Generalized Entropy Regularization or: There's Nothing Special about Label Smoothing

Clara Meister[ʕ] **Elizabeth Salesky**[ʒ] **Ryan Cotterell**[ð,ʕ]
[ʕ]ETH Zürich [ʒ]Johns Hopkins University [ð]University of Cambridge
clara.meister@inf.ethz.ch esalesky@jhu.edu
ryan.cotterell@inf.ethz.ch

Abstract

Prior work has explored directly regularizing the output distributions of probabilistic models to alleviate peaky (i.e. over-confident) predictions, a common sign of overfitting. This class of techniques, of which label smoothing is one, has a connection to entropy regularization. Despite the consistent success of label smoothing across architectures and data sets in language generation tasks, two problems remain open: (1) there is little understanding of the underlying effects entropy regularizers have on models, and (2) the full space of entropy regularization techniques is largely unexplored. We introduce a parametric family of entropy regularizers, which includes label smoothing as a special case, and use it to gain a better understanding of the relationship between the entropy of a model and its performance on language generation tasks. We also find that variance in model performance can be explained largely by the resulting entropy of the model. Lastly, we find that label smoothing provably does not allow for sparsity in an output distribution, an undesirable property for language generation models, and therefore advise the use of other entropy regularization methods in its place. Our code is available online at https://github.com/rycolab/entropyRegularization.

1 Introduction

When training large neural networks with millions of parameters, regularization of some form is needed to prevent overfitting, even when large amounts of data are used; models for language generation are no exception. In probabilistic modeling, e.g. when the final layer of the neural network is a softmax, overfitting often manifests itself in overconfident placement of most of the probability mass on a few candidates, resulting in peaky (low-entropy) probability distributions over the vocabulary. Specifically for language generation tasks, this behavior leads to the output of repetitive or frequently occurring but unrelated text, which is detrimental to the generalization abilities of the model (Chorowski and Jaitly, 2017; Holtzman et al., 2020). A natural regularizer to consider is, therefore, one that penalizes overconfidence, encouraging higher entropy in the learned distribution. Indeed, the literature has ascribed gains of ≈ 1 BLEU point in machine translation to label smoothing, one such technique (Chen et al., 2018).

Despite the clear relationship between low entropy and overfitting, only a handful of distinct entropy regularizers have been explored. To fill this gap, we introduce **generalized entropy regularization** (GER), a unified framework for understanding and exploring a broad range of entropy-inducing regularizers. GER is based on the skew-Jensen family of divergences $\mathrm{J}_{\alpha,G}$ (Nielsen and Boltz, 2011) and thus may be generalized to any Bregman divergence through the choice of generator function G. For the negative entropy generator function, GER recovers label smoothing (Szegedy et al., 2015) as $\alpha \rightarrow 1$, and the confidence penalty (Pereyra et al., 2017) as $\alpha \rightarrow 0$. We provide formal properties of GER in §3, proving these special-case equivalences among other properties of GER. We then use GER to empirically examine the relationship between entropy and the evaluation metrics in two language generation tasks: neural machine translation (NMT) and abstractive summarization.

GER encompasses a large family of regularizers, which allows us to directly compare label smoothing to other forms of entropy regularization. By studying the relationship between different regularizers on the performance of natural language generation (NLG) systems, we can better understand not just *when* but also *why* label smoothing aids language generation tasks. Through our analysis, we gain the following insights:

(i) With tuning of the regularizer's coefficient, *any* choice of α can yield similar performance, i.e. there is nothing special about label smoothing. In fact, our results suggest that label smoothing ($\alpha \to 1$) makes it more difficult to tune the regularizer's coefficient.

(ii) Label smoothing assigns infinite cost to sparse output distributions, which may be an undesirable behavior for language generation tasks.

(iii) There is a strong (quadratic) relationship between a model's performance on the evaluation metric and its (average) entropy, offering a hint as to why these regularizers are so effective for NLG.

In summary, entropy-inducing regularizers are a boon to probabilistic NLG systems, which benefit from higher entropy output distributions. Label smoothing works *because* it forces the model towards a higher-entropy solution, but we recommend the confidence penalty and other entropy regularizers ($\alpha < 1$) for reasons (i) and (ii) above.

2 Preliminaries

In this work, we consider conditional probability models $p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})$ for natural language generation; such models assign probability to a target sequence $\mathbf{y} \in \mathcal{Y}$ given a source sequence $\mathbf{x}$. Specifically, our target sequence $\mathbf{y} = \langle y_1, \ldots, y_n \rangle$ of arbitrary length n is a sequence of target words[1] y_i from our vocabulary Y. The set of all complete target sequences, which are padded with distinguished beginning- and end-of-sentence symbols, BOS and EOS, is then defined as $\mathcal{Y} := \{\text{BOS} \circ \mathbf{y} \circ \text{EOS} \mid \mathbf{y} \in Y^*\}$. For language generation tasks, $p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})$ is typically a neural network with parameters $\boldsymbol{\theta}$; this network is trained to approximate $\tilde{p}(\mathbf{y} \mid \mathbf{x})$, the empirical distribution (i.e. the distribution of the data). In this work, we focus on locally normalized models; in such models $p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x})$ is factored as:

$$p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x}) = p_{\boldsymbol{\theta}}(y_1 \mid \mathbf{x}) \cdots p_{\boldsymbol{\theta}}(y_n \mid \mathbf{x}, \mathbf{y}_{<n}) \quad (1)$$

where $p_{\boldsymbol{\theta}}(y_i \mid \mathbf{x}, \mathbf{y}_{<i})$ is represented by a softmax over the output of the final fully connected layer of the network. Generation is performed using greedy search, beam search or a sampling scheme. Of the candidate sequences generated, the one with the highest probability under the model $p_{\boldsymbol{\theta}}$ is returned as the model's prediction.

One way of selecting the parameters $\boldsymbol{\theta}$ is to minimize the KL-divergence between the empirical distribution and the model. This yields the cross-entropy loss (plus an additive constant):[2]

$$\mathcal{L}(\boldsymbol{\theta}) = \mathrm{KL}(\tilde{p} \mid\mid p_{\boldsymbol{\theta}}) \quad (2)$$
$$= \underbrace{\mathrm{H}(\tilde{p}, p_{\boldsymbol{\theta}})}_{\text{cross-entropy loss}} - \underbrace{\mathrm{H}(\tilde{p})}_{\text{constant w.r.t. } \boldsymbol{\theta}} \quad (3)$$

However, fitting a model that perfectly approximates the empirical distribution is, in general, fraught with problems (Hastie et al., 2001). The goal of learning is to generalize beyond observed data. Exactly fitting the empirical distribution, often termed overfitting, is therefore not an ideal goal and for language generation models specifically, does not go hand-in-hand with the ability of a model to generate desirable text (Bengio et al., 2015). Consequently, it is advisable to minimize a regularized objective to prevent overfitting:

$$\mathcal{L}(\boldsymbol{\theta}) + \beta\, \mathcal{R}(\boldsymbol{\theta}) \quad (4)$$

where $\mathcal{R}(\boldsymbol{\theta})$ is a regularizer defined over the model with "strength" coefficient $\beta > 0$.

2.1 Entropy Regularization

Overfitting can manifest itself as peakiness in $p_{\boldsymbol{\theta}}$ (Williams and Peng, 1991; Mnih et al., 2016; Pereyra et al., 2017). In other words, $p_{\boldsymbol{\theta}}$ overconfidently places most of the probability mass on very few candidates. While this overconfidence improves training loss, it hurts generalization. Entropy regularization is one technique that directly combats such overconfidence by encouraging more entropic (less peaky) distributions.

The entropy of the model $p_{\boldsymbol{\theta}}$ is defined as

$$\mathrm{H}\left(p_{\boldsymbol{\theta}}\right) := -\sum_{\mathbf{y} \in \mathcal{Y}} p_{\boldsymbol{\theta}}(\mathbf{y}) \log p_{\boldsymbol{\theta}}(\mathbf{y}) \quad (5)$$

where we remove dependence on $\mathbf{x}$ for notational simplicity. However, the sum in eq. (5) over $\mathcal{Y}$ generally renders its computation intractable.[3] Instead, regularization is performed on the conditional distribution over Y at each time step, which can be interpreted as an approximation of the true model entropy. For ease of notation, we define a higher-order function D_f over our training corpus $\mathcal{C}$ consisting of $\langle \mathbf{x}, \mathbf{y} \rangle$ pairs that maps a function f over

[1]Targets y_i may also be characters or subwords; our experiments use byte-pair encoding (Sennrich et al., 2016)

[2]$\mathrm{H}(p, q) := -\sum_{z \in \mathcal{Z}} p(z) \log q(z)$ is cross-entropy and $\mathrm{H}(p) := \mathrm{H}(p, p) = -\sum_{z \in \mathcal{Z}} p(z) \log p(z)$ is the Shannon entropy, for which $\log = \log_2$ and $\mathcal{Z} = \mathrm{supp}(p)$.

