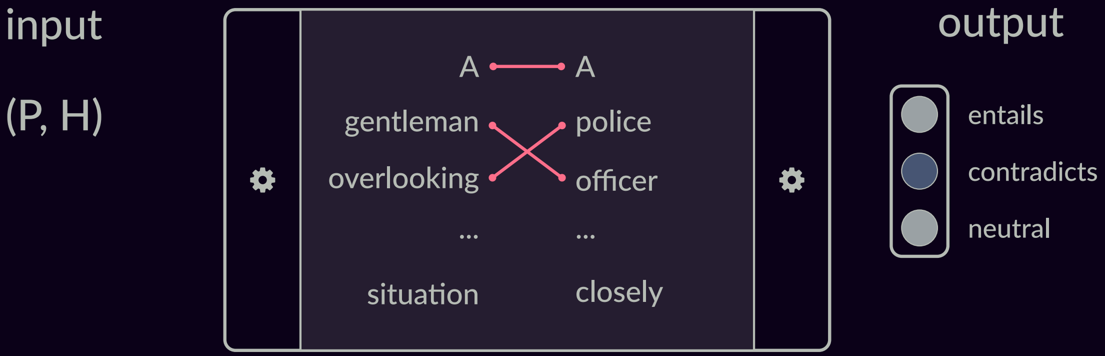



(Model: decomposable attention (Parikh et al., 2016))

# Structured Attention for Alignments

NLI

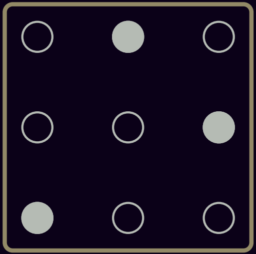
premise: A gentleman overlooking a neighborhood situation.
hypothesis: A police officer watches a situation closely.




(Proposed model: global structured alignment.)

# Structured Alignment Models

***matching***

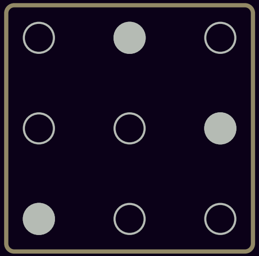


SparseMAP w/ Kuhn-Munkres

(Kuhn, 1955)

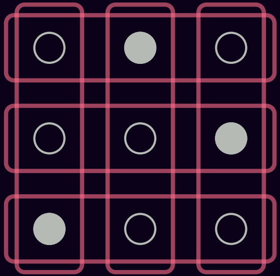
# Structured Alignment Models

***matching***



SparseMAP w/ Kuhn-Munkres
(Kuhn, 1955)

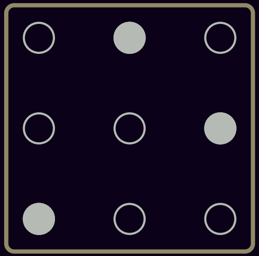
***LP-matching***



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

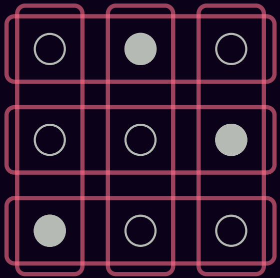
# Structured Alignment Models

### *matching*



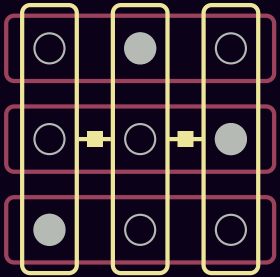
SparseMAP w/ Kuhn-Munkres
(Kuhn, 1955)

### *LP-matching*



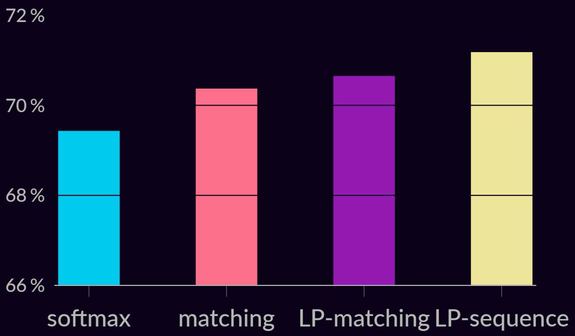
LP-SparseMAP w/ XORs
(equivalent; different solver)

