Beyond Softmax: Adaptively Sparse Transformers

André Martins

AthNLP, Athens, 24/9/19

Transformers Are Big Bulldozers

Very powerful, but a brute force solution.

Can't be used for gardening: don't distinguish brambles from a rare flower.

NLP Today: Muppets Driving Bulldozers

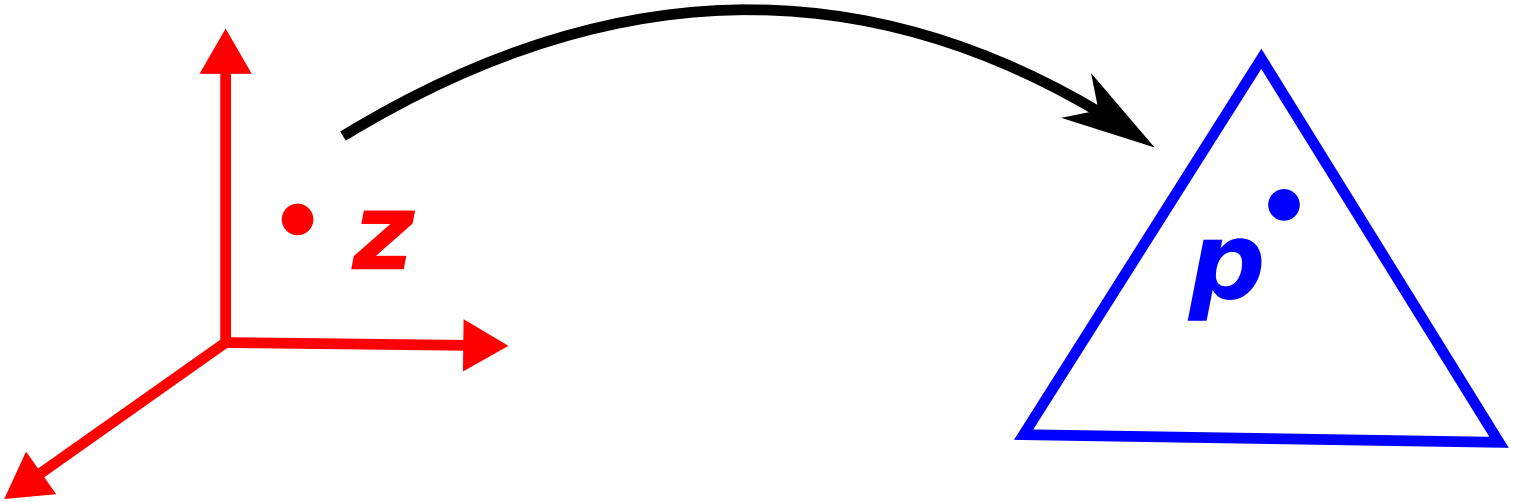
Great for leaderboards! But not much insight about how machines can understand language.

This Talk: Sparse Bulldozers

What's inside a bulldozer? Can we redesign its components?

I'll argue that **sparse modeling** is a great tool for discovering linguistic structure, and can be an integrated component of complex systems.

This talk is about...



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

Joint work with Ben Peters, Gonçalo Correia, Vlad Niculae, Chaitanya Malaviya, Pedro Ferreira, Julia Kreutzer, Mathieu Blondel, Claire Cardie, Ramon Astudillo.

What Does This Have To Do With NLP?

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)
- Adaptively sparse
- Can be used at hidden or output layers.

Outline

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 and Argmax

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

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

- **Fully dense:** $\text{softmax}(\boldsymbol{z}) > \mathbf{0}, \forall \boldsymbol{z}$
- Used both as a loss function (cross-entropy) and for attention

Recap: Softmax and Argmax

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

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

- **Fully dense:** $\text{softmax}(\boldsymbol{z}) > \mathbf{0}, \forall \boldsymbol{z}$
- Used both as a loss function (cross-entropy) and for attention
- Argmax can be written as:

$$\text{argmax}(\boldsymbol{z}) := \underset{\boldsymbol{p} \in \Delta^{K-1}}{\text{argmax}}\, \boldsymbol{z}^\top \boldsymbol{p}.$$

- Retrieves a **one-hot vector** for the highest scored index.
- Sometimes used as hard attention, but not differentiable!

Sparsemax (Martins and Astudillo, 2016)

- We propose as an alternative:

$$\begin{aligned} \text{sparsemax}(\boldsymbol{z}) &:= \operatorname*{argmin}_{\boldsymbol{p} \in \Delta^{K-1}} \|\boldsymbol{p} - \boldsymbol{z}\|^2 \\ &= \operatorname*{argmax}_{\boldsymbol{p} \in \Delta^{K-1}} \boldsymbol{z}^\top \boldsymbol{p} - \frac{1}{2}\|\boldsymbol{p}\|^2. \end{aligned}$$

- In words: Euclidean projection of $\boldsymbol{z}$ onto the probability simplex
- Likely to hit the boundary of the simplex, in which case 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:

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

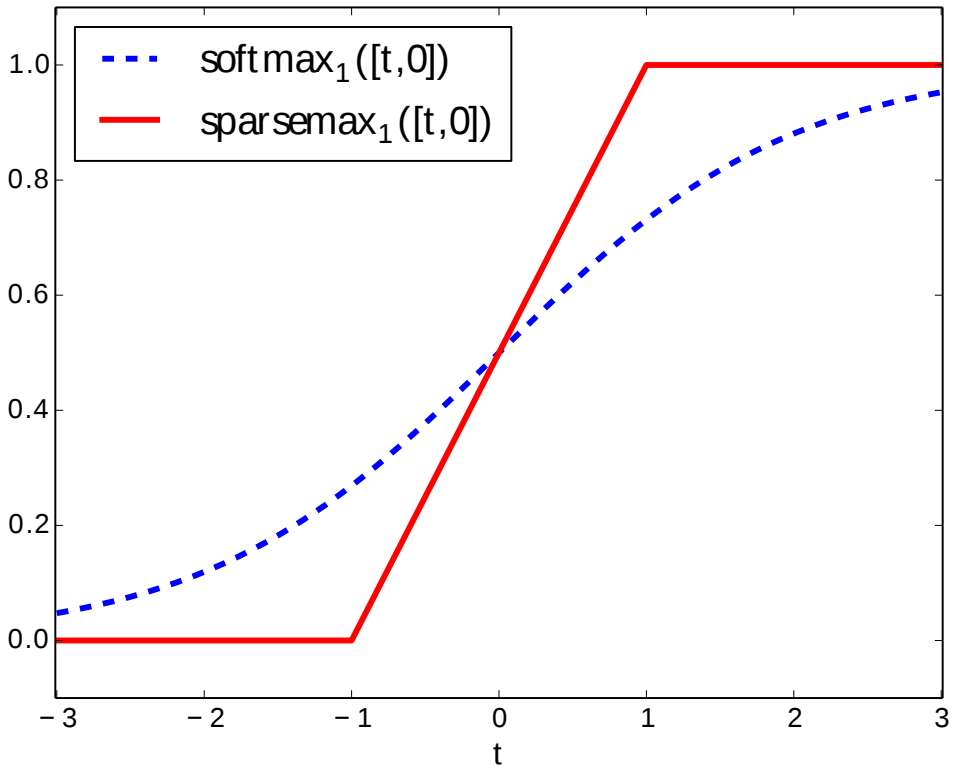
  where τ is a normalizing constant such that $\sum_j \max\{0, z_j - \tau\} = 1$
- To evaluate the sparsemax, all we need is to compute τ
- Forward pass: $O(K)$ (Pardalos and Kovoor, 1990), **same as softmax**
- Backprop: sublinear, **better than softmax!**

Two Dimensions

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

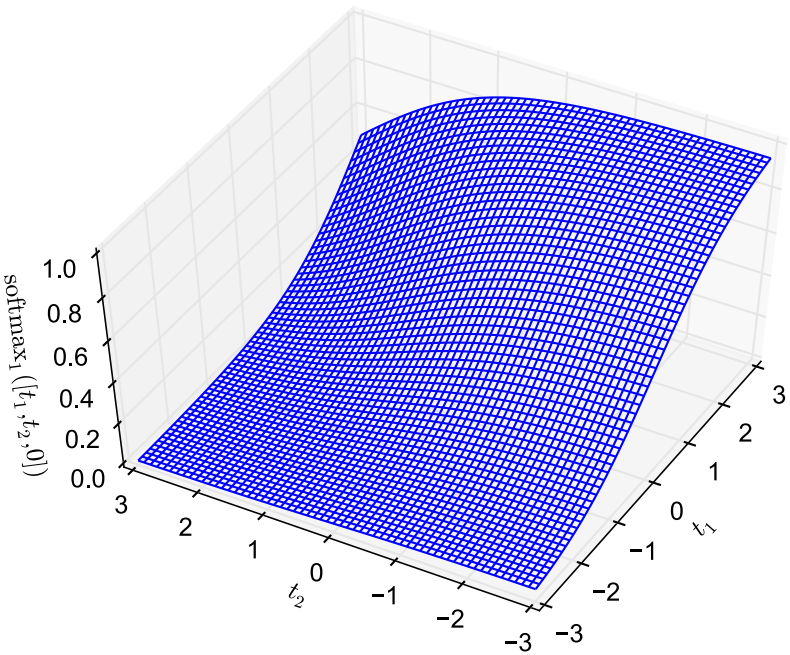
$$\text{softmax}_1(\boldsymbol{z}) = (1 + \exp(-t))^{-1}$$

