Sparse Sequence-to-Sequence Models

Ben Peters[†] **Vlad Niculae**[†] and **André F. T. Martins**[†‡]
[†]Instituto de Telecomunicações, Lisbon, Portugal
[‡]Unbabel, Lisbon, Portugal
benzurdopeters@gmail.com, vlad@vene.ro, andre.martins@unbabel.com

Abstract

Sequence-to-sequence models are a powerful workhorse of NLP. Most variants employ a softmax transformation in both their attention mechanism and output layer, leading to dense alignments and strictly positive output probabilities. This density is wasteful, making models less interpretable and assigning probability mass to many implausible outputs. In this paper, we propose *sparse* sequence-to-sequence models, rooted in a new family of α-entmax transformations, which includes softmax and sparsemax as particular cases, and is sparse for any $\alpha > 1$. We provide fast algorithms to evaluate these transformations and their gradients, which scale well for large vocabulary sizes. Our models are able to produce sparse alignments and to assign nonzero probability to a short list of plausible outputs, sometimes rendering beam search exact. Experiments on morphological inflection and machine translation reveal consistent gains over dense models.

1 Introduction

Attention-based sequence-to-sequence (*seq2seq*) models have proven useful for a variety of NLP applications, including machine translation (Bahdanau et al., 2015; Vaswani et al., 2017), speech recognition (Chorowski et al., 2015), abstractive summarization (Chopra et al., 2016), and morphological inflection generation (Kann and Schütze, 2016), among others. In part, their strength comes from their flexibility: many tasks can be formulated as transducing a source sequence into a target sequence of possibly different length.

However, conventional *seq2seq* models are **dense**: they compute both attention weights and output probabilities with the **softmax function** (Bridle, 1990), which always returns positive values. This results in *dense attention alignments*, in which each source position is attended to at each

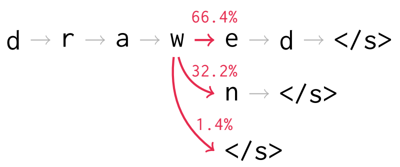

Figure 1: The full beam search of our best performing morphological inflection model when generating the past participle of the verb "draw". The model gives nonzero probability to exactly three hypotheses, including the correct form ("drawn") and the form that would be correct if "draw" were regular ("drawed").

target position, and in *dense output probabilities*, in which each vocabulary type always has nonzero probability of being generated. This contrasts with traditional statistical machine translation systems, which are based on sparse, hard alignments, and decode by navigating through a sparse lattice of phrase hypotheses. Can we transfer such notions of sparsity to modern neural architectures? And if so, do they improve performance?

In this paper, we provide an affirmative answer to both questions by proposing **neural sparse *seq2seq* models** that replace the softmax transformations (both in the attention and output) by **sparse transformations**. Our innovations are rooted in the recently proposed sparsemax transformation (Martins and Astudillo, 2016) and Fenchel-Young losses (Blondel et al., 2019). Concretely, we consider a family of transformations (dubbed **α-entmax**), parametrized by a scalar α, based on the Tsallis entropies (Tsallis, 1988). This family includes softmax ($\alpha = 1$) and sparsemax ($\alpha = 2$) as particular cases. Crucially, entmax transforms are sparse for all $\alpha > 1$.

Our models are able to produce both **sparse attention**, a form of inductive bias that increases focus on relevant source words and makes alignments more interpretable, and **sparse output probabilities**, which together with auto-regressive mod-

This	92.9%	**is another**	view	49.8%	**at**	95.7%	**the tree of life .**
So	5.9%		**look**	27.1%	on	5.9%	
And	1.3%		glimpse	19.9%	,	1.3%	
Here	<0.1%		kind	2.0%			
			looking	0.9%			
			way	0.2%			
			vision	<0.1%			
			gaze	<0.1%			

Figure 2: Forced decoding using sparsemax attention and 1.5-entmax output for the German source sentence, "Dies ist ein weiterer Blick auf den Baum des Lebens." Predictions with nonzero probability are shown at each time step. All other target types have probability exactly zero. When consecutive predictions consist of a single word, we combine their borders to showcase *auto-completion* potential. The selected gold targets are in boldface.

els can lead to probability distributions that are nonzero only for a finite subset of all possible strings. In certain cases, a short list of plausible outputs can be enumerated without ever exhausting the beam (Figure 1), rendering beam search **exact**. Sparse output *seq2seq* models can also be used for adaptive, sparse next word suggestion (Figure 2).

Overall, our contributions are as follows:

- We propose an **entmax sparse output layer**, together with a natural loss function. In large-vocabulary settings, sparse outputs avoid wasting probability mass on unlikely outputs, substantially improving accuracy. For tasks with little output ambiguity, entmax losses, coupled with beam search, can often produce **exact finite sets** with only one or a few sequences. To our knowledge, this is the first study of sparse output probabilities in *seq2seq* problems.

- We construct **entmax sparse attention**, improving interpretability at no cost in accuracy. We show that the entmax gradient has a simple form (Proposition 2), revealing an insightful missing link between softmax and sparsemax.

- We derive a novel exact algorithm for the case of 1.5-entmax, achieving processing speed close to softmax on the GPU, even with large vocabulary sizes. For arbitrary α, we investigate a GPU-friendly approximate algorithm.[1]

We experiment on two tasks: one character-level with little ambiguity (**morphological inflection generation**) and another word-level, with more ambiguity (**neural machine translation**). The results show clear benefits of our approach, both in terms of accuracy and interpretability.

2 Background

The underlying architecture we focus on is an RNN-based *seq2seq* with global attention and input-feeding (Luong et al., 2015). We provide a brief description of this architecture, with an emphasis on the attention mapping and the loss function.

Notation. Scalars, vectors, and matrices are denoted respectively as a, $\boldsymbol{a}$, and $\boldsymbol{A}$. We denote the d–probability simplex (the set of vectors representing probability distributions over d choices) by $\triangle^d := \{\boldsymbol{p} \in \mathbb{R}^d \colon \boldsymbol{p} \geq 0, \|\boldsymbol{p}\|_1 = 1\}$. We denote the positive part as $[a]_+ := \max\{a, 0\}$, and by $[\boldsymbol{a}]_+$ its elementwise application to vectors. We denote the indicator vector $\boldsymbol{e}_y := [0, \dots, 0, \underbrace{1}_{y}, 0, \dots, 0]$.

Encoder. Given an input sequence of tokens $\boldsymbol{x} := [x_1, \dots, x_J]$, the encoder applies an embedding lookup followed by K layered bidirectional LSTMs (Hochreiter and Schmidhuber, 1997), resulting in encoder states $[\boldsymbol{h}_1, \dots, \boldsymbol{h}_J]$.

Decoder. The decoder generates output tokens $y_1, \dots, y_T$, one at a time, terminated by a stop symbol. At each time step t, it computes a probability distribution for the next generated word y_t, as follows. Given the current state $\boldsymbol{s}_t$ of the decoder LSTM, an **attention mechanism** (Bahdanau et al., 2015) computes a focused, fixed-size summary of the encodings $[\boldsymbol{h}_1, \dots, \boldsymbol{h}_J]$, using $\boldsymbol{s}_t$ as a query vector. This is done by computing token-level scores $z_j := \boldsymbol{s}_t^\top \boldsymbol{W}^{(z)} \boldsymbol{h}_j$, then taking a weighted average

$$\boldsymbol{c}_t := \sum_{j=1}^{J} \pi_j \boldsymbol{h}_j, \text{ where } \boldsymbol{\pi} := \mathsf{softmax}(\boldsymbol{z}). \quad (1)$$

[1]Our standalone Pytorch entmax implementation is available at `https://github.com/deep-spin/entmax`.

The contextual output is the non-linear combination $\boldsymbol{o}_t := \mathsf{tanh}(\boldsymbol{W}^{(o)}[\boldsymbol{s}_t;\boldsymbol{c}_t] + \boldsymbol{b}^{(o)})$, yielding the predictive distribution of the next word

$$p(y_t = \cdot \mid \boldsymbol{x}, y_1, ..., y_{t-1}) := \mathsf{softmax}(\boldsymbol{V}\boldsymbol{o}_t + \boldsymbol{b}). \tag{2}$$

The output $\boldsymbol{o}_t$, together with the embedding of the predicted $\widehat{y}_t$, feed into the decoder LSTM for the next step, in an auto-regressive manner. The model is trained to maximize the likelihood of the correct target sentences, or equivalently, to minimize

$$\mathcal{L} = \sum_{(x,y)\in\mathcal{D}} \sum_{t=1}^{|y|} \underbrace{(-\log \mathsf{softmax}(\boldsymbol{V}\boldsymbol{o}_t))_{y_t}}_{\mathsf{L}_{\mathsf{softmax}}(y_t, \boldsymbol{V}\boldsymbol{o}_t)}. \tag{3}$$

A central building block in the architecture is the transformation $\mathsf{softmax}\colon \mathbb{R}^d \to \triangle^d$,

$$\mathsf{softmax}(\boldsymbol{z})_j := \frac{\mathsf{exp}(z_j)}{\sum_i \mathsf{exp}(z_i)}, \tag{4}$$

which maps a vector of scores $\boldsymbol{z}$ into a probability distribution (*i.e.*, a vector in $\triangle^d$). As seen above, the softmax mapping plays **two crucial roles** in the decoder: first, in computing normalized attention weights (Eq. 1), second, in computing the predictive probability distribution (Eq. 2). Since $\exp \gneqq 0$, softmax never assigns a probability of zero to any word, so we may never fully rule out non-important input tokens from attention, nor unlikely words from the generation vocabulary. While this may be advantageous for dealing with uncertainty, it may be preferrable to avoid dedicating model resources to irrelevant words. In the next section, we present a strategy for **differentiable sparse probability mappings**. We show that our approach can be used to learn powerful *seq2seq* models with sparse outputs and sparse attention mechanisms.

