Beyond Softmax:
Sparsity, Constraints, Latent Structure

... all end-to-end differentiable!!

André Martins

Unbabel

instituto de telecomunicações

TÉCNICO LISBOA

FOTRAN Workshop, Helsinki, 28/9/18

In a Nutshell

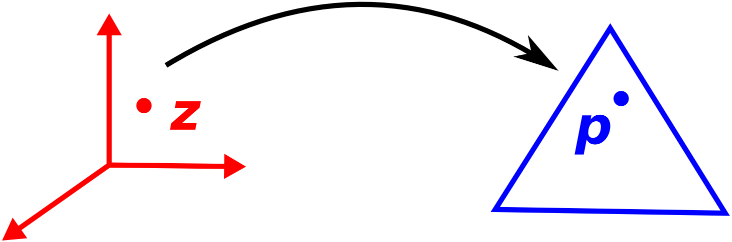
The **softmax** transfomation is prevalent in language generation:

1. Softmax over the vocabulary to obtain a distribution over words
2. Attention mechanisms to condition of some property of the input (Bahdanau et al., 2015; Sukhbaatar et al., 2015)

This talk: new transformations that capture **sparsity**, **constraints**, and **structure**

- Sparsemax, Constrained Softmax/Sparsemax, SparseMAP
- All differentiable (efficient forward and backward propagation)
- Can be used at hidden or output layers.

This talk is about...



Transformations from the Euclidean space $\mathbb{R}^K$ to the simplex.

Joint work with Ramon Astudillo, Julia Kreutzer, Chaitanya Malaviya, Pedro Ferreira, Vlad Niculae, Mathieu Blondel, and Claire Cardie.

Outline

1. **Sparsity**
2. Constraints
3. Latent Structure
4. Conclusions

Sparse Attention with Sparsemax

André F. T. Martins and Ramon Astudillo.

"From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification."

ICML 2016.

Recap: Softmax

- The transformation $\text{softmax} : \mathbb{R}^K \to \Delta^{K-1}$ is defined as:

$$\text{softmax}_i(\boldsymbol{z}) = \frac{\exp(z_i)}{\sum_{k=1}^{K} \exp(z_k)}$$

- Resulting distribution has full support: $\text{softmax}(\boldsymbol{z}) > \mathbf{0}, \forall \boldsymbol{z}$
- A disadvantage if a *sparse* probability distribution is desired
- Common workaround: threshold and truncate

Sparsemax (Martins and Astudillo, 2016)

- We propose as an alternative:

$$\mathrm{sparsemax}(\boldsymbol{z}) := \operatorname*{argmin}_{\boldsymbol{p} \in \Delta^{K-1}} \|\boldsymbol{p} - \boldsymbol{z}\|^2.$$

- In words: Euclidean projection of $\boldsymbol{z}$ onto the probability simplex
- Likely to hit the boundary of the simplex, in which case $\mathrm{sparsemax}(\boldsymbol{z})$ becomes sparse (hence the name)
- We'll see that sparsemax retains many of the properties of softmax, having in addition the ability of producing sparse distributions!

Sparsemax in Closed Form

- Projecting onto the simplex amounts to a soft-thresholding operation:

$$\mathrm{sparsemax}_i(\boldsymbol{z}) \;=\; \max\{0, z_i - \tau\}$$

  where τ is a normalizing constant such that $\sum_j \max\{0, z_j - \tau\} = 1$
- To evaluate the sparsemax, all we need is to compute τ
- Runtime is $O(K \log K)$ with a naive sort; $O(K)$ using linear-time selection (Pardalos and Kovoor, 1990; Duchi et al., 2008)
- Evaluating softmax costs $O(K)$ too

What about Backprop?

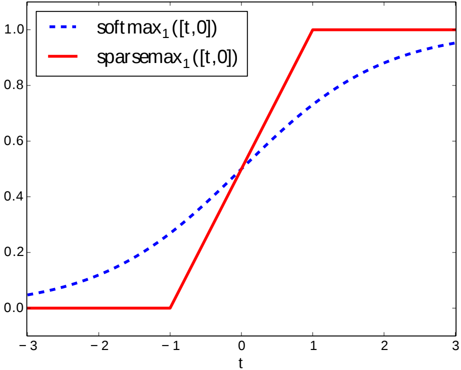
- Sparsemax is differentiable almost everywhere
- Backprop is **more efficient** than softmax: runtime linear in the **number of nonzeros**
- See Martins and Astudillo (2016) for details

Two Dimensions

- Parametrize $\boldsymbol{z} = (t, 0)$
- The 2D softmax is the logistic (sigmoid) function:

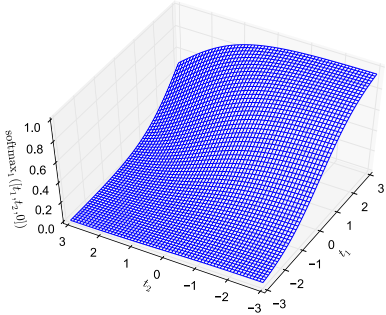
$$\mathrm{softmax}_1(\boldsymbol{z}) = (1 + \exp(-t))^{-1}$$

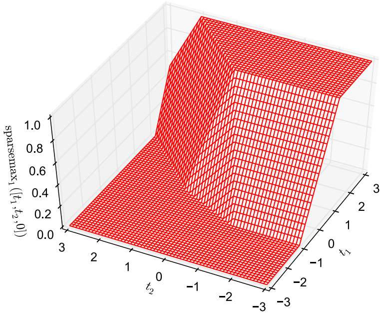
- The 2D sparsemax is the "hard" version of the sigmoid:



Three Dimensions

- Parameterize $\boldsymbol{z} = (t_1, t_2, 0)$ and plot $\text{softmax}_1(\boldsymbol{z})$ and $\text{sparsemax}_1(\boldsymbol{z})$ as a function of t_1 and t_2
- sparsemax is piecewise linear, but asymptotically similar to softmax

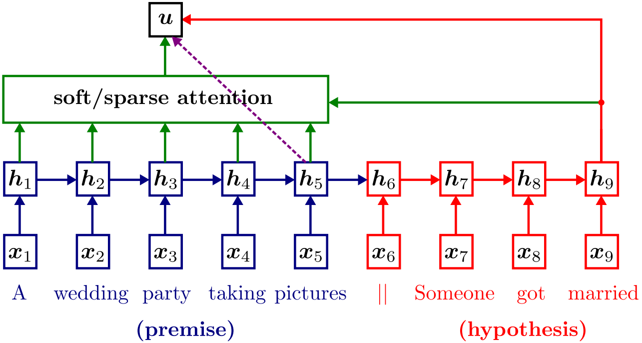




- This gives us all the ingredients to use sparsemax inside a neural network (e.g. a "sparse" attention mechanism)

Neural Networks with Attention Mechanisms

- SNLI corpus (Bowman et al., 2015): 570K sentence pairs (a premise and an hypothesis), labeled as **entailment**, **contradiction**, or **neutral**
- We used an attention-based architecture as Rocktäschel et al. (2015)



Experimental Results

Four neural attention strategies:

- **NoAttention**, a RNN-based system without attention
- **LogisticAttention**, which uses independent logistic activations
- **SoftAttention**, using a softmax attention-based system
- **SparseAttention**, using a sparsemax attention-based system

	Dev Acc.	Test Acc.
NoAttention	81.84	80.99
LogisticAttention	82.11	80.84
SoftAttention	82.86	82.08
SparseAttention	82.52	**82.20**

- Soft and sparse-activated attention systems perform similarly
- Both outperform the **NoAttention** and **LogisticAttention** systems

Some Examples

- *In blue*, the premise words selected by **SparseAttention**
- In red, the hypothesis
- Only a few words are selected, which are key for the system's decision
- The sparsemax activation yields a compact and more interpretable selection, which can be particularly useful in long sentences

A boy *rides on* a *camel* in a crowded area while talking on his cellphone.	
—— A boy is riding an animal.	[entailment]
A young girl wearing *a pink coat* plays with a *yellow* toy golf club.	
—— A girl is wearing a blue jacket.	[contradiction]
Two black dogs are *frolicking* around the *grass together*.	
—— Two dogs swim in the lake.	[contradiction]
A man wearing a yellow striped shirt *laughs* while *seated next* to another *man* who is wearing a light blue shirt and *clasping* his hands together.	
—— Two mimes sit in complete silence.	[contradiction]

More: Sparsemax as a Loss Function

- Sparsemax can also be used in the **output layer**, replacing logistic/cross-entropy loss
- There is a continuous family of transformations that includes both softmax and sparsemax
- The corresponding loss functions are called **Fenchel-Young losses**

Mathieu Blondel, André F. T. Martins, and Vlad Niculae.

"Learning Classifiers with Fenchel-Young Losses: Generalized Entropies, Margins, and Algorithms."

Arxiv preprint 2018.

Outline

Sparse and Constrained Attention

- André F. T. Martins and Julia Kreutzer.
  "Fully Differentiable Neural Easy-First Taggers."
  EMNLP 2017

- Chaitanya Malaviya, Pedro Ferreira, and André F. T. Martins.
  "Sparse and Constrained Attention for Neural Machine Translation."
  ACL 2018.