[3]The notation used by Pereyra et al. (2017) is imprecise.

Training Method	Loss Function	Alternate Formulation
Cross Entropy	$\mathcal{L}(\boldsymbol{\theta}) = \mathrm{H}(\tilde{p}, p_{\boldsymbol{\theta}})$	$= \mathrm{KL}(\tilde{p} \,\|\|\, p_{\boldsymbol{\theta}}) + \mathrm{H}(\tilde{p})$
Confidence Penalty, D_{J_0}	$\mathcal{L}_{CP}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \beta\, D_{\mathrm{KL}}(p_{\boldsymbol{\theta}} \,\|\|\, u)$	$= \mathcal{L}(\boldsymbol{\theta}) - \beta\, D_{\mathrm{H}}(p_{\boldsymbol{\theta}}) + C$
Label Smoothing, D_{J_1}	$\mathcal{L}_{LS}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \beta\, D_{\mathrm{KL}}(u \,\|\|\, p_{\boldsymbol{\theta}})$	$= \mathcal{L}(\boldsymbol{\theta}) + \beta\, D_{\mathrm{H}}(u, p_{\boldsymbol{\theta}}) + C$
Generalized Entropy Regularization, D_{J_α}	$\mathcal{L}_{GER}(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + \beta\, D_{\mathrm{J}_\alpha}(u \,\|\|\, p_{\boldsymbol{\theta}})$	—

Table 1: Loss functions and their alternate formulations for different training methods; the latter three are entropy regularization techniques that augment the standard loss function in row 1. C denotes a constant with respect to $\boldsymbol{\theta}$.

distributions p, q as follows below:

$$D_f(p \,||\, q) = \sum_{\langle \mathbf{x},\mathbf{y}\rangle \in \mathcal{C}} \sum_{t=1}^{|\mathbf{y}|} f(p(\cdot \mid \mathbf{x}, \mathbf{y}_{<t}) \,||\, q(\cdot \mid \mathbf{x}, \mathbf{y}_{<t})) \tag{6}$$

The function D_f allows us to describe in notation how entropy regularization is typically employed in the training of language generation systems.[4]

Label Smoothing. Label smoothing, first introduced as a regularizer for neural networks by Szegedy et al. (2015), is so named because the technique smooths hard target distributions. One such distribution, the empirical distribution, is represented with a set of one-hot encoded vectors (hard targets) where for each data point, the correct label (e.g., vocabulary index of a word) has value 1 and all other labels have value 0. Label smoothing with strength coefficient γ is an add-γ smoothing scheme on the distribution over labels at every time step. Interestingly, minimizing the cross entropy between this modified distribution and the model $p_{\boldsymbol{\theta}}$ is equivalent to adding the weighted KL divergence between the uniform distribution and the model $p_{\boldsymbol{\theta}}$ in our original objective function with the same strength coefficient:

$$\mathcal{L}(\boldsymbol{\theta})_{\gamma}^{LS} := (1-\gamma)\mathcal{L}(\boldsymbol{\theta}) + \gamma\, D_{\mathrm{KL}}(u \,||\, p_{\boldsymbol{\theta}}) \tag{7}$$

While the loss function is often scaled as above, it is nonetheless equivalent to $\mathcal{L}(\boldsymbol{\theta})_{\beta}^{LS} = \mathcal{L}(\boldsymbol{\theta}) + \beta\, D_{\mathrm{KL}}(u \,||\, p_{\boldsymbol{\theta}})$[5] – we use this form for consistency.

Confidence Penalty. The confidence penalty, formally explored in the supervised learning setting by Pereyra et al. (2017), aims to penalize a low-entropy model. This is done by subtracting a weighted term for the entropy of the model's prediction $p_{\boldsymbol{\theta}}(\cdot)$ from the loss function, thereby encouraging a more entropic model. This is equivalent to adding the KL divergence between the model $p_{\boldsymbol{\theta}}$ and the uniform distribution:

$$\mathcal{L}(\boldsymbol{\theta})_{\beta}^{CP} := \mathcal{L}(\boldsymbol{\theta}) + \beta\, D_{\mathrm{KL}}(p_{\boldsymbol{\theta}} \,||\, u) \tag{8}$$

While Pereyra et al. (2017) found that label smoothing performed better than the confidence penalty for NMT, they only searched coarsely over a small range of β's for both regularizers. Our findings in §4 suggest an alternate conclusion.

3 Generalized Entropy Regularization

The positive effect of both label smoothing and the confidence penalty on model performance in language generation tasks motivates further exploration of entropy-promoting regularizers. To this end, we construct a parameterized family of regularizers with label smoothing and the confidence penalty as special cases. We discuss the formal properties of a subset of this family, providing upper and lower bounds for it. We show divergence only occurs in one case for this subset ($\alpha \to 1$), which directly implies that no sparse solution exists when label smoothing is used as a regularizer.

3.1 A Family of Entropy Regularizers

We derive a family of regularizers from the skew-Jensen divergence $\mathrm{J}_{\alpha,G}$ (Nielsen and Boltz, 2011), which is defined below as:

$$\mathrm{J}_{\alpha,G}(q \,||\, p_{\boldsymbol{\theta}}) := \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)G(q) + \alpha G(p_{\boldsymbol{\theta}}) - G((1-\alpha)q + \alpha p_{\boldsymbol{\theta}})\Big) \tag{9}$$

for a strictly convex generator function $G : \Omega \to \mathbb{R}$ and $\alpha \in (0,1)$ where Ω is a closed convex set. In this paper, we restrict Ω to be the $|Y|$-simplex. Note that $\mathrm{J}_{\alpha,G}(q \,||\, p_{\boldsymbol{\theta}}) \neq \mathrm{J}_{\alpha,G}(p_{\boldsymbol{\theta}} \,||\, q)$ in general, although this is true for some choices of G and α.

We define the generalized entropy regularizer as $\mathcal{R}(\boldsymbol{\theta}) = D_{\mathrm{J}_{\alpha,G}}(u \,||\, p_{\boldsymbol{\theta}})$ where u is the uniform

[4] Note that the standard loss function in eq. (3) can be written in this form when computed over $\mathcal{C}$, i.e. $\mathrm{KL}(\tilde{p} \,||\, p_{\boldsymbol{\theta}}) = D_{\mathrm{KL}}(\tilde{p} \,||\, p_{\boldsymbol{\theta}})$, since the reference $\mathbf{y}$ is the only value in $\mathrm{supp}(\tilde{p})$.

[5] up to multiplicative factor $1-\gamma$ when $\beta = \gamma/(1-\gamma)$

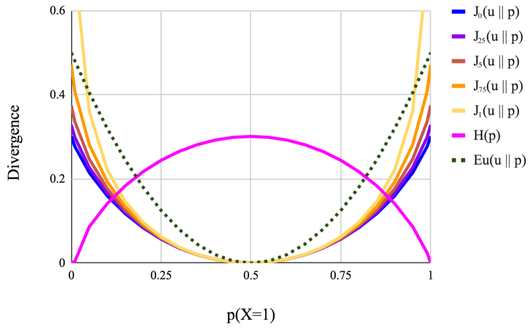

Figure 1: Different divergence measures between u, the uniform distribution and p, a probability distribution over a Bernoulli random variable X. Note that the **confidence penalty** is equivalent to $\mathrm{KL}(p \mid\mid u) = \mathrm{J}_0$ and **label smoothing** is equivalent to $\mathrm{KL}(u \mid\mid p) = \mathrm{J}_1$ (see §3.1). We include entropy $\mathrm{H}(p)$ and $\mathrm{Eu}(u \mid\mid p) = \mathrm{J}_{\alpha,G}(u \mid\mid p)$ for $\alpha = 0.5$ and $G(p) = ||p||_2^2$.