3 Sparse Attention and Outputs

3.1 The sparsemax mapping and loss

To pave the way to a more general family of sparse attention and losses, we point out that softmax (Eq. 4) is only one of many possible mappings from $\mathbb{R}^d$ to $\triangle^d$. Martins and Astudillo (2016) introduce **sparsemax**: an alternative to softmax which tends to yield **sparse probability distributions**:

$$\mathsf{sparsemax}(\boldsymbol{z}) := \underset{\boldsymbol{p}\in\triangle^d}{\mathsf{argmin}} \|\boldsymbol{p} - \boldsymbol{z}\|^2. \tag{5}$$

Since Eq. 5 is a projection onto $\triangle^d$, which tends to yield sparse solutions, the predictive distribution $\boldsymbol{p}^\star := \mathsf{sparsemax}(\boldsymbol{z})$ is likely to assign **exactly zero** probability to low-scoring choices. They also propose a corresponding **loss function** to replace the negative log likelihood loss $\mathsf{L}_{\mathsf{softmax}}$ (Eq. 3):

$$\mathsf{L}_{\mathsf{sparsemax}}(y, \boldsymbol{z}) := \frac{1}{2}\left(\|\boldsymbol{e}_y - \boldsymbol{z}\|^2 - \|\boldsymbol{p}^\star - \boldsymbol{z}\|^2\right), \tag{6}$$

This loss is smooth and convex on $\boldsymbol{z}$ and has a **margin**: it is zero if and only if $\boldsymbol{z}_y \geq \boldsymbol{z}_{y'} + 1$ for any $y' \neq y$ (Martins and Astudillo, 2016, Proposition 3). Training models with the sparsemax loss requires its gradient (*cf.* Appendix A.2):

$$\nabla_{\boldsymbol{z}} \mathsf{L}_{\mathsf{sparsemax}}(y, \boldsymbol{z}) = -\boldsymbol{e}_y + \boldsymbol{p}^\star.$$

For using the sparsemax *mapping* in an attention mechanism, Martins and Astudillo (2016) show that it is differentiable almost everywhere, with

$$\frac{\partial\, \mathsf{sparsemax}(\boldsymbol{z})}{\partial \boldsymbol{z}} = \mathsf{diag}(\boldsymbol{s}) - \frac{1}{\|\boldsymbol{s}\|_1}\boldsymbol{s}\boldsymbol{s}^\top,$$

where $s_j = 1$ if $p^\star_j > 0$, otherwise $s_j = 0$.

Entropy interpretation. At first glance, sparsemax appears very different from softmax, and a strategy for producing other sparse probability mappings is not obvious. However, the connection becomes clear when considering the variational form of softmax (Wainwright and Jordan, 2008):

$$\mathsf{softmax}(\boldsymbol{z}) = \underset{\boldsymbol{p}\in\triangle^d}{\mathsf{argmax}}\, \boldsymbol{p}^\top\boldsymbol{z} + \mathsf{H}^{\mathsf{S}}(\boldsymbol{p}), \tag{7}$$

where $\mathsf{H}^{\mathsf{S}}(\boldsymbol{p}) := -\sum_j p_j \log p_j$ is the well-known Gibbs-Boltzmann-Shannon entropy with base e.

Likewise, letting $\mathsf{H}^{\mathsf{G}}(\boldsymbol{p}) := \frac{1}{2}\sum_j p_j(1 - p_j)$ be the **Gini entropy**, we can rearrange Eq. 5 as

$$\mathsf{sparsemax}(\boldsymbol{z}) = \underset{\boldsymbol{p}\in\triangle^d}{\mathsf{argmax}}\, \boldsymbol{p}^\top\boldsymbol{z} + \mathsf{H}^{\mathsf{G}}(\boldsymbol{p}), \tag{8}$$

crystallizing the connection between softmax and sparsemax: they only differ in the choice of **entropic regularizer**.

3.2 A new entmax mapping and loss family

The parallel above raises a question: can we find **interesting interpolations between softmax and sparsemax?** We answer affirmatively, by considering a generalization of the Shannon and Gini entropies proposed by Tsallis (1988): a family of entropies parametrized by a scalar $\alpha > 1$ which we call **Tsallis α-entropies**:

$$\mathsf{H}^{\mathsf{T}}_\alpha(\boldsymbol{p}) := \begin{cases} \frac{1}{\alpha(\alpha-1)} \sum_j \left(p_j - p_j^\alpha\right), & \alpha \neq 1, \\ \mathsf{H}^{\mathsf{S}}(\boldsymbol{p}), & \alpha = 1. \end{cases} \tag{9}$$

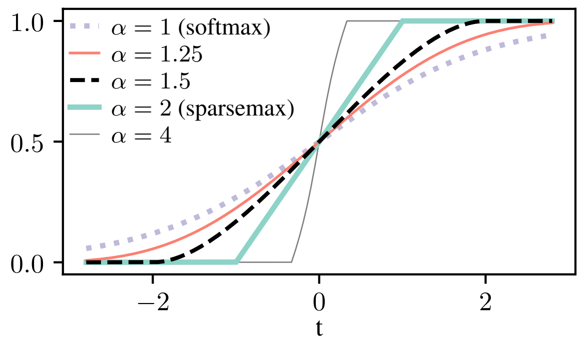

Figure 3: Illustration of entmax in the two-dimensional case $\alpha\text{-entmax}([t, 0])_1$. All mappings except softmax saturate at $t = \pm {}^1/_{\alpha-1}$. While sparsemax is piecewise linear, mappings with $1 < \alpha < 2$ have smooth corners.

This family is continuous, *i.e.*, $\lim_{\alpha\to1} \mathsf{H}^{\mathsf{T}}_\alpha(\boldsymbol{p}) = \mathsf{H}^{\mathsf{S}}(\boldsymbol{p})$ for any $\boldsymbol{p} \in \triangle^d$ (*cf.* Appendix A.1). Moreover, $\mathsf{H}^{\mathsf{T}}_2 \equiv \mathsf{H}^{\mathsf{G}}$. Thus, Tsallis entropies interpolate between the Shannon and Gini entropies. Starting from the Tsallis entropies, we construct a probability mapping, which we dub **entmax**:

$$\alpha\text{-entmax}(\boldsymbol{z}) := \underset{\boldsymbol{p}\in\triangle^d}{\operatorname{argmax}}\, \boldsymbol{p}^\top \boldsymbol{z} + \mathsf{H}^{\mathsf{T}}_\alpha(\boldsymbol{p}), \tag{10}$$

and, denoting $\boldsymbol{p}^\star := \alpha\text{-entmax}(\boldsymbol{z})$, a loss function

$$\mathsf{L}_\alpha(y, \boldsymbol{z}) := (\boldsymbol{p}^\star - \boldsymbol{e}_y)^\top \boldsymbol{z} + \mathsf{H}^{\mathsf{T}}_\alpha(\boldsymbol{p}^\star) \tag{11}$$

The motivation for this loss function resides in the fact that it is a **Fenchel-Young loss** (Blondel et al., 2019), as we briefly explain in Appendix A.2. Then, 1-entmax $\equiv$ softmax and 2-entmax $\equiv$ sparsemax. Similarly, L_1 is the negative log likelihood, and L_2 is the sparsemax loss. For all $\alpha > 1$, entmax tends to produce **sparse probability distributions**, yielding a function family continuously interpolating between softmax and sparsemax, *cf.* Figure 3. The gradient of the entmax loss is

$$\nabla_{\boldsymbol{z}} \mathsf{L}_\alpha(y, \boldsymbol{z}) = -\boldsymbol{e}_y + \boldsymbol{p}^\star. \tag{12}$$

Tsallis entmax losses have useful properties including convexity, differentiability, and a hinge-like separation margin property: the loss incurred becomes zero when the score of the correct class is separated by the rest by a margin of ${}^1/_{\alpha-1}$. When separation is achieved, $\boldsymbol{p}^\star = \boldsymbol{e}_y$ (Blondel et al., 2019). This allows entmax *seq2seq* models to be **adaptive to the degree of uncertainty** present: decoders may make fully confident predictions at "easy" time steps, while preserving **sparse uncertainty** when a few choices are possible (as exemplified in Figure 2).

Tsallis entmax probability mappings have not, to our knowledge, been used in attention mechanisms. They inherit the desirable sparsity of sparsemax, while exhibiting smoother, differentiable curvature, whereas sparsemax is piecewise linear.

3.3 Computing the entmax mapping

Whether we want to use α-entmax as an attention mapping, or L_α as a loss function, we must be able to efficiently compute $\boldsymbol{p}^\star = \alpha\text{-entmax}(\boldsymbol{z})$, *i.e.*, to solve the maximization in Eq. 10. For $\alpha = 1$, the closed-form solution is given by Eq. 4. For $\alpha > 1$, given $\boldsymbol{z}$, we show that there is a unique threshold τ such that (Appendix C.1, Lemma 2):

$$\alpha\text{-entmax}(\boldsymbol{z}) = [(\alpha-1)\boldsymbol{z} - \tau\mathbf{1}]_+^{1/\alpha-1}, \tag{13}$$

i.e., entries with score $z_j \le {}^\tau/_{\alpha-1}$ get zero probability. For sparsemax ($\alpha = 2$), the problem amounts to Euclidean projection onto $\triangle^d$, for which two types of algorithms are well studied:

i. exact, based on sorting (Held et al., 1974; Michelot, 1986),

ii. iterative, bisection-based (Liu and Ye, 2009).

The bisection approach searches for the optimal threshold τ numerically. Blondel et al. (2019) generalize this approach in a way applicable to α-entmax. The resulting algorithm is (*cf.* Appendix C.1 for details):

Algorithm 1 Compute α-entmax by bisection.

```
1 Define $\boldsymbol{p}(\tau) := [\boldsymbol{z} - \tau]_+^{1/\alpha-1}$, set $\boldsymbol{z} \leftarrow (\alpha-1)\boldsymbol{z}$
2 $\tau_{\min} \leftarrow \max(\boldsymbol{z}) - 1$; $\tau_{\max} \leftarrow \max(\boldsymbol{z}) - d^{1-\alpha}$
3 for $t \in 1, \ldots, T$ do
4     $\tau \leftarrow (\tau_{\min} + \tau_{\max})/2$
5     $Z \leftarrow \sum_j p_j(\tau)$
6     if $Z < 1$ then $\tau_{\max} \leftarrow \tau$ else $\tau_{\min} \leftarrow \tau$
7 return ${}^1/_Z\, \boldsymbol{p}(\tau)$
```

Algorithm 1 works by iteratively narrowing the interval containing the exact solution by exactly half. Line 7 ensures that approximate solutions are valid probability distributions, *i.e.*, that $\boldsymbol{p}^\star \in \triangle^d$.