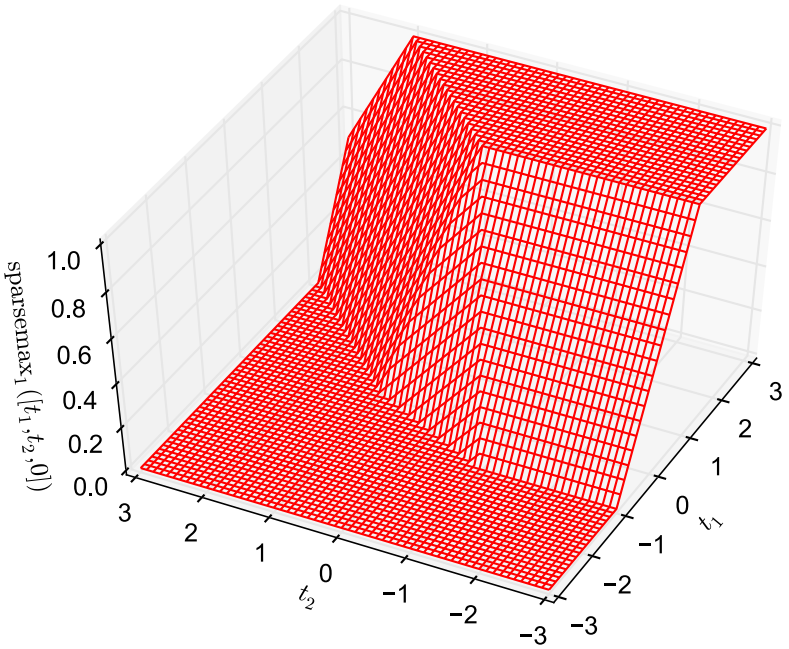
- The 2D sparsemax is the "hard" version of the sigmoid:



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





Properties of Softmax/Sparsemax

$\rho \in \{\text{softmax}, \text{sparsemax}\}$ have similar behaviour and invariances:

1. $\rho(\mathbf{0})$ is the **uniform distribution**
2. $\lim_{t \to +\infty} \rho(t\boldsymbol{z})$ is a **delta distribution** (for sparsemax, t is finite!)
3. **Invariance to adding constants:**

$$\rho(\boldsymbol{z}) = \rho(\boldsymbol{z} + c\mathbf{1}), \qquad \text{for any } c \in \mathbb{R}.$$

Example: Sparse Word Selection In SNLI

- *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]

Sparsemax Loss

- Sparsemax can also be used as a loss in the **output layer** (to replace logistic/cross-entropy loss)
- However, not expressed as a log-likelihood (which could lead to log(0) problems due to sparsity)
- Instead, we build a sparsemax loss inspired by **Fenchel-Young losses**.

Fenchel-Young Losses

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

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

AISTATS 2019.

Ω-Regularized Argmax

For convex Ω, define the Ω-**regularized argmax prediction**:

$$\operatorname{argmax}_\Omega(\boldsymbol{z}) := \operatorname*{argmax}_{\boldsymbol{p}\in\Delta^{K-1}} \boldsymbol{z}^\top \boldsymbol{p} - \Omega(\boldsymbol{p}).$$

- **Argmax** corresponds to **no regularization**, $\Omega \equiv 0$
- **Softmax** amounts to **entropic regularization**, $\Omega(\boldsymbol{p}) = \sum_{i=1}^K p_i \log p_i$
- **Sparsemax** amounts to ℓ_2-**regularization**, $\Omega(\boldsymbol{p}) = \frac{1}{2}\|\boldsymbol{p}\|^2$

Is there something in-between?

Tsallis Entropies (Tsallis, 1988)

A family of entropies parametrized by $\alpha \geq 0$:

$$H_\alpha(\boldsymbol{p}) := \frac{1}{1-\alpha}\left(1 - \sum_{i=1}^{K} p_i^\alpha\right)$$

Includes Shannon entropy (limit case when $\alpha \to 1$)

Setting $\Omega = -H_\alpha/\alpha$:

- **Argmax** corresponds to $\alpha \to \infty$
- **Softmax** amounts to $\alpha \to 1$
- **Sparsemax** amounts to $\alpha = 2$

We call this transformation α-**entmax** (Peters et al., 2019).

α-Entmax: Summary

- Includes argmax, softmax, sparsemax as particular cases
- Forward pass for general α can be done with a bissection algorithm (Blondel et al., 2018)
- Backward pass runs in sublinear time
- Always sparse for $\alpha > 1$, sparsity increases with α
- Special case: 1.5-entmax (specialized forward pass algo)

Fenchel-Young Losses

Groundtruth $\boldsymbol{q} \in \Delta^{K-1}$, scores $\boldsymbol{z} \in \mathbb{R}^K$, $\Omega^*(\boldsymbol{z}) := \max_{\boldsymbol{p} \in \Delta^{K-1}} \boldsymbol{z}^\top \boldsymbol{p} - \Omega(\boldsymbol{p})$

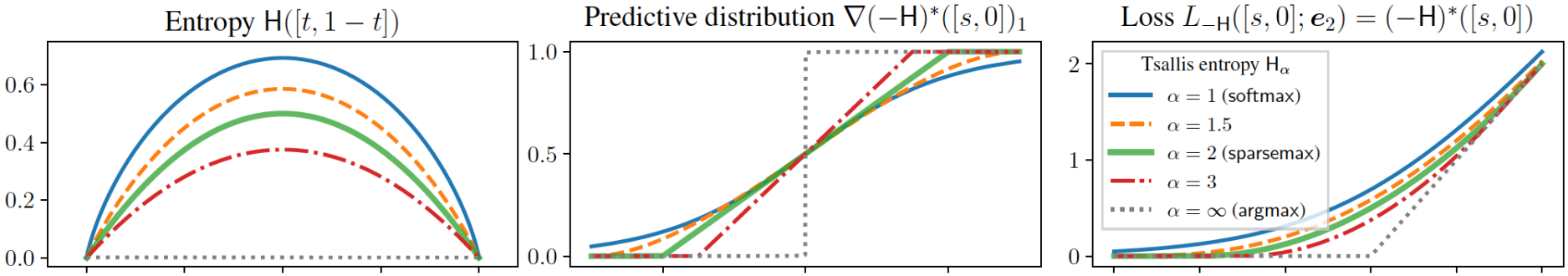
$$L_\Omega(\boldsymbol{z}, \boldsymbol{q}) := \Omega^*(\boldsymbol{z}) + \Omega(\boldsymbol{q}) - \boldsymbol{z}^\top \boldsymbol{q}$$

For any strictly convex Ω:

- $L_\Omega(\boldsymbol{z}, \boldsymbol{q}) \geq 0$ (automatic from **Fenchel-Young inequality**)
- $L_\Omega(\boldsymbol{z}, \boldsymbol{q}) = 0$ iff $\boldsymbol{q} = \operatorname{argmax}_\Omega(\boldsymbol{z})$
- L_Ω is convex and differentiable with $\nabla L_\Omega(\boldsymbol{z}, \boldsymbol{q}) = \operatorname{argmax}_\Omega(\boldsymbol{z}) - \boldsymbol{q}$

Recovers **cross-entropy loss**, **sparsemax loss**, and many other known losses

Tsallis Entropies and their Losses



- Key result: for all $\alpha > 1$, all transformations are **sparse** and lead to losses with **margins**!
- The **margin size** is related to the **slope** of the entropy in the simplex corners!
- See paper for details!

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 (Martins and Kreutzer, 2017)

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** $\boldsymbol{u} \in \mathbb{R}^K$:

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

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 $\boldsymbol{p} = \boldsymbol{u}$

Constrained Sparsemax (Malaviya et al., 2018)

Similar idea, but replacing softmax by sparsemax:

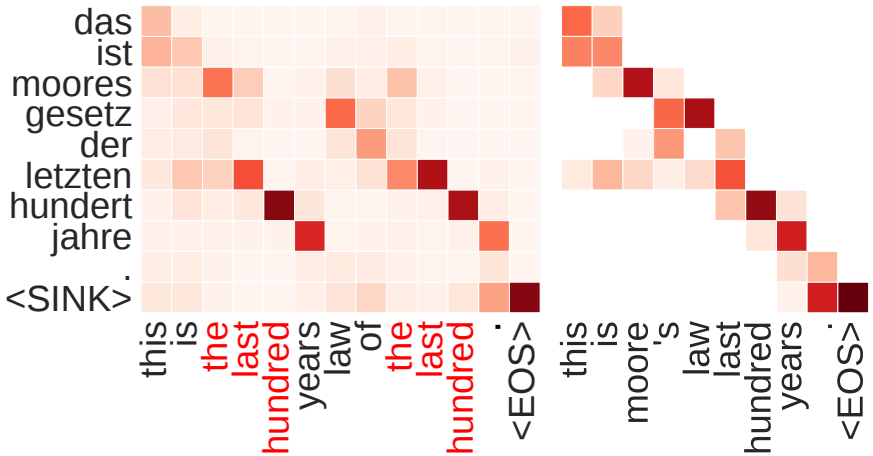
$$\begin{aligned} \text{csparsemax}(\boldsymbol{z}; \boldsymbol{u}) \quad = \quad & \underset{\boldsymbol{p} \in \Delta^{K-1}}{\text{argmin}} \; \|\boldsymbol{p} - \boldsymbol{z}\|^2 \\ & \text{s.t.} \quad \boldsymbol{p} \leq \boldsymbol{u} \end{aligned}$$

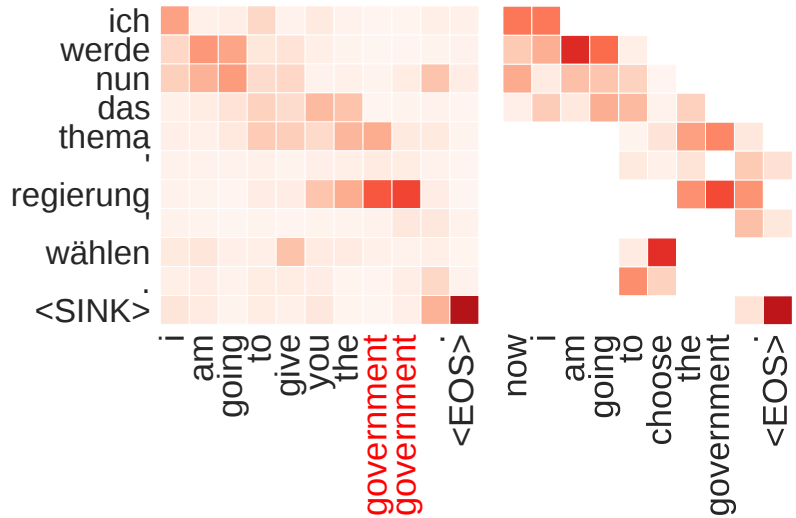
- Both sparse and upper bounded
- $\boldsymbol{u} \geq \mathbf{1} \quad \Longrightarrow \quad$ becomes sparsemax
- Forward pass can be done in $O(K)$ (Pardalos and Kovoor, 1990)
- Backprop can be done in **sublinear** time
- Malaviya et al. (2018) used this to model **fertility** in NMT (a la IBM Model 2).

