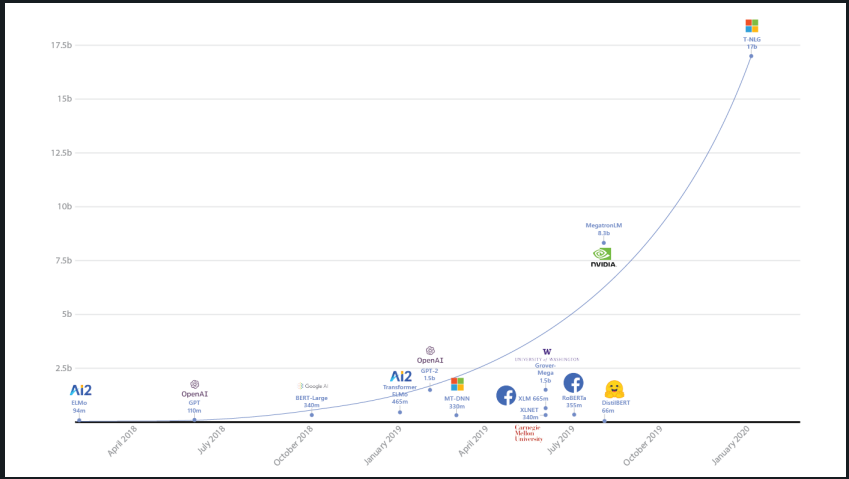
Thesis Proposal:

Neural Machine Translation with Latent Context

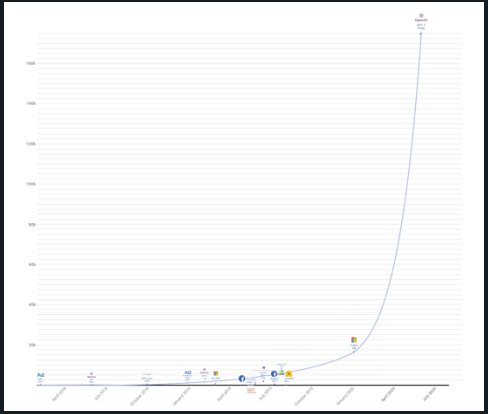
Gonçalo Correia

Committee: André Martins, Mário Figueiredo, Vlad Niculae, Wilker Aziz

Neural networks: big, uninterpretable, and powerful black-boxes?



Neural networks: big, uninterpretable, and powerful black-boxes?



Key concepts of this thesis

- Transparency

Key concepts of this thesis

- Transparency
- Sparsity

Key concepts of this thesis

- Transparency
- Sparsity
- Compactness

Key concepts of this thesis

- Transparency
- Sparsity
- Compactness
- Data efficiency

(Brown et al., 1993)

Key concepts of this thesis

- Transparency
- Sparsity
- Compactness
- Data efficiency

Neural Machine Translation (NMT)

- Advances in optimization → ditch structure in favour of direct decomposition

Neural Machine Translation (NMT)

- Advances in optimization → ditch structure in favour of direct decomposition
- Good for high-resource language, fails in low-resource settings

(Bahdanau et al., 2015)

A bit of context... On Seq2Seq

United Nations elections end today

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

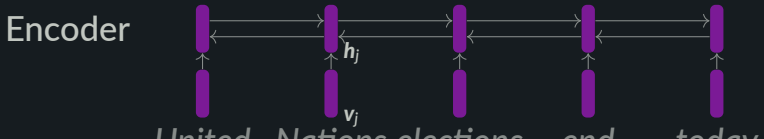
A bit of context… On Seq2Seq

Encoder

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

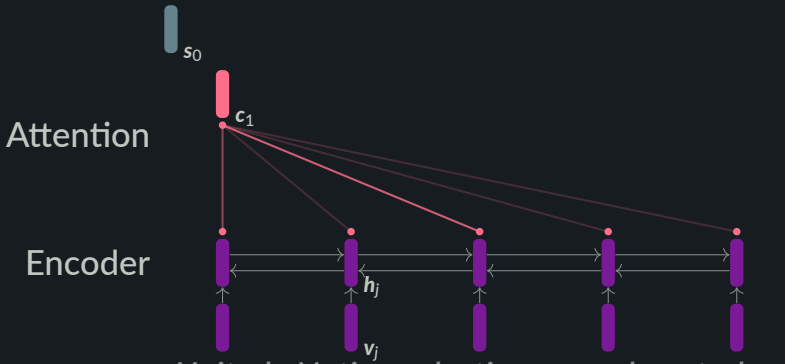
A bit of context... On Seq2Seq



Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq



attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

$$\pi_j = \text{softmax}_j(z)$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context... On Seq2Seq

attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

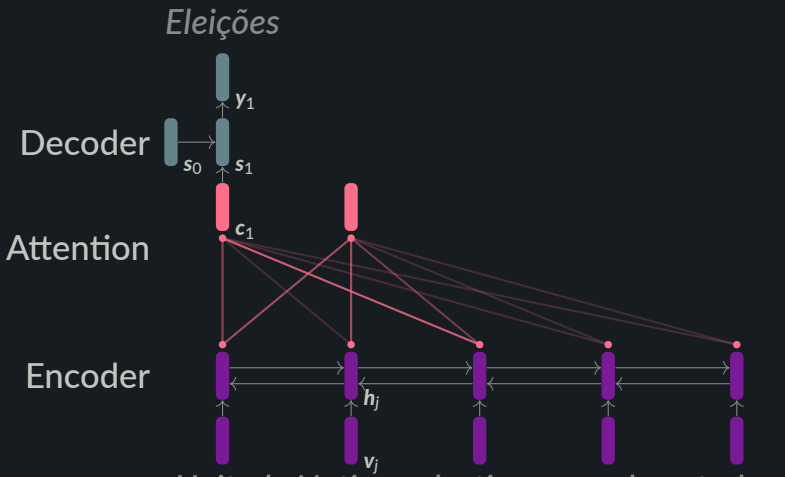
$$\pi_j = \text{softmax}_j(\mathbf{z})$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq



attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

$$\pi_j = \text{softmax}_j(z)$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq

attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

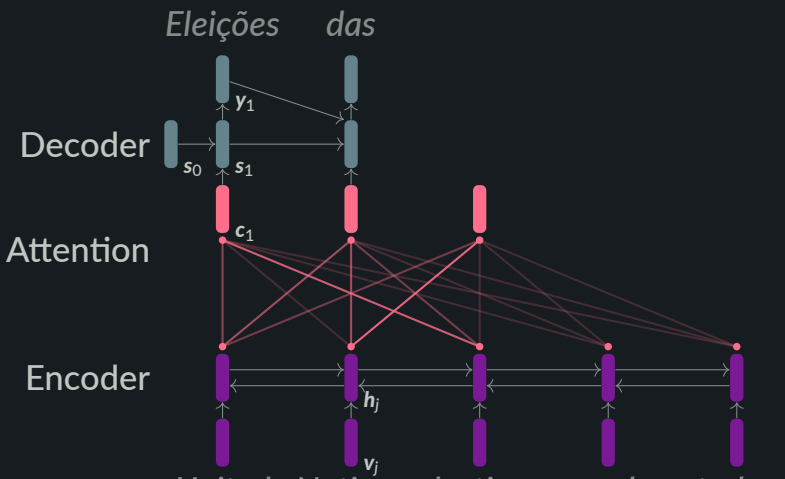
$$\pi_j = \text{softmax}_j(\mathbf{z})$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq



attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

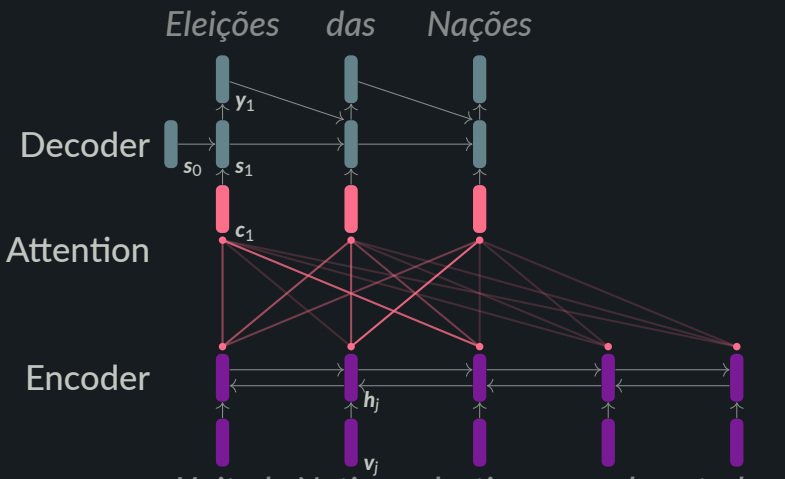
$$\pi_j = \text{softmax}_j(\mathbf{z})$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq



attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

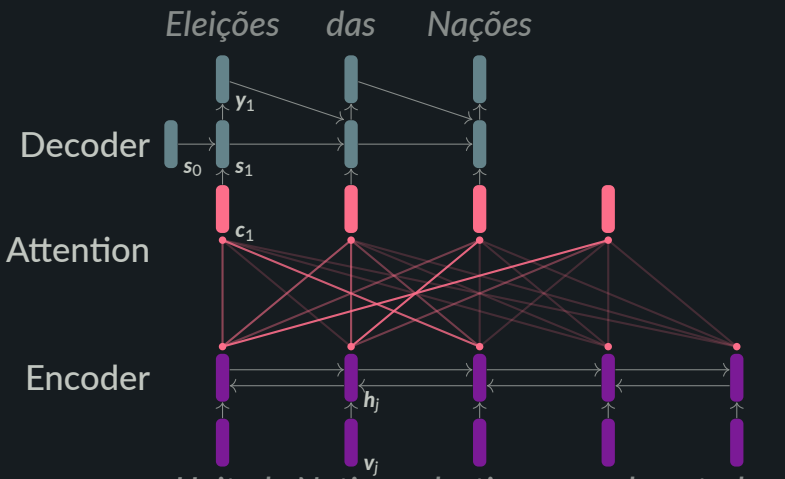
$$\pi_j = \text{softmax}_j(\mathbf{z})$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq



attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

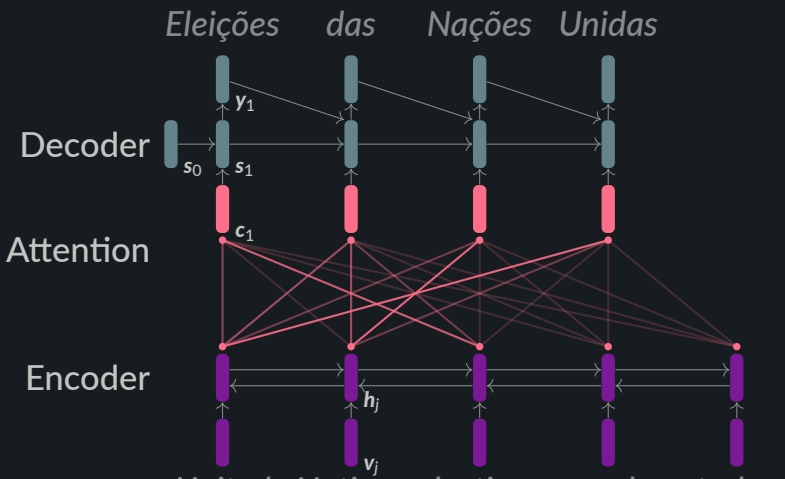
$$\pi_j = \text{softmax}_j(z)$$

$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Bahdanau et al., 2015)

A bit of context… On Seq2Seq



attention weights computed with *softmax*:

for some decoder state s_t, compute contextually weighted average of input c_t:

$$z_j = s_t^\top W^{(a)} h_j$$

$$\pi_j = \text{softmax}_j(z)$$

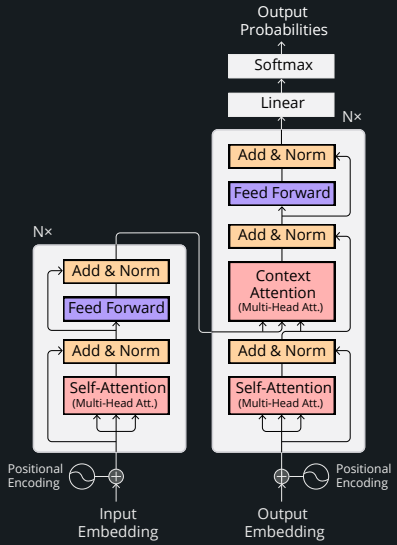
$$c_t = \sum_j \pi_j h_j$$

Slide Credit: Vlad Niculae

(Vaswani et al., 2017)

A bit of context... on Transformers

What if... Attention is all you need?

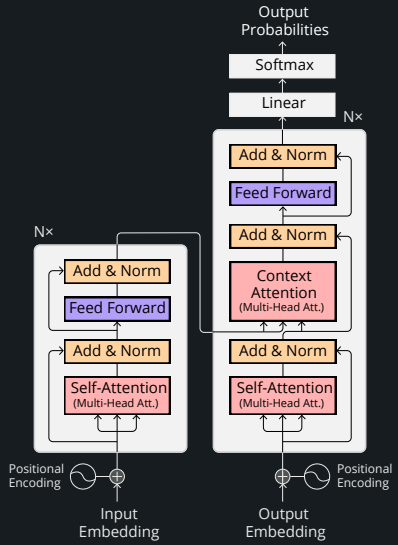


(Vaswani et al., 2017)

A bit of context... on Transformers

What if... Attention is all you need?

Key idea: Instead of Recurrent Neural Networks (RNNs), let's use attention mechanisms!



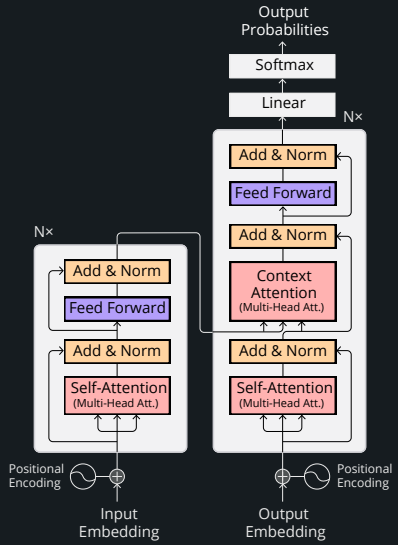
(Vaswani et al., 2017)

A bit of context... on Transformers

What if... Attention is all you need?

Key idea: Instead of Recurrent Neural Networks (RNNs), let's use attention mechanisms!