Although bisection is simple and effective, an exact sorting-based algorithm, like for sparsemax, has the potential to be faster and more accurate. Moreover, as pointed out by Condat (2016), when exact solutions are required, it is possible to construct inputs $\boldsymbol{z}$ for which bisection requires arbitrarily many iterations. To address these issues, we propose a

novel, exact algorithm for 1.5-entmax, halfway between softmax and sparsemax.

Algorithm 2 Compute 1.5-entmax($\boldsymbol{z}$) exactly.

```
1 Sort $\boldsymbol{z}$, yielding $z_{[d]} \leq \cdots \leq z_{[1]}$; set $\boldsymbol{z} \leftarrow z/2$
2 for $\rho \in 1, \ldots, d$ do
3     $M(\rho) \leftarrow {}^1/_\rho \sum_{j=1}^{\rho} z_{[j]}$
4     $S(\rho) \leftarrow \sum_{j=1}^{\rho} \left(z_{[j]} - M(\rho)\right)^2$
5     $\tau(\rho) \leftarrow M(\rho) - \sqrt{{}^1/_\rho\,(1 - S(\rho))}$
6     if $z_{[\rho+1]} \leq \tau(\rho) \leq z_{[\rho]}$ then
7         return $\boldsymbol{p}^\star = [\boldsymbol{z} - \tau\,\mathbf{1}]_+^2$
```

We give a full derivation in Appendix C.2. As written, Algorithm 2 is $O(d \log d)$ because of the sort; however, in practice, when the solution $\boldsymbol{p}^\star$ has no more than k nonzeros, we do not need to fully sort $\boldsymbol{z}$, just to find the k largest values. Our experiments in §4.2 reveal that a partial sorting approach can be very efficient and competitive with softmax on the GPU, even for large d. Further speed-ups might be available following the strategy of Condat (2016), but our simple incremental method is very easy to implement on the GPU using primitives available in popular libraries (Paszke et al., 2017).

Our algorithm resembles the aforementioned sorting-based algorithm for projecting onto the simplex (Michelot, 1986). Both algorithms rely on the optimality conditions implying an analytically-solvable equation in τ: for sparsemax ($\alpha = 2$), this equation is linear, for $\alpha = 1.5$ it is quadratic (Eq. 48 in Appendix C.2). Thus, exact algorithms may not be available for general values of α.

3.4 Gradient of the entmax mapping

The following result shows how to compute the backward pass through α-entmax, a requirement when using α-entmax as an attention mechanism.

Proposition 1. *Let $\alpha \geq 1$. Assume we have computed $\boldsymbol{p}^\star = \alpha$-entmax$(\boldsymbol{z})$, and define the vector*

$$s_i = \begin{cases} (p_i^\star)^{2-\alpha}, & p_i^\star > 0, \\ 0, & \textit{otherwise}. \end{cases}$$

Then, $$\frac{\partial\, \alpha\text{-entmax}(\boldsymbol{z})}{\partial \boldsymbol{z}} = \operatorname{diag}(\boldsymbol{s}) - \frac{1}{\|\boldsymbol{s}\|_1} \boldsymbol{s}\boldsymbol{s}^\top.$$

Proof: The result follows directly from the more general Proposition 2, which we state and prove in Appendix B, noting that $\left(\frac{t^\alpha - t}{\alpha(\alpha-1)}\right)'' = t^{\alpha-2}$. ■

The gradient expression recovers the softmax and sparsemax Jacobians with $\alpha = 1$ and $\alpha = 2$, respectively (Martins and Astudillo, 2016, Eqs. 8 and 12), thereby providing another relationship between the two mappings. Perhaps more interestingly, Proposition 1 shows why **the sparsemax Jacobian depends only on the support** and not on the actual values of $\boldsymbol{p}^\star$: the sparsemax Jacobian is equal for $\boldsymbol{p}^\star = [.99, .01, 0]$ and $\boldsymbol{p}^\star = [.5, .5, 0]$. This is not the case for α-entmax with $\alpha \neq 2$, suggesting that the gradients obtained with other values of α may be more informative. Finally, we point out that the gradient of entmax *losses* involves the entmax *mapping* (Eq. 12), and therefore Proposition 1 also gives the *Hessian* of the entmax loss.

4 Experiments

The previous section establishes the computational building blocks required to train models with entmax sparse attention and loss functions. We now put them to use for two important NLP tasks, morphological inflection and machine translation. These two tasks highlight the characteristics of our innovations in different ways. Morphological inflection is a **character-level** task with mostly monotonic alignments, but the evaluation demands exactness: the predicted sequence must match the gold standard. On the other hand, machine translation uses a **word-level** vocabulary orders of magnitude larger and forces a sparse output layer to confront more ambiguity: any sentence has several valid translations and it is not clear beforehand that entmax will manage this well.

Despite the differences between the tasks, we keep the architecture and training procedure as similar as possible. We use two layers for encoder and decoder LSTMs and apply dropout with probability 0.3. We train with Adam (Kingma and Ba, 2015), with a base learning rate of 0.001, halved whenever the loss increases on the validation set. We use a batch size of 64. At test time, we select the model with the best validation accuracy and decode with a beam size of 5. We implemented all models with OpenNMT-py (Klein et al., 2017).[2]

In our primary experiments, we use three α values for the attention and loss functions: $\alpha = 1$ (softmax), $\alpha = 1.5$ (to which our novel Algorithm 2 applies), and $\alpha = 2$ (sparsemax). We also investigate the effect of tuning α with increased granularity.

[2]Our experiment code is at `https://github.com/deep-spin/OpenNMT-entmax`.

α output	α attention	high (avg.)	high (ens.)	medium (avg.)	medium (ens.)
1	1	93.15	94.20	82.55	85.68
	1.5	92.32	93.50	83.20	85.63
	2	90.98	92.60	83.13	85.65
1.5	1	94.36	94.96	84.88	86.38
	1.5	94.44	95.00	84.93	86.55
	2	94.05	94.74	84.93	86.59
2	1	**94.59**	**95.10**	84.95	86.41
	1.5	94.47	95.01	**85.03**	**86.61**
	2	94.32	94.89	84.96	86.47
UZH (2018)			96.00		86.64

Table 1: Average per-language accuracy on the test set (CoNLL–SIGMORPHON 2018 task 1) averaged or ensembled over three runs.

4.1 Morphological Inflection

The goal of morphological inflection is to produce an inflected word form (such as "drawn") given a lemma ("draw") and a set of morphological tags ({verb, past, participle}). We use the data from task 1 of the CoNLL–SIGMORPHON 2018 shared task (Cotterell et al., 2018). shared task

Training. We train models under two data settings: *high* (approximately 10,000 samples per language in 86 languages) and *medium* (approximately 1,000 training samples per language in 102 languages). We depart from previous work by using **multilingual training**: each model is trained on the data from all languages in its data setting. This allows parameters to be shared between languages, eliminates the need to train language-specific models, and may provide benefits similar to other forms of data augmentation (Bergmanis et al., 2017). Each sample is presented as a pair: the source contains the lemma concatenated to the morphological tags and a special language identification token (Johnson et al., 2017; Peters et al., 2017), and the target contains the inflected form. As an example, the source sequence for Figure 1 is `english␣verb␣participle␣past␣d␣r␣a␣w`. Although the set of inflectional tags is not sequential, treating it as such is simple to implement and works well in practice (Kann and Schütze, 2016). All models use embedding and hidden state sizes of 300. We validate at the end of every epoch in the *high* setting and only once every ten epochs in *medium* because of its smaller size.

Accuracy. Results are shown in Table 1. We report the official metric of the shared task, word accuracy averaged across languages. In addition to the average results of three individual model runs, we use an ensemble of those models, where we decode by averaging the raw probabilities at each time step. Our best sparse loss models beat the softmax baseline by nearly a full percentage point with ensembling, and up to two and a half points in the medium setting without ensembling. The choice of attention has a smaller impact. In both data settings, our best model on the validation set outperforms all submissions from the 2018 shared task except for UZH (Makarov and Clematide, 2018), which uses a more involved imitation learning approach and larger ensembles. In contrast, our only departure from standard *seq2seq* training is the drop-in replacement of softmax by entmax.

Sparsity. Besides their accuracy, we observed that entmax models made very sparse predictions: the best configuration in Table 1 concentrates all probability mass into a single predicted sequence in 81% validation samples in the *high* data setting, and 66% in the more difficult *medium* setting. When the model *does* give probability mass to more than one sequence, the predictions reflect reasonable ambiguity, as shown in Figure 1. Besides enhancing interpretability, sparsity in the output also has attractive properties for beam search decoding: when the beam covers all nonzero-probability hypotheses, we have a **certificate** of globally optimal decoding, **rendering beam search exact**. This is the case on 87% of validation set sequences in the *high* setting, and 79% in *medium*. To our knowledge, this is the first instance of a neural *seq2seq* model that can offer optimality guarantees.

4.2 Machine Translation

We now turn to a highly different *seq2seq* regime in which the vocabulary size is much larger, there is a great deal of ambiguity, and sequences can generally be translated in several ways. We train models for three language pairs in both directions:

- IWSLT 2017 German ↔ English (DE↔EN, Cettolo et al., 2017): training size 206,112.
- KFTT Japanese ↔ English (JA↔EN, Neubig, 2011): training size of 329,882.
- WMT 2016 Romanian ↔ English (RO↔EN, Bojar et al., 2016): training size 612,422, diacritics removed (following Sennrich et al., 2016b).

method	DE→EN	EN→DE	JA→EN	EN→JA	RO→EN	EN→RO
softmax	25.70 ± 0.15	21.86 ± 0.09	20.22 ± 0.08	25.21 ± 0.29	29.12 ± 0.18	28.12 ± 0.18
1.5-entmax	**26.17** ± 0.13	**22.42** ± 0.08	**20.55** ± 0.30	**26.00** ± 0.31	**30.15** ± 0.06	**28.84** ± 0.10
sparsemax	24.69 ± 0.22	20.82 ± 0.19	18.54 ± 0.11	23.84 ± 0.37	29.20 ± 0.16	28.03 ± 0.16

Table 2: Machine translation comparison of softmax, sparsemax, and the proposed 1.5-entmax as both attention mapping and loss function. Reported is tokenized test BLEU averaged across three runs (higher is better).