Attention Maps

Each source word has a **budget** of how much attention it can receive, which prevents repetitions.

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 .

SparseMAP

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

"SparseMAP: Differentiable Sparse Structured Inference."

ICML 2018.

SparseMAP (Niculae et al., 2018)

- 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!

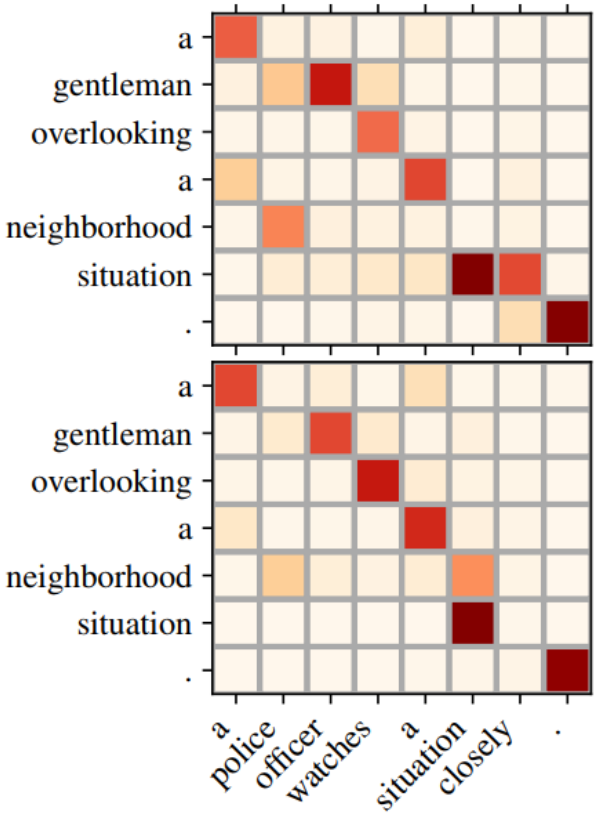
Structured Inference

Unstructured	Structured
argmax	MAP inference
softmax	Marginal inference
sparsemax	?

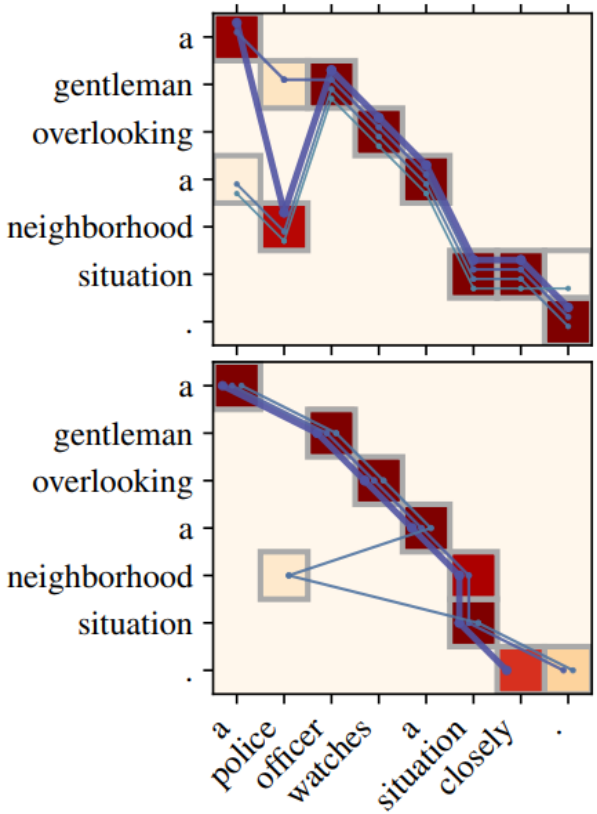
Structured Inference

Unstructured	Structured
argmax	MAP inference
softmax	Marginal inference
sparsemax	SparseMAP

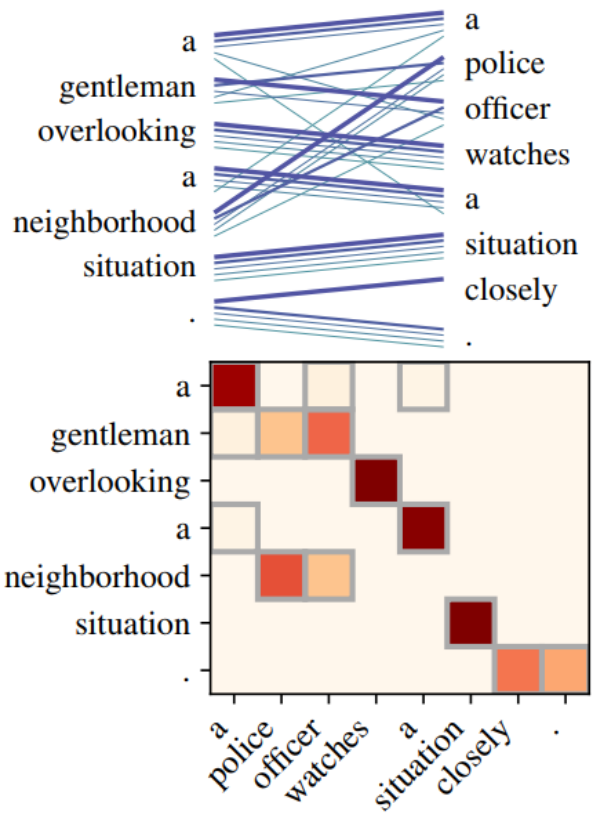
Example: Latent Structured Alignments in SNLI



(a) softmax



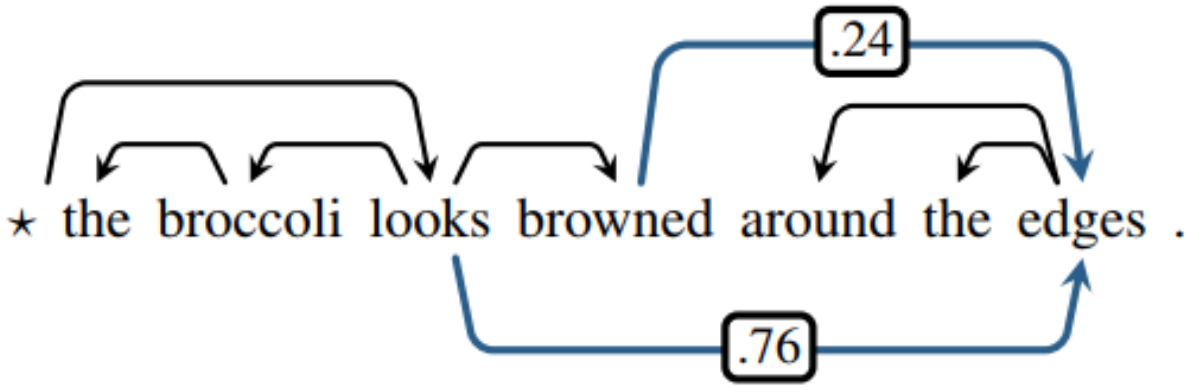
(b) sequence



(c) matching

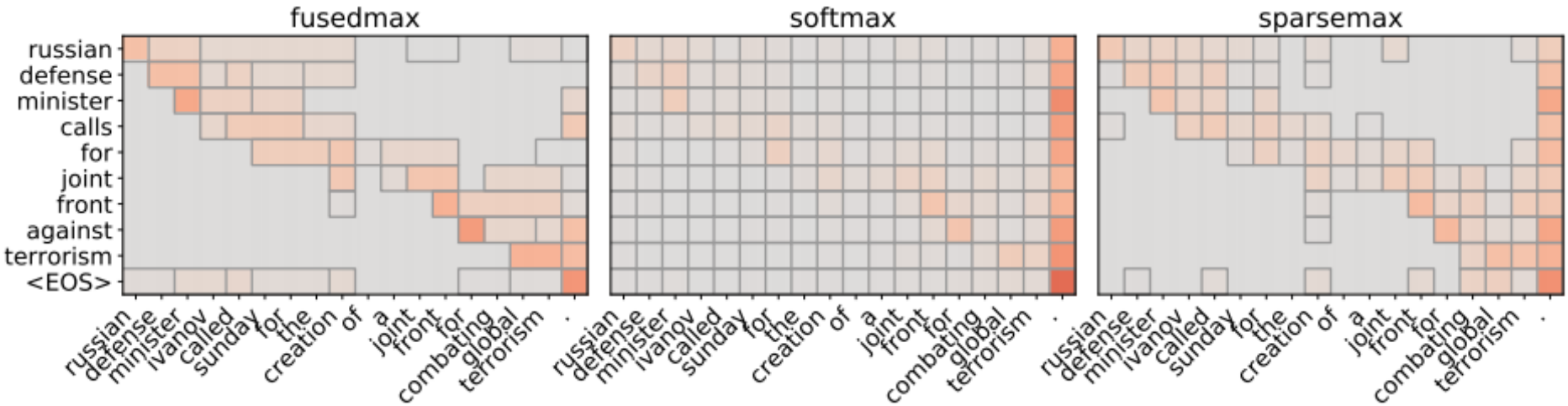
Example: Dependency Parsing

- Suitable for capturing ambiguity in natural language!



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

Sparse Sequence to Sequence

Ben Peters, Vlad Niculae, and André F. T. Martins.

"Sparse Sequence-to-Sequence Models."

ACL 2019.

Sparse Sequence-to-Sequence (Peters et al., 2019)

Backbone: an RNN-based model with attention.