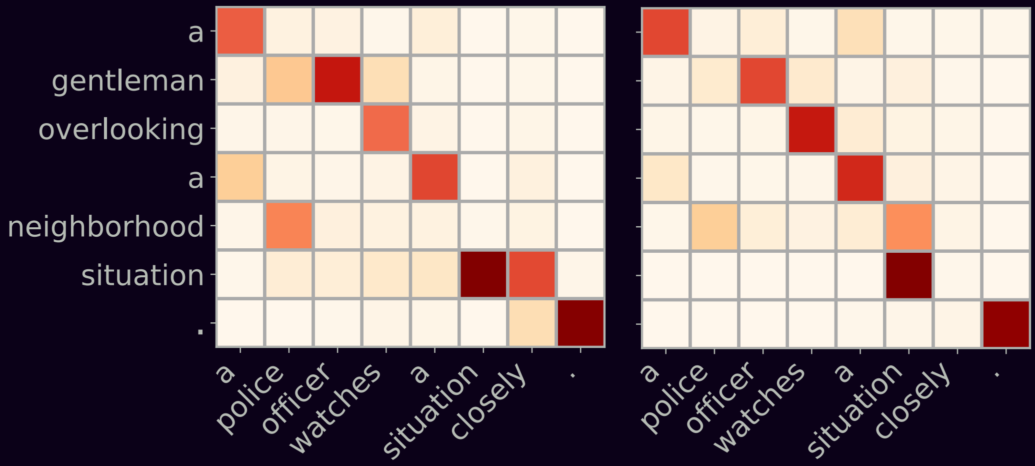
### *LP-sequence*



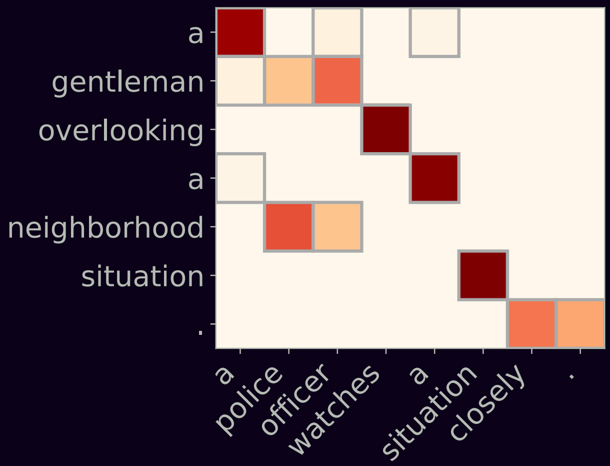
additional score
for *contiguous alignments*
$(i, j) - (i + 1, j \pm 1)$

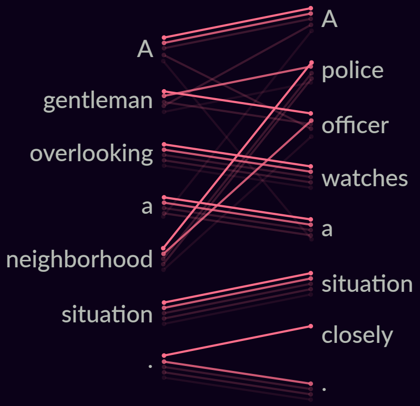
## **MultiNLI** (Williams et al., 2017)





a
gentleman
overlooking
a
neighborhood
situation
.
a
police
officer
watches
a
situation
closely
.
a
police
officer
watches
a
situation
closely
.


a
gentleman
overlooking
a
neighborhood
situation
.
a
police
officer
watches
a
situation
closely
.

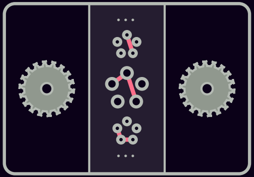

A
gentleman
overlooking
a
neighborhood
situation
.
A
police
officer
watches
a
situation
closely
.

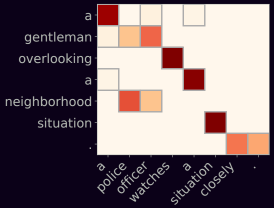
# Conclusions

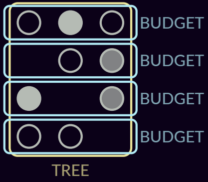
Differentiable & sparse
structured inference

Generic, extensible, efficient algorithms

Interpretable **structured attention**






vlad@vene.ro
https://vene.ro
github.com/deep-spin/lp-sparsemap
@vnfrombucharest

# Conclusions

Differentiable & sparse
structured inference

Generic, extensible, efficient algorithms

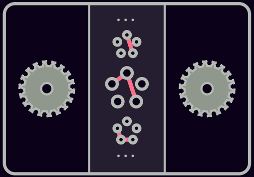
Interpretable **structured attention**

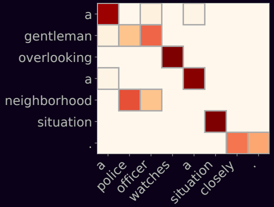
# Future work

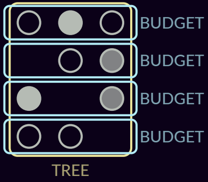
Structure beyond NLP

Weak & semi-supervision

Generative latent structure models






vlad@vene.ro
https://vene.ro
github.com/deep-spin/lp-sparsemap
@vnfrombucharest

# 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.

# Sparsemax

$$\operatorname{sparsemax}(\boldsymbol{\theta}) = \operatorname*{arg\,max}_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{\theta} - 1/2 \|\boldsymbol{p}\|_2^2$$

$$= \operatorname*{arg\,min}_{\boldsymbol{p} \in \triangle} \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2$$

# Sparsemax

$$\mathrm{sparsemax}(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta} - 1/2\|\boldsymbol{p}\|_2^2$$
$$= \arg\min_{\boldsymbol{p}\in\triangle} \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2$$

**Computation:**

$$\boldsymbol{p}^\star = [\boldsymbol{\theta} - \tau\mathbf{1}]_+$$
$$\theta_i > \theta_j \Rightarrow p_i \geq p_j$$
$\mathcal{O}(d)$ via partial sort

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

# Sparsemax

$$\text{sparsemax}(\boldsymbol{\theta}) = \operatorname*{arg\,max}_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top\boldsymbol{\theta} - {}^1\!/\!{}_2\|\boldsymbol{p}\|_2^2$$

$$= \operatorname*{arg\,min}_{\boldsymbol{p}\in\triangle} \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2$$

**Computation:**

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

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

$\mathcal{O}(d)$ via partial sort

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

**Backward pass:**

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

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

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

(Martins and Astudillo, 2016)

# Sparsemax

$$\text{sparsemax}(\boldsymbol{\theta}) = \arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta} - 1/2\|\boldsymbol{p}\|_2^2$$

$$= \arg\min_{\boldsymbol{p}\in\triangle} \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2$$

**Computation:**

$\boldsymbol{p}^\star = [($

$\theta_i > \theta_j$

$\mathcal{O}(d)$ via

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

**argmin differentiation**

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

**Backward pass:**

$g(\boldsymbol{s}) - \frac{1}{|\mathcal{S}|}\boldsymbol{s}\boldsymbol{s}^\top$

$: p_j^\star > 0\},$

$\in \mathcal{S}]$

(Martins and Astudillo, 2016)