Training. We use byte pair encoding (BPE; Sennrich et al., 2016a) to ensure an open vocabulary. We use separate segmentations with 25k merge operations per language for RO↔EN and a joint segmentation with 32k merges for the other language pairs. DE↔EN is validated once every 5k steps because of its smaller size, while the other sets are validated once every 10k steps. We set the maximum number of training steps at 120k for RO↔EN and 100k for other language pairs. We use 500 dimensions for word vectors and hidden states.

Evaluation. Table 2 shows BLEU scores (Papineni et al., 2002) for the three models with $\alpha \in \{1, 1.5, 2\}$, using the same value of α for the attention mechanism and loss function. We observe that the 1.5-entmax configuration consistently performs best across all six choices of language pair and direction. These results support the notion that the optimal function is somewhere between softmax and sparsemax, which motivates a more fine-grained search for α; we explore this next.

Fine-grained impact of α. Algorithm 1 allows us to further investigate the marginal effect of varying the attention α and the loss α, while keeping the other fixed. We report DE→EN validation accuracy on a fine-grained α grid in Figure 4. On this dataset, moving from softmax toward sparser **attention** (left) has a very small positive effect on accuracy, suggesting that the benefit in interpretability does not hurt accuracy. The impact of the **loss function** α (right) is much more visible: there is a distinct optimal value around $\alpha = 1.33$, with performance decreasing for too large values. Interpolating between softmax and sparsemax thus inherits the benefits of both, and our novel Algorithm 2 for $\alpha = 1.5$ is confirmed to strike a good middle ground. This experiment also confirms that bisection is effective in practice, despite being inexact. Extrapolating beyond the sparsemax loss ($\alpha > 2$) does not seem to perform well.

Sparsity. In order to form a clearer idea of how sparse entmax becomes, we measure the average number of nonzero indices on the DE→EN validation set and show it in Table 3. As expected, 1.5-entmax is less sparse than sparsemax as both an attention mechanism and output layer. In the attention mechanism, 1.5-entmax's increased support size does not come at the cost of much interpretability, as Figure 5 demonstrates. In the output layer, 1.5-entmax assigns positive probability to only 16.13 target types out of a vocabulary of 17,993, meaning that the supported set of words often has an intuitive interpretation. Figure 2 shows the sparsity of the 1.5-entmax output layer in practice: the support becomes completely concentrated when generating a phrase like "the tree of life", but grows when presenting a list of synonyms ("view", "look", "glimpse", and so on). This has potential practical applications as a **predictive translation** system (Green et al., 2014), where the model's support set serves as a list of candidate auto-completions at each time step.

method	# attended	# target words
softmax	24.25	17993
1.5-entmax	5.55	16.13
sparsemax	3.75	7.55

Table 3: Average number of nonzeros in the attention and output distributions for the DE→EN validation set.

Training time. Importantly, the benefits of sparsity do **not** come at a high computational cost. Our proposed Algorithm 2 for 1.5-entmax runs on the GPU at near-softmax speeds (Figure 6). For other α values, bisection (Algorithm 1) is slightly more costly, but practical even for large vocabulary sizes. On DE→EN, bisection is capable of processing about 10,500 target words per second on a single Nvidia GeForce GTX 1080 GPU, compared to 13,000 words per second for 1.5-entmax with Algorithm 2 and 14,500 words per second with softmax. On the smaller-vocabulary morphology datasets, Algorithm 2 is nearly as fast as softmax.

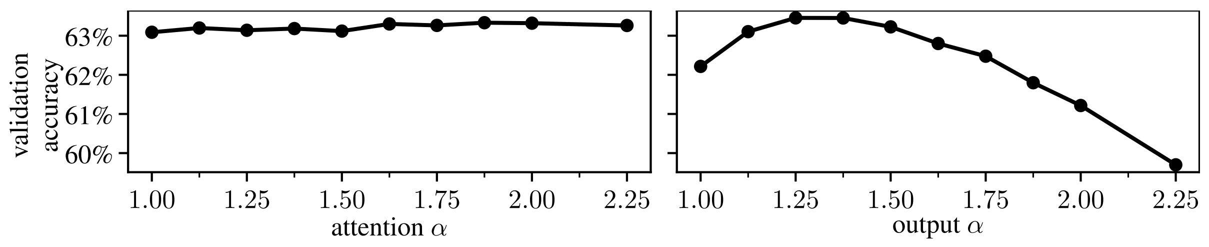

Figure 4: Effect of tuning α on DE→EN, for attention (left) and for output (right), while keeping the other $\alpha = 1.5$.

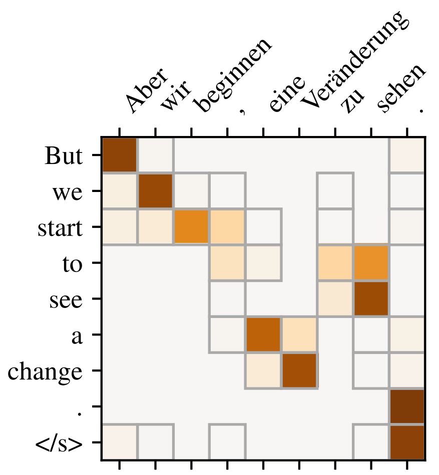

Figure 5: Attention weights produced by the DE→EN 1.5-entmax model. Nonzero weights are outlined.

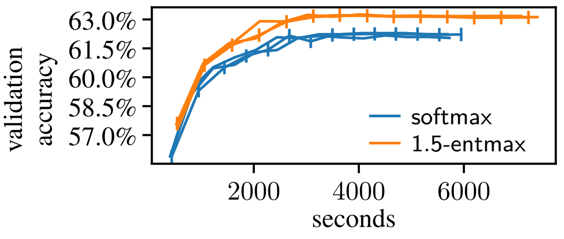

Figure 6: Training timing on three DE→EN runs. Markers show validation checkpoints for one of the runs.

5 Related Work

Sparse attention. Sparsity in the attention and in the output have different, but related, motivations. Sparse attention can be justified as a form of inductive bias, since for tasks such as machine translation one expects only a few source words to be relevant for each translated word. Dense attention probabilities are particularly harmful for long sequences, as shown by Luong et al. (2015), who propose "local attention" to mitigate this problem. Combining sparse attention with fertility constraints has been recently proposed by Malaviya et al. (2018). Hard attention (Xu et al., 2015; Aharoni and Goldberg, 2017; Wu et al., 2018) selects exactly one source token. Its discrete, non-differentiable nature requires imitation learning or Monte Carlo policy gradient approximations, which drastically complicate training. In contrast, entmax is a differentiable, easy to use, drop-in softmax replacement. A recent study by Jain and Wallace (2019) tackles the limitations of attention probabilities to provide interpretability. They only study dense attention in classification tasks, where attention is less crucial for the final predictions. In their conclusions, the authors defer to future work exploring sparse attention mechanisms and *seq2seq* models. We believe our paper can foster interesting investigation in this area.

Losses for *seq2seq* models. Mostly motivated by the challenges of large vocabulary sizes in *seq2seq*, an important research direction tackles replacing the cross-entropy loss with other losses or approximations (Bengio and Senécal, 2008; Morin and Bengio, 2005; Kumar and Tsvetkov, 2019). While differently motivated, some of the above strategies (*e.g.*, hierarchical prediction) could be combined with our proposed sparse losses. Niculae et al. (2018) use sparsity to predict interpretable sets of structures. Since auto-regressive *seq2seq* makes no factorization assumptions, their strategy cannot be applied without approximations, such as in Edunov et al. (2018).

6 Conclusion and Future Work

We proposed sparse sequence-to-sequence models and provided fast algorithms to compute their attention and output transformations. Our approach yielded consistent improvements over dense models on morphological inflection and machine translation, while inducing interpretability in both attention and output distributions. Sparse output layers also provide exactness when the number of possible hypotheses does not exhaust beam search.

Given the ubiquity of softmax in NLP, entmax has many potential applications. A natural next step is to apply entmax to self-attention (Vaswani et al., 2017). In a different vein, the strong morphological inflection results point to usefulness in other tasks where probability is concentrated in a small number of hypotheses, such as speech recognition.

Acknowledgments

This work was supported by the European Research Council (ERC StG DeepSPIN 758969), and by the Fundação para a Ciência e Tecnologia through contracts UID/EEA/50008/2019 and CMUPERI/TIC/0046/2014 (GoLocal). We thank Mathieu Blondel, Nikolay Bogoychev, Gonçalo Correia, Erick Fonseca, Pedro Martins, Tsvetomila Mihaylova, Miguel Rios, Marcos Treviso, and the anonymous reviewers, for helpful discussion and feedback.

References

Roee Aharoni and Yoav Goldberg. 2017. Morphological inflection generation with hard monotonic attention. In *Proc. ACL*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. ICLR*.

Yoshua Bengio and Jean-Sébastien Senécal. 2008. Adaptive importance sampling to accelerate training of a neural probabilistic language model. *IEEE Transactions on Neural Networks*, 19(4):713–722.

Toms Bergmanis, Katharina Kann, Hinrich Schütze, and Sharon Goldwater. 2017. Training data augmentation for low-resource morphological inflection. In *Proc. CoNLL–SIGMORPHON*.

Mathieu Blondel, André FT Martins, and Vlad Niculae. 2019. Learning classifiers with Fenchel-Young losses: Generalized entropies, margins, and algorithms. In *Proc. AISTATS*.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *ACL WMT*.

John S Bridle. 1990. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer.