Constrained Softmax

Constrained softmax resembles softmax, but it allows imposing hard constraints on the maximal probability assigned to each word

- Given scores $\boldsymbol{z} \in \mathbb{R}^K$ and **upper bounds** $\boldsymbol{u} \in \mathbb{R}^K$:

$$\begin{aligned} \mathrm{csoftmax}(\boldsymbol{z}; \boldsymbol{u}) \quad = \quad & \operatorname*{argmin}_{\boldsymbol{p} \in \Delta^{K-1}} \mathbf{KL}(\boldsymbol{p} \,\|\, \mathrm{softmax}(\boldsymbol{z})) \\ & \text{s.t.} \quad \boldsymbol{p} \leq \boldsymbol{u} \end{aligned}$$

- Related to **posterior regularization** (Ganchev et al., 2010)

Particular cases:

- If $\boldsymbol{u} \geq \mathbf{1}$, all constraints are loose and this reduces to softmax
- If $\boldsymbol{u} \in \Delta^{K-1}$, they are tight and we must have $\boldsymbol{p} = \boldsymbol{u}$

How to Evaluate?

Forward computation takes $O(K \log K)$ **time** (Martins and Kreutzer, 2017):

- Let $\mathcal{A} = \{i \in [K] \mid p_i^\star < u_i\}$ be the **constraints that are met strictly**
- Then by writing the KKT conditions we can express the solution as:

$$p_i^\star = \min\left\{\frac{\exp(z_i)}{Z}, u_i\right\} \quad \forall i \in [K], \quad \text{where } Z = \frac{\sum_{i \in \mathcal{A}} \exp(z_i)}{1 - \sum_{i \notin \mathcal{A}} u_i}.$$

- Identifying the set $\mathcal{A}$ can be done in $O(K \log K)$ time with a sort

How to Backpropagate?

We need to compute gradients with respect to both $\boldsymbol{z}$ and $\boldsymbol{u}$

Can be done in $O(K)$ **time** (Martins and Kreutzer, 2017):

- Let $L(\boldsymbol{\theta})$ be a loss function, $\mathrm{d}\boldsymbol{p} = \nabla_{\boldsymbol{\alpha}} L(\boldsymbol{\theta})$ be the output gradient, and $\mathrm{d}\boldsymbol{z} = \nabla_{\boldsymbol{z}} L(\boldsymbol{\theta})$ and $\mathrm{d}\boldsymbol{u} = \nabla_{\boldsymbol{u}} L(\boldsymbol{\theta})$ be the input gradients
- Then, the input gradients are given as:

$$\begin{aligned} \mathrm{d}z_i &= \mathbb{1}(i \in \mathcal{A}) p_i (\mathrm{d}p_i - m) \\ \mathrm{d}u_i &= \mathbb{1}(i \notin \mathcal{A}) (\mathrm{d}p_i - m), \end{aligned}$$

where $m = (\sum_{i \in \mathcal{A}} p_i \, \mathrm{d}p_i) / (1 - \sum_{i \notin \mathcal{A}} u_i)$.

This opens the door for using constrained softmax attention in a neural network, backpropagating through the scores and the upper bounds...

Martins and Kreutzer (2017): usage as a module in **neural easy-first decoders**.

Constrained Sparsemax (Malaviya et al., 2018)

Similar idea, but replacing softmax by sparsemax:

- Given scores $\boldsymbol{z} \in \mathbb{R}^K$ and **upper bounds** $\boldsymbol{u} \in \mathbb{R}^K$:

$$\begin{aligned} \text{csparsemax}(\boldsymbol{z}; \boldsymbol{u}) \quad = \quad & \underset{\boldsymbol{p} \in \Delta^{K-1}}{\text{argmin}} \|\boldsymbol{p} - \boldsymbol{z}\|^2 \\ & \text{s.t.} \quad \boldsymbol{p} \leq \boldsymbol{u} \end{aligned}$$

- Both sparse and upper bounded
- If $\boldsymbol{u} \geq \mathbf{1}$, all constraints are loose and this reduces to sparsemax
- If $\boldsymbol{u} \in \Delta^{K-1}$, they are tight and we must have $\boldsymbol{p} = \boldsymbol{u}$

How to Evaluate?