- In place of the RNNs, use self-attention



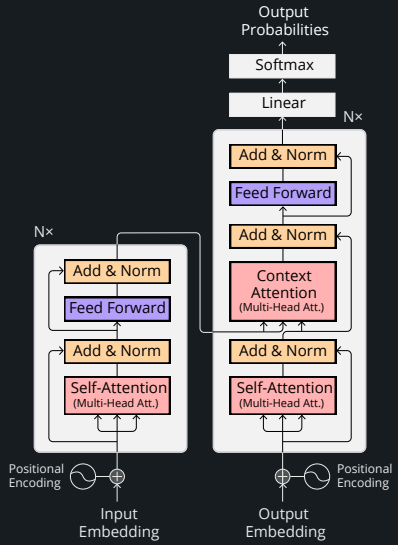
(Vaswani et al., 2017)

A bit of context... on Transformers

What if... Attention is all you need?

Key idea: Instead of Recurrent Neural Networks (RNNs), let's use attention mechanisms!

- In place of the RNNs, use self-attention
- Do this with multiple heads (i.e. attention mechanisms in parallel)



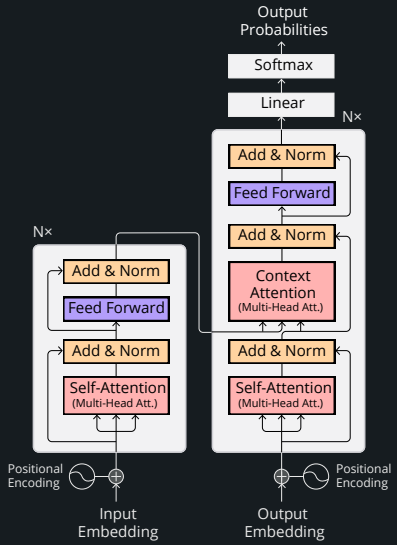
(Vaswani et al., 2017)

A bit of context... on Transformers

What if... Attention is all you need?

Key idea: Instead of Recurrent Neural Networks (RNNs), let's use attention mechanisms!

- In place of the RNNs, use self-attention
- Do this with multiple heads (i.e. attention mechanisms in parallel)
- ... and do it through several layers



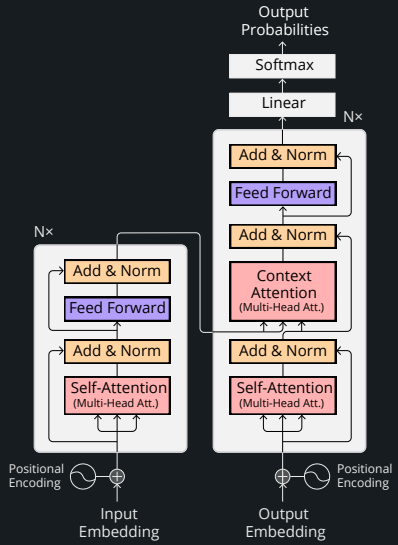
(Devlin et al., 2019)

A bit of context... on Transformers

What if... Attention is all you need?

Key idea: Instead of Recurrent Neural Networks (RNNs), let's use attention mechanisms!

- In place of the RNNs, use self-attention
- Do this with multiple heads (i.e. attention mechanisms in parallel)
- ... and do it through several layers
- Inspiration for big general-purpose models like BERT!



Outline

(Correia and Martins, 2019)

Outline

- Automatic Post-Editing (APE) using BERT (ACL 2019)

(Correia, Niculae, and Martins, 2019)

Outline

- Automatic Post-Editing (APE) using BERT (ACL 2019)
- Letting Transformer learn how sparse it wants its attention (EMNLP 2019)

(Correia, Niculae, Aziz, et al., 2020)

Outline

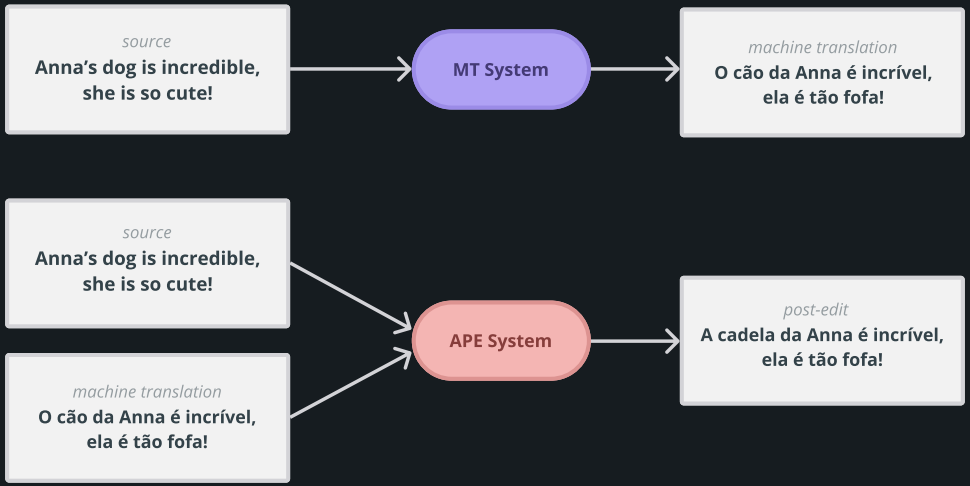
- Automatic Post-Editing (APE) using BERT (ACL 2019)
- Letting Transformer learn how sparse it wants its attention (EMNLP 2019)
- General strategy for efficient training discrete latent variables (NeurIPS 2020)

Outline

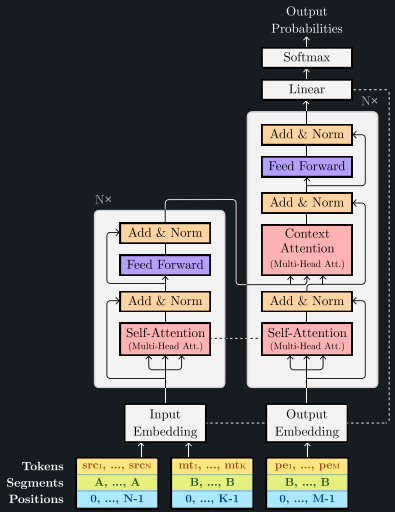
- Automatic Post-Editing (APE) using BERT (ACL 2019)
- Letting Transformer learn how sparse it wants its attention (EMNLP 2019)
- General strategy for efficient training discrete latent variables (NeurIPS 2020)
- Future work: powerful structured latent variable models for NMT

Table of Contents

What is APE?

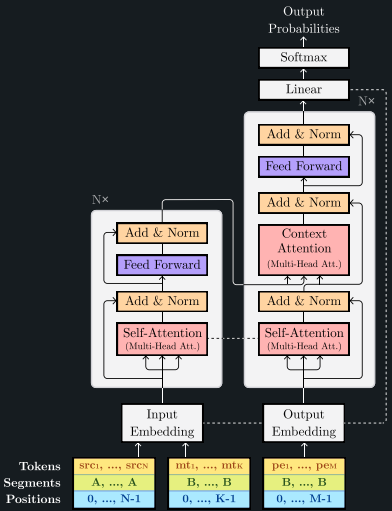

source
Anna's dog is incredible, she is so cute!
MT System
machine translation
O cão da Anna é incrível, ela é tão fofa!
source
Anna's dog is incredible, she is so cute!
machine translation
O cão da Anna é incrível, ela é tão fofa!
APE System
post-edit
A cadela da Anna é incrível, ela é tão fofa!