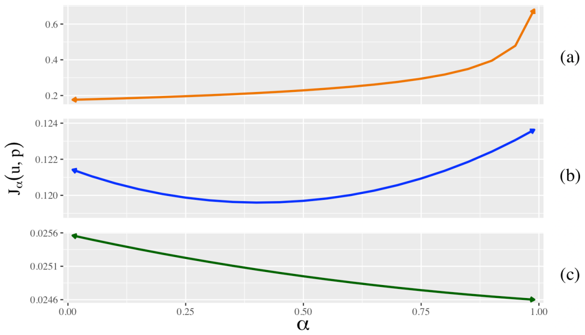

Figure 2: $\mathrm{J}_\alpha(u \mid\mid p)$ as a function of α for u, the uniform distribution, and p, a probability distribution over a 3-way categorical random variable, where for (a) $p = (0.0001, 0.49995, 0.49995)$ (b) $p = (0.15, 0.15, 0.7)$ and (c) $p = (0.25, 0.25, 0.5)$. There is no standard trend for J_α as purely a function of $\alpha \in (0, 1)$.

distribution.[6] These regularizers promote entropy because they push the model p_θ towards u, which is the maximum-entropy distribution with an entropy of $\log|Y|$. Throughout the rest of this paper, we primarily use the generator function[7] $G(p) = -\mathrm{H}(p)$. We use J_α as shorthand for $\mathrm{J}_{\alpha,-\mathrm{H}}$.

We note J_α is equivalent to quadruple the Jensen–Shannon (JS) divergence and asymptotically approaches the Kullback–Leibler (KL) divergence for certain values of α. Specifically, we have:

$$\lim_{\alpha \to 0} \mathrm{J}_\alpha(q \mid\mid p_\theta) = \mathrm{KL}(p_\theta \mid\mid q) \tag{10}$$

$$\lim_{\alpha \to 1} \mathrm{J}_\alpha(q \mid\mid p_\theta) = \mathrm{KL}(q \mid\mid p_\theta) \tag{11}$$

$$\mathrm{J}_{1/2}(q \mid\mid p_\theta) = 4 \cdot \mathrm{JS}(q \mid\mid p_\theta) \tag{12}$$

We prove these relationships in App. A and App. B. For ease, we define $\mathrm{J}_1 := lim_{\alpha \to 1}\mathrm{J}_\alpha$ and $\mathrm{J}_0 := \lim_{\alpha \to 0} \mathrm{J}_\alpha$. We note the following two equivalences for these special cases.

Proposition 1. $\nabla_\theta \mathrm{J}_1(u \mid\mid p_\theta) = \nabla_\theta \mathrm{H}(q, p_\theta)$. *In words, the gradient of the loss with GER as $\alpha \to 1$ is equivalent to the gradient of the loss augmented with* ***label smoothing****.*

Proposition 2. $\nabla_\theta \mathrm{J}_0(u \mid\mid p_\theta) = \nabla_\theta \mathrm{H}(p_\theta)$. *In words, the gradient of the loss with GER as $\alpha \to 0$ is equivalent to the gradient of the loss augmented with the* ***confidence penalty****.*

See App. C and App. D for proofs.

[6] Distributions other than u may also be used. See §5.
[7] We also experiment with $G(z) = ||z||_2^2$.

3.2 Formal Properties of J_α

When fitting a model p_θ, we generally optimize the inclusive KL, i.e. $\mathrm{KL}(\tilde{p} \mid\mid p_\theta)$, so that, among other reasons, p_θ has support everywhere that $\tilde{p}$ has support. However, it is unclear what relationships we want to encourage between the model p_θ and the uniform distribution u during regularization as complete support of u implies no word can ever have non-zero probability. Here we explore formal properties of J_α as a regularizer to gain insight into how, as a function of α, these regularizers affect the learned distribution.

Magnitude. Figure 1 shows the different divergence measures between u and p_θ. We see that $\mathrm{J}_1 = \mathrm{KL}(u \mid\mid p_\theta)$ **(label smoothing)** is much larger than $\mathrm{J}_0 = \mathrm{KL}(p_\theta \mid\mid u)$ **(confidence penalty)** at values of p_θ farther from u. This indicates that J_1 would be a stronger regularizer than $\mathrm{J}_{<1}$, i.e. penalize values of p_θ far from u more heavily, given the same strength coefficient β. Note that it is not always the case that $\mathrm{J}_{<1}(u \mid\mid p) \le \mathrm{J}_1(u \mid\mid p)$ for fixed p. We can, however, bound J_α from above and below by other quantities.

Proposition 3. *The divergence* $\mathrm{J}_\alpha(u \mid\mid p)$ *is* not *a monotonic function of α for all distributions p.*

A proof by counter example is shown in Figure 2.

Proposition 4. *For fixed p, J_α has bounds:* $0 \le \mathrm{J}_\alpha(u \mid\mid p) \le \mathrm{KL}(u \mid\mid p) + \mathrm{KL}(p \mid\mid u)$.

See App. E for a proof.

Sparsity. Sparsity is generally a desirable trait in probabilistic models; specifically for structured prediction, it leads to improvements in performance and interpretability (Martins et al., 2011; Niculae

	WMT'14 De-En				IWSLT'14 De-En				MTTT Fr-En			
	α	β	H	BLEU	α	β	H	BLEU	α	β	H	BLEU
No Regularization	–	0	0.11	31.1	–	0	0.1	35.7	–	0	0.15	35.2
Label Smoothing D_{J_1} $(\gamma=0.1)$	1	0.11	0.23	31.3 **+0.2**	1	0.11	0.18	36.9 **+1.2**	1	0.11	0.18	36.5 **+0.8**
Label Smoothing D_{J_1}	1	0.35	0.38	31.7 **+0.6**	1	0.50	0.40	37.2 **+1.5**	1	0.693	0.47	37.5 **+2.3**
Confidence Penalty D_{J_0}	0	0.28	0.55	31.6 **+0.5**	0	0.76	0.81	37.5 **+1.8**	0	0.95	0.86	37.4 **+2.2**
GER D_{J_α}	0.7	0.65	0.47	32.0 **+0.9**	0.5	1.00	0.56	37.5 **+1.8**	0.85	0.52	0.37	37.6 **+2.4**

Table 2: BLEU scores and normalized entropy $\mathrm{H}(p_{\boldsymbol{\theta}})$ on the test sets for WMT'14 De-En, WMT'14 De-En, and MTTT Fr-En. Results include baseline models with no (entropy) regularization and standard label smoothing with γ=0.1 (equivalent to $\beta \approx 0.11$). We report scores from the best model found (on validation set) for D_{J_0}, D_{J_1}, and D_{J_α} over all α, β pairs. BLEU standard deviation across random seeds was typically < 0.1 and always < 0.16.[8] Results for MTTT Ja-En and convolutional architectures can be found in App. H.

et al., 2018). For example, Martins and Astudillo (2016) showed the benefits of using sparsemax, which induces sparsity in an output distribution, for natural language inference tasks. There are also intuitive reasons for allowing $p_{\boldsymbol{\theta}}$ to be sparse. Part of modeling language generations tasks is learning when particular sequences *cannot*, or at least should not, occur (e.g. are grammatically or syntactically incorrect). In these cases, a model should be able to assign 0 probability mass to that sequence. However, there is no sparse optimal solution $p_{\boldsymbol{\theta}}$ when using label smoothing.

Proposition 5. $\mathrm{J}_\alpha(u \mid\mid p)$ *is finite for any* $p \in \Omega$ *and any* $\alpha < 1$. *As* $\alpha \to 1$, $\mathrm{J}_\alpha(u \mid\mid p)$ *diverges iff* $\exists y \in \mathrm{supp}(u)$ *for which* $p(y) = 0$.

See App. F for a proof.

4 Experiments

We evaluate our family of entropy regularizers on two language generation tasks: machine translation and abstractive summarization. We then analyze trends in model performance as a function of α and model entropy[9] and explore how this entropy affects other properties of language generation models. In the following experiments, each model is trained using eq. (4) where $\mathcal{R}(\boldsymbol{\theta}) = D_{\mathrm{J}_\alpha}(\tilde{p} \mid\mid p_{\boldsymbol{\theta}})$. We conduct searches over α and β using Bayesian optimization (Snoek et al., 2012) to find the combination of regularizer D_{J_α} and strength coefficient β that lead to the best evaluation score on the respective task.[10] We additionally do a more fine-grained grid search over β for J_0 (confidence penalty) and J_1 (label smoothing) for completeness. All other model hyperparameters are held constant. We run experiments on multiple architectures and across several data sets to ensure trends are general.