Forward computation can be done with a sort in $O(K \log K)$ time

Can be reduced to $O(K)$ (Malaviya et al., 2018; Pardalos and Kovoor, 1990):

- Let $\mathcal{A} = \{i \in [K] \mid 0 < p_i^\star < u_i\}$ be the **constraints that are met strictly**
- Let $\mathcal{A}_R = \{i \in [K] \mid p_i^\star = u_i\}$
- Then by writing the KKT conditions we can express the solution as:

$$p_i^\star = \max\{0, \min\{u_i, z_i - \tau\}\} \quad \forall i \in [K], \quad \text{where } \tau \text{ is a constant.}$$

- Identifying the sets $\mathcal{A}$ and $\mathcal{A}_R$ can be done in $O(K \log K)$ time with a sort

How to Backpropagate?

We need to compute gradients with respect to both $\boldsymbol{z}$ and $\boldsymbol{u}$

Can be done in sublinear time $O(|\mathcal{A}| + |\mathcal{A}_R|)$ (Malaviya et al., 2018):

- Let $L(\boldsymbol{\theta})$ be a loss function, $\mathrm{d}\boldsymbol{p} = \nabla_{\boldsymbol{\alpha}} L(\boldsymbol{\theta})$ be the output gradient, and $\mathrm{d}\boldsymbol{z} = \nabla_{\boldsymbol{z}} L(\boldsymbol{\theta})$ and $\mathrm{d}\boldsymbol{u} = \nabla_{\boldsymbol{u}} L(\boldsymbol{\theta})$ be the input gradients
- Then, the input gradients are given as:

$$\begin{aligned} \mathrm{d}z_i &= \mathbb{1}(i \in \mathcal{A})(\mathrm{d}p_i - m) \\ \mathrm{d}u_i &= \mathbb{1}(i \in \mathcal{A}_R)(\mathrm{d}p_i - m), \end{aligned}$$

where $m = \frac{1}{|\mathcal{A}|} \sum_{i \in \mathcal{A}} \mathrm{d}p_i$.

Next, we show how to use these constrained attentions in neural machine translation decoders, inspired by the idea of **fertility** (IBM Model 2)...

Modeling Fertility in NMT

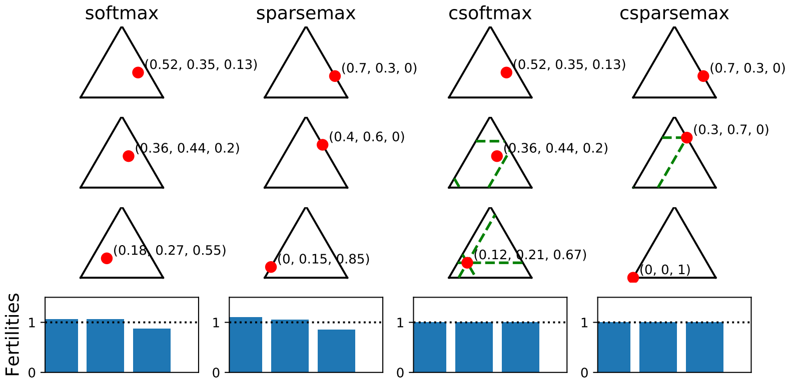
We do the following procedure:

1. Align the training data with `fast_align`
2. Train a separate BILSTM to predict fertility f_i for each word
3. At each decoder step, use upper bound $u_i = f_i - \beta_i$ for each word, where β_i is the cumulative attention

See Malaviya et al. (2018) for more details.

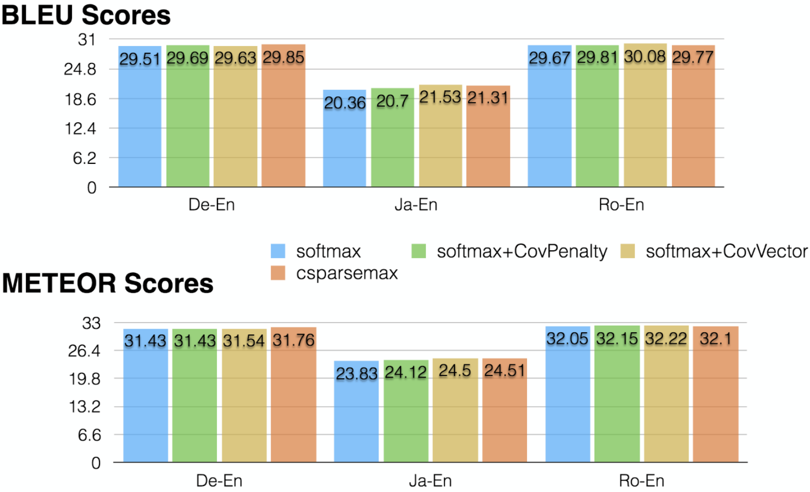
Example: Source Sentence with Three Words

Assume each word is given fertility 1:



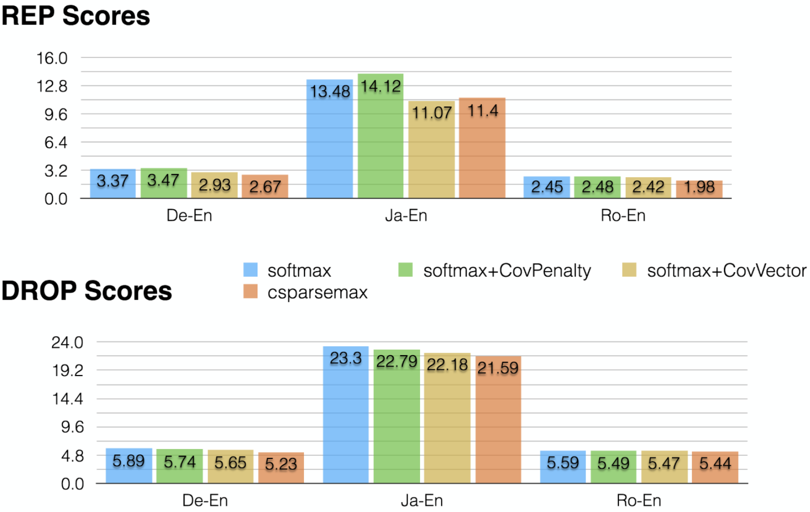
BLEU Scores

Baselines are softmax and two other coverage models (Wu et al., 2016; Tu et al., 2016)



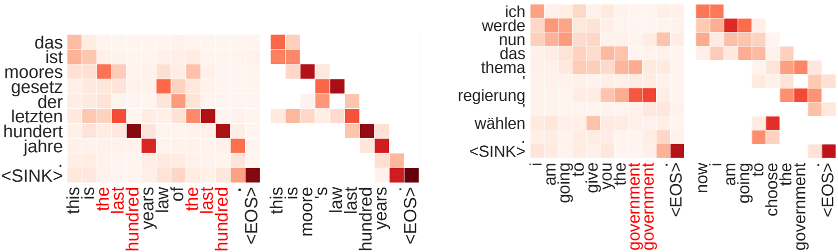
Coverage Scores

Account for **repetitions** and **dropped** source words (lower is better):



Attention Maps

Softmax (left) vs Constrained Sparsemax (right) for De-En:



Sentence Examples

input	so ungefähr , sie wissen schon .
reference	*like that , you know* .
softmax	**so , you know , you know** .
sparsemax	**so , you know , you know** .
csoftmax	**so , you know , you know** .
csparsemax	like that , you know .

input	und wir benutzen dieses wort mit solcher verachtung .
reference	and we say that word *with such contempt* .
softmax	and we use this word with such **contempt contempt** .
sparsemax	and we use this word with such contempt .
csoftmax	and we use this word with **like this** .
csparsemax	and we use this word with such contempt .

Code (Pytorch + OpenNMT):

www.github.com/Unbabel/sparse_constrained_attention

Outline

1. Sparsity
2. Constraints
3. **Latent Structure**
4. Conclusions

SparseMAP

Vlad Niculae, André F. T. Martins, Mathieu Blondel, and Claire Cardie.

"SparseMAP: Differentiable Sparse Structured Inference."

ICML 2018.

SparseMAP

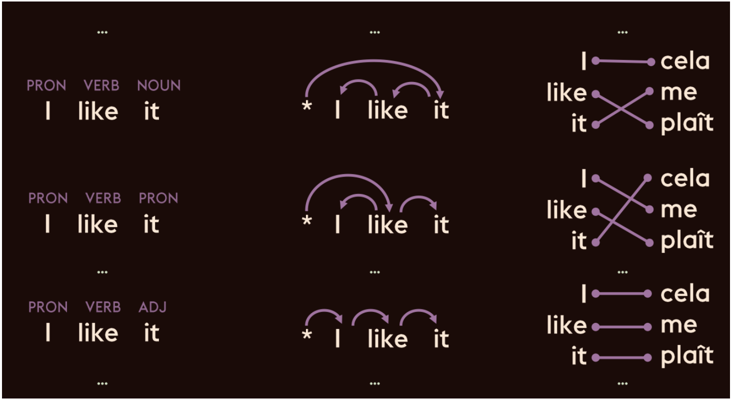
- Generalizes sparsemax to **sparse structured prediction**
- Works both as output layer and hidden layer
- With latent models, similar to structured attention networks (Kim et al., 2017), but **sparse**
- Efficient forward/backprop requiring only an argmax (MAP) oracle!