BERT for APE


Output Probabilities
Softmax
Linear
N×
Add & Norm
Feed Forward
Add & Norm
Context Attention
(Multi-Head Att.)
Add & Norm
Self-Attention
(Multi-Head Att.)
N×
Add & Norm
Feed Forward
Add & Norm
Self-Attention
(Multi-Head Att.)
Input Embedding
Output Embedding
Tokens
Segments
Positions
src1, ..., srcN
A, ..., A
0, ..., N-1
mt1, ..., mtK
B, ..., B
0, ..., K-1
pe1, ..., peM
B, ..., B
0, ..., M-1

BERT for APE

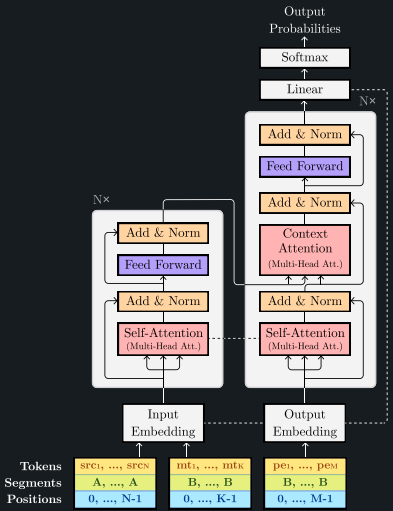
Key idea: Use BERT to do APE



BERT for APE

Key idea: Use BERT to do APE

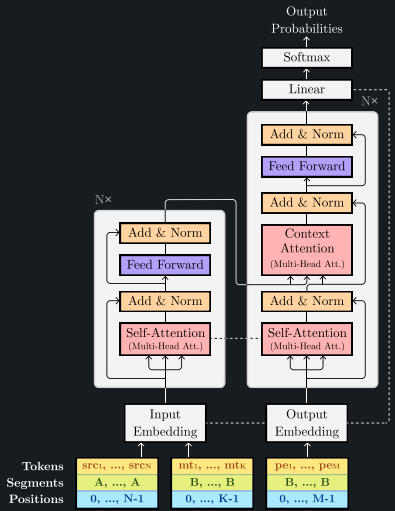
- Prior to this work, BERT was mainly used for simple classification tasks



BERT for APE

Key idea: Use BERT to do APE

- Prior to this work, BERT was mainly used for simple classification tasks
- We introduced an effective method to use BERT in a generation task (APE)



BERT for APE

Key idea: Use BERT to do APE

- Prior to this work, BERT was mainly used for simple classification tasks
- We introduced an effective method to use BERT in a generation task (APE)
- Smart parameter sharing between encoder and decoder

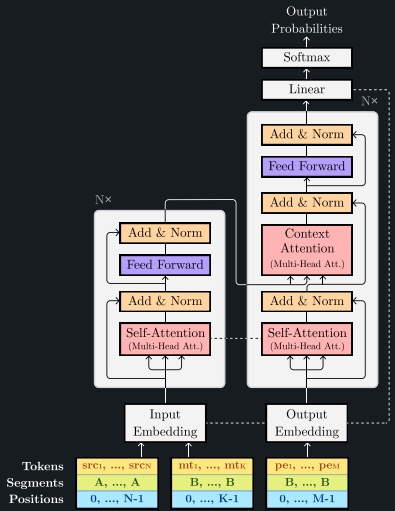


Table of Contents

Getting to know attention heads better

Attention heads may aid visualization but they are completely dense.

Getting to know attention heads better

Attention heads may aid visualization but they are completely dense.

Our solution is to bet on sparsity:

- for interpretability
- for discovering linguistic structure
- for efficiency

Getting to know attention heads better

Attention heads may aid visualization but they are completely dense.

Our solution is to bet on sparsity:

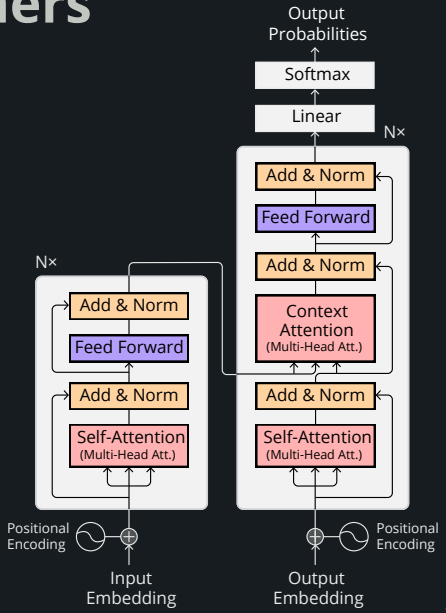
- for interpretability
- for discovering linguistic structure
- for efficiency

(Vaswani et al., 2017)

Transformers

In each attention head:

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



(Vaswani et al., 2017)

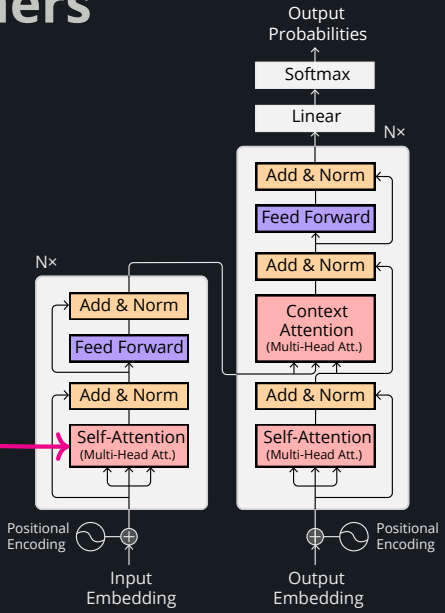
Transformers

In each attention head:

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

Attention in three places:

- Self-attention in the encoder



(Vaswani et al., 2017)

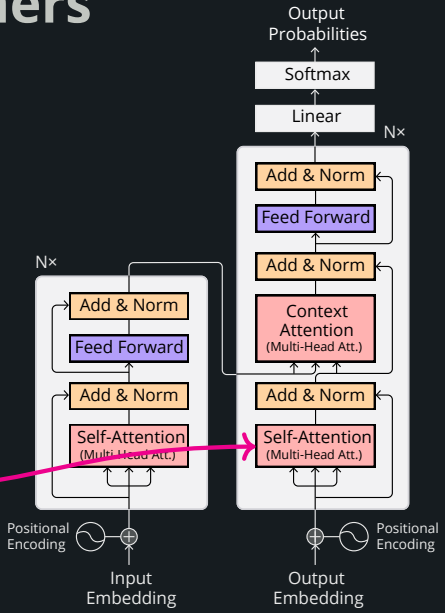
Transformers

In each attention head:

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

Attention in three places:

- Self-attention in the encoder
- Self-attention in the decoder



(Vaswani et al., 2017)

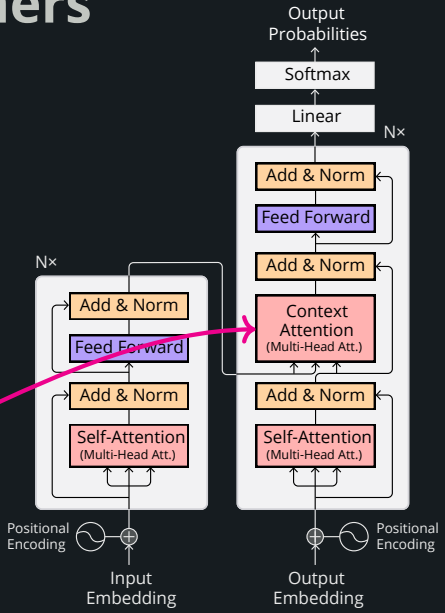
Transformers

In each attention head:

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

Attention in three places:

- Self-attention in the encoder
- Self-attention in the decoder
- Contextual attention



Sparse Transformers

Sparse Transformers

Key idea: replace softmax in attention heads by a sparse normalizing function! 🙌

Adaptively Sparse Transformers

Key idea: replace softmax in attention heads by a sparse normalizing function! 🙌

Another key idea: use a normalizing function that is adaptively sparse via a learnable α! 🙌 🙌 🙌

What is softmax?