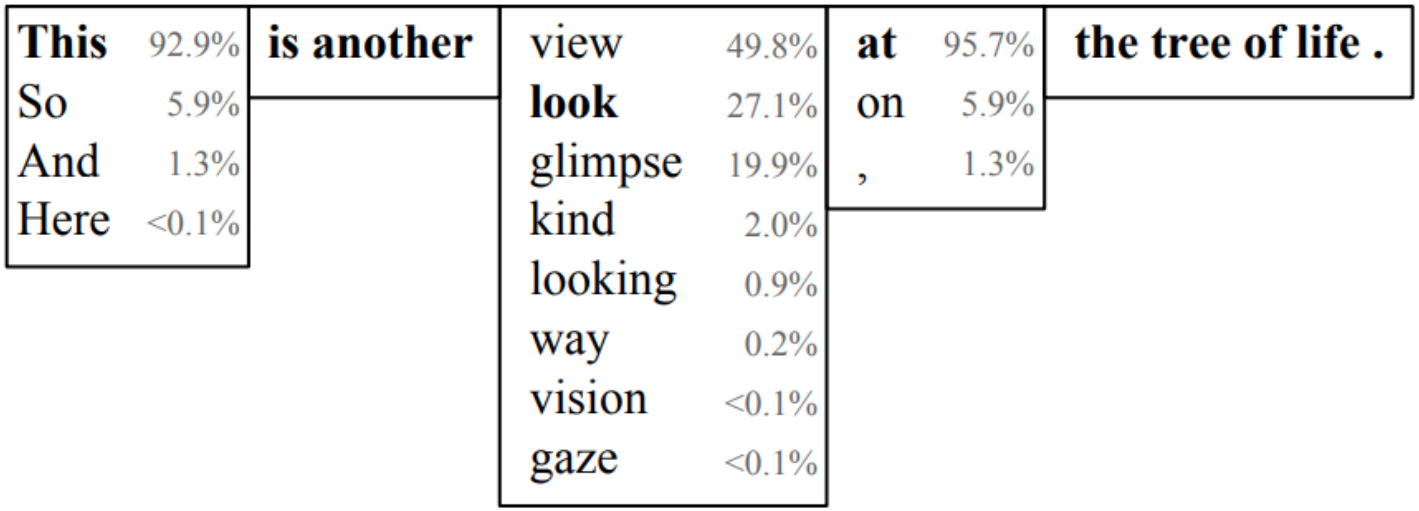
Key idea:

- Replace all instances of softmax by sparsemax or α-entmax.
- We consider both sparsity in the **attention mechanism** and sparsity in the **output layer**

Two tasks:

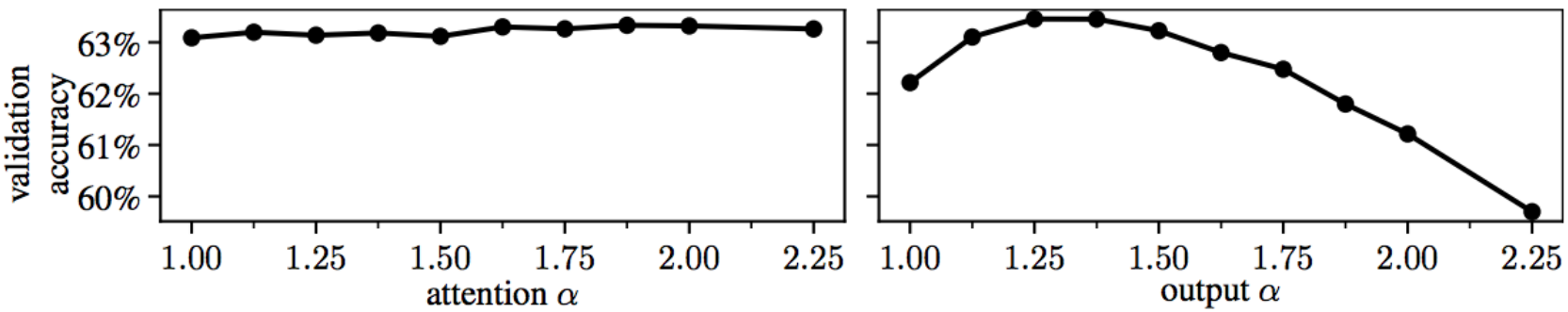
- Machine translation (word-based)
- Morphological inflection (character-based).

Sparsity in Forced Decoding (Peters et al., 2019)



- Only a few words get non-zero probability at each time step
- Auto-completion when several words in a row have probability 1
- Useful for predictive translation when interacting with a user

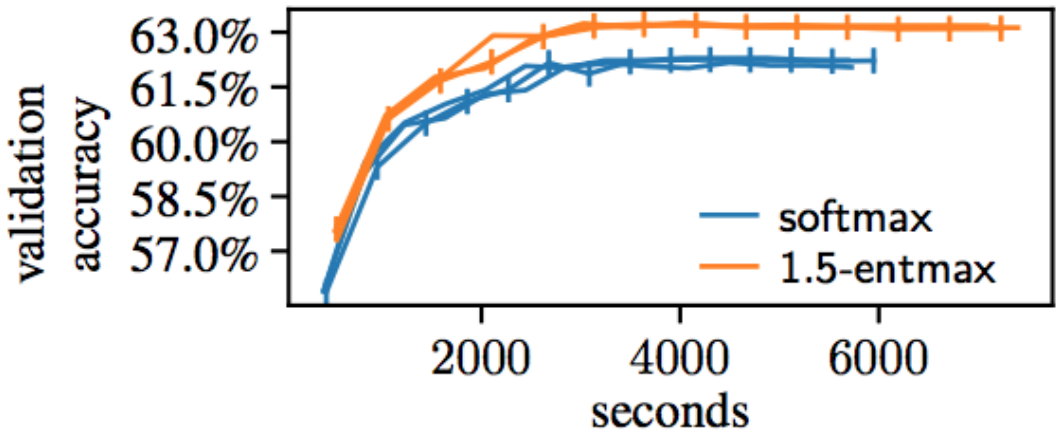
Sparsity in Attention and in Output Later



The impact on accuracy is bigger when sparsity is used in the output layer

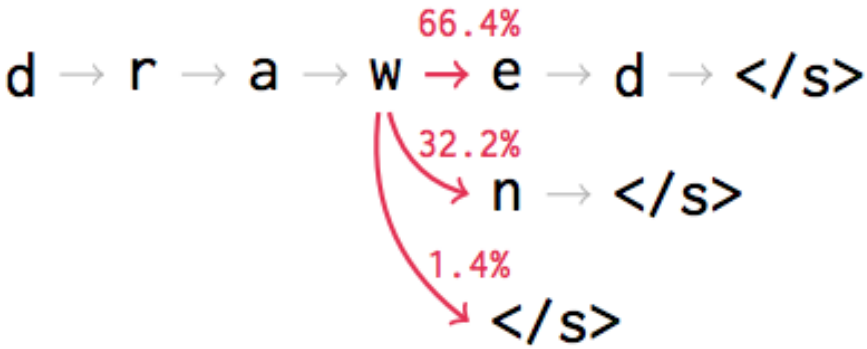
Sparsity in attention doesn't impact accuracy, but leads to interpretable alignments.

Training Time vs Accuracy



1.5-entmax attains better performance faster.

Example: Morphological Inflection

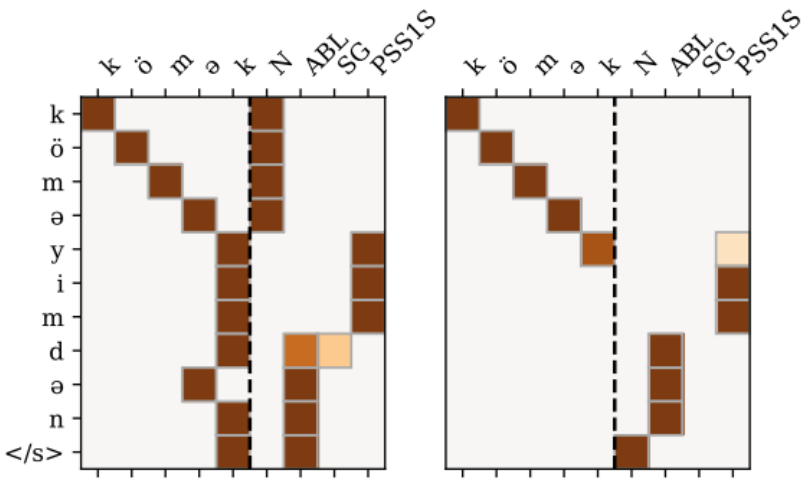


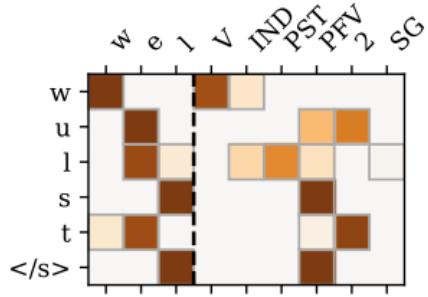
Only a few inflected words get nonzero probability.

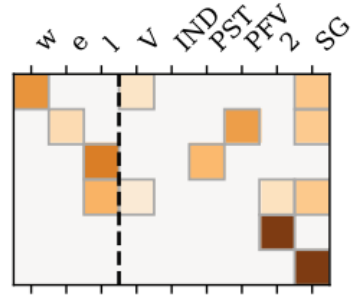
Example: Morphological Inflection

We developed variants of these models for a SIGMORPHON submission

A double attention model and a gated attention model, where the gates decides whether to read information from the lemma or the inflectional tags.







Outline

Adaptively Sparse Transformers

Gonçalo Correia, Vlad Niculae, and André F. T. Martins.

"Adaptively Sparse Transformers."

EMNLP 2019 (to appear).

Transformer (Vaswani et al., 2017)

- **Key idea:** instead of RNN/CNNs, use self-attention in the encoder
- Each word state attends to all the other words
- Each self-attention is followed by a feed-forward transformation
- Do several layers of this
- Do the same for the decoder, attending only to already generated words.