Two Scenarios:

- Structured output prediction
- Latent structured inference

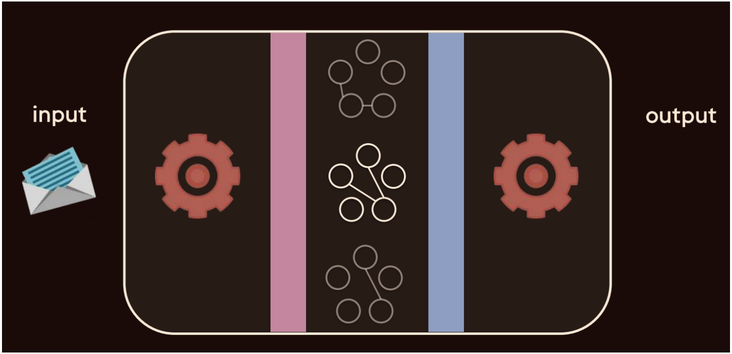
Structured Output Prediction

- Many NLP tasks require predicting linguistic structure as output
- Examples: sequence tagging, dependency parsing, alignments



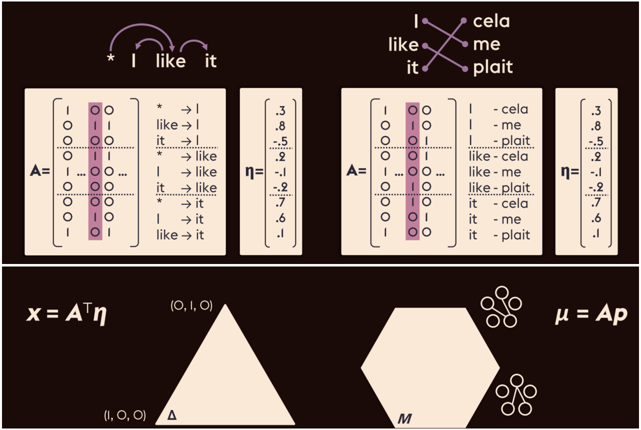
Latent Structured Inference

- Sometimes it's convenient to induce linguistic structure as a latent variable for some downstream task
- Examples: latent syntax for MT; latent alignments for NLI



Marginal Polytope

- Vertices are codewords of combinatorial structures
- Points correspond to marginal distributions over those structures



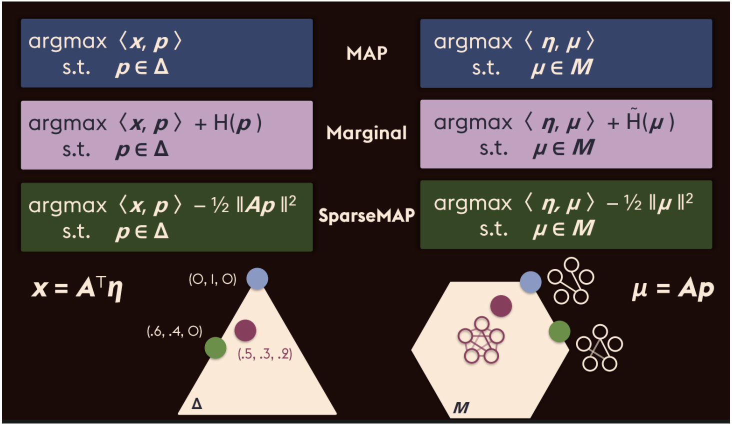
Structured Inference

Unstructured	Structured
argmax	MAP inference
softmax	Marginal inference
sparsemax	?

Structured Inference

Unstructured	Structured
argmax	MAP inference
softmax	Marginal inference
sparsemax	SparseMAP

Sparse Structured Prediction

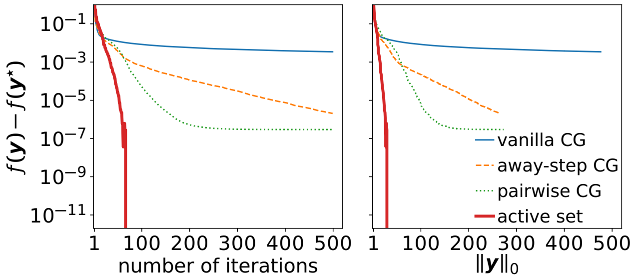


SparseMAP yields a **sparse combination of vertices**, hence it selects only a small number of structures (out of exponentially many)