Softmax exponentiates and normalizes:

$$\textbf{softmax}(z_i) := \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

What is softmax?

Softmax exponentiates and normalizes:

$$\textbf{softmax}(z_i) := \frac{\exp(z_i)}{\sum_j \exp(z_j)}$$

It's fully dense: $\textbf{softmax}(\boldsymbol{z}) > \mathbf{0}$

(Peters et al., 2019)

α-Entmax

Parametrized by $\alpha \geq 0$:

(Peters et al., 2019)

α-Entmax

Parametrized by $\alpha \geq 0$:

- **Argmax** corresponds to $\alpha \rightarrow \infty$

(Peters et al., 2019)

α-Entmax

Parametrized by $\alpha \geq 0$:

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

(Peters et al., 2019)

α-Entmax

Parametrized by $\alpha \geq 0$:

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

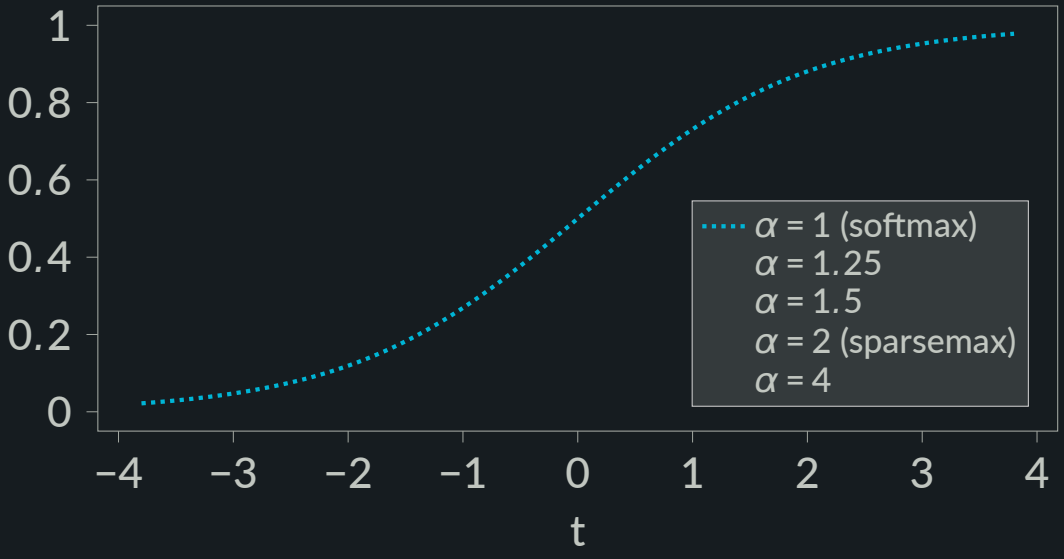
(Peters et al., 2019)

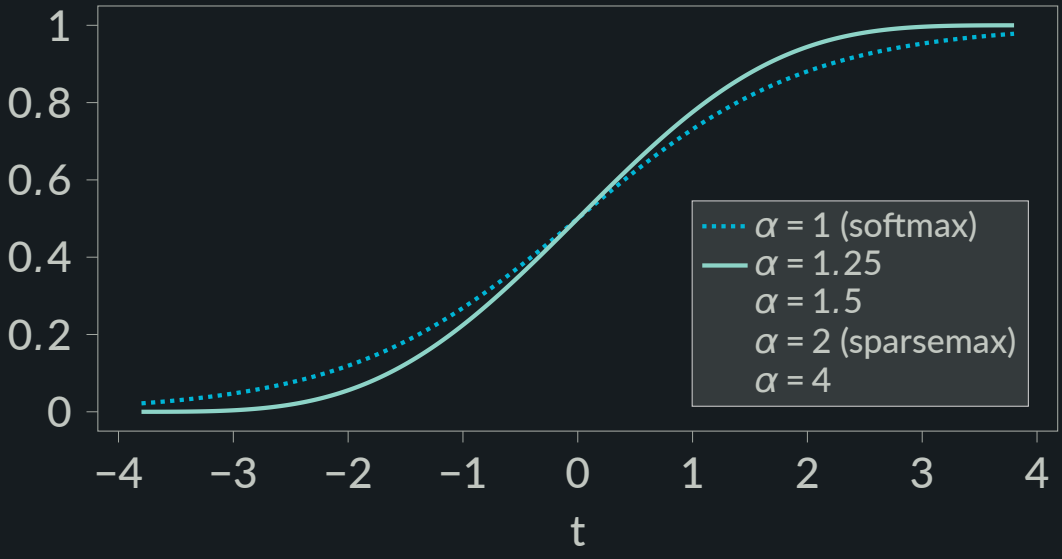
α-Entmax

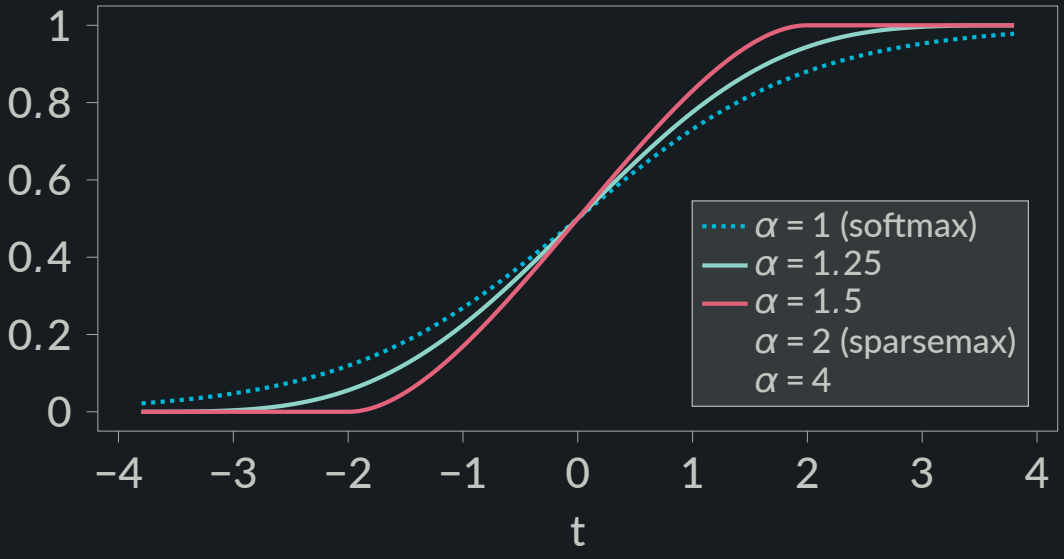
Parametrized by $\alpha \geq 0$:

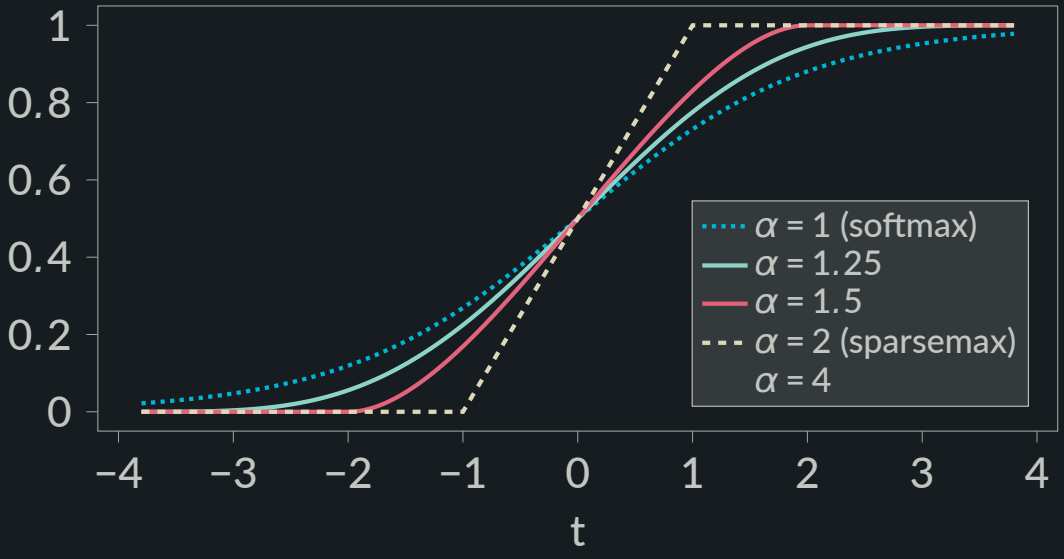
- **Argmax** corresponds to $\alpha \to \infty$
- **Softmax** amounts to $\alpha \to 1$
- **Sparsemax** amounts to $\alpha = 2$.

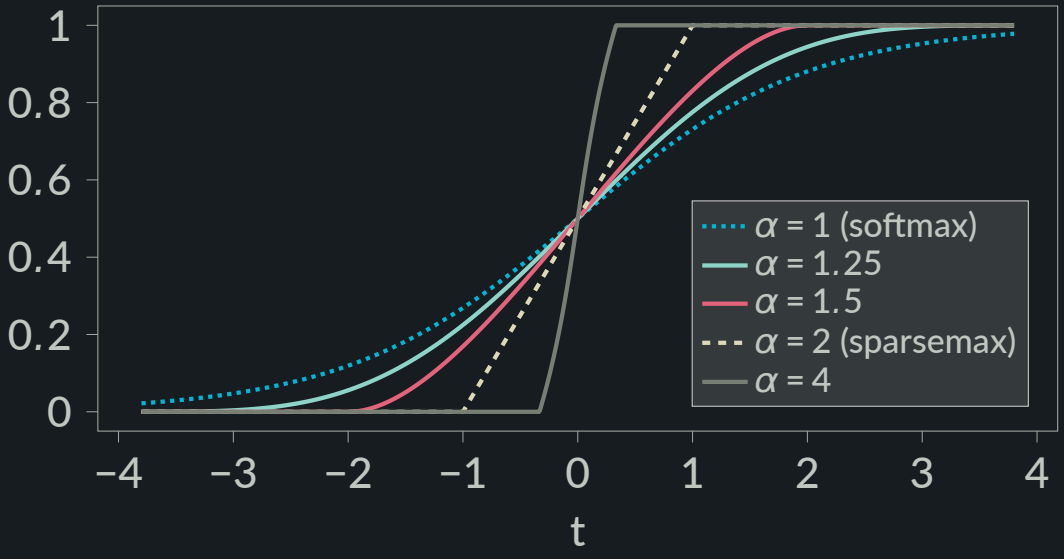
Key result: **can be sparse for** $\alpha > 1$, propensity for sparsity increases with α.


1
0.8
0.6
0.4
0.2
0
−4
−3
−2
−1
0
1
2
3
4
t
α = 1 (softmax)
α = 1.25
α = 1.5
α = 2 (sparsemax)
α = 4


1
0.8
0.6
0.4
0.2
0
−4
−3
−2
−1
0
1
2
3
4
t
α = 1 (softmax)
α = 1.25
α = 1.5
α = 2 (sparsemax)
α = 4


1
0.8
0.6
0.4
0.2
0
−4
−3
−2
−1
0
1
2
3
4
t
α = 1 (softmax)
α = 1.25
α = 1.5
α = 2 (sparsemax)
α = 4


1
0.8
0.6
0.4
0.2
0
−4
−3
−2
−1
0
1
2
3
4
t
α = 1 (softmax)
α = 1.25
α = 1.5
α = 2 (sparsemax)
α = 4


1
0.8
0.6
0.4
0.2
0
−4
−3
−2
−1
0
1
2
3
4
t
α = 1 (softmax)
α = 1.25
α = 1.5
α = 2 (sparsemax)
α = 4

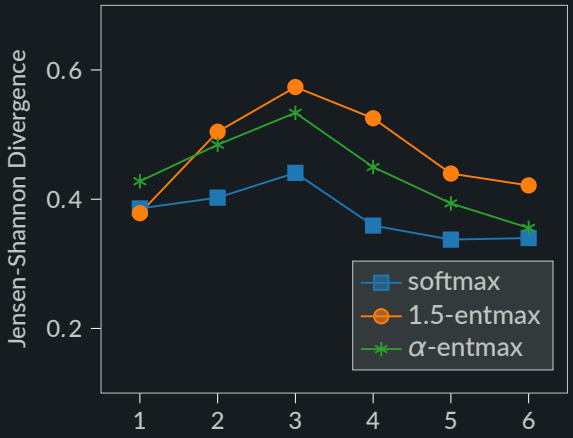
Learning α

Learning α

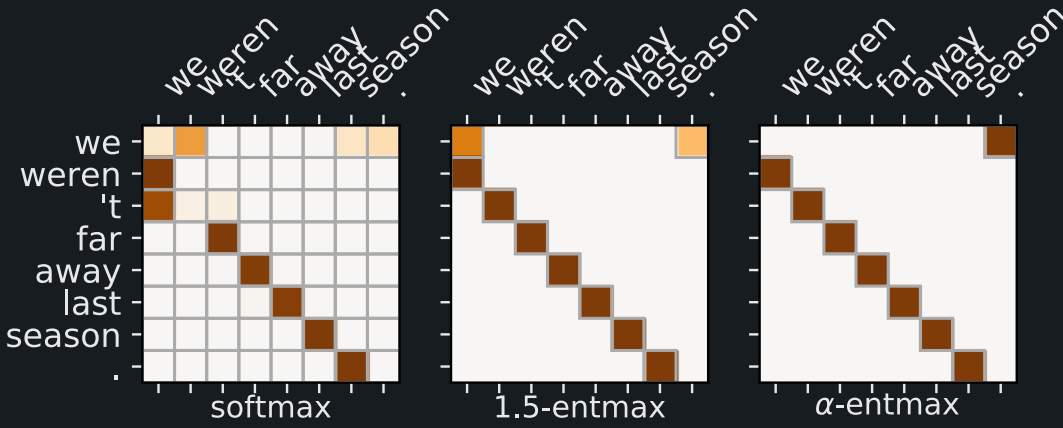
Key contribution:

a closed-form expression for $\frac{\partial\, \alpha\text{-entmax}(\boldsymbol{z})}{\partial \alpha}$ 😱

Head Diversity per Layer

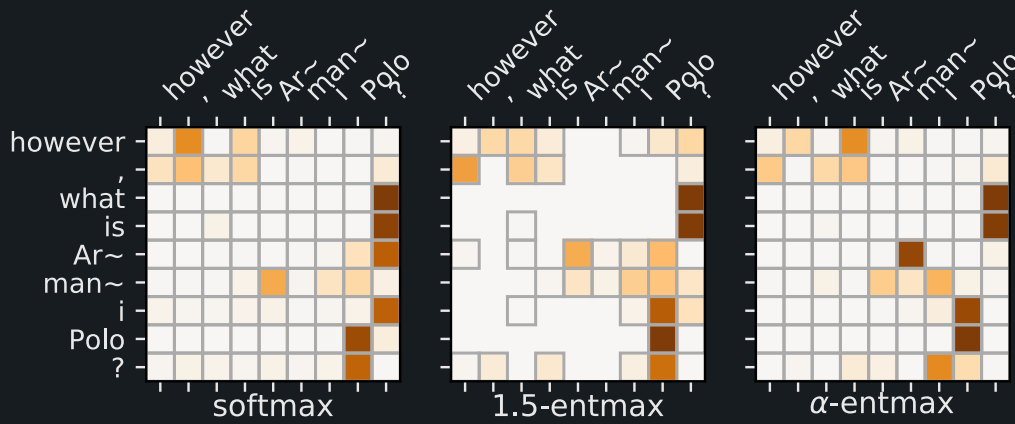


Previous Position Head



This head role was also found in Voita et al. (2019)! Learned $\alpha = 1.91$.