# Fusedmax

$$\mathrm{fusedmax}(\boldsymbol{\theta}) = \underset{\boldsymbol{p}\in\triangle}{\arg\max}\, \boldsymbol{p}^\top\boldsymbol{\theta} - 1/2\|\boldsymbol{p}\|_2^2 - \sum_{2\le j\le d} |p_j - p_{j-1}|$$

$$= \underset{\boldsymbol{p}\in\triangle}{\arg\min}\, \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2 + \sum_{2\le j\le d} |p_j - p_{j-1}|$$

$$\mathrm{prox}_{\mathrm{fused}}(\boldsymbol{\theta}) = \underset{\boldsymbol{p}\in\mathbb{R}^d}{\arg\min}\, \|\boldsymbol{p} - \boldsymbol{\theta}\|_2^2 + \sum_{2\le j\le d} |p_j - p_{j-1}|$$

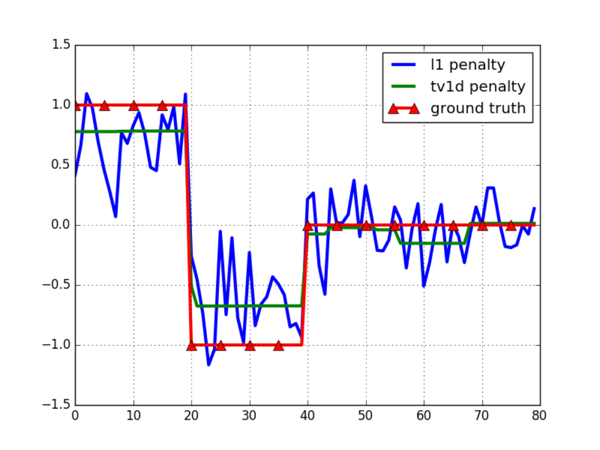
**Proposition:** $\mathrm{fusedmax}(\boldsymbol{\theta}) = \mathrm{sparsemax}\left(\mathrm{prox}_{\mathrm{fused}}(\boldsymbol{\theta})\right)$

(Niculae and Blondel, 2017)

fusedmax( … $|p_j - p_{j-1}|$

prox$_{\text{fused}}$( … $_{j-1}|$

… $_{j-1}|$




"Fused Lasso" a.k.a. 1-d Total Variation

(Tibshirani et al., 2005)

**Proposi** … $_{\text{used}}(\boldsymbol{\theta}))$

(Niculae and Blondel, 2017)

(Danskin, 1966; Prop. B.25 in Bertsekas, 1999)

# Danskin's Theorem

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

$$\partial \max_{\boldsymbol{z} \in \mathcal{Z}} \phi(\boldsymbol{x}, \boldsymbol{z}) = \operatorname{conv}\{\nabla_{\boldsymbol{x}} \phi(\boldsymbol{x}, \boldsymbol{z}^\star) \mid \boldsymbol{z}^\star \in \arg\max_{\boldsymbol{z} \in \mathcal{Z}} \phi(\boldsymbol{x}, \boldsymbol{z})\}.$$

**Example: maximum of a vector**

(Danskin, 1966; Prop. B.25 in Bertsekas, 1999)

# Danskin's Theorem

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

$$\partial \max_{\boldsymbol{z} \in \mathcal{Z}} \phi(\boldsymbol{x}, \boldsymbol{z}) = \operatorname{conv}\{\nabla_{\boldsymbol{x}} \phi(\boldsymbol{x}, \boldsymbol{z}^\star) \mid \boldsymbol{z}^\star \in \arg\max_{\boldsymbol{z} \in \mathcal{Z}} \phi(\boldsymbol{x}, \boldsymbol{z})\}.$$

**Example: maximum of a vector**

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

(Danskin, 1966; Prop. B.25 in Bertsekas, 1999)

# Danskin's Theorem

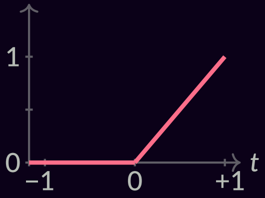
Let $\phi : \mathbb{R}^d \times \mathcal{Z} \to \mathbb{R}$, $\mathcal{Z} \subset \mathbb{R}^d$ compact.

$$\partial \max_{\boldsymbol{z}\in\mathcal{Z}} \phi(\boldsymbol{x}, \boldsymbol{z}) = \operatorname{conv}\left\{ \nabla_{\boldsymbol{x}} \phi(\boldsymbol{x}, \boldsymbol{z}^\star) \mid \boldsymbol{z}^\star \in \arg\max_{\boldsymbol{z}\in\mathcal{Z}} \phi(\boldsymbol{x}, \boldsymbol{z}) \right\}.$$

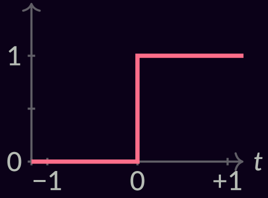
## Example: maximum of a vector

$$\begin{aligned}
\partial \max_{j\in[d]} \theta_j &= \partial \max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{\theta} \\
&= \partial \max_{\boldsymbol{p}\in\triangle} \phi(\boldsymbol{p}, \boldsymbol{\theta}) \\
&= \operatorname{conv}\left\{ \nabla_{\boldsymbol{\theta}} \phi(\boldsymbol{p}^\star, \boldsymbol{\theta}) \right\} \\
&= \operatorname{conv}\left\{ \boldsymbol{p}^\star \right\}
\end{aligned}$$

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



$\max_j \theta_j$



$\{ g_1 \mid \boldsymbol{g} \in \partial \max_j \theta_j \}$

# Dynamically inferring the computation graph

So far: a structured hidden layer

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

Network must handle “soft” combinations of structures.