Efficiently Computing SparseMAP

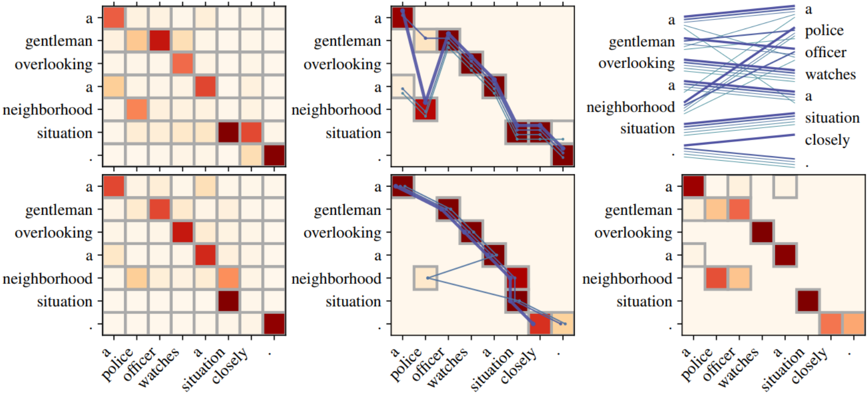
Boils down to **projecting onto the marginal polytope**

Key Result: can be solved as a (small) sequence of argmax (MAP) calls



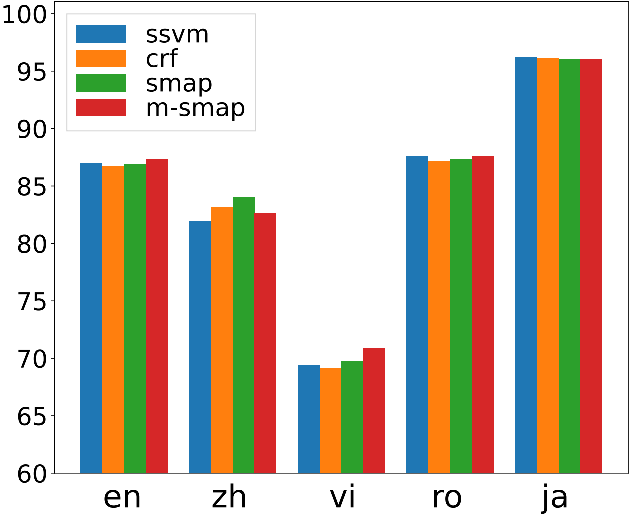
Gradient backprop comes for free once we have done forward!

Example: Latent Structured Alignments in SNLI



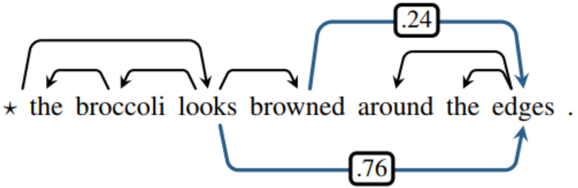
(a) softmax (b) sequence (c) matching

Example: Dependency Parsing

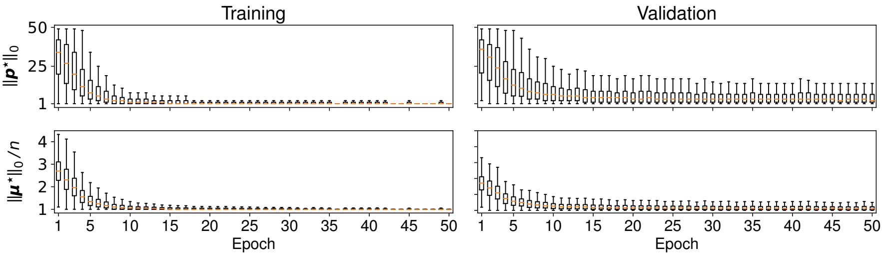


Example: Dependency Parsing

- Suitable for capturing ambiguity in natural language!

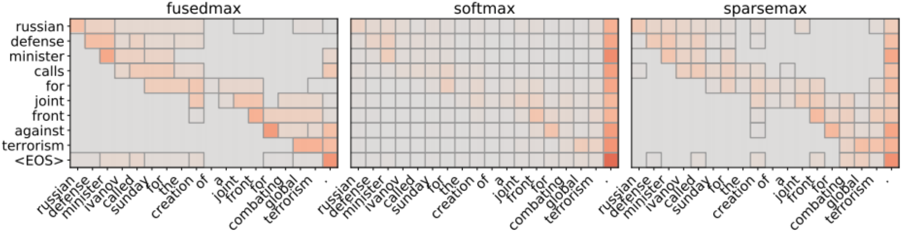


Learning to be Sparse



Related Work

- Structured attention networks (Kim et al., 2017): not sparse
- SPIGOT (Peng et al., 2018): different framework, same building blocks (our active set algo for polytope projection applies there too)
- ... but SPIGOT gradients are *inexact* while ours are exact
- Fusedmax (and other structured sparse) attention (Niculae and Blondel, 2017):