Interrogation-Detecting Head



Learned $\alpha = 1.05$.

Table of Contents

Latent Variable Models

Latent variable z can be

Latent Variable Models

Latent variable z can be continuous

Source: Bouges et al., 2013

Latent Variable Models

Latent variable z can be continuous, discrete

Source: valleyeyecareaz.com

Latent Variable Models

Latent variable z can be continuous, discrete, or structured

Source: Liu et al., 2015

Training Discrete or Structured Latent Variable Models

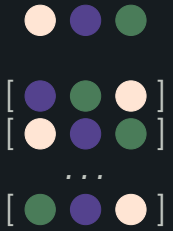
Latent variable z can be

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete

Training Discrete or Structured Latent Variable Models

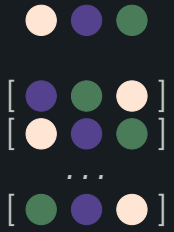
Latent variable z can be discrete or structured



Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

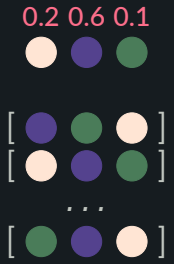
$\pi(z|x, \theta)$: distribution over possible z



Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z



Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

To train, we need to compute the following expectation:

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

To train, we need to compute the following expectation:

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta)$$

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

To train, we need to compute the following expectation:

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta)$$

If $\mathcal{Z}$ is large, this sum can get very expensive due to $\ell(x, z; \theta)$! 😱

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

To train, we need to compute the following expectation:

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta)$$

If $\mathcal{Z}$ is combinatorial, this can be intractable to compute! 😱 😱 😱

Current Solutions

If $\mathcal{Z}$ is large, exact gradient computation is prohibitive

Current Solutions

If $\mathcal{Z}$ is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance

Current Solutions

If $\mathcal{Z}$ is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance
Another option: Gumbel-Softmax → continuous relaxation, biased estimation

Current Solutions

If $\mathcal{Z}$ is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance
Another option: Gumbel-Softmax → continuous relaxation, biased estimation

New option: use sparsity! 🙌

Current Solutions

If $\mathcal{Z}$ is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance
Another option: Gumbel-Softmax → continuous relaxation, biased estimation

New option: use sparsity! 🙌

no need for sampling → no variance

Current Solutions

If $\mathcal{Z}$ is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance
Another option: Gumbel-Softmax → continuous relaxation, biased estimation

New option: use sparsity! 🙌

no need for sampling → no variance
no relaxation into the continuous space

Taking a step back...

Does the expectation over possible z need to be expensive?

Taking a step back...

Does the expectation over possible z need to be expensive?

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta) \\ &= \pi(z_1|x, \theta)\, \ell(x, z_1; \theta) + \pi(z_2|x, \theta)\, \ell(x, z_2; \theta) + \ldots \\ &+ \pi(z_i\,|x, \theta)\, \ell(x, z_i\,; \theta) + \ldots + \pi(z_N|x, \theta)\, \ell(x, z_N; \theta)\end{aligned}$$

Taking a step back...

Does the expectation over possible z need to be expensive?

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta) \\ &= \pi(z_1|x, \theta)\, \ell(x, z_1; \theta) + \pi(z_2|x, \theta)\, \ell(x, z_2; \theta) + \dots \\ &+ \pi(z_i\,|x, \theta)\, \ell(x, z_i\,; \theta) + \dots + \pi(z_N|x, \theta)\, \ell(x, z_N; \theta)\end{aligned}$$

Usually we normalize π with softmax $\propto \exp(s) \Rightarrow \pi(z_i\,|x, \theta) > 0$

(Martins and Astudillo, 2016; Niculae et al., 2018)

Sparse normalizers

We use sparsemax, top-k sparsemax and SparseMAP to allow efficient marginalization

(Martins and Astudillo, 2016; Niculae et al., 2018)

Sparse normalizers

We use sparsemax, top-k sparsemax and SparseMAP to allow efficient marginalization

These functions are able to assign **probabilities of exactly zero**!

(Martins and Astudillo, 2016; Niculae et al., 2018)

Sparse normalizers

We use sparsemax, top-k sparsemax and SparseMAP to allow efficient marginalization

These functions are able to assign **probabilities of exactly zero**!

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta) \\ &= \pi(z_1|x, \theta)\, \ell(x, z_1; \theta) + \underbrace{\pi(z_2|x, \theta)}_{=0}\, \ell(x, z_2; \theta) + \dots \\ &+ \pi(z_i|x, \theta)\, \ell(x, z_i; \theta) + \dots + \underbrace{\pi(z_N|x, \theta)}_{=0}\, \ell(x, z_N; \theta)\end{aligned}$$

(Martins and Astudillo, 2016; Niculae et al., 2018)

Sparse normalizers

We use sparsemax, top-k sparsemax and SparseMAP to allow efficient marginalization

These functions are able to assign **probabilities of exactly zero**!

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta)\, \ell(x, z; \theta)$$

$$= \pi(z_1|x, \theta)\, \ell(x, z_1; \theta) + \underbrace{\pi(z_2|x, \theta)}_{=0}\, \cancel{\ell(x, z_2; \theta)} + \dots$$

$$+ \pi(z_i|x, \theta)\, \ell(x, z_i; \theta) + \dots + \underbrace{\pi(z_N|x, \theta)}_{=0}\, \cancel{\ell(x, z_N; \theta)}$$

No need for computing $\ell(x, z; \theta)$ for all $z \in \mathcal{Z}$!

Results

We test our methods for models with discrete latent variables,

Results

We test our methods for models with discrete latent variables,

- Semi-Supervised VAE

Results

We test our methods for models with discrete latent variables,

- Semi-Supervised VAE
- Emergent communication

Results

We test our methods for models with discrete latent variables,

- Semi-Supervised VAE
- Emergent communication

but also in models with an exponentially large set of $\mathcal{Z}$,

Results

We test our methods for models with discrete latent variables,

- Semi-Supervised VAE
- Emergent communication

but also in models with an exponentially large set of $\mathcal{Z}$,

- Bit-vector VAE

Results

We test our methods for models with discrete latent variables,

- Semi-Supervised VAE
- Emergent communication

but also in models with an exponentially large set of $\mathcal{Z}$,

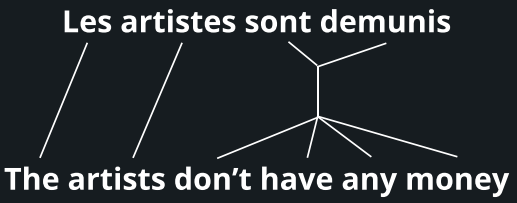
- Bit-vector VAE

Our methods are top-performers and efficient!