4.1 Neural Machine Translation

We explore performance of the regularizer D_{J_α} on NMT systems using three language pairs and corpora of two different sizes on the following tasks: WMT'14 German-to-English (De-En) (Bojar et al., 2014), IWSLT'14 German-to-English (De-En) (Cettolo et al., 2012), and Multitarget Ted Talks Task (MTTT) French-to-English (Fr–En) and Japanese-to-English (Ja-En) tasks (Duh, 2018). For the larger WMT data set, we train fewer models using coarser-grained α and β ranges. We perform experiments for both Transformers (Vaswani et al., 2017) and convolutional sequence-to-sequence models (Gehring et al., 2017).

For reproducibility and comparability, we use the data pre-processing scripts provided by fairseq (Ott et al., 2019) and follow recommended hyperparameter settings from previous work (Vaswani et al., 2017; Gehring et al., 2017) for baseline models. We use SacreBLEU (Post, 2018) to calculate BLEU scores (Papineni et al., 2002). Specific data pre-processing steps and model hyperparameter details are provided in App. G. Decoding is performed with length-normalized beam search with a beam size of 5 unless otherwise stated. Early stopping was used during training; model parameters were taken from the checkpoint with the best validation set BLEU.

[8] We have $\alpha \approx 1$ as an exception; the standard deviation is slightly higher for larger values of β.

[9] Model entropy is estimated as an average of the entropies of distributions at each time step during decoding. Entropy is normalized by the maximum possible entropy for the given vocabulary size ($\log |Y|$) in all figures and tables to control for the fact that languages have vocabularies of different sizes.

[10] We only report results with generator function $G = -\mathrm{H}$ as results using $G(z) = ||z||_2^2$ were consistently worse and often did not improve on the baseline; these results may be seen in App. H.

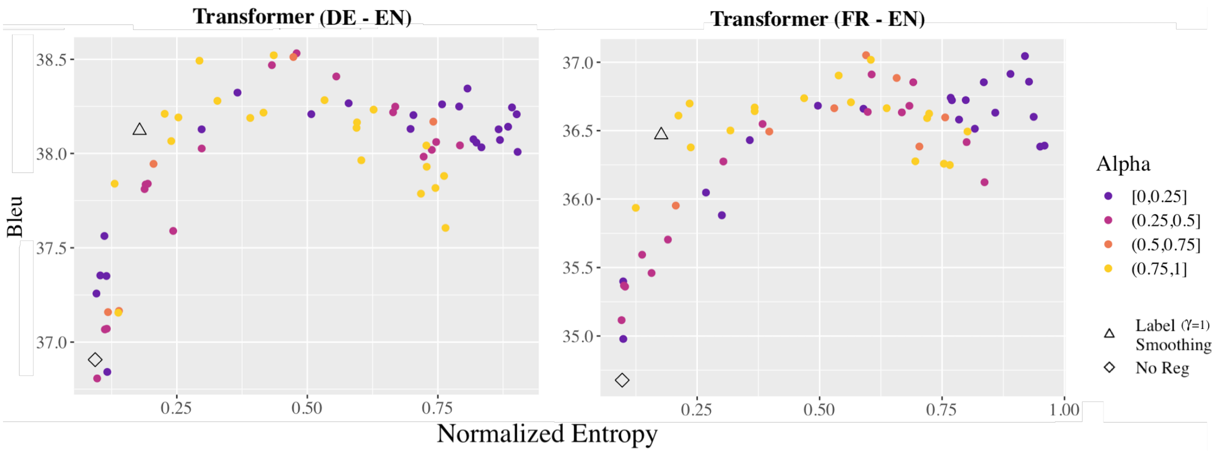

Figure 3: Model entropy $\mathrm{H}(p_{\theta})$ vs. BLEU on IWSLT'14 German to English (De-En) and Multitarget Ted Talks Task French to English (Fr-En) using a Transformer architecture; each point is a fully trained model, regularized with D_{J_α} for varying α and β. Label smoothing at standard $\gamma = 0.1$ and no (entropy) regularization are marked.

	α	β	$\mathrm{H}(p)$	ROUGE-L
No Regularization	–	–	0.08	40.5
Confidence Penalty D_{J_0}	0	0.15	0.19	40.9 **+0.4**
Label Smoothing D_{J_1}	1	0.1	0.2	40.9 **+0.4**
GER D_{J_α}	0.5	0.35	0.19	40.8 **+0.3**

Table 3: ROUGE-L on test set for CNN/DailyMail abstractive summarization task. Note that we replicate their reported result (achieved with label smoothing).

Results of our experiments are shown in Table 2 and Figure 3. We see the same relation between model entropy and BLEU with both Transformer and convolutional architectures and between different language pairs. We show results for the Transformer architectures inline as they are the current standard for many NLP tasks; results for convolutional architectures are in App. H. Our results show better performance is achieved with values of α and β other than those that correspond to label smoothing with $\gamma = 0.1$, which is the commonly used value for the strength coefficient (Vaswani et al., 2017; Edunov et al., 2018). Moreover, the relationship between model entropy and evaluation performance is quite strong, following the same trend for all values of α, which suggests tuning a model for a specific entropy rather than α, β may be a better method in practice. We discuss trends in §4.3.

4.2 Abstractive Summarization

We fine-tune BART (Lewis et al., 2019) on the CNN/DailyMail abstractive summarization task (Hermann et al., 2015) with regularizer D_{J_α}. Data pre-processing and other hyperparameter settings follow Lewis et al. (2019). Results in Table 3 show that optimal values of ROUGE-L, the evaluation metric, can be achieved by regularizing with D_{J_α} for different values of α. Notably, the entropy is virtually the same for the models that achieve top performance, demonstrating the closer relationship of performance with model entropy than with α, discussed further in §4.3.

4.3 Significance of α and Model Entropy

We look at the strength of the relationship between the evaluation metrics and both α and the model's entropy. Figure 3 shows a quadratic relationship between model entropy and BLEU. On the other hand, the relationship between α (coloring of points) and BLEU is not an obvious one; the best performing models are regularized with various values of α.

As correlation only tells us about linear relationships, we report mutual information to measure the strength of the relationship between α, model entropy, and BLEU. Mutual information shows the proportion of entropy of a variable that is "explained" by another and is often used as a generalized correlation measure i.e. for nonlinear relationships (Song et al., 2012). We see in Figure 4 that model entropy has a much stronger relationship with BLEU than α. Indeed, the normalized mutual information (NMI) between α and BLEU is ≈ 0.05 compared to ≈ 0.25 between model entropy and BLEU—implying that any flavor of entropy regularization can lead to similar performance.

While the relationship between α and BLEU is weak, it is still statistically significant. Some evidence of this exists in Figure 3 where a closer examination reveals that each level of α has a

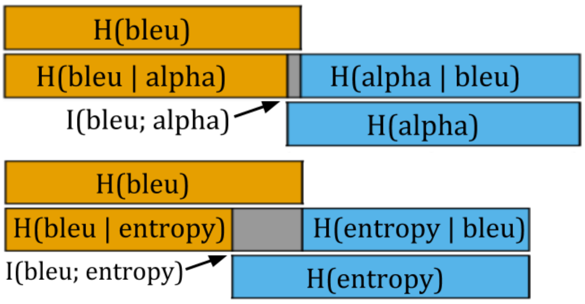

Figure 4: Entropy $\mathrm{H}(\cdot)$, Conditional Entropy $\mathrm{H}(\cdot \mid \cdot)$ and Mutual Information $\mathrm{I}(\cdot;\cdot)$ for BLEU with alpha (α) and model entropy, respectively. Model entropy explains a greater portion of variability in BLEU than α does. Non-parametric estimates are used for all values (Beirlant et al., 1997). Data from IWSLT'14 De-En Transformer models.

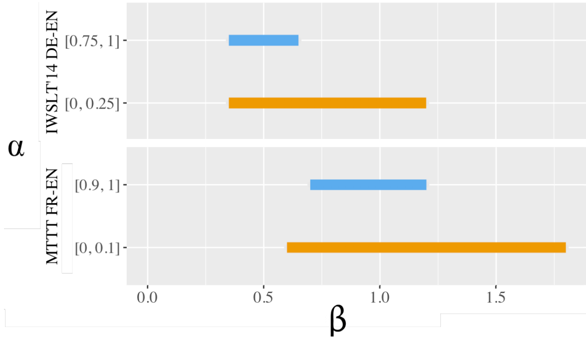

Figure 5: Each line represents the range of β for which D_{J_α} leads to performance within $\approx 1\%$ (< 0.4 BLEU) of the best overall model for the task. For α close to 1, (which includes label smoothing) D_{J_α} has a smaller optimal range, and so is harder to tune.

similar quadratic trend, albeit with a different offset. Specifically, the performance of models trained with D_{J_α} for $\alpha \in [0.75, 1]$ (which includes label smoothing) starts to degrade at lower levels of entropy than models trained with D_{J_α} for $\alpha \in [0, 0.25]$ (confidence penalty). As quantitative validation of this observation, we (i) run a conditional independence test to see whether BLEU and α are conditionally independent given model entropy and (ii) look at the range of β for which D_{J_α} leads to good performance for different α.