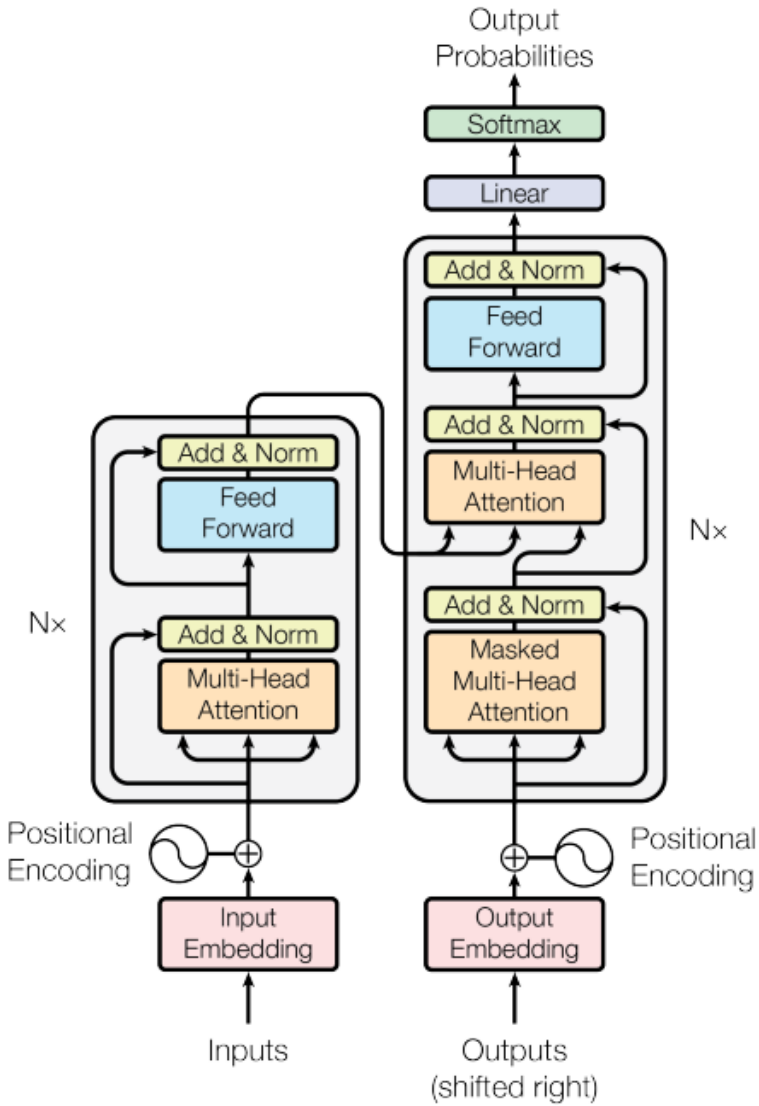


Figure 1: The Transformer - model architecture.

Transformer Basics

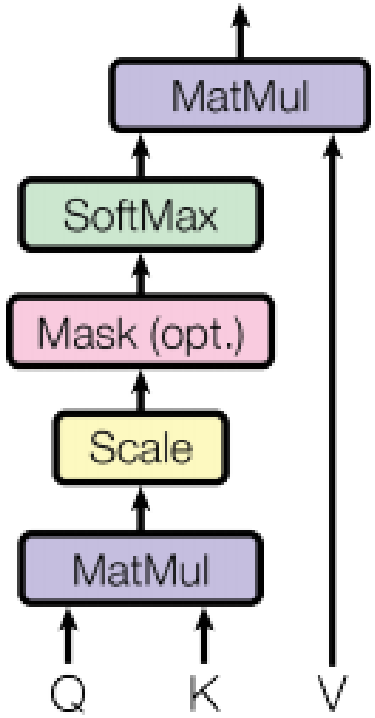
Let's define the basic building blocks of transformer networks first: new attention layers!

Two innovations:

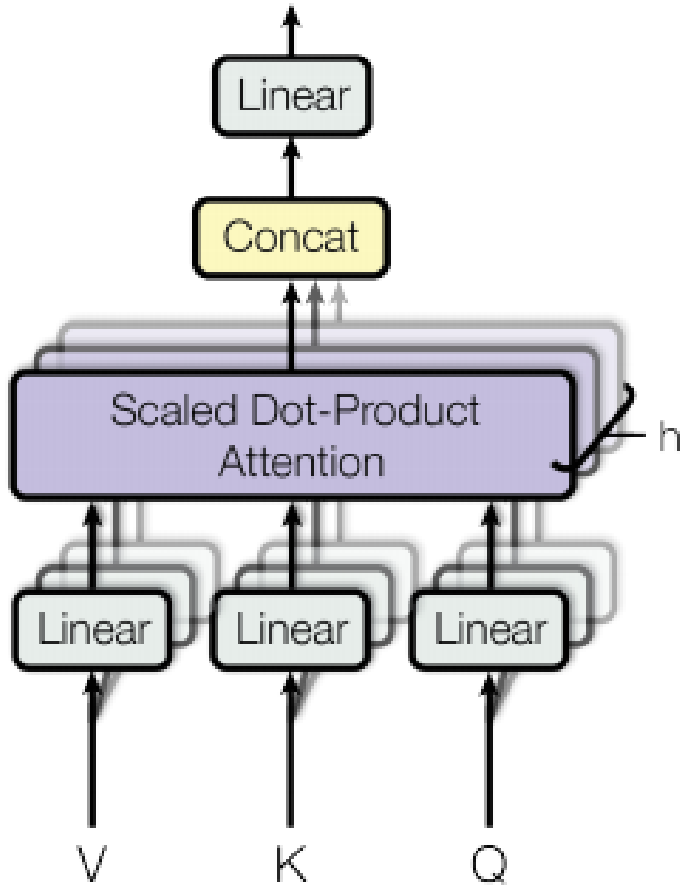
- scaled dot-product attention
- multi-head attention

Scaled Dot-Product and Multi-Head Attention

Scaled Dot-Product Attention



Multi-Head Attention



(Vaswani et al., 2017)

Scaled Dot-Product Attention

Inputs:

- A query vector $\boldsymbol{q}$ (e.g. the decoder state)
- A matrix $\mathbf{K}$ whose columns are key vectors (e.g. the encoder states)
- A matrix $\mathbf{V}$ whose columns are value vectors (e.g. the encoder states)

When discussing attention with RNNs, we assume the key and value vectors were the same, but they don't need to!

Output: the weighted sum of values, where each weight is computed by a dot product between the query and the corresponding key:

$$\boldsymbol{a} = \text{softmax}(\mathbf{K}\boldsymbol{q}), \qquad \bar{\boldsymbol{v}} = \mathbf{V}\boldsymbol{a}.$$

With multiple queries,

$$\bar{\mathbf{V}} = \text{softmax}(\mathbf{Q}\mathbf{K}^\top)\mathbf{V}, \qquad \mathbf{Q} \in \mathbb{R}^{|Q| \times d_k}, \mathbf{K} \in \mathbb{R}^{|K| \times d_k}, \mathbf{V} \in \mathbb{R}^{|K| \times d_v}.$$

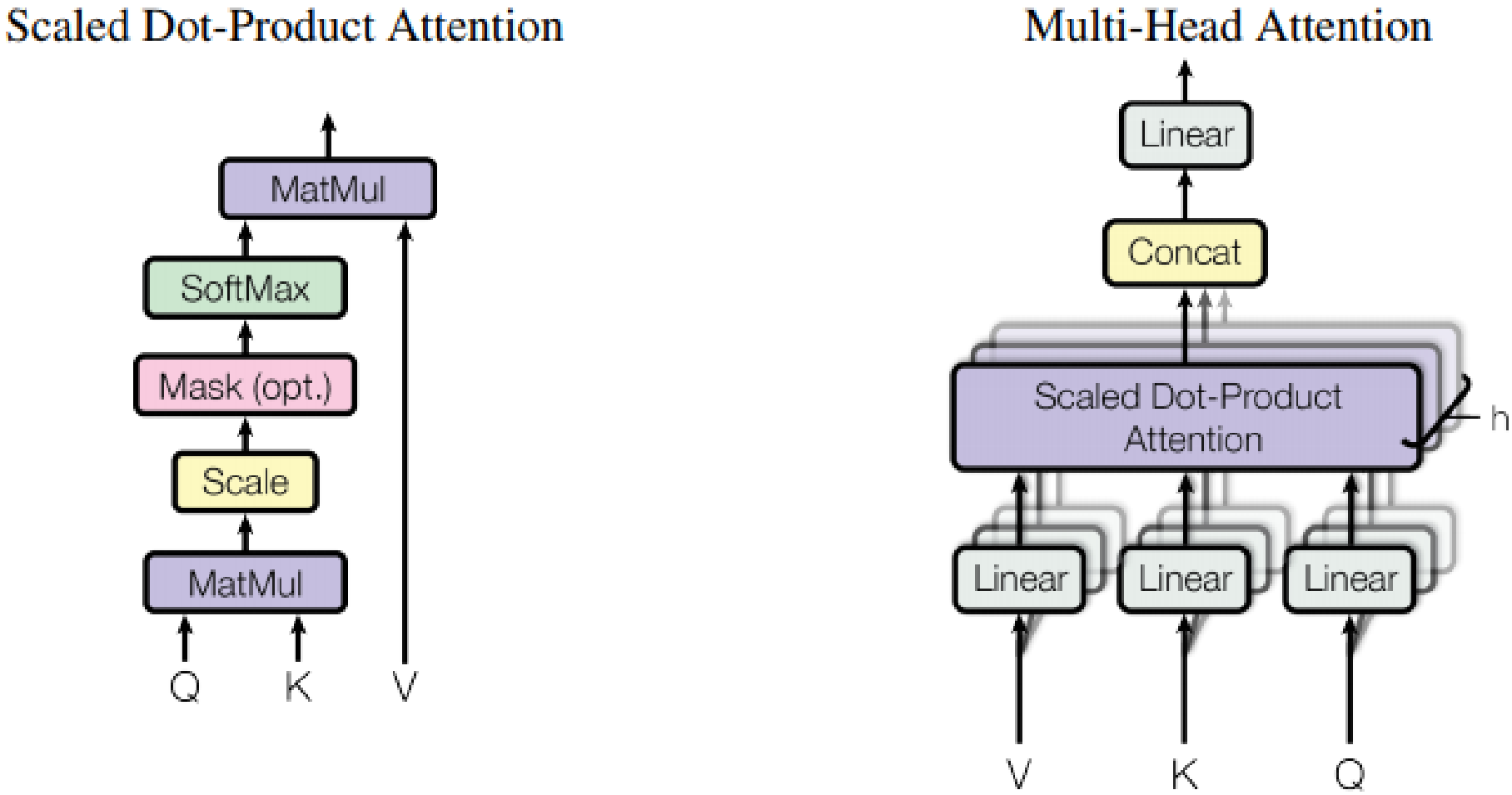
Scaled Dot-Product Attention

Problem: As d_k gets large, the variance of $\boldsymbol{q}^\top \boldsymbol{k}$ increases, the softmax gets very peaked, hence its gradient gets smaller.