M Cettolo, M Federico, L Bentivogli, J Niehues, S Stüker, K Sudoh, K Yoshino, and C Federmann. 2017. Overview of the IWSLT 2017 evaluation campaign. In *Proc. IWSLT*.

Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proc. NAACL-HLT*.

Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Proc. NeurIPS*.

Laurent Condat. 2016. Fast projection onto the simplex and the ℓ_1 ball. *Mathematical Programming*, 158(1-2):575–585.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, Gėraldine Walther, Ekaterina Vylomova, Arya D McCarthy, Katharina Kann, Sebastian Mielke, Garrett Nicolai, Miikka Silfverberg, et al. 2018. The CoNLL–SIGMORPHON 2018 shared task: Universal morphological reinflection. *Proc. CoNLL–SIGMORPHON*.

John M Danskin. 1966. The theory of max-min, with applications. *SIAM Journal on Applied Mathematics*, 14(4):641–664.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, et al. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proc. NAACL-HLT*.

Spence Green, Sida I Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proc. EMNLP*.

Michael Held, Philip Wolfe, and Harlan P Crowder. 1974. Validation of subgradient optimization. *Mathematical Programming*, 6(1):62–88.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. In *Proc. NAACL-HLT*.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU system for the SIGMORPHON 2016 shared task on morphological reinflection. In *Proc. SIGMORPHON*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proc. ICLR*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. *arXiv e-prints*.

Sachin Kumar and Yulia Tsvetkov. 2019. Von Mises-Fisher loss for training sequence to sequence models with continuous outputs. In *Proc. ICLR*.

Jun Liu and Jieping Ye. 2009. Efficient Euclidean projections in linear time. In *Proc. ICML*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proc. EMNLP*.

Peter Makarov and Simon Clematide. 2018. UZH at CoNLL–SIGMORPHON 2018 shared task on universal morphological reinflection. *Proc. CoNLL–SIGMORPHON*.

Chaitanya Malaviya, Pedro Ferreira, and André FT Martins. 2018. Sparse and constrained attention for neural machine translation. In *Proc. ACL*.

André FT Martins and Ramón Fernandez Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proc. of ICML*.

Christian Michelot. 1986. A finite algorithm for finding the projection of a point onto the canonical simplex of $\mathbb{R}^n$. *Journal of Optimization Theory and Applications*, 50(1):195–200.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In *Proc. AISTATS*.

Graham Neubig. 2011. The Kyoto free translation task. `http://www.phontron.com/kftt`.

Vlad Niculae and Mathieu Blondel. 2017. A regularized framework for sparse and structured neural attention. In *Proc. NeurIPS*.

Vlad Niculae, André FT Martins, Mathieu Blondel, and Claire Cardie. 2018. SparseMAP: Differentiable sparse structured inference. In *Proc. ICML*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *Proc. NeurIPS Autodiff Workshop*.

Ben Peters, Jon Dehdari, and Josef van Genabith. 2017. Massively multilingual neural grapheme-to-phoneme conversion. In *Proc. Workshop on Building Linguistically Generalizable NLP Systems*.

R Tyrrell Rockafellar. 1970. *Convex Analysis*. Princeton University Press.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Neural machine translation of rare words with subword units. In *Proc. ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Edinburgh neural machine translation systems for WMT 16. In *Proc. WMT*.

Constantino Tsallis. 1988. Possible generalization of Boltzmann-Gibbs statistics. *Journal of Statistical Physics*, 52:479–487.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. NeurIPS*.

Martin J Wainwright and Michael I Jordan. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305.

Shijie Wu, Pamela Shapiro, and Ryan Cotterell. 2018. Hard non-monotonic attention for character-level transduction. In *Proc. EMNLP*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *Proc. ICML*.

A Background

A.1 Tsallis entropies

Recall the definition of the Tsallis family of entropies in Eq. 9 for $\alpha \geq 1$,

$$\mathsf{H}^{\mathsf{T}}_{\alpha}(\boldsymbol{p}) := \begin{cases} \frac{1}{\alpha(\alpha-1)} \sum_j \Big(p_j - p_j^{\alpha}\Big), & \text{if } \alpha \neq 1, \\ \mathsf{H}^{\mathsf{S}}(\boldsymbol{p}), & \text{if } \alpha = 1. \end{cases} \tag{14}$$

This family is **continuous** in α, *i.e.*,

$$\lim_{\alpha \to 1} \mathsf{H}^{\mathsf{T}}_{\alpha}(\boldsymbol{p}) = \mathsf{H}^{\mathsf{T}}_{1}(\boldsymbol{p})$$

for any $\boldsymbol{p} \in \triangle^d$.

Proof: We rewrite $\mathsf{H}^{\mathsf{T}}_{\alpha}$ in separable form:

$$\mathsf{H}^{\mathsf{T}}_{\alpha}(\boldsymbol{p}) = \sum_j h_{\alpha}(p_j), \quad \text{with}$$
$$h_{\alpha}(t) := \begin{cases} \frac{t - t^{\alpha}}{\alpha(\alpha-1)}, & \alpha > 1, \\ -t \log t, & \alpha = 1. \end{cases} \tag{15}$$

It suffices to show that $\lim_{\alpha \to 1} h_{\alpha}(t) = h_1(t)$ for $t \in [0,1]$. Let $f(\alpha) := t - t^{\alpha}$, and $g(\alpha) := \alpha(\alpha-1)$. Note that $\frac{f(1)}{g(1)} = {}^0/_0$, leading to an indetermination. Take the derivatives of f and g:

$$\begin{aligned} f'(\alpha) &= 0 - (\exp(\log t^{\alpha}))' \\ &= -\exp(\log t^{\alpha}) \cdot (\alpha \log t)' \\ &= -t^{\alpha} \log t, \end{aligned} \tag{16}$$

and $g'(\alpha) = 2\alpha - 1$. From l'Hôpital's rule,

$$\lim_{\alpha \to 1} \frac{f(\alpha)}{g(\alpha)} = \lim_{\alpha \to 1} \frac{f'(\alpha)}{g'(\alpha)} = -t \log t = h_1(t).$$

■

Note also that, as $\alpha \to \infty$, the denominator grows unbounded, so $\mathsf{H}^{\mathsf{T}}_{\infty} \equiv 0$.

A.2 Fenchel-Young losses

In this section, we recall the definitions and properties essential for our construction of α-entmax. The concepts below were formalized by Blondel et al. (2019) in more generality; we present below a less general version, sufficient for our needs.

Definition 1 (Probabilistic prediction function regularized by Ω)**.** *Let $\Omega\colon \triangle^d \to \mathbb{R} \cup \{\infty\}$ be a strictly convex regularization function. We define the prediction function $\boldsymbol{\pi}_{\Omega}$ as*

$$\boldsymbol{\pi}_{\Omega}(\boldsymbol{z}) \in \underset{\boldsymbol{p} \in \triangle^d}{\operatorname{argmax}} \left(\boldsymbol{p}^{\top}\boldsymbol{z} - \Omega(\boldsymbol{p})\right) \tag{17}$$

Definition 2 (Fenchel-Young loss generated by Ω)**.** *Let $\Omega\colon \triangle^d \to \mathbb{R} \cup \{\infty\}$ be a strictly convex regularization function. Let $\boldsymbol{y} \in \triangle$ denote a ground-truth label (for example, $\boldsymbol{y} = \boldsymbol{e}_y$ if there is a unique correct class y). Denote by $\boldsymbol{z} \in \mathbb{R}^d$ the prediction scores produced by some model, and by $\boldsymbol{p}^{\star} := \boldsymbol{\pi}_{\Omega}(\boldsymbol{z})$ the probabilistic predictions. The* **Fenchel-Young loss** *$\mathsf{L}_{\Omega}\colon \mathbb{R}^d \times \triangle \to \mathbb{R}_+$ generated by Ω is*

$$L_{\Omega}(\boldsymbol{z}; \boldsymbol{y}) := \Omega(\boldsymbol{y}) - \Omega(\boldsymbol{p}^{\star}) + \boldsymbol{z}^{\top}(\boldsymbol{p}^{\star} - \boldsymbol{y}). \tag{18}$$

This justifies our choice of entmax mapping and loss (Eqs. 10–11), as $\boldsymbol{\pi}_{-\mathsf{H}^{\mathsf{T}}_{\alpha}} = \alpha\text{-entmax}$ and $\mathsf{L}_{-\mathsf{H}^{\mathsf{T}}_{\alpha}} = \mathsf{L}_{\alpha}$.

Properties of Fenchel-Young losses.

1. **Non-negativity.** $L_{\Omega}(\boldsymbol{z}; \boldsymbol{y}) \geq 0$ for any $\boldsymbol{z} \in \mathbb{R}^d$ and $\boldsymbol{y} \in \triangle^d$.
2. **Zero loss.** $L_{\Omega}(\boldsymbol{z}; \boldsymbol{y}) = 0$ if and only if $\boldsymbol{y} = \boldsymbol{\pi}_{\Omega}(\boldsymbol{z})$, *i.e.*, the prediction is exactly correct.
3. **Convexity.** L_{Ω} is convex in $\boldsymbol{z}$.
4. **Differentiability.** L_{Ω} is differentiable with $\nabla \mathsf{L}_{\Omega}(\boldsymbol{z}; \boldsymbol{y}) = \boldsymbol{p}^{\star} - \boldsymbol{y}$.
5. **Smoothness.** If Ω is strongly convex, then L_{Ω} is smooth.
6. **Temperature scaling.** For any constant $t > 0$, $\mathsf{L}_{t\Omega}(\boldsymbol{z}; \boldsymbol{y}) = t\mathsf{L}_{\Omega}({}^{\boldsymbol{z}}/_{t}; \boldsymbol{y})$.