Conditional Independence. Conditional independence of α and BLEU implies that the value of α does not supply any additional information about the value BLEU given model entropy, i.e. α does not matter when using the regularizer D_{J_α}. We use a Monte Carlo permutation test where the null hypothesis is that no relationship between α and BLEU exists.[11] However, this test rejects the null hypothesis with p-value < 0.05, supporting the alternate hypothesis that α and BLEU are *not* conditionally independent.

Tuning β. On the tasks for which we trained > 60 models, we take the subset of models for which performance is within $\approx 1\%$ (< 0.4 BLEU) of the best overall model. We then look at the range of β used with the regularizer D_{J_α} for these models. The range of β that meets the above criterion is much larger for α close to 0 than for for α close to 1 (see Figure 5). We contend this implies that D_{J_α} is easier to tune (i.e. it is more robust) for $\alpha \approx 0$ while for $\alpha \approx 1$, D_{J_α} is relatively sensitive to β.

[11] The underlying distributions of random variables are assumed to be Gaussian. See Legendre (2000) for more details.

	Sparsity Threshold e^{-10}	Sparsity Threshold e^{-15}
Label Smoothing D_{J_1}	$38\% \pm 0.01\%$	$0.0\% \pm$ 5e-5%
Confidence Penalty D_{J_0}	$54\% \pm$ 5e-3%	$0.7\% \pm$ 4e-4%

Table 4: Percentage of words with $< \epsilon$ probability mass at different values of ϵ (below which we consider as functionally 0) for models trained with D_{J_1} and D_{J_0}. To control for entropy, all models used in the calculation have entropy within the same 1%.

4.4 Sparsity

We take a subset of models trained with regularizers D_{J_0} and D_{J_1} and examine the sparsity of $p_{\boldsymbol{\theta}}$. Results in Table 4 support our formal analysis regarding the sparsity of D_{J_0} and D_{J_1} in §3.2; D_{J_1} steeply penalizes sparsity while D_{J_α} for $\alpha < 1$ allows words to be assigned probability ≈ 0.

4.5 Sequence Likelihood

We look at how the probability of the reference solution changes with model entropy. Previously, Müller et al. (2019) found that training with label smoothing rather than hard targets resulted in worse log-likelihoods for the reference sequence. However, this observation has not been put in the appropriate context; since the most probable sequence (relative to $p_{\boldsymbol{\theta}}$) is returned, a more relevant calculation would be that of the overall *ranking* of the reference sequence or of the log-likelihood of the reference sequence *relative* to the most probable sequence. Since the former is typically impossible to calculate exactly due to the size of $\mathcal{Y}$, we approximate it by looking at the average rank of each word in the reference sequence.

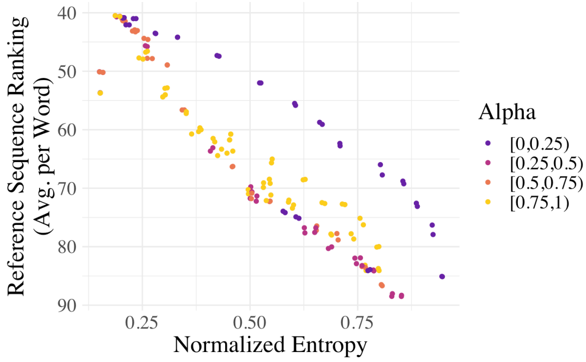

Figure 6: Average ranking in p_θ of words in the reference sequence on the test set for IWSLT '14 (De-En) plotted against model entropy. Overall trends show a decrease in the ranking of the reference for models with more entropy regularization. Notably, the reference is generally ranked higher for models regularized with D_{J_α} for $\alpha \approx 0$ than for $\alpha \in [0.25, 1)$.

	Normalized Entropy [0,0.3]	[0.3,0.45]	[0.45,0.6]	[0.6,0.75]	[0.75,1]
k = 1	32.8	33.6	33.6	33.4	33.1
k = 3	34.2	34.9	35.0	34.8	34.5
k = 5	34.3	35.0	35.1	34.9	34.6
k = 10	34.4	35.1	35.2	35.0	34.7
Random Sampling	26.0	15.2	11.3	5.1	1.5

Figure 7: BLEU scores on IWSLT'14 De-En validation set with the convolutional architecture by decoding strategy and model entropy. The trend in BLEU stays remarkably constant for beam search as the beam width is varied. Performance declines drastically for higher entropy models when random sampling is used. Color reflects average distance from baseline model.

In Figure 6, we see that higher entropy models, i.e. models with more entropy regularization, generally rank the reference sequence lower than low entropy models. Despite this, we know that models with entropy regularization still tend to perform better in terms of downstream evaluation metrics (see Figure 3), a phenomenon similar to that seen by Müller et al. (2019). Another behavior that Figure 6 exhibits is the tendency for the reference sequence to be ranked relatively low across *all* models, supporting findings seen in Ott et al. (2018).

In Figure 8, we see that while lower entropy models will place more probability mass on the reference sequence, the reference sequence is still far from the most probable. For models with higher entropy, the ratio of log-likelihoods of the reference sequence to the highest-probability sequence is larger, showing that in this context, higher entropy correlates with higher *relative* log-likelihood.

4.6 Decoding

In language generation tasks, estimated distributions are fed to decoding algorithms to create sequence predictions. To fully understand how model entropy affects performance for these tasks, we must explore the potential interactions between model entropy and the decoding strategy.

Chorowski and Jaitly (2017) saw that with label smoothing, prediction accuracy improved and so using a wider beam during beam search did not give further improvements; however, our results suggest otherwise. As shown in Figure 7, the trend in BLEU vs. model entropy stays remarkably constant for beam search as the beam width is varied, including for greedy decoding (beam size of 1). Perhaps unsurprisingly though, higher entropy is detrimental to the performance of decoding with random sampling (with temperature $T = 1$). However, this phenomenon could potentially be remedied by decreasing the temperature during decoding, a common practice for avoiding sampling from the tail of the distribution (Kirkpatrick et al., 1983).

5 Discussion

Our experiments show entropy regularization has a number of beneficial effects on natural language generation models. Foremost, we observe that low-entropy predictions, which are more aligned with the empirical distribution (Figure 8), lead to worst performance on evaluation metrics than the higher entropy predictions produced by more regularized models (Figure 3). Further, our findings show that closely approximating the empirical distribution appears to be at ends with a well calibrated model, i.e. a model $p_\theta(y \mid \mathbf{x})$ that matches the true, underlying probabilities $p(y \mid \mathbf{x})$.[12] Namely, Figure 8 demonstrates an increase in the log-likelihood of the reference sequence relative to the highest probability sequence for higher entropy models.

Decoding. Overconfident predictions inhibit the ability to recover after a poor choice of words during decoding; Chorowski and Jaitly (2017) suggest that higher-entropy models p_θ, like the ones resulting from regularization with label smoothing, would alleviate this problem. Results throughout this paper support this hypothesis not just for la-

[12] This is different than the empirical distribution $\tilde{p}(\mathbf{y} \mid \mathbf{x})$.

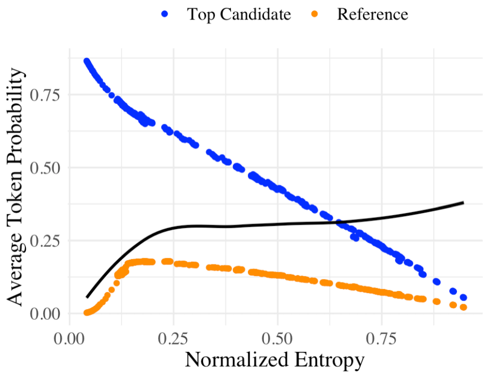

Figure 8: Average word probability of the reference and the most probable (for beam search with $k = 5$) sequences plotted against model entropy on test set for IWSLT '14 (De-En). The black line is a smoothed estimate of their ratio.

bel smoothing, but for the D_{J_α} family of entropy regularizers as well.