Table of Contents

Proposed work: key concepts of this thesis

- Transparency
- Sparsity
- Compactness
- Data efficiency



Proposed work: key concepts of this thesis

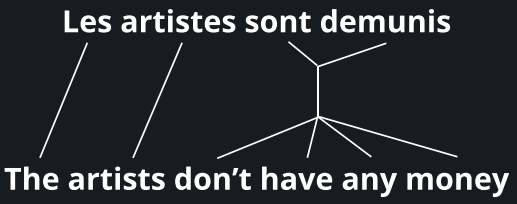
- Transparency
- Sparsity
- Compactness
- Data efficiency

- Inspired by, and data efficient like APE

Proposed work: key concepts of this thesis

- Transparency
- Sparsity
- Compactness
- Data efficiency

- Inspired by, and data efficient like APE
- Transparent like sparse transformers



Proposed work: key concepts of this thesis

- Transparency
- Sparsity
- Compactness
- Data efficiency

- Inspired by, and data efficient like APE
- Transparent like sparse transformers
- Compact thanks to latent variables

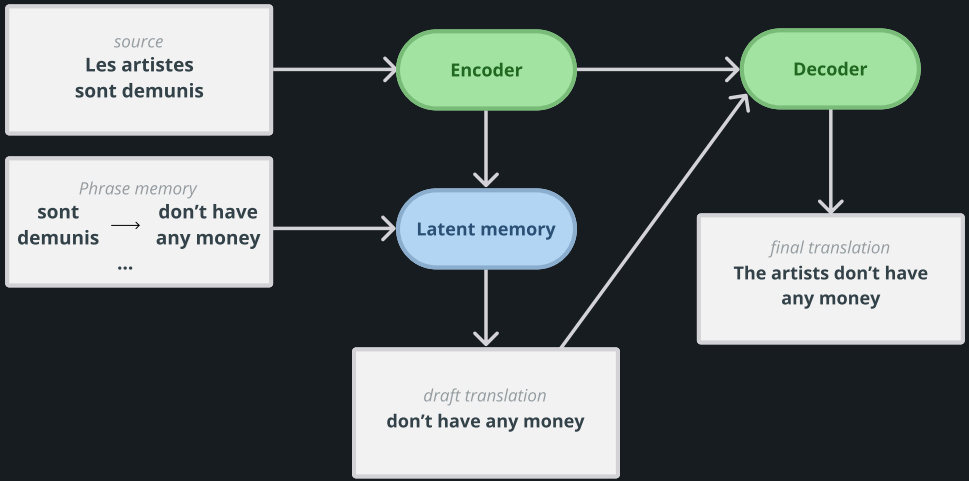
Proposed work: key concepts of this thesis

- Transparency
- Sparsity
- Compactness
- Data efficiency

- Inspired by, and data efficient like APE
- Transparent like sparse transformers
- Compact thanks to latent variables
- Using sparsity as a training method

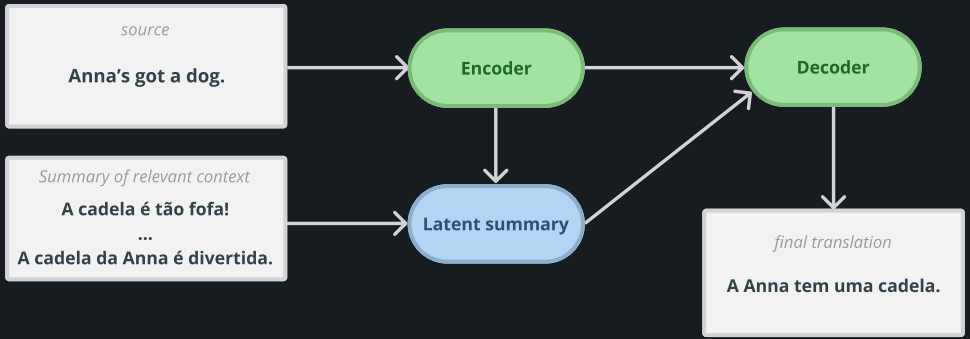
Idea: create a latent, draft translation

Idea: create a latent, draft translation



Another Idea: using latent summaries in NMT

Another Idea: using latent summaries in NMT



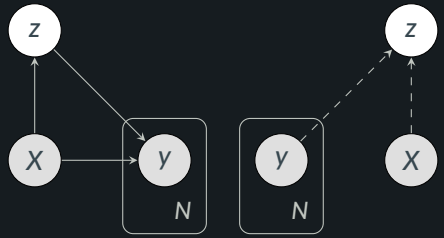
References I

- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). “Neural machine translation by jointly learning to align and translate”. In: *Proc. of ICLR*.
- Bouges, Pierre, Thierry Chateau, Christophe Blanc, and Gaëlle Loosli (Dec. 2013). “Handling missing weak classifiers in boosted cascade: application to multiview and occluded face detection”. In: *EURASIP Journal on Image and Video Processing* 2013, p. 55. DOI: 10.1186/1687-5281-2013-55.
- Brown, Peter F, Stephen A Della Pietra, Vincent J Della Pietra, and Robert L Mercer (1993). “The mathematics of statistical machine translation: Parameter estimation”. In: *Computational linguistics* 19.2, pp. 263–311.
- Correia, Gonçalo M, Vlad Niculae, and André FT Martins (2019). “Adaptively sparse transformers”. In: *Proc. EMNLP*.
- Correia, Gonçalo M. and André F. T. Martins (July 2019). “A Simple and Effective Approach to Automatic Post-Editing with Transfer Learning”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 3050–3056. DOI: 10.18653/v1/P19-1292. URL: https://www.aclweb.org/anthology/P19-1292.
- Correia, Gonçalo M., Vlad Niculae, Wilker Aziz, and André F. T. Martins (2020). “Efficient Marginalization of Discrete and Structured Latent Variables via Sparsity”. In: *Proc. NeurIPS*. URL: https://arxiv.org/abs/2007.01919.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). “BERT: Pre-training of deep bidirectional transformers for language understanding”. In: *Proc. NAACL-HLT*.

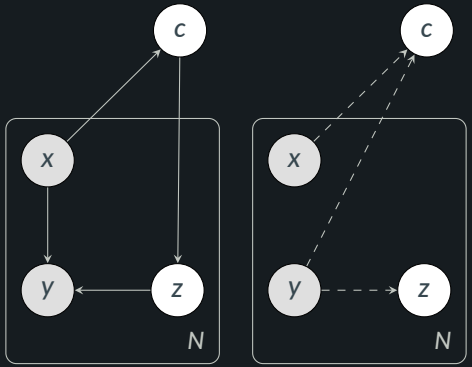
References II

- Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (Dec. 2015). “Deep Learning Face Attributes in the Wild”. In: *Proceedings of International Conference on Computer Vision (ICCV)*.
- 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*.
- Niculae, Vlad, André FT Martins, Mathieu Blondel, and Claire Cardie (2018). “SparseMAP: Differentiable sparse structured inference”. In: *Proc. of ICML*.
- Peters, Ben, Vlad Niculae, and André F. T. Martins (2019). “Sparse Sequence-to-Sequence Models”. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). “Attention Is All You Need”. In: *Proc. of NeurIPS*.
- Voita, Elena, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov (2019). “Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned”. In: *Proc. ACL*.

Using latent summaries in NMT



Multi-domain NMT



Document-level NMT