Solution: scale by length of query/key vectors:

$$\bar{\mathbf{V}} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}.$$

Scaled Dot-Product and Multi-Head Attention



(Vaswani et al., 2017)

Multi-Head Attention

Self-attention lets each word state form a query vector and attend to the other words' key vectors

This is vaguely similar to a 1D convolution, but where the filter weights are "dynamic" is the window size spans the entire sentence!

Problem: only one channel for words to interact with one-another

Solution: multi-head attention!

- first project $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ into lower dimensional spaces
- then apply attention in multiple channels, concatenate the outputs and pipe through linear layer:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O,$$

where $\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$.

Other Tricks

- Self-attention blocks are repeated 6 times
- Residual connections on each attention block
- Positional encodings (to distinguish word positions)
- Layer normalization

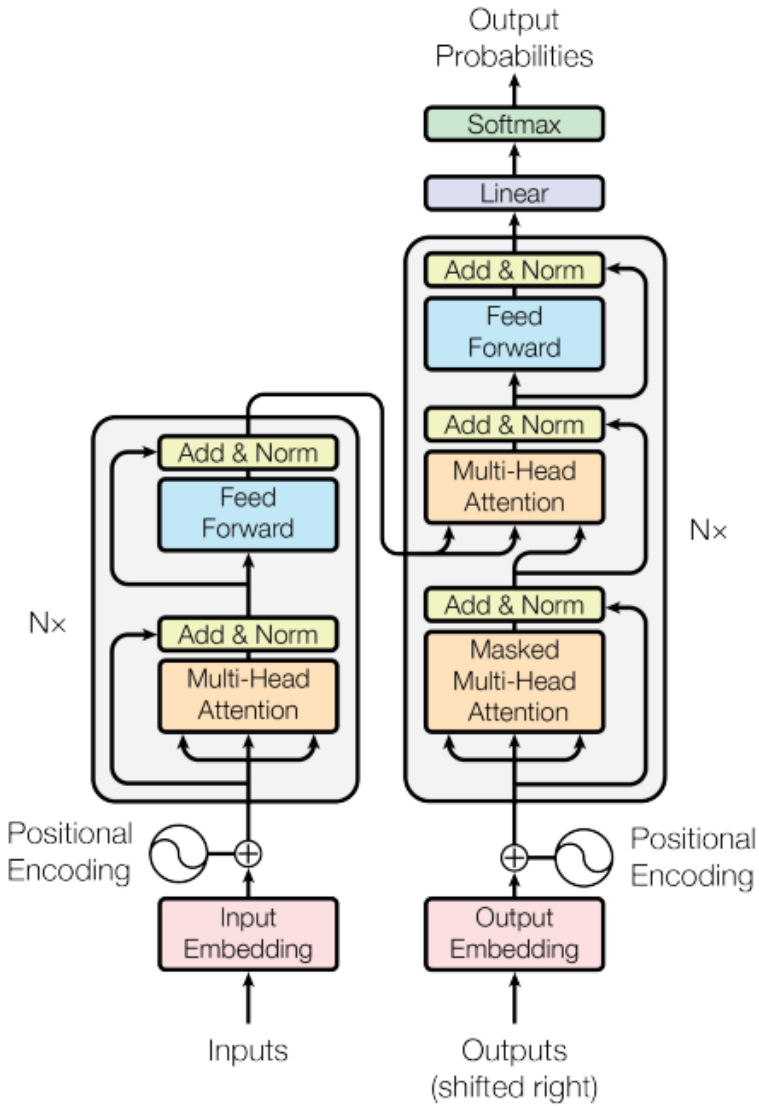


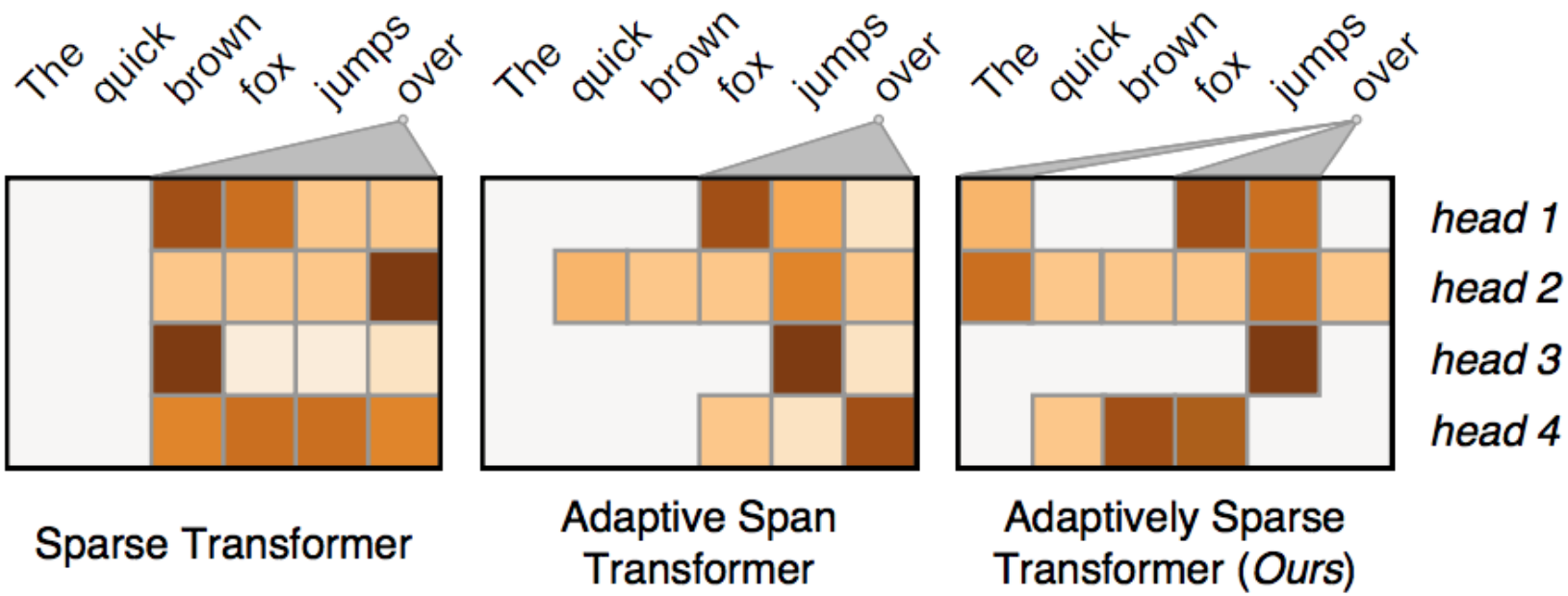
Figure 1: The Transformer - model architecture.

Adaptively Sparse Transformers (Correia et al., 2019)

Key idea: replace softmax in attention heads by α-entmax!

- We saw that a scalar parameter (α) influences how sparse the distribution will be
- Similar to a temperature parameter, but more stable
- Can we learn α?
- Idea: learn adaptively how sparse transformer attentions should be
- One α for each attention head and each layer

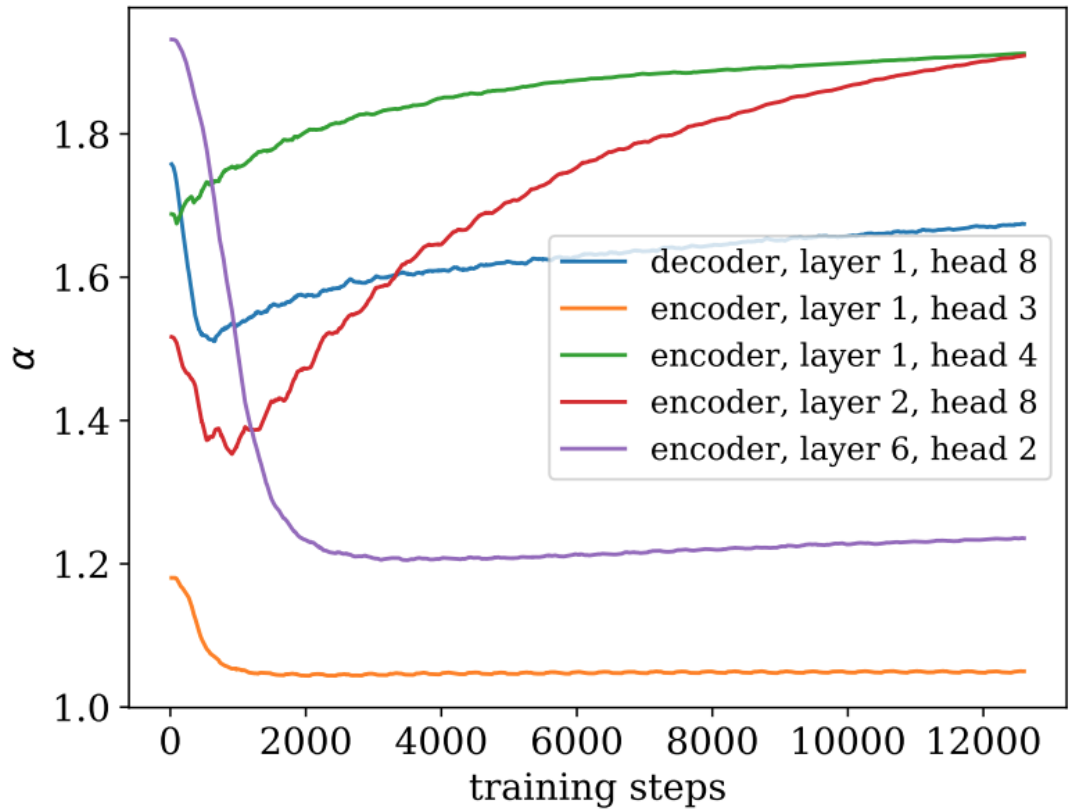
Related Work: Other Sparse Transformers



Sparse Transformer (Child et al., 2019) and the adaptive span Transformer (Sukhbaatar et al., 2019) only attend to words within a contiguous span