Choosing the baseline distribution. Throughout this work, we use the uniform distribution u as our baseline distribution for the regularizer D_{J_α}. However, one could also use some other distribution defined over the vocabulary such as the unigram (Chorowski and Jaitly, 2017) or a function of word embedding distance with the target word (Kumar and Tsvetkov, 2019; Li et al., 2020). Both have proven to be more effective than u when used with label smoothing and the confidence penalty. However, using distributions other than u with D_{J_α} leads to indirect forms of entropy regularization. Specifically, the mathematical relationship to entropy regularization becomes more convoluted. Therefore, we leave the application of GER to other distributions as a topic for future work.

6 Related Work

Entropy regularization has a long history in reinforcement learning (Williams and Peng, 1991; Mnih et al., 2016; Fox et al., 2016; Haarnoja et al., 2018) where it has provided substantial improvements in exploration. Such methods have since been adapted for supervised learning where they have proven to be reliable forms of regularization for various probabilistic modeling tasks (Grandvalet and Bengio, 2005; Smith and Eisner, 2007).

More recently, interpolating between exclusive and inclusive KL divergences has been explored in NMT by Xiao et al. (2019). However, this method was used for the objective function (i.e. between $\tilde{p}$ and p_θ) and not as a regularization technique (i.e. between a baseline distribution q and p_θ). Li et al. (2020) construct a baseline distribution q as a function of word embedding distances to to use in place of the uniform distribution u in the label smoothing equation. This work is complementary to ours, as q can similarly be used in place of u with GER. Finally, our work is closest to that of Müller et al. (2019), which attempts to find the circumstances under which label smoothing has a positive effect on model performance. However, they do not explore entropy regularization on the whole nor do they attempt to provide an explanation for why label smoothing works. We attempt to answer the "why" question through a quantitative analysis of label smoothing and empirical exploration of the relationship between model entropy and performance.

7 Conclusion

We discuss the properties of generalized entropy regularization and provide empirical results on two language generation tasks. We find entropy regularization leads to improvements over baseline systems on evaluation metrics for all values of the parameter α with our regularizer D_{J_α}. Theoretical and empirical evidence show label smoothing adds undesirable constraints to the model and is the hardest to tune of the regularizers tested. We therefore advocate the use of alternate forms of entropy regularization for language generation tasks.

References

J. Beirlant, E. Dudewicz, L. Gyor, and E. C. Meulen. 1997. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences*, 6.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, pages 1171–1179.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit[3]: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 76–86.

Jan Chorowski and Navdeep Jaitly. 2017. Towards better decoding and language model integration in sequence to sequence models. In *Proceedings of Interspeech*.

Kevin Duh. 2018. The multitarget TED talks task. `http://www.cs.jhu.edu/~kevinduh/a/multitarget-tedtalks/`.

Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. 2018. Understanding back-translation at scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 489–500, Brussels, Belgium. Association for Computational Linguistics.

Roy Fox, Ari Pakman, and Naftali Tishby. 2016. Taming the noise in reinforcement learning via soft updates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 202–211.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1243–1252.

Yves Grandvalet and Yoshua Bengio. 2005. Semi-supervised learning by entropy minimization. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 529–536. MIT Press.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *International Conference on Learning Representations*.

Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. 1983. Optimization by simulated annealing. *Science*, 220(4598):671–680.

Sachin Kumar and Yulia Tsvetkov. 2019. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In *International Conference on Learning Representations*.

Pierre Legendre. 2000. Comparison of permutation methods for the partial correlation and partial mantel tests. *Journal of Statistical Computation and Simulation*, 67(1):37–73.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *arXiv:1910.13461*.

Zuchao Li, Rui Wang, Kehai Chen, Masso Utiyama, Eiichiro Sumita, Zhuosheng Zhang, and Hai Zhao. 2020. Data-dependent Gaussian prior objective for language generation. In *International Conference on Learning Representations*.

André Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1614–1623.

André F. T. Martins, Noah A. Smith, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2011. Structured sparsity in structured prediction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1500–1511.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA. PMLR.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems 32*, pages 4696–4705.

Vlad Niculae, André Martins, Mathieu Blondel, and Claire Cardie. 2018. SparseMAP: Differentiable sparse structured inference. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 3799–3808.

Frank Nielsen and Sylvain Boltz. 2011. The Burbea-Rao and Bhattacharyya centroids. *IEEE Transactions on Information Theory*, 57(8):5455–5466.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In *International Conference on Machine Learning*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, pages 48–53.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *Proceedings of the International Conference on Learning Representations*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 1715–1725.

David A. Smith and Jason Eisner. 2007. Bootstrapping feature-rich dependency parsers with entropic priors. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 667–677, Prague, Czech Republic. Association for Computational Linguistics.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Proceedings of the 25th International Conference on Neural Information Processing Systems*, volume 2, pages 2951–2959.

Lin Song, Peter Langfelder, and Steve Horvath. 2012. Comparison of co-expression measures: mutual information, correlation, and model based indices. *BMC Bioinformatics*, 13(1):328.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N. Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2tensor for neural machine translation. *CoRR*.

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 *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008.

Ronald Williams and Jing Peng. 1991. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3:241–268.

Fengshun Xiao, Yingting Wu, Hai Zhao, Rui Wang, and Shu Jiang. 2019. Dual skew divergence loss for neural machine translation.

A α-Jensen to KL

For reference, we repeat eq. (9), the definition of the skew Jensen divergence for some strictly convex function $G : \Omega \to \mathbb{R}$ and probability distributions p, q:

$$\mathrm{J}_{\alpha,G}(p \,||\, q) := \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)G(p) + \alpha G(q) - \nabla G((1-\alpha)p + \alpha q)\Big)$$

We can rewrite the α-Jensen divergence with convex generator function G in terms of the Bregman divergence

$$\begin{aligned}
\mathrm{J}_{\alpha,G}(p \,||\, q) &= \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)G(p) + \alpha G(q) - G((1-\alpha)p + \alpha q)\Big) \\
&= \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)G(p) + \alpha G(q) - G((1-\alpha)p + \alpha q) \\
&\quad \underbrace{-\alpha(1-\alpha)\langle p-q, \nabla G((1-\alpha)p+\alpha q)\rangle - \alpha(1-\alpha)\langle q-p, \nabla G((1-\alpha)p+\alpha q)\rangle}_{=\,0,\ \text{note } p-q \text{ in first inner product and } q-p \text{ in second}}\Big) \\
&= \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)G(p) + \alpha G(q) - G((1-\alpha)p + \alpha q) \\
&\quad - \underbrace{(1-\alpha)\langle \alpha(p-q), \nabla G((1-\alpha)p+\alpha q)\rangle}_{\text{bring } \alpha \text{ inside the inner product since } b\langle v,w\rangle = \langle b\cdot v, w\rangle} \\
&\quad - \underbrace{\alpha\langle (1-\alpha)(q-p), \nabla G((1-\alpha)p+\alpha q)\rangle}_{\text{likewise, bring } (1-\alpha) \text{ inside the inner product}}\Big) \\
&= \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)G(p) + \alpha G(q) - G((1-\alpha)p + \alpha q) \\
&\quad - (1-\alpha)\underbrace{\langle p - ((1-\alpha)p+\alpha q), \nabla G((1-\alpha)p+\alpha q)\rangle}_{\text{distribute } \alpha \text{ and rewrite}} \\
&\quad - \alpha\underbrace{\langle q - ((1-\alpha)p+\alpha q), \nabla G((1-\alpha)p+\alpha q)\rangle}_{\text{distribute } (1-\alpha) \text{ and rewrite}}\Big) \\
&= \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)[G(p) - G((1-\alpha)p+\alpha q) - \langle p - ((1-\alpha)p+\alpha q), \nabla G((1-\alpha)p+\alpha q)\rangle] \\
&\quad \underbrace{+\alpha[G(q) - G((1-\alpha)p+\alpha q) - \langle q - ((1-\alpha)p+\alpha q), \nabla G((1-\alpha)p+\alpha q)\rangle]}_{\text{regroup terms based on multiplier (either } \alpha \text{ or } 1-\alpha\text{) so we can rewrite equation as two Bregman divergences}}\Big) \\
&= \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)D_G(p, (1-\alpha)p+\alpha q) + \alpha D_G(q, (1-\alpha)p+\alpha q)\Big)
\end{aligned}$$

We look at the behavior of $D_{\mathrm{J}_{\alpha,G}}(p \,||\, q)$ as $\alpha \to \{0, 1\}$

$$\begin{aligned}
&\lim_{\alpha\to 0} \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)D_G(p, (1-\alpha)p+\alpha q) + \alpha D_G(q, (1-\alpha)p+\alpha q)\Big) \\
&= \lim_{\alpha\to 0} \frac{1}{\alpha(1-\alpha)}\Big((1-\alpha)\underbrace{D_G(p,p)}_{=\,0} + \alpha D_G(q,p)\Big) \\
&= \lim_{\alpha\to 0} \frac{1}{(1-\alpha)} D_G(q,p) \\
&= D_G(q,p)
\end{aligned}$$