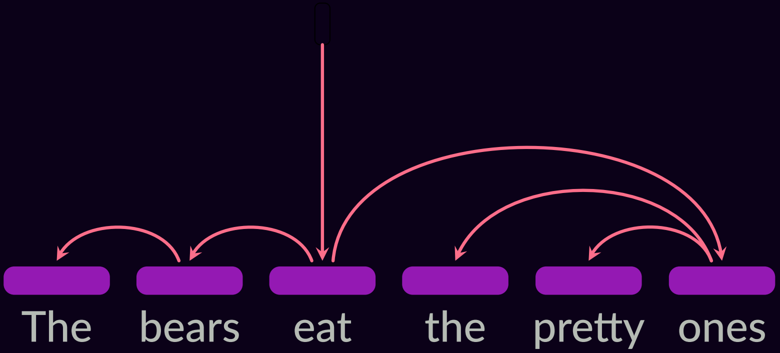
Fine for attention, but can be limiting.

# Dependency TreeLSTM

(Tai et al., 2015)

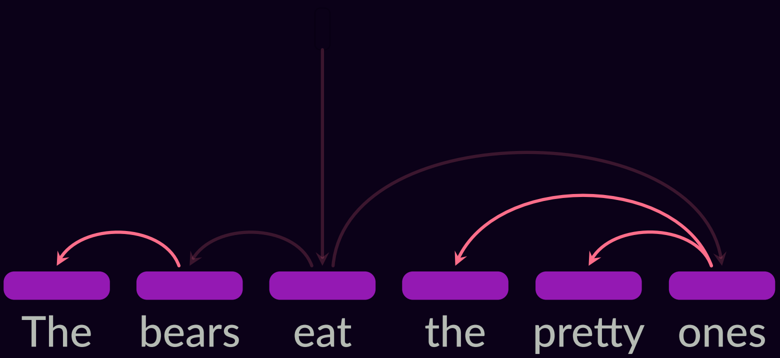
# Dependency TreeLSTM

(Tai et al., 2015)



# Dependency TreeLSTM

(Tai et al., 2015)

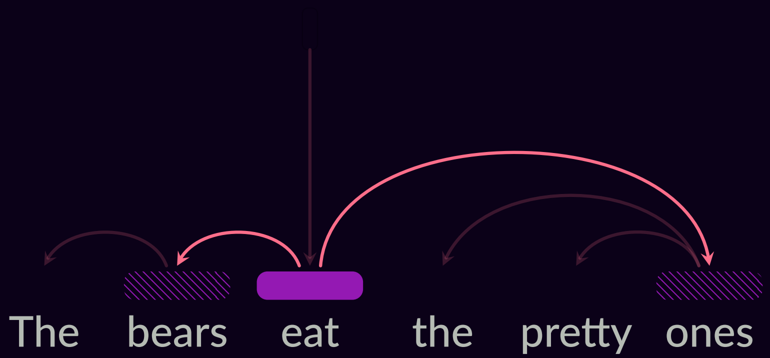


# Dependency TreeLSTM

(Tai et al., 2015)

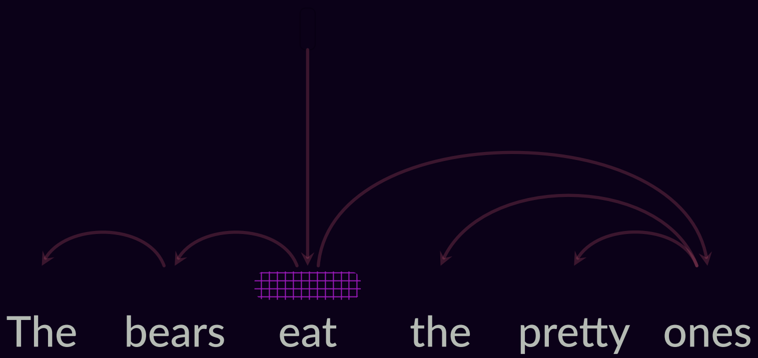
# Dependency TreeLSTM

(Tai et al., 2015)



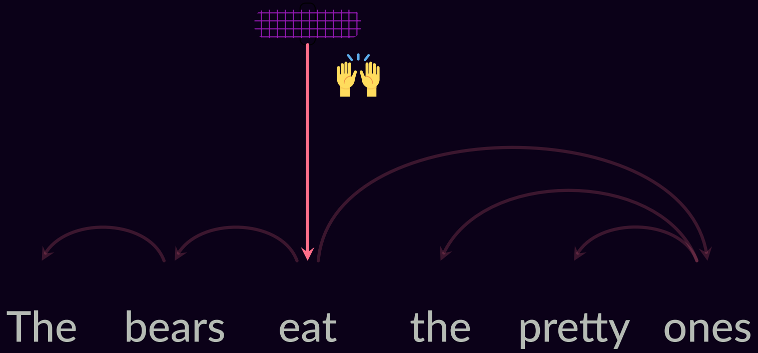
# Dependency TreeLSTM

(Tai et al., 2015)



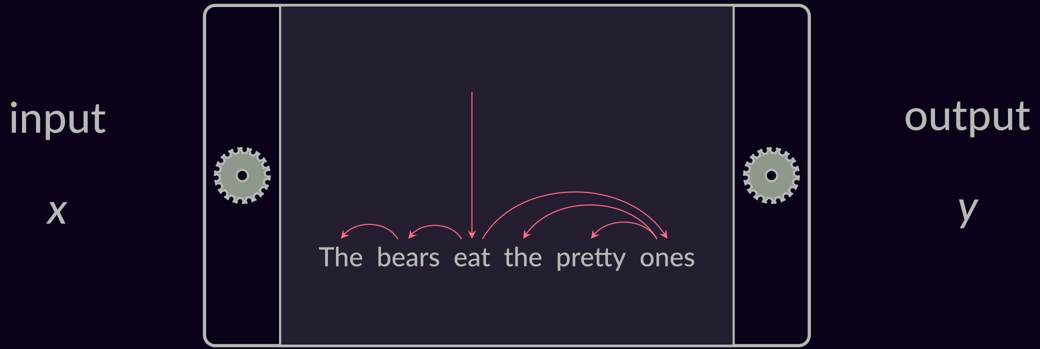
# Dependency TreeLSTM

(Tai et al., 2015)