Characterizing the solution $\boldsymbol{p}^{\star}$ of $\boldsymbol{\pi}_{\Omega}(\boldsymbol{z})$. To shed light on the generic probability mapping in Eq. 17, we derive below the optimality conditions characterizing its solution. The optimality conditions are essential not only for constructing algorithms for computing $\boldsymbol{p}^{\star}$ (Appendix C), but also for deriving the Jacobian of the mapping (Appendix B). The Lagrangian of the maximization in Eq. 17 is

$$\mathcal{L}(\boldsymbol{p}, \boldsymbol{\nu}, \tau) = \Omega(\boldsymbol{p}) - (\boldsymbol{z} + \boldsymbol{\nu})^{\top}\boldsymbol{p} + \tau(\mathbf{1}^{\top}\boldsymbol{p} - 1). \tag{19}$$

with subgradient

$$\partial_{\boldsymbol{p}}\mathcal{L}(\boldsymbol{p}, \boldsymbol{\nu}, \tau) = \partial\Omega(\boldsymbol{p}) - \boldsymbol{z} - \boldsymbol{\nu} + \tau\mathbf{1}. \tag{20}$$

The subgradient KKT conditions are therefore:

$$\begin{cases} \boldsymbol{z} + \boldsymbol{\nu} - \tau\mathbf{1} \in \partial\Omega(\boldsymbol{p}) & (21) \\ \boldsymbol{p}^{\top}\boldsymbol{\nu} = 0 & (22) \\ \boldsymbol{p} \in \triangle^d & (23) \\ \boldsymbol{\nu} \geq 0. & (24) \end{cases}$$

Connection to softmax and sparsemax. We may now directly see that, when $\Omega(\boldsymbol{p}) := \sum_j p_j \log p_j$, Eq. 21 becomes $\log p_j = \boldsymbol{z}_j + \nu_j - \tau - 1$, which can only be satisfied if $p_j > 0$, thus $\boldsymbol{\nu} = \boldsymbol{0}$. Then, $p_j = {}^{\exp(z_j)}/_{Z}$, where $Z := \exp(\tau + 1)$. From Eq. 23, Z must be such that p_j sums to 1, yielding the well-known softmax expression. In the case of sparsemax, note that for any $\boldsymbol{p} \in \triangle^d$, we have

$$\begin{aligned}\Omega(\boldsymbol{p}) &= -\mathsf{H}^{\mathsf{G}}(\boldsymbol{p}) \\ &= {}^{1}/_{2}\sum_j p_j(p_j - 1) \\ &= {}^{1}/_{2}\|\boldsymbol{p}\|^2 - \frac{1}{2}\underbrace{\sum_j p_j}_{=1} \\ &= {}^{1}/_{2}\|\boldsymbol{p}\|^2 + \text{const.}\end{aligned} \tag{25}$$

Thus,

$$\begin{aligned}&\operatorname*{argmax}_{\boldsymbol{p}\in\triangle^d} \boldsymbol{p}^\top\boldsymbol{z} + \mathsf{H}^{\mathsf{G}}(\boldsymbol{p}) \\ &= \operatorname*{argmin}_{\boldsymbol{p}\in\triangle^d} 0.5\Big(\|\boldsymbol{p}\|^2 - 2\boldsymbol{p}^\top\boldsymbol{z} (+\|z\|^2)\Big) \\ &= \operatorname*{argmin}_{\boldsymbol{p}\in\triangle^d} \|\boldsymbol{p} - \boldsymbol{z}\|^2.\end{aligned} \tag{26}$$

B Backward pass for generalized sparse attention mappings

When a mapping $\boldsymbol{\pi}_\Omega$ is used inside the computation graph of a neural network, the Jacobian of the mapping has the important role of showing how to propagate error information, necessary when training with gradient methods. In this section, we derive a new, simple expression for the Jacobian of generalized sparse mappings $\boldsymbol{\pi}_\Omega$. We apply this result to obtain a simple form for the Jacobian of α-entmax mappings.

The proof is in two steps. First, we prove a lemma that shows that Jacobians are zero outside of the support of the solution. Then, to complete the result, we characterize the Jacobian on the support.

Lemma 1 (Sparse attention mechanisms have sparse Jacobians)**.** *Let $\Omega : \mathbb{R}^d \to \mathbb{R}$ be strongly convex. The attention mapping $\boldsymbol{\pi}_\Omega$ is differentiable almost everywhere, with Jacobian $\frac{\partial \boldsymbol{\pi}_\Omega}{\partial \boldsymbol{z}}$ symmetric and satisfying*

$$\frac{\partial(\boldsymbol{\pi}_\Omega(\boldsymbol{z}))_i}{\partial z_j} = 0$$

if $\big(\boldsymbol{\pi}_\Omega(\boldsymbol{z})\big)_i = 0$ or $\big(\boldsymbol{\pi}_\Omega(\boldsymbol{z})\big)_j = 0$.

Proof: Since Ω is strictly convex, the argmax in Eq. 17 is unique. Using Danskin's theorem (Danskin, 1966), we may write

$$\boldsymbol{\pi}_\Omega(\boldsymbol{z}) = \nabla \max_{\boldsymbol{p}\in\triangle}\Big(\boldsymbol{p}^\top\boldsymbol{z} - \Omega(\boldsymbol{p})\Big) = \nabla\Omega^*(\boldsymbol{z}).$$

Since Ω is strongly convex, the gradient of its conjugate Ω^* is differentiable almost everywhere (Rockafellar, 1970). Moreover, $\frac{\partial\boldsymbol{\pi}_\Omega}{\partial\boldsymbol{z}}$ is the Hessian of Ω^*, therefore symmetric, proving the first two claims.

Recall the definition of a partial derivative,

$$\frac{\partial(\boldsymbol{\pi}_\Omega(\boldsymbol{z}))_i}{\partial z_j} = \lim_{\varepsilon\to0}\frac{1}{\varepsilon}\left(\boldsymbol{\pi}_\Omega(\boldsymbol{z} + \varepsilon\boldsymbol{e}_j)_i - \boldsymbol{\pi}_\Omega(\boldsymbol{z})_i\right).$$

Denote by $\boldsymbol{p}^\star := \boldsymbol{\pi}_\Omega(\boldsymbol{z})$. We will show that for any j such that $p^\star_j = 0$, and any $\varepsilon \geq 0$,

$$\boldsymbol{\pi}_\Omega(\boldsymbol{z} - \varepsilon\boldsymbol{e}_j) = \boldsymbol{\pi}_\Omega(\boldsymbol{z}) = \boldsymbol{p}^\star.$$

In other words, we consider only one side of the limit, namely **subtracting** a small non-negative ε. A vector $\boldsymbol{p}^\star$ solves the optimization problem in Eq. 17 if and only if there exists $\boldsymbol{\nu}^\star \in \mathbb{R}^d$ and $\tau^\star \in \mathbb{R}$ satisfying Eqs. 21–24. Let $\boldsymbol{\nu}_\varepsilon := \boldsymbol{\nu}^\star + \varepsilon\boldsymbol{e}_j$. We verify that $(\boldsymbol{p}^\star, \boldsymbol{\nu}_\varepsilon, \tau^\star)$ satisfies the optimality conditions for $\boldsymbol{\pi}_\Omega(\boldsymbol{z} - \varepsilon\boldsymbol{e}_j)$, which implies that $\boldsymbol{\pi}_\Omega(\boldsymbol{z} - \varepsilon\boldsymbol{e}_j) = \boldsymbol{\pi}_\Omega(\boldsymbol{z})$. Since we add a non-negative quantity to $\boldsymbol{\nu}^\star$, which is non-negative to begin with, $(\boldsymbol{\nu}_\varepsilon)_j \geq 0$, and since $p^\star_j = 0$, we also satisfy $p^\star_j(\boldsymbol{\nu}_\varepsilon)_j = 0$. Finally,

$$\begin{aligned}&\boldsymbol{z} - \varepsilon\boldsymbol{e}_j + \boldsymbol{\nu}_\varepsilon - \tau^\star\boldsymbol{1} \\ =&\boldsymbol{z} - \varepsilon\boldsymbol{e}_j + \boldsymbol{\nu}^\star + \varepsilon\boldsymbol{e}_j - \tau^\star\boldsymbol{1} \\ \in&\partial\Omega(\boldsymbol{p}^\star).\end{aligned} \tag{27}$$

It follows that

$$\lim_{\varepsilon\to0_-}\frac{1}{\varepsilon}\left(\boldsymbol{\pi}_\Omega(\boldsymbol{z} + \varepsilon\boldsymbol{e}_j)_i - \boldsymbol{\pi}_\Omega(\boldsymbol{z})_i\right) = 0. \tag{28}$$

If $\boldsymbol{\pi}_\Omega$ is differentiable at $\boldsymbol{z}$, this one-sided limit must agree with the derivative. Otherwise, the sparse one-sided limit is a generalized Jacobian. ■

Proposition 2. *Let $\boldsymbol{p}^\star := \boldsymbol{\pi}_\Omega(\boldsymbol{z})$, with strongly convex and differentiable Ω. Denote the support of $\boldsymbol{p}^\star$ by $\mathcal{S} = \big\{j \in \{1, \ldots, d\} : p_j > 0\big\}$. If the second derivative $h_{ij} = \frac{\partial^2\Omega}{\partial p_i \partial p_j}(\boldsymbol{p}^\star)$ exists for any $i, j \in \mathcal{S}$, then*

$$\frac{\partial\boldsymbol{\pi}_\Omega}{\partial\boldsymbol{z}} = \boldsymbol{S} - \frac{1}{\|\boldsymbol{s}\|_1}\boldsymbol{s}\boldsymbol{s}^\top \tag{29}$$

where

$$S_{ij} = \begin{cases} H^{-1}_{ij}, & i,j \in \mathcal{S}, \\ 0, & o.w. \end{cases}, \quad \textit{and } s = S\mathbf{1}. \tag{30}$$

In particular, if $\Omega(\boldsymbol{p}) = \sum_j g(p_j)$ *with* g *twice differentiable on* $(0,1]$*, we have*

$$\frac{\partial \boldsymbol{\pi}_\Omega}{\partial \boldsymbol{z}} = \mathsf{diag}\, \boldsymbol{s} - \frac{1}{\|\boldsymbol{s}\|_1} \boldsymbol{s}\boldsymbol{s}^\top \tag{31}$$

where

$$s_i = \begin{cases} \left(g''(p_i^\star)\right)^{-1}, & i \in \mathcal{S}, \\ 0, & o.w. \end{cases} \tag{32}$$