If we expand $D_G(q, p)$ using our generator function $G(p) = \sum_i p(i) \log p(i)$, we get

$$
\begin{aligned}
&D_G(q, p) \\
&= \sum_i q(i) \log q(i) - \sum_i p(i) \log p(i) - \langle q - p, log(p) - 1 \rangle \\
&= \sum_i q(i) \log q(i) - \sum_i p(i) \log p(i) + \sum_i p(i) \log p(i) - \sum_i q(i) \log p(i) \\
&\qquad \underbrace{- \sum_i p(i) + \sum_i q(i)}_{\text{=0 since } q, p \text{ are both probability distributions summing to 1}} \\
&= \sum_i q(i) \log q(i) - \sum_i q(i) \log p(i) \\
&= \mathrm{KL}(q \mid\mid p)
\end{aligned}
$$

Similarly, we can show $\lim_{\alpha \to 1} \mathrm{J}_\alpha = \mathrm{KL}(p \mid\mid q)$

B α-Jensen to Jensen–Shannon

The proof that the α-Jensen divergence is proportional to the Jensen–Shannon divergence is quite straightforward. If we evaluate $\mathrm{J}_\alpha(p \mid\mid q)$ at $G = x \log x$ and $\alpha = \frac{1}{2}$

$$
\begin{aligned}
\mathrm{J}_\alpha(p \mid\mid q) &= \frac{1}{\alpha(1-\alpha)} \Big((1-\alpha) G(p) + \alpha G(q) - G((1-\alpha)p + \alpha q) \Big) \\
&= 4 \cdot \Big(\frac{1}{2} G(p) + \frac{1}{2} G(q) - G(\frac{1}{2} p + \frac{1}{2} q) \Big) \\
&= 4 \cdot \Big(\frac{1}{2} p \log(p) + \frac{1}{2} q \log(q) - \frac{p+q}{2} \log(\frac{p+q}{2}) \Big) \\
&= 4 \cdot \Big(\frac{1}{2} (p \log(p) - p \log(\frac{p+q}{2})) + \frac{1}{2} (q \log(q) - q \log(\frac{p+q}{2})) \Big) \\
&= 4 \cdot \Big(\frac{1}{2} \mathrm{KL}(p \mid\mid \frac{p+q}{2}) + \frac{1}{2} \mathrm{KL}(p \mid\mid \frac{p+q}{2}) \Big) \\
&= 4 \cdot \mathrm{JS}(p \mid\mid q)
\end{aligned}
$$

C Label Smoothing

For the case that $\alpha \to 1$, $p = u$, and $q = p_{\boldsymbol{\theta}}$, we have

$$
\begin{aligned}
\lim_{\alpha \to 1} \mathrm{J}_\alpha(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x})) &= \mathrm{KL}(u \mid\mid p_{\boldsymbol{\theta}}) \\
&= \sum_{y \in Y} u(y) \log \frac{u(y)}{p_{\boldsymbol{\theta}}(y \mid \mathbf{x})} \\
&= \sum_{y \in Y} u(y) \log u(y) - \sum_{y \in Y} u(y) \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \\
&= \log |Y| - \sum_{y \in Y} u(y) \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \\
&= - \sum_{y \in Y} u(y) \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) + N
\end{aligned}
$$

When $\mathrm{J}_1(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x}))$ is used as a regularizer for maximum likelihood training, we get the loss function

$$\begin{aligned}
&\mathcal{L}(\theta) \\
&= \mathrm{KL}\left(\tilde{p}(\cdot \mid \mathbf{x}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x})\right) + \beta \cdot \mathrm{KL}\left(u(\cdot) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x})\right) \\
&= \underbrace{-\sum_{y \in Y} \Big(\tilde{p}(y \mid \mathbf{x}) + \beta \cdot u(y)\Big) \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x})}_{\textit{unnormalized label-smoothed cross-entropy loss}} + N
\end{aligned}$$

where N is constant with respect to $\boldsymbol{\theta}$.

D Classical Entropy Regularization

For the case that $\alpha \to 0$, $p = u$, and $q = p_{\boldsymbol{\theta}}$, we have

$$\begin{aligned}
\lim_{\alpha \to 0} \mathrm{J}_\alpha(q \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x})) &= \mathrm{KL}(p_{\boldsymbol{\theta}} \mid\mid u) \\
&= \sum_{y \in Y} p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \log \frac{p_{\boldsymbol{\theta}}(y \mid \mathbf{x})}{u(y)} \\
&= \sum_{y \in Y} p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \log p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) - \sum_{y \in Y} p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \log u(y) \\
&= -\mathrm{H}(p_{\boldsymbol{\theta}}(y \mid \mathbf{x})) - \log \frac{1}{|Y|} \sum_{y \in Y} p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \\
&= -\mathrm{H}(p_{\boldsymbol{\theta}}(y \mid \mathbf{x})) - \log \frac{1}{|Y|} \\
&= -\mathrm{H}(p_{\boldsymbol{\theta}}(y \mid \mathbf{x})) - N
\end{aligned}$$

When $\mathrm{J}_0(u \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x}))$ is used as a regularizer for maximum likelihood training, we get the loss function

$$\begin{aligned}
&\mathcal{L}(\theta) \\
&= \mathrm{KL}\left(\tilde{p}(\cdot \mid \mathbf{x}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x})\right) + \beta \cdot \mathrm{KL}\left(p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x} \mid\mid u(\cdot))\right) \\
&= \underbrace{\mathrm{KL}\left(\tilde{p}(\cdot \mid \mathbf{x}) \mid\mid p_{\boldsymbol{\theta}}(\cdot \mid \mathbf{x})\right) - \beta \cdot H(p_{\boldsymbol{\theta}}(y \mid \mathbf{x}))}_{\textit{confidence penalty cross-entropy loss}} + \beta \cdot N
\end{aligned}$$

E Bounds of J_α

Upper bound of J_α:

First note that $\mathrm{KL}(p \mid\mid q)$ is convex in q when $\mathrm{supp}(p) \subseteq \mathrm{supp}(q)$, which must be true since $(1-\alpha)u+\alpha p$ has support everywhere both p and u do for $\alpha \in (0, 1)$. Therefore for $\alpha \in (0, 1)$

$$\begin{aligned}
\mathrm{KL}(p \mid\mid (1-\alpha)u + \alpha p) &\leq (1-\alpha)\mathrm{KL}(p \mid\mid u) + \alpha \mathrm{KL}(p \mid\mid p) \\
&= (1-\alpha)\mathrm{KL}(p \mid\mid u)
\end{aligned}$$

similarly,

$$\mathrm{KL}(u \mid\mid (1-\alpha)u + \alpha p) \leq \alpha \mathrm{KL}(u \mid\mid p)$$

We then have:

$$\begin{aligned}
\mathrm{J}_\alpha(u \,||\, p) &= \frac{\alpha}{\alpha(1-\alpha)}\mathrm{KL}(p \,||\, (1-\alpha)u + \alpha p) + \frac{1-\alpha}{\alpha(1-\alpha)}\mathrm{KL}(u \,||\, (1-\alpha)u + \alpha p) \\
&\leq \frac{\alpha(1-\alpha)}{\alpha(1-\alpha)}\mathrm{KL}(p \,||\, u) + \frac{\alpha(1-\alpha)}{\alpha(1-\alpha)}\mathrm{KL}(u \,||\, p) \\
&= \mathrm{KL}(p \,||\, u) + \mathrm{KL}(u \,||\, p)
\end{aligned}$$

Lower bound of J_α:

The bound from below is trivial given the definition of J_α, however, it can more easily be seen by expressing J_α as the sum of KL divergences as above:

$$\mathrm{J}_\alpha(u \,||\, p) = \frac{\alpha}{\alpha(1-\alpha)}\mathrm{KL}(p \,||\, (1-\alpha)u + \alpha p) + \frac{1-\alpha}{\alpha(1-\alpha)}\mathrm{KL}(u \,||\, (1-\alpha)u + \alpha p)$$

Since $\alpha > 0$ and necessarily $\mathrm{KL}(\cdot \,||\, \cdot) \geq 0$, the lower bound $0 \leq \mathrm{J}_\alpha(u \,||\, p)$ follows.