# Latent Dependency TreeLSTM

(Niculae, Martins, and Cardie, 2018)



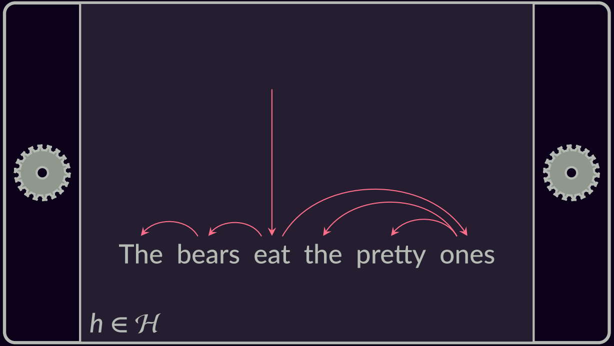
# Latent Dependency TreeLSTM

(Niculae, Martins, and Cardie, 2018)

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

input

$x$




output

$y$

# Structured Latent Variable Models

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

# Structured Latent Variable Models

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

# Structured Latent Variable Models

e.g., a TreeLSTM defined by $h$

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

# Structured Latent Variable Models

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

Exponentially large sum!

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

idea 1

idea 2

idea 3

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

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

idea 1

idea 2

idea 3

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

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

**How to define $p_{\boldsymbol{\pi}}$?**

idea 1

idea 2

idea 3

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

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

idea 1 $p_{\boldsymbol{\pi}}(h \mid x) = 1 \text{ if } h = h^\star \text{ else } 0$ argmax

idea 2

idea 3

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

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

idea 1 $p_{\boldsymbol{\pi}}(h \mid x) = 1$ if $h = h^\star$ else $0$ argmax 😊

idea 2

idea 3

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

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

😊 😟

idea 1 $p_{\boldsymbol{\pi}}(h \mid x) = 1$ if $h = h^{\star}$ else 0 argmax

idea 2

idea 3

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

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

| | | | | |
|---|---|---|---|---|
| idea 1 | $p_{\boldsymbol{\pi}}(h \mid x) = 1$ if $h = h^\star$ else $0$ | argmax | 😊 | 😟 |
| idea 2 | $p_{\boldsymbol{\pi}}(h \mid x) \propto \exp\big(\text{score}_{\boldsymbol{\pi}}(h; x)\big)$ | softmax | | |
| idea 3 | | | | |

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

| | | | $\sum_{h \in \mathcal{H}}$ | $\frac{\partial p(y \mid x)}{\partial \boldsymbol{\pi}}$ |
|---|---|---|---|---|
| idea 1 | $p_{\boldsymbol{\pi}}(h \mid x) = 1$ if $h = h^\star$ else $0$ | argmax | 😊 | 😟 |
| idea 2 | $p_{\boldsymbol{\pi}}(h \mid x) \propto \exp\left(\text{score}_{\boldsymbol{\pi}}(h; x)\right)$ | softmax | | 😊 |
| idea 3 | | | | |

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\mathrm{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

| | | | $\sum_{h \in \mathcal{H}}$ | $\frac{\partial p(y \mid x)}{\partial \boldsymbol{\pi}}$ |
|---|---|---|---|---|
| idea 1 | $p_{\boldsymbol{\pi}}(h \mid x) = 1$ if $h = h^\star$ else 0 | argmax | 😊 | 😟 |
| idea 2 | $p_{\boldsymbol{\pi}}(h \mid x) \propto \exp\left(\mathrm{score}_{\boldsymbol{\pi}}(h; x)\right)$ | softmax | 😱 | 😊 |
| idea 3 | | | | |

# Structured Latent Variable Models

sum over
all possible trees

*e.g.*, a TreeLSTM defined by $h$

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

parsing model,
using some $\text{score}_{\boldsymbol{\pi}}(h; x)$

**How to define $p_{\boldsymbol{\pi}}$?**

| | | | $\sum_{h \in \mathcal{H}}$ | $\frac{\partial p(y \mid x)}{\partial \boldsymbol{\pi}}$ |
|---|---|---|---|---|
| idea 1 | $p_{\boldsymbol{\pi}}(h \mid x) = 1$ if $h = h^{\star}$ else 0 | argmax | 😊 | 😟 |
| idea 2 | $p_{\boldsymbol{\pi}}(h \mid x) \propto \exp\big(\text{score}_{\boldsymbol{\pi}}(h; x)\big)$ | softmax | 😱 | 😊 |
| idea 3 | | SparseMAP | 😊 | 😊 |

# SparseMAP

 = .7  + .3 

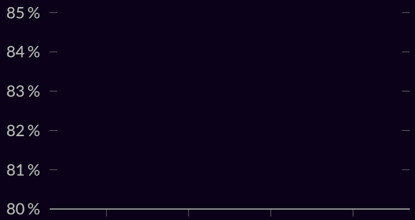
# SparseMAP

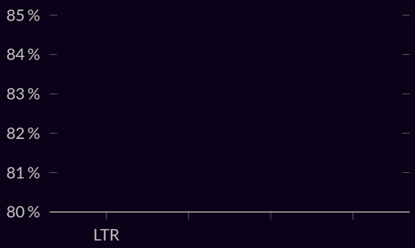
 = .7  + .3  + 0  + ...