Our model: different and **not necessarily contiguous** sparsity patterns for each attention head; learn it **adaptively**

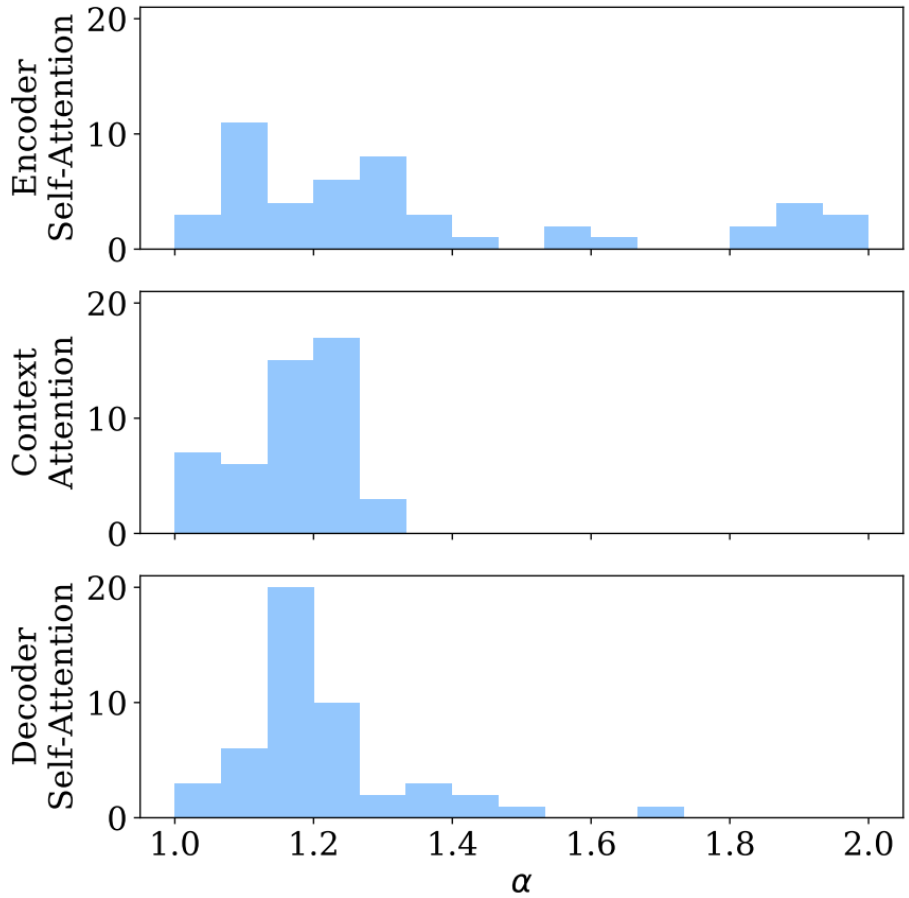
Trajectories of α During Training



Most heads become denser in the beginning, before converging.

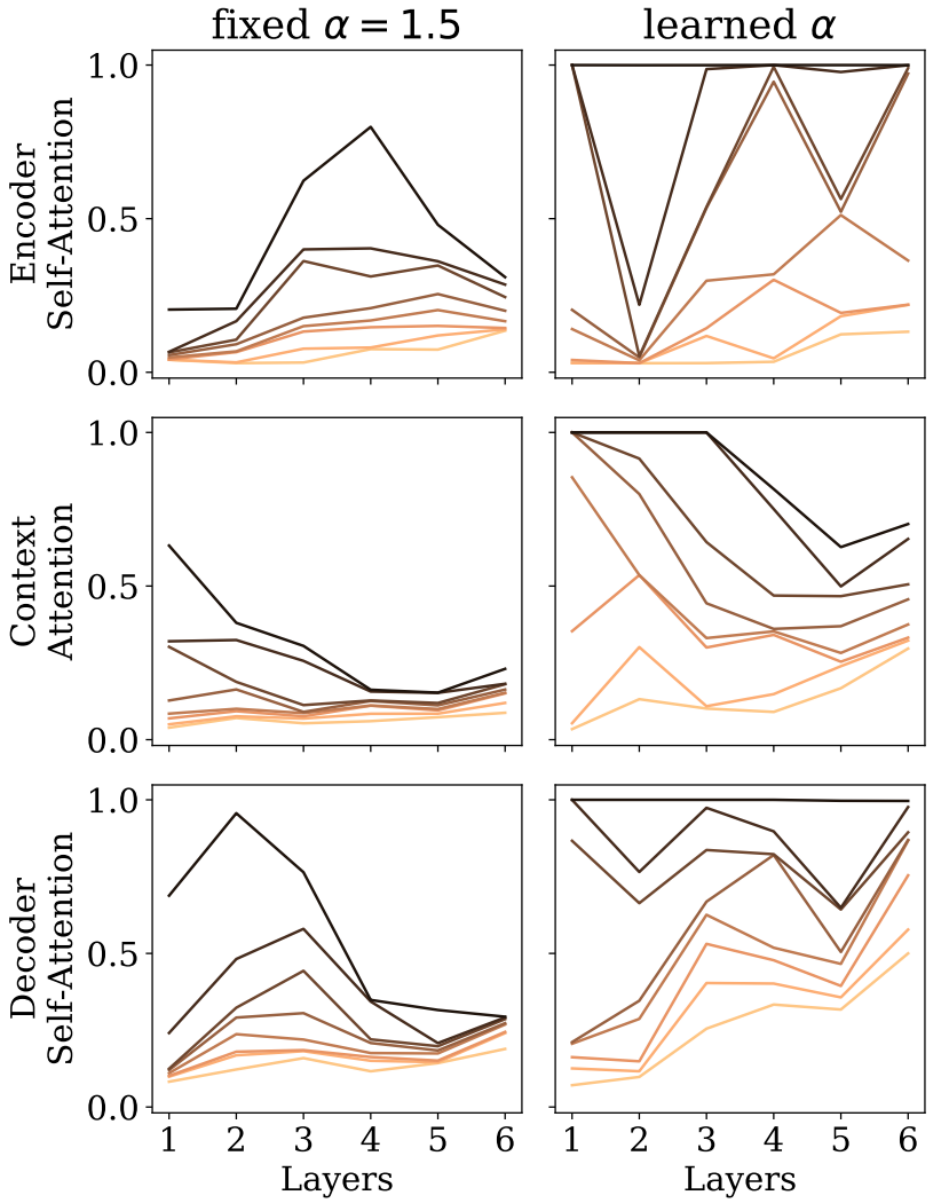
This suggests that dense attention may be more beneficial while the network is still uncertain, being replaced by sparse attention afterwards.

Learned α

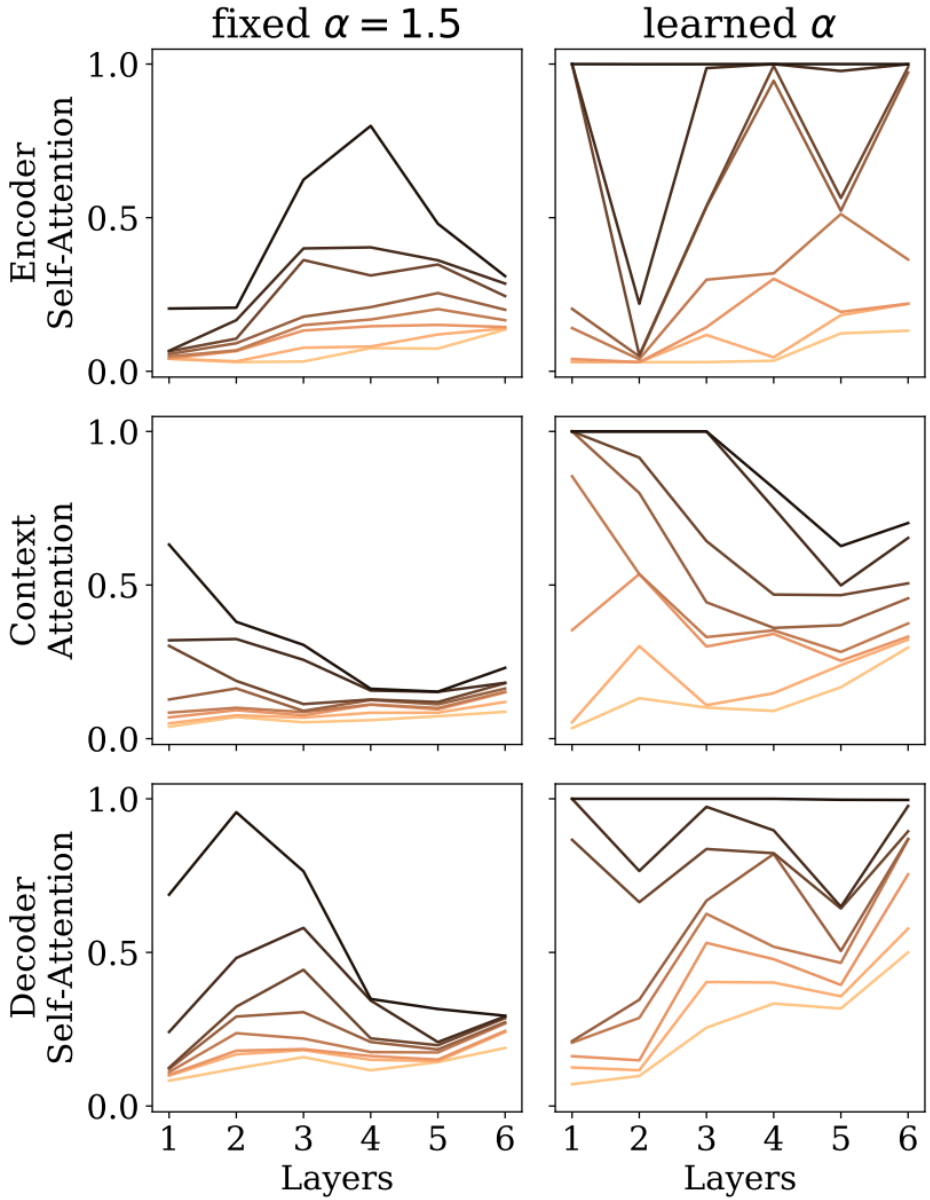


Bimodal for the encoder, mostly unimodal for the decoder.