F No Sparse Solution for J_1

Proof. By definition, for any distribution p over a vocabulary Y:

$$\mathrm{J}_1(u \,||\, p) = -\frac{1}{|Y|}\sum_{y \in Y} \log p(y) + \log|Y| \tag{13}$$

Thus, if $p_{\boldsymbol{\theta}}(y \mid \mathbf{x}) \to 0$ for some $y \in Y$ and some $\mathbf{x} \in \mathcal{X}$, we have $\mathrm{J}_1(u \,||\, p) = \mathrm{KL}(u \,||\, p_{\boldsymbol{\theta}}) \to \infty$. This means that label smoothing enforces $p_{\boldsymbol{\theta}}$ has support everywhere $u > 0$, i.e. over all words $y \in Y$. For any $\alpha < 1$, J_α allows for sparse solutions since $\lim_{x \to 0} x \log x = 0$. $\square$

G Data Pre-Processing and Hyperparameter Settings

For training with convolutional architectures we set hyperparameters, e.g. dropout, learning rate, etc., following Gehring et al. (2017). On IWSLT'14 and MTTT tasks, we follow the recommended Transformer settings for IWSLT'14 in fairseq.[13] Hyperparameters for models trained on the WMT task are set following version 3 of the Tensor2Tensor toolkit (Vaswani et al., 2018). We use byte-pair encoding (BPE; Sennrich et al. 2016) for all languages. Vocabulary sizes for WMT and IWSLT'14 are set from recommendations for the respective tasks in fairseq; for the MTTT tasks, vocabulary sizes are tuned on models with standard label smoothing regularization.

Similarly, the CNN/DailyMail data set is pre-processed and uses BPE following the same steps as (Lewis et al., 2019). Hyperparameters are the same as for their model fine-tuned on CNN/DailyMail. Details are available on the fairseq website.[14]

H Additional Results

[13] https://github.com/pytorch/fairseq/tree/master/examples/translation
[14] https://github.com/pytorch/fairseq/blob/master/examples/bart/README.cnn.md

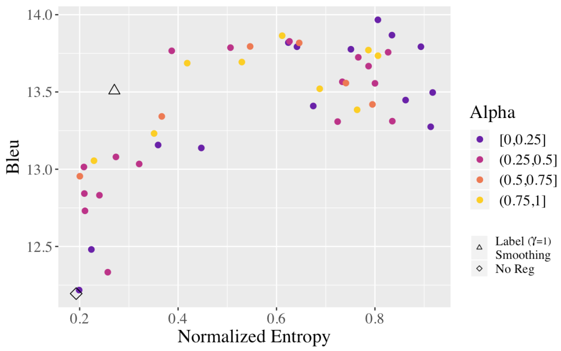

Figure 9: Model entropy vs. BLEU (validation set) on Multitarget Ted Talks Task Japanese to English (Ja-En) using a Transformer architecture; see Figure 3 for additional information.

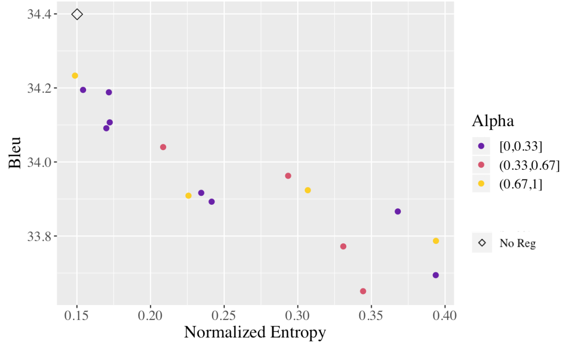

Figure 10: Model entropy vs. BLEU (validation set) on IWSLT'14 German to English (De-En) using a convolutional architecture and generator function $G(z) = ||z||_2^2$; see Figure 3 for additional information.

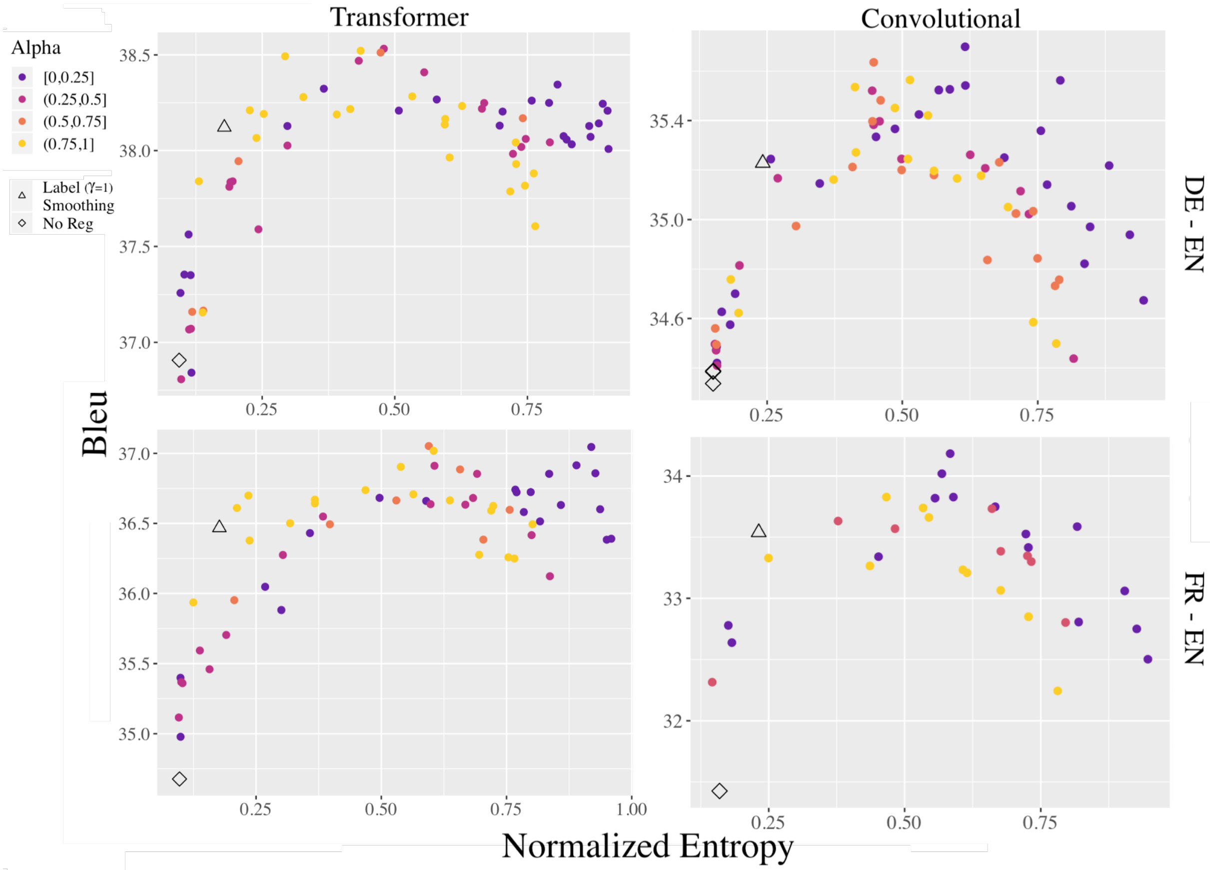

Figure 11: Model entropy vs. BLEU (validation set) on IWSLT'14 German to English (De-En) and Multitarget Ted Talks Task French to English (Fr-En) using Transformer and convolutional architectures; see Figure 3 for additional information.

	WMT'14 De-En (Convolutional)				MTTT Ja-En (Transformer)			
	α	β	$\mathrm{H}(p_{\boldsymbol{\theta}})$	BLEU	α	β	$\mathrm{H}(p_{\boldsymbol{\theta}})$	BLEU
No Regularization	-	0	0.15	33.2	-	0	0.19	13.8
Label Smoothing D_{J_1} $(\gamma = 0.1)$	1	0.11	0.25	34.1 **+0.9**	1	0.11	0.27	15.2 **+1.4**
Label Smoothing D_{J_1}	1	0.35	0.42	34.6 **+1.4**	1	0.96	0.61	16.2 **+2.4**
Confidence Penalty D_{J_0}	0	0.60	0.79	34.7 **+1.5**	0	0.65	0.80	15.9 **+2.1**
GER D_{J_α}	0.75	0.60	0.45	34.8 **+1.6**	0.42	1.7	0.76	15.9 **+2.1**

Table 5: Test BLEU for IWSLT'14 German-to-English using a convolutional architecture and for MTTT Japanese-to-English using a Transformer architecture; see Table 2 for additional information.