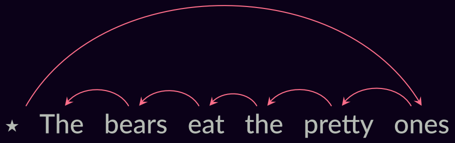
# SparseMAP

 $= \ .7$  $+ \ .3$  $+ \ 0$  $+ \ \ldots$

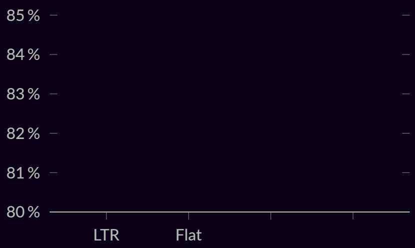
$p(y \mid x) = \ .7 \ p_{\boldsymbol{\phi}}(y \mid$  $) + .3 \ p_{\boldsymbol{\phi}}(y \mid$  $)$

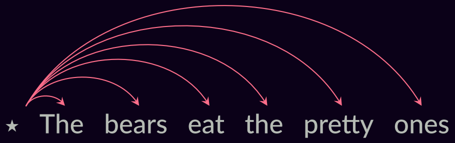

85 %
84 %
83 %
82 %
81 %
80 %



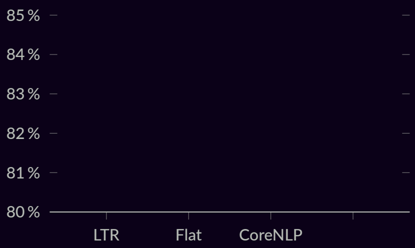




Left-to-right: regular LSTM



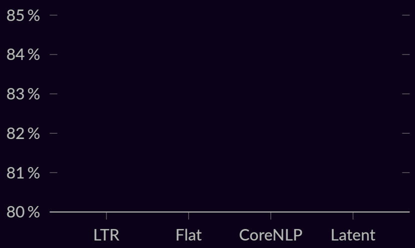




Flat: bag-of-words–like

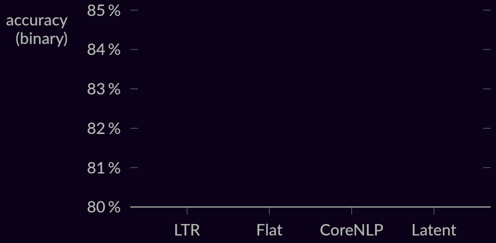




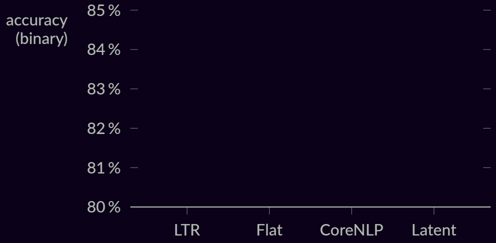
CoreNLP: off-line parser


85 %
84 %
83 %
82 %
81 %
80 %
LTR
Flat
CoreNLP
Latent

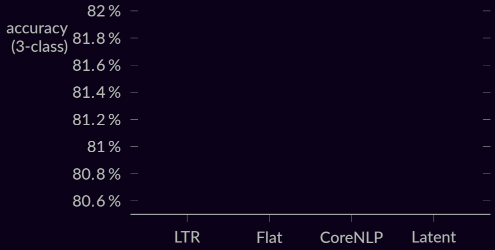
# **Sentiment classification** (SST)



**Sentiment classification** (SST)



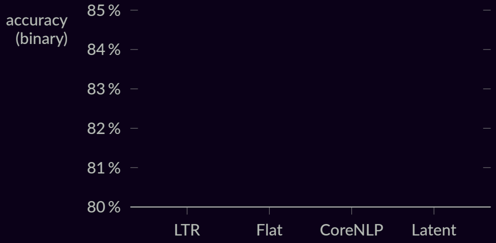

**Natural Language Inference** (SNLI)



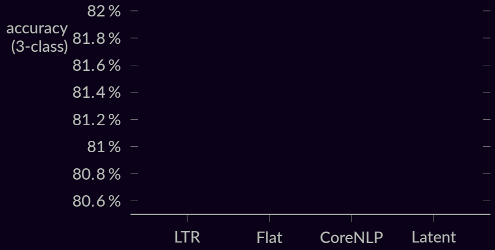

Sentence pair classification $(P, H)$

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

**Sentiment classification** (SST)




**Natural Language Inference** (SNLI)

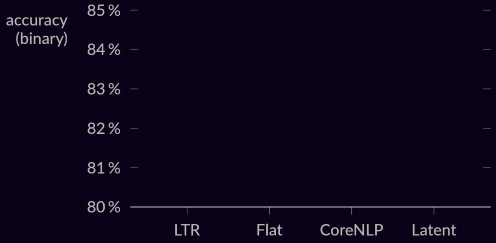



## Reverse dictionary lookup

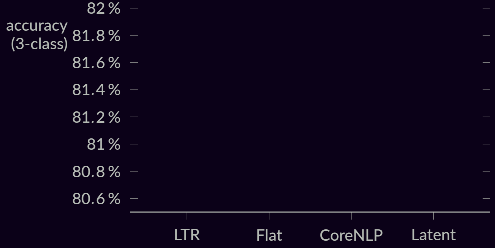
given word description, predict word embedding (Hill et al., 2016)

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

## **Sentiment classification** (SST)

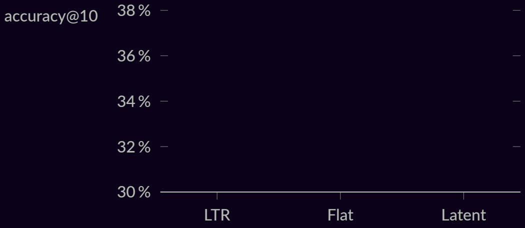



## **Natural Language Inference** (SNLI)

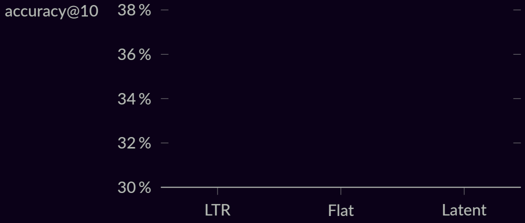



## **Reverse dictionary lookup**

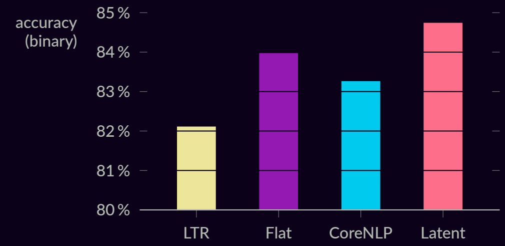
(definitions)