Proof: Lemma 1 verifies that $\frac{\partial(\boldsymbol{\pi}_\Omega)_i}{\partial z_j} = 0$ for $i,j \notin \mathcal{S}$. It remains to find the derivatives w.r.t. $i,j \in \mathcal{S}$. Denote by $\bar{\boldsymbol{p}}^\star, \bar{\boldsymbol{z}}$ the restriction of the corresponding vectors to the indices in the support $\mathcal{S}$. The optimality conditions on the support are

$$\begin{cases} \boldsymbol{g}(\bar{\boldsymbol{p}}) + \tau\mathbf{1} & = \bar{\boldsymbol{z}} \\ \mathbf{1}^\top \bar{\boldsymbol{p}} & = 1 \end{cases} \tag{33}$$

where $\boldsymbol{g}(\bar{\boldsymbol{p}}) := \left(\nabla\Omega(\boldsymbol{p})\right)\big|_{\mathcal{S}}$, so $\frac{\partial \boldsymbol{g}}{\partial \bar{\boldsymbol{p}}}(\bar{\boldsymbol{p}}^\star) = \boldsymbol{H}$. Differentiating w.r.t. $\bar{\boldsymbol{z}}$ at $\boldsymbol{p}^\star$ yields

$$\begin{cases} \boldsymbol{H}\frac{\partial \bar{\boldsymbol{p}}}{\partial \bar{\boldsymbol{z}}} + \mathbf{1}\frac{\partial \tau}{\partial \bar{\boldsymbol{z}}} & = \boldsymbol{I} \\ \mathbf{1}^\top \frac{\partial \bar{\boldsymbol{p}}}{\partial \bar{\boldsymbol{z}}} & = 0 \end{cases} \tag{34}$$

Since Ω is strictly convex, $\boldsymbol{H}$ is invertible. From Gaussian elimination (*i.e.*, the Schur complement),

$$\frac{\partial \tau}{\partial \bar{\boldsymbol{z}}} = -\frac{1}{\mathbf{1}^\top \boldsymbol{H}^{-1}\mathbf{1}}\mathbf{1}^\top \boldsymbol{H}^{-1},$$

which can then be used to solve for $\frac{\partial \bar{\boldsymbol{p}}}{\partial \bar{\boldsymbol{z}}}$ giving

$$\frac{\partial \bar{\boldsymbol{p}}}{\partial \bar{\boldsymbol{z}}} = \boldsymbol{H}^{-1} - \frac{1}{\mathbf{1}^\top \boldsymbol{H}^{-1}\mathbf{1}}\boldsymbol{H}^{-1}\mathbf{1}\mathbf{1}^\top \boldsymbol{H}^{-1},$$

yielding the desired result. When Ω is separable, $\boldsymbol{H}$ is diagonal, with $H_{ii} = g''(p_i^\star)$, yielding the simplified expression which completes the proof. ■

Connection to other differentiable attention results. Our result is similar, but simpler than Niculae and Blondel (2017, Proposition 1), especially in the case of separable Ω. Crucially, our result does not require that the second derivative exist outside of the support. As such, unlike the cited work, our result is applicable in the case of α-entmax, where either $g''(t) = t^{\alpha-2}$ or its reciprocal may not exist at $t = 0$.

C Algorithms for entmax

C.1 General thresholded form for bisection algorithms.

The following lemma provides a simplified form for the solution of α-entmax.

Lemma 2. *For any* $\boldsymbol{z} \in \mathbb{R}^d$*, there is a unique* $\tau^\star \in \mathbb{R}$ *such that*

$$\alpha\text{-}\mathsf{entmax}(\boldsymbol{z}) = [(\alpha-1)\boldsymbol{z} - \tau^\star\mathbf{1}]_+^{1/\alpha-1}. \tag{35}$$

Proof: We use the regularized prediction functions defined in Appendix A.2. From both definitions,

$$\alpha\text{-}\mathsf{entmax}(\boldsymbol{z}) \equiv \boldsymbol{\pi}_{-\mathsf{H}^\mathsf{T}_\alpha}(\boldsymbol{z}).$$

We first note that for all $\boldsymbol{p} \in \triangle^d$,

$$-(\alpha-1)\mathsf{H}^\mathsf{T}_\alpha(\boldsymbol{p}) = \frac{1}{\alpha}\sum_{i=1}^d p_i^\alpha + \text{const}. \tag{36}$$

From the constant invariance and scaling of $\boldsymbol{\pi}_\Omega$ (Blondel et al., 2019, Proposition 1, items 4–5),

$$\boldsymbol{\pi}_{-\mathsf{H}^\mathsf{T}_\alpha}(\boldsymbol{z}) = \boldsymbol{\pi}_\Omega((\alpha-1)\boldsymbol{z}), \tag{37}$$

with

$$\Omega(\boldsymbol{p}) = \sum_{j=1}^d g(p_j), \quad g(t) = \frac{t^\alpha}{\alpha}. \tag{38}$$

Using (Blondel et al., 2019, Proposition 5), noting that $g'(t) = t^{\alpha-1}$ and $(g')^{-1}(u) = u^{1/\alpha-1}$, yields

$$\boldsymbol{\pi}_\Omega(\boldsymbol{z}) = [\boldsymbol{z} - \tau^\star\mathbf{1}]_+^{1/\alpha-1}, \tag{39}$$

and therefore

$$\alpha\text{-}\mathsf{entmax}(\boldsymbol{z}) = [(\alpha-1)\boldsymbol{z} - \tau^\star\mathbf{1}]_+^{1/\alpha-1}. \tag{40}$$

Uniqueness of $\tau^\star$ follows from the fact that α-entmax has a unique solution $\boldsymbol{p}^\star$, and Eq. 40 implies a one-to-one mapping between $\boldsymbol{p}^\star$ and $\tau^\star$, as long as $\boldsymbol{p}^\star \in \triangle$. ■

Corollary 2.1. *For* $\alpha = 1.5$*, Lemma 2 implies existence of a unique* $\tau^\star$ *such that*

$$1.5\text{-}\mathsf{entmax}(\boldsymbol{z}) = [\boldsymbol{z}/2 - \tau^\star\mathbf{1}]_+^2.$$

C.2 An exact algorithm for entmax with $\alpha = 1.5$: Derivation of Algorithm 2.

In this section, we derive an exact, sorting-based algorithm for 1.5-entmax. The key observation is that the solution can be characterized by the size of its support, $\rho^\star = \|\boldsymbol{p}^\star\|_0$. Then, we can simply enumerate all possible values of $\rho \in \{1, \dots, d\}$ until the solution verifies all optimality conditions. The challenge, however, is expressing the threshold τ as a function of the support size ρ; for this, we rely on $\alpha = 1.5$.

Proposition 3. *Exact computation of* $\mathsf{1.5\text{-}entmax}(\boldsymbol{z})$
Let $z_{[d]} \leq \dots \leq z_{[1]}$ denote the sorted coordinates of $\boldsymbol{z}$, and, for convenience, let $z_{[d+1]} := -\infty$. Define the top-ρ mean, unnormalized variance, and induced threshold for $\rho \in \{1, \dots, d\}$ as

$$M_{\boldsymbol{z}}(\rho) := \frac{1}{\rho}\sum_{j=1}^{\rho} z_{[j]},$$

$$S_{\boldsymbol{z}}(\rho) := \sum_{j=1}^{\rho}\left(z_{[j]} - M_{\boldsymbol{z}}(\rho)\right)^2,$$

$$\tau_{\boldsymbol{z}}(\rho) := \begin{cases} M_{\boldsymbol{z}}(\rho) - \sqrt{\frac{1-S_{\boldsymbol{z}}(\rho)}{\rho}}, & S_{\boldsymbol{z}}(\rho) \leq 1, \\ +\infty, & S_{\boldsymbol{z}}(\rho) > 1. \end{cases}$$

Then,

$$(\mathsf{1.5\text{-}entmax}(\boldsymbol{z}))_i = \left[\frac{z_i}{2} - \tau_{\boldsymbol{z}/2}(\rho)\right]_+^2, \qquad (41)$$

for any ρ satisfying $\tau_{\boldsymbol{z}}(\rho) \in [z_{[\rho+1]}, z_{[\rho]}]$.

Proposition 3 implies the correctness of Algorithm 2. To prove it, we first show the following.

Lemma 3. *Define $\tau(\rho)$ as in Proposition 3. Then, τ is non-decreasing, and there exists $\rho_{\mathit{max}} \in \{1, \dots, d\}$ such that τ is finite for $1 \leq \rho \leq \rho_{\mathit{max}}$, and infinite for $\rho > \rho_{\mathit{max}}$.*

The proof is slightly more technical, and we defer it to *after* the proof of the proposition.