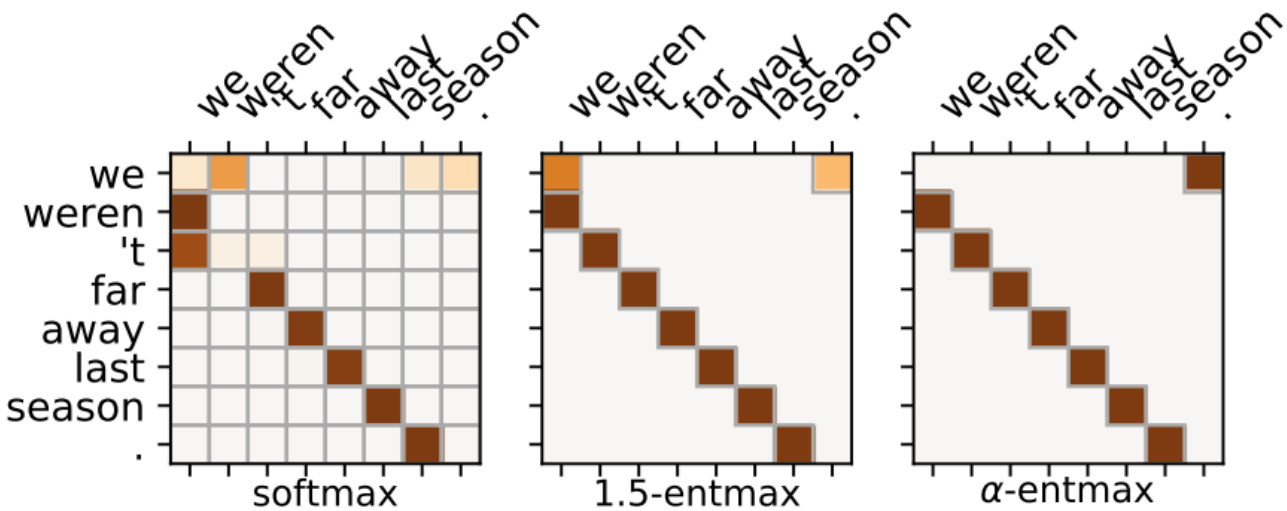
Head Density Per Layer



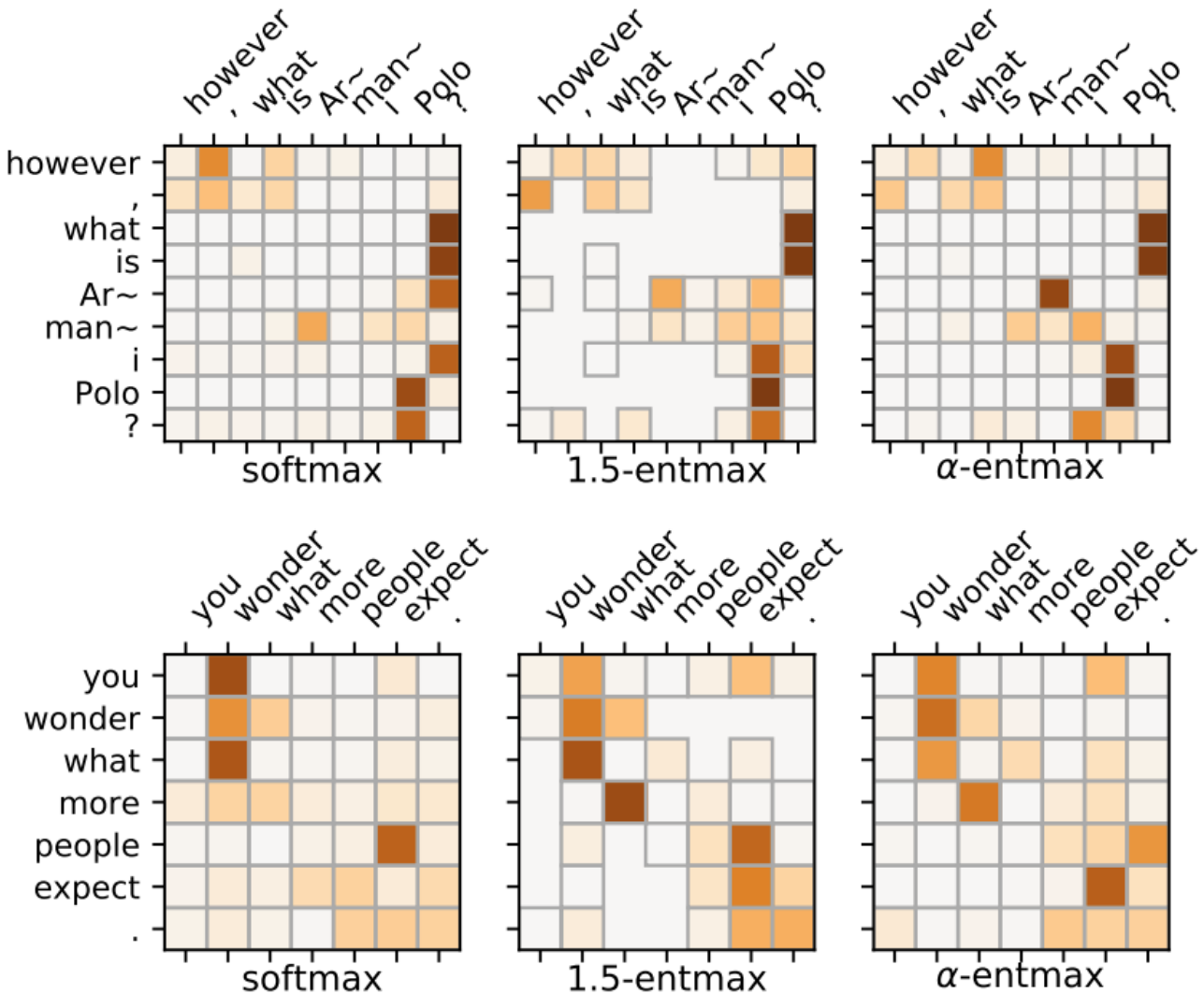
Jensen-Shannon Divergence Between Heads



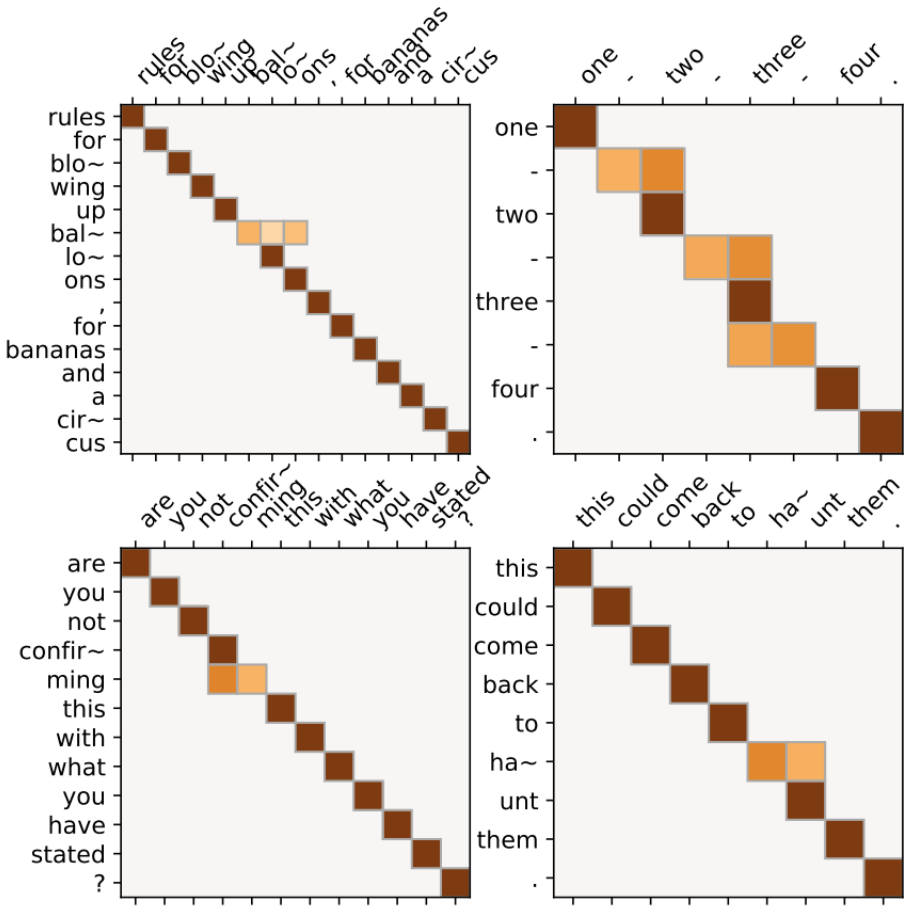
Previous Position Head



Interrogation-Detecting Head



Subword-Merging Head



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
- The sparsity can be adaptive
- Various experiments in NMT (RNN and Transformers) with improved interpretability

We're Hiring!

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

Unbabel

DeepSPIN

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

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

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*.

Blondel, M., Martins, A. F. T., and Niculae, V. (2018). Learning Classifiers with Fenchel-Young Losses: Generalized Entropies, Margins, and Algorithms. *AISTATS*.

Correia, G., Niculae, V., and Martins, A. F. T. (2019). Adaptively sparse transformers. In *Proceedings of the Empirical Methods for Natural Language Processing*.

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

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*.

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*.

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

Niculae, V., Martins, A. F. T., Blondel, M., and Cardie, C. (2018). SparseMAP: Differentiable Sparse Structured Inference. In *Proc. of the International Conference on Machine Learning*.

References II

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*.

Peters, B., Niculae, V., and Martins, A. F. T. (2019). Sparse sequence-to-sequence models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

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.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.