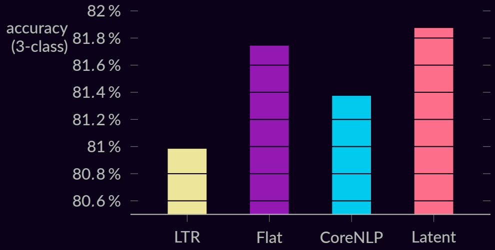

(concepts)



## **Sentiment classification** (SST)

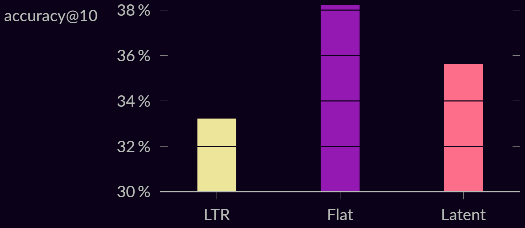



## **Natural Language Inference** (SNLI)

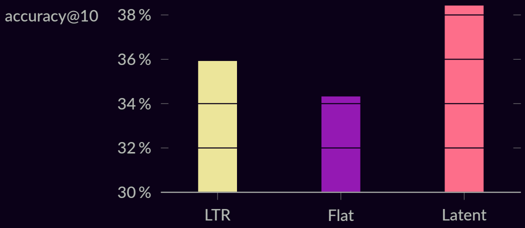



## **Reverse dictionary lookup**

(definitions)




(concepts)



# Syntax vs. Composition Order

CoreNLP parse, $p = 21.4\%$

# Syntax vs. Composition Order

$p = 22.6\%$

⋆ lovely and poignant .

CoreNLP parse, $p = 21.4\%$

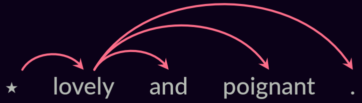
⋆ lovely and poignant .

. . .

# Syntax vs. Composition Order

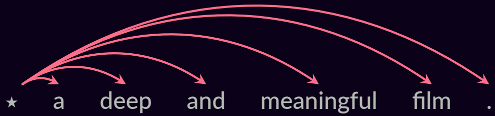
$p = 22.6\%$


CoreNLP parse, $p = 21.4\%$




. . .

$p = 15.33\%$




$p = 15.27\%$


. . .

CoreNLP parse, $p = 0\%$

# Structured Output Prediction

SparseMAP

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

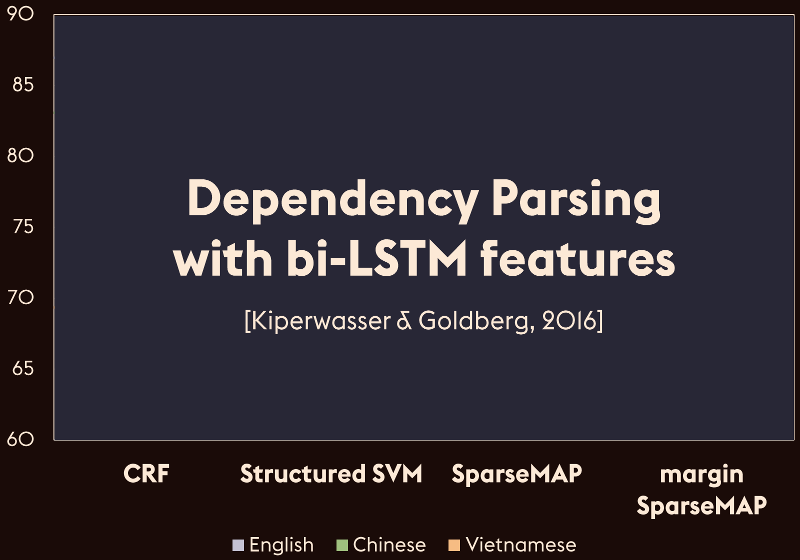
Instance of a structured Fenchel-Young loss, like CRF, SVM, etc. (Blondel, Martins, and Niculae, 2019b)

# Structured Output Prediction

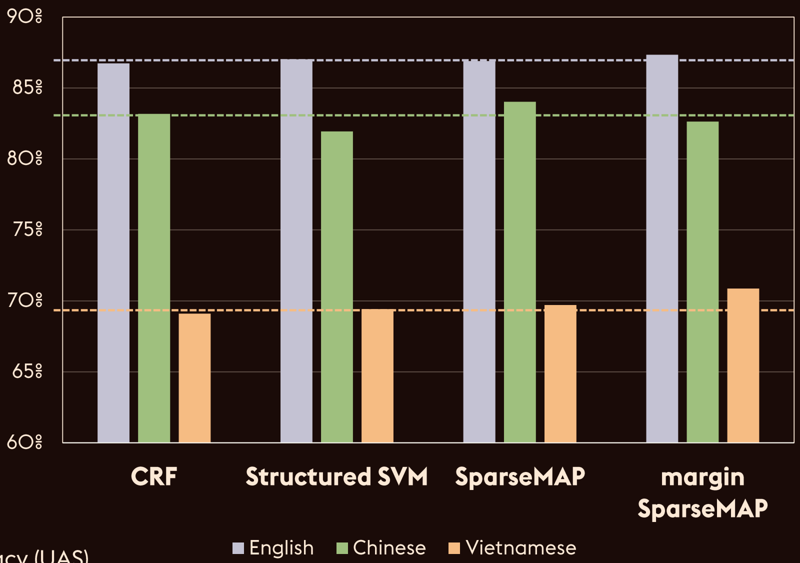
SparseMAP
$$L_A(\boldsymbol{\eta}, \bar{\boldsymbol{\mu}}) = \max_{\boldsymbol{\mu} \in \mathcal{M}} \left\{ \boldsymbol{\eta}^\top \boldsymbol{\mu} - 1/2 \|\boldsymbol{\mu}\|^2 \right\} - \boldsymbol{\eta}^\top \bar{\boldsymbol{\mu}} + 1/2 \|\bar{\boldsymbol{\mu}}\|^2$$