Proof of Proposition 3. First, using Corollary 2.1 we reduce the problem of computing 1.5-entmax to

$$\boldsymbol{\pi}_\Omega(\boldsymbol{z}) := \mathop{\mathsf{argmax}}_{\boldsymbol{p} \in \triangle^d} \boldsymbol{p}^\top \boldsymbol{z} - \sum_j {}^2\!/\!_3\, p_j^{3/2}. \qquad (42)$$

Denote by $\tau^\star$ the optimal threshold as defined in the corollary. We will show that $\tau^\star = \tau(\rho)$ for any ρ satisfying $\tau(\rho) \in [z_{[\rho+1]}, z_{[\rho]}]$, where we assume, for convenience, $z_{[d+1]} = -\infty$. The generic stationarity condition in Eq. 21, applied to the problem in Eq. 42, takes the form

$$\sqrt{p_j} = \nu_j + z_j - \tau \quad \forall\, 0 < j \leq d \qquad (43)$$

Since Ω is symmetric, $\boldsymbol{\pi}_\Omega$ is permutation-preserving (Blondel et al., 2019, Proposition 1, item 1), so we may assume w.l.o.g. that $\boldsymbol{z}$ is sorted non-increasingly, i.e., $z_1 \geq \dots \geq z_d$; in other words, $z_j = z_{[j]}$. Therefore, the optimal $\boldsymbol{p}$ is also non-increasing. Denote by ρ an index such as $p_j \geq 0$ for $1 \leq j \leq \rho$, and $p_j = 0$ for $j > \rho$. From the complementary slackness condition (22), $\nu_j = 0$ for $1 \leq j \leq \rho$, thus we may split the stationarity conditions (43) into

$$\begin{cases} \sqrt{p_j} = z_j - \tau, & \forall\, 1 \leq j \leq \rho, & (44) \\ \nu_j = \tau - z_j, & \forall\, \rho < j \leq d. & (45) \end{cases}$$

For (44) to have solutions, the r.h.s. must be non-negative, i.e., $\tau \leq z_j$ for $j \leq \rho$, so $\tau \leq z_\rho$. At the same time, from dual feasability (24) we have $\nu_j = \tau - z_j \geq 0$ for $j > \rho$, therefore

$$\tau(\rho) \in [z_{\rho+1}, z_\rho]. \qquad (46)$$

Given ρ, we can solve for τ using (44) and primal feasability (23)

$$1 = \sum_{j=1}^{d} p_j = \sum_{j=1}^{\rho} (z_j - \tau)^2. \qquad (47)$$

Expanding the squares and dividing by 2ρ yields the quadratic equation

$$\frac{1}{2}\tau^2 - \frac{\sum_{j=1}^{\rho} z_j}{\rho}\tau + \frac{\sum_{j=1}^{\rho} z_j^2 - 1}{2\rho} = 0, \qquad (48)$$

with discriminant

$$\Delta(\rho) = \left(M(\rho)\right)^2 - \frac{\sum_{j=1}^{\rho} z_j^2}{\rho} + \frac{1}{\rho} = \frac{1 - S(\rho)}{\rho}. \qquad (49)$$

where we used the variance expression $\mathbb{E}\left[(X - \mathbb{E}[X])^2\right] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$. If $S(\rho) > 1$, $\Delta(\rho) < 0$, so there must exist an optimal ρ satisfying $S(\rho) \in [0, 1]$. Therefore, (48) has the two solutions $\tau_\pm(\rho) = M(\rho) \pm \sqrt{\frac{1-S(\rho)}{\rho}}$. However, τ_+ leads to a contradiction: The mean $M(\rho)$ is never smaller than the smallest averaged term, so $M(\rho) \geq z_\rho$, and thus $\tau_+ \geq z_\rho$. At the same time, from (46), $\tau \leq z_\rho$, so τ must equal z_ρ, which can only happen if $M(\rho) = z_\rho$ and

$S(\rho) = 1$. But $M(\rho) = z_\rho$ only if $z_1 = \cdots = z_\rho$, in which case $S(\rho) = 0$ (contradiction).

Therefore, $\tau^\star = \tau(\rho) = M(\rho) - \sqrt{\frac{1-S(\rho)}{\rho}}$ for *some* ρ verifying (46). It remains to show that *any* such ρ leads to the same value of $\tau(\rho)$. Pick any $\rho_1 < \rho_2$, both verifying (46). Therefore, $\rho_1 + 1 \leq \rho_2$ and

$$\begin{aligned} z_{\rho_1+1} &\underbrace{\leq}_{(46) \text{ for } \rho_1} \tau(\rho_1) \\ &\underbrace{\leq}_{\text{Lemma 3}} \tau(\rho_2) \\ &\underbrace{\leq}_{(46) \text{ for } \rho_2} z_{\rho_2} \\ &\underbrace{\leq}_{\boldsymbol{z} \text{ sorted}} z_{\rho_1+1}, \end{aligned} \tag{50}$$

thus $\tau(\rho_1) = \tau(\rho_2)$, and so any ρ verifying (46) satisfies $\tau^\star = \tau(\rho)$, concluding the proof. □

Proof of Lemma 3. We regard $\tau(\rho)$ as an extended-value sequence, *i.e.*, a function from $\mathbb{N} \to \mathbb{R} \cup \infty$. The lemma makes a claim about the *domain* of the sequence τ, and a claim about its monotonicity. We prove the two in turn.

Domain of τ. The threshold $\tau(\rho)$ is only finite for $\rho \in T := \big\{\rho \in \{1, \ldots, d\} \colon S(\rho) \leq 1\big\}$, i.e., where ${}^{(1-S(\rho))}/_{\rho} \geq 0$. We show there exists ρ_{max} such that $T = \{1, \ldots, \rho_{\mathsf{max}}\}$. Choose ρ_{max} as the largest index satisfying $S(\rho_{\mathsf{max}}) \leq 1$. By definition, $\rho > \rho_{\mathsf{max}}$ implies $\rho \notin T$. Remark that $S(1) = 0$, and $S(\rho{+}1) - S(\rho) = (\cdot)^2 \geq 0$. Therefore, S is nondecreasing and, for any $1 \leq \rho \leq \rho_{\mathsf{max}}$, $0 \leq S(\rho) \leq 1$.

Monotonicity of τ. Fix $\rho \in [\rho_{\mathsf{max}} - 1]$, assume w.l.o.g. that $M_{\boldsymbol{z}}(\rho) = 0$, and define $\tilde{\boldsymbol{z}}$ as

$$\tilde{z}_{[j]} = \begin{cases} x, & j = \rho + 1, \\ z_{[j]}, & \text{otherwise.} \end{cases}$$

The ρ highest entries of $\tilde{\boldsymbol{z}}$ are the same as in $\boldsymbol{z}$, so $M_{\tilde{\boldsymbol{z}}}(\rho) = M_{\boldsymbol{z}}(\rho) = 0$, $S_{\tilde{\boldsymbol{z}}}(\rho) = S_{\boldsymbol{z}}(\rho)$, and $\tau_{\tilde{\boldsymbol{z}}}(\rho) = \tau_{\boldsymbol{z}}(\rho)$. Denote $\widetilde{\tau}(x) := \tau_{\tilde{\boldsymbol{z}}}(\rho + 1)$, and analogously $\widetilde{M}(x)$ and $\widetilde{S}(x)$. Then,

$$\tau_{\boldsymbol{z}}(\rho{+}1) = \widetilde{\tau}(z_{[\rho+1]}) \geq \min_{x \colon \widetilde{S}(x) \in [0,1]} \widetilde{\tau}(x) =: \widetilde{\tau}(x^\star) \tag{51}$$

We seek the lower bound $\widetilde{\tau}(x^\star)$ and show that $\widetilde{\tau}(x^\star) \geq \tau_{\boldsymbol{z}}(\rho)$. From (51), this implies $\tau_{\boldsymbol{z}}(\rho + 1) \geq \tau_{\boldsymbol{z}}(\rho)$ and, by transitivity, the monotonicity of $\tau_{\boldsymbol{z}}$.

It is easy to verify that the following incremental update expressions hold.

$$\widetilde{M}(x) = \frac{x}{\rho+1}, \quad \widetilde{S}(x) = S_{\boldsymbol{z}}(\rho) + \frac{\rho}{\rho+1} x^2. \tag{52}$$

We must solve the optimization problem

$$\text{minimize}_x \quad \widetilde{\tau}(x) \quad \text{subject to} \quad \widetilde{S}(x) \in [0, 1]. \tag{53}$$

The objective value is

$$\begin{aligned} \widetilde{\tau}(x) &= \widetilde{M}(x) - \sqrt{\frac{1 - \widetilde{S}(x)}{\rho + 1}} \\ &= \frac{1}{\rho+1}\left(x - \sqrt{\big(1 - S_{\boldsymbol{z}}(\rho)\big)(\rho+1) - \rho x^2}\right) \end{aligned} \tag{54}$$

Ignoring the constraint for a moment and setting the gradient to 0 yields the solution

$$\begin{aligned} 0 &= \widetilde{\tau}'(x^\star) \\ &= \frac{1}{\rho+1}\left(1 + \frac{\rho x^\star}{\sqrt{\big(1 - S_{\boldsymbol{z}}(\rho)\big)(\rho+1) - \rho x^{\star 2}}}\right) \\ \iff \rho x^\star &= -\sqrt{\big(1 - S_{\boldsymbol{z}}(\rho)\big)(\rho+1) - \rho x^{\star 2}}, \end{aligned} \tag{55}$$

implying $x^\star < 0$. Squaring both sides and rearranging yields the solution of the unconstrained optimization problem,

$$x^\star = -\sqrt{\frac{1 - S_{\boldsymbol{z}}(\rho)}{\rho}}. \tag{56}$$

We verify that $x^\star$ readily satisfies the constraints, thus it is a solution to the minimization in Eq. 53. Since

$$\widetilde{S}(x^\star) = S_{\boldsymbol{z}}(\rho) + \frac{1 - S_{\boldsymbol{z}}(\rho)}{\rho + 1}, \tag{57}$$

we have

$$\widetilde{S}(x^\star) \geq S_{\boldsymbol{z}}(\rho) \geq 0 \tag{58}$$

and

$$\widetilde{S}(x^\star) \leq S_{\boldsymbol{z}}(\rho) + \frac{1 - S_{\boldsymbol{z}}(\rho)}{2} \leq 1. \tag{59}$$

Plugging $x^\star$ into the objective yields

$$\begin{aligned}
&\widetilde{\tau}(x^\star) \\
&= \frac{1}{\rho+1}\left(-\sqrt{\frac{1-S_{\boldsymbol{z}}(\rho)}{\rho}} - \sqrt{\rho\big(1-S_{\boldsymbol{z}}(\rho)\big)}\right) \\
&= -\sqrt{\frac{1-S_{\boldsymbol{z}}(\rho)}{\rho}}\frac{1}{\rho+1}(1+\rho) \\
&= -\sqrt{\frac{1-S_{\boldsymbol{z}}(\rho)}{\rho}} = \tau_{\boldsymbol{z}}(\rho).
\end{aligned} \tag{60}$$

Therefore, $\widetilde{\tau}(x) \geq \tau_{\boldsymbol{z}}(\rho)$ for any valid x, proving that $\tau_{\boldsymbol{z}}(\rho) \leq \tau_{\boldsymbol{z}}(\rho+1)$.