Outline

Conclusions

- Transformations from real numbers to distributions are ubiquitous
- We introduced new transformations that handle **sparsity**, **constraints**, and **structure**
- All are differentiable and their gradients are efficient to compute
- Can be used as hidden layers or as output layers
- Various experiments in NMT and sentence pair tasks, with improved interpretability
- **Recent work:** dynamically determining the computation graph based on the SparseMAP selected structures

To Appear

Vlad Niculae, André F. T. Martins and Claire Cardie

"Towards Dynamic Computation Graphs via Sparse Latent Structure"

EMNLP 2018

DeepSPIN

ERC project **DeepSPIN** (Deep Structured Prediction in NLP)

- ERC starting grant, started in 2018
- Post-doc positions may open next year
- Topics: deep learning, structured prediction, NLP, machine translation
- Involving Unbabel and the University of Lisbon
- More details: https://deep-spin.github.io

Unbabel instituto de telecomunicações TÉCNICO LISBOA

We're Hiring at Unbabel!

Excited about MT, NLP, and Lisbon? ⇒ jobs@unbabel.com.

Open positions: **ML/NLP Software Engineer**, **Sr Research Scientist**

Unbabel

Lisbon 5 Day Weather

4:20 pm WEST Print

DAY	DESCRIPTION	HIGH / LOW	PRECIP	WIND	HUMIDITY
TONIGHT SEP 27	Clear	--/21°	0%	NNW 23 km/h	57%
FRI SEP 28	Mostly Sunny	31°/18°	0%	NNE 22 km/h	54%
SAT SEP 29	Sunny	30°/17°	0%	N 13 km/h	55%
SUN SEP 30	Sunny	30°/18°	10%	NNW 16 km/h	59%
MON OCT 1	Partly Cloudy	30°/18°	0%	NNW 19 km/h	49%
TUE OCT 2	Sunny	31°/17°	0%	ENE 18 km/h	33%

References I

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conference on Learning Representations*.

Bowman, S. R., Angeli, G., Potts, C., and Manning, C. D. (2015). A Large Annotated Corpus for Learning Natural Language Inference. In *Proc. of Empirical Methods in Natural Language Processing*.

Duchi, J., Shalev-Shwartz, S., Singer, Y., and Chandra, T. (2008). Efficient Projections onto the L1-Ball for Learning in High Dimensions. In *Proc. of International Conference of Machine Learning*.

Ganchev, K., Graça, J. a., Gillenwater, J., and Taskar, B. (2010). Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*, 11:2001–2049.

Huber, P. J. (1964). Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101.

Kim, Y., Denton, C., Hoang, L., and Rush, A. M. (2017). Structured attention networks. *arXiv preprint arXiv:1702.00887*.

Liu, D. C. and Nocedal, J. (1989). On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical programming*, 45(1-3):503–528.

Malaviya, C., Ferreira, P., and Martins, A. F. T. (2018). Sparse and Constrained Attention for Neural Machine Translation. In *Proc. of the Annual Meeting of the Association for Computation Linguistics*.

References II

Martins, A. F. T. and Astudillo, R. (2016). From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification. In *Proc. of the International Conference on Machine Learning*.

Martins, A. F. T. and Kreutzer, J. (2017). Fully differentiable neural easy-first taggers. In *Proc. of Empirical Methods for Natural Language Processing*.

Nesterov, Y. (1983). A Method of Solving a Convex Programming Problem with Convergence Rate $O(1/k^2)$. *Soviet Math. Doklady*, 27:372–376.

Niculae, V. and Blondel, M. (2017). A regularized framework for sparse and structured neural attention. *arXiv preprint arXiv:1705.07704*.

Pardalos, P. M. and Kovoor, N. (1990). An Algorithm for a Singly Constrained Class of Quadratic Programs Subject to Upper and Lower Bounds. *Mathematical Programming*, 46(1):321–328.

Peng, H., Thomson, S., and Smith, N. A. (2018). Backpropagating through Structured Argmax using a SPIGOT. In *Proc. of the Annual Meeting of the Association for Computation Linguistics*.

Rocktäschel, T., Grefenstette, E., Hermann, K. M., Kočiskỳ, T., and Blunsom, P. (2015). Reasoning about Entailment with Neural Attention. *arXiv preprint arXiv:1509.06664*.

Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). End-to-End Memory Networks. In *Advances in Neural Information Processing Systems*, pages 2431–2439.

Tsallis, C. (1988). Possible generalization of boltzmann-gibbs statistics. *Journal of Statistical Physics*, 52:479–487.

References III

Tu, Z., Lu, Z., Liu, Y., Liu, X., and Li, H. (2016). Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Zhang, T. (2004). Statistical Behavior and Consistency of Classification Methods Based on Convex Risk Minimization. *Annals of Statistics*, pages 56–85.