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)



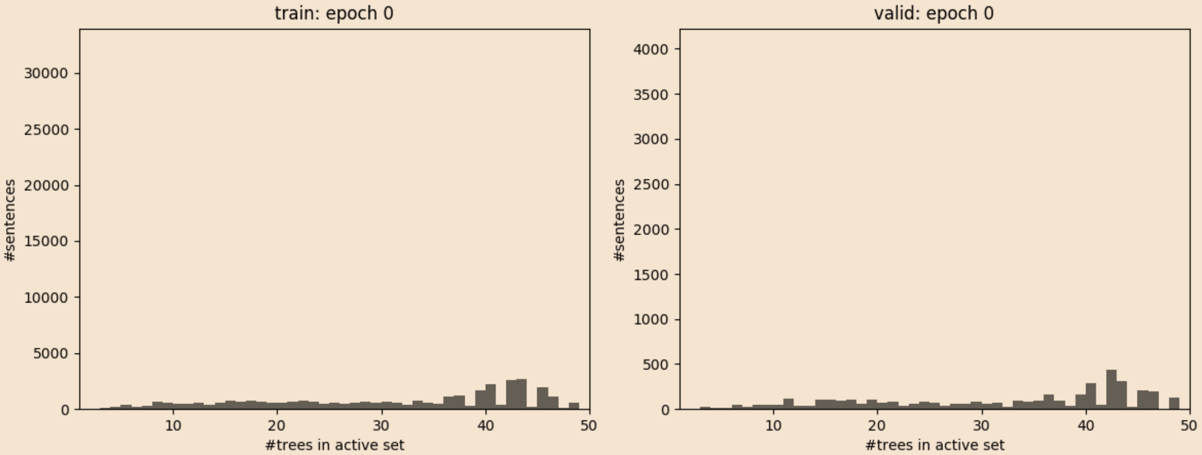
90
85
80
75
70
65
60
Dependency Parsing
with bi-LSTM features
[Kiperwasser & Goldberg, 2016]
CRF
Structured SVM
SparseMAP
margin SparseMAP
English
Chinese
Vietnamese




Unlabeled Accuracy (UAS)
Universal Dependencies dataset

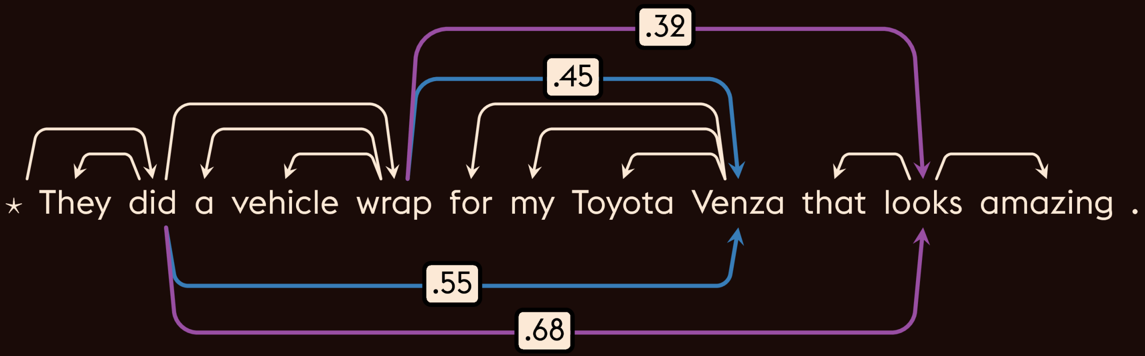
# Sparse Structured Output Prediction

## As models train, inference gets sparser!



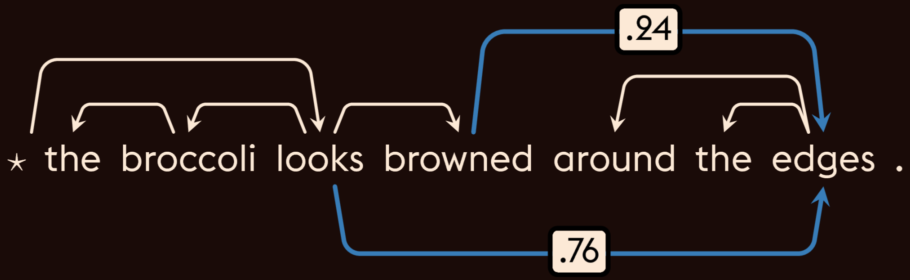
# Sparse Structured Output Prediction

**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 $\ell_1$ ball". In: *Mathematical Programming* 158.1-2, pp. 575–585.

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*.

Danskin, John M (1966). "The theory of max-min, with applications". In: *SIAM Journal on Applied Mathematics* 14.4, pp. 641–664.

# References II


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*.

Kipf, Thomas and Max Welling (2017). "Semi-supervised classification with graph convolutional networks". In: *Proc. of ICLR*.

# References III


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.

Lacoste-Julien, Simon and Martin Jaggi (2015). “On the global linear convergence of Frank-Wolfe optimization variants”. In: *Proc. of NeurIPS*.

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*.

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*.

# References IV


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*.

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*.

# References V


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.

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.