Studying the Effect of Generalized Entropy Regularization on Hierarchical Story Generation

Anya Trivedi
aht324@nyu.edu

Vishal Kumar
vk2161@nyu.edu

Mahima Gaur
mg6827@nyu.edu

Abstract

A common problem faced by Probabilistic Natural Language Generation models is that they are prone to produce "peaky" distributions, where a large proportion of the weights is associated with only a few candidates. This is a tell-tale sign of overfitting, and such models are known to have low entropy. An intuitive way to combat this overfitting is to force models to have higher entropy (or less peaks in their distribution), an approach known as Entropy Regularization. Common regularization techniques such as label smoothing and confidence penalties can be thought of as specific cases of a family of Entropy Regularizers. In this project, we aim to study the effect of entropy regularizers on Hierarchical Story Generation, where stories are built based on generated prompts. Our method uses a hierarchical sequence model to generate stories, and we study the effect of the degree of entropy regularization on model perplexity and BLEU Scores. We find that both BLEU Scores and Model Perplexity increase with Model Entropy, indicating that increasing the entropy of NLG models can help combat overfitting, and lead to enhanced model performance.

1 Introduction

A common problem faced by probabilistic Natural Language Generation (NLG) models is their proneness to produce “peaky" distributions, or distributions that overconfidently place most of their probability mass on a few candidates. This is a sign of overfitting, and such models are known to have low entropy. An intuitive way to then combat such overfitting would be to train models that have higher entropy, and thus generalize better. In their seminal paper, (Meister et al., 2020) introduce a family of Generalized Entropy Regularizers (GER), which penalise models with low entropy. Specific members of this family include Label Smoothing (Szegedy et al., 2015) and Confidence Penalties (Pereyra et al., 2017), both of which allow some probability mass to be “reserved" for unseen words, improving the generalization power of a model. Since they tend to bring the approximate empirical distribution of a model close to the uniform distribution (the distribution of maximum entropy), these methods can be thought of methods of increasing entropy.

While the authors of (Meister et al., 2020) study the effect of entropy regularizers on two common NLG tasks- Machine Translation and Abstract Summarization, we aim to study the effect of this family of regularization techniques on another common language generation task- Story Generation. NLG tasks often lead to sparse distributions (for example, we would want the model to assign very low probability to grammatically incorrect statements), and (Meister et al., 2020) argues that label smoothing does not allow for sparse distributions, and recommend using other regularizers in its place. We aim to study whether entropy regularization would improve model performance on Hierarchical Story Generation (Fan et al., 2018), where a *story* is generated by conditioning on a small description describing the story content, called a *prompt*. We find that the unique requirements of grounded, consistent stories make this task an interesting case study.

We find that the BLEU scores of the generated stories increase with increasing model entropy. We also find that model perplexity increases with model entropy, proving that entropy regularization helps improve model performance.

2 Methodology

2.1 Problem Statement

Entropy Regularization is a regularization technique which favours high-entropy models, by bringing the approximate empirical distribution of a model close to the uniform distribution. This helps

combat overfitting in the form of peaky, overconfident distributions, with skewed weight distributions. This form of regularization is proven to improve the performance of language models for Neural Machine Translation and Abstractive Summarization tasks (Meister et al., 2020). Another interesting NLG task is Hierarchical Story generation, where a story is generated from a plot(Fan et al., 2018). The nature of Story Generatiion tasks is that they are required to model large range dependencies, and must generate consistent, non-repetitve plots, which may benefit from entropy regularization. We aim to answer the question- does Entropy Regularization affect model performance for Hierarchical Story Generation?

2.2 Approach

Language models model the conditional probability $p_\theta(\mathbf{y}|\mathbf{x})$- assigning probability to target sequence $\mathbf{y}$ given a source sequence x. For NLG tasks, $p_\theta(\mathbf{y}|\mathbf{x})$ is a neural network with parameters θ. Commonly, the network is trained to approximate $\tilde{p}(\mathbf{y}|\mathbf{x})$, the empirical distribution of the data. One way of estimating θ is to minimize the KL-divergence between the empirical distribution and the model, with an additional regularization term to prevent overfitting. Thus, the loss function to be optimized is described as

$$L(\theta) + \beta R(\theta) \tag{1}$$

where

$$L(\theta) = \mathrm{KL}(\tilde{p}||p_\theta) \tag{2}$$
$$= H(\tilde{p}, p_\theta) - H(\tilde{p}) \tag{3}$$

Here $H(\tilde{p}, p_\theta)$ is the cross entropy loss, $H(\tilde{p})$ is the constant with respect to θ, β is the "strength" (regularization) coefficient and $R(\theta)$ is the regularization function.

Generalized Entropy Regularization can be used to penalize low-entropy language models, which overfit by placing most probability mass on a few candidates (Chorowski and Jaitly, 2016). This leads to models that generate repetitive or unrelated text, which harms the model's generalization ability (Holtzman et al., 2019). GER is based on the skew Jensen family of divergences $J_{\alpha,G}$ (Nielsen and Boltz, 2011)and thus may be generalized through the choice of generator function G and weighted density α, mathematically defined below:

$$J_{\alpha,G}(q||p_\theta) = \frac{1}{\alpha(1-\alpha)}((1-\alpha)G(q)+ \alpha G(p_\theta) - G((1-\alpha)q + \alpha p_\theta)) \tag{4}$$

Two common instances of GER are label smoothing (Szegedy et al., 2015) and Confidence penalties (Pereyra et al., 2017). Both of these methods aim to bring the approximate empirical distribution of a model close to the uniform distribution (the distribution of maximum entropy), and can thus be thought of as methods to increase model entropy. Based on the definition of GER, label smoothing occurs when $\alpha \to 1$, while confidence penalties occur when $\alpha \to 0$, when entropy is used as the Generator function,

$$G(p_\theta) = -H(p_\theta) = \sum_{y \epsilon \Upsilon} p_\theta(\mathbf{y}) \log p_\theta(\mathbf{y}) \tag{5}$$

This project aims to study the effect of entropy regularization on Hierarchical Story Generation, an NLG task that generates a story conditioned on a small description of the story content, called a prompt(Fan et al., 2018). We generate a set of candidate entropy regularizers, by changing the values of α and β, and study how these changes affect model performance.

Following the approach of (Meister et al., 2020), evaluation is done using perplexity values, which indicate how fluently the model can produce the correct next word given the preceding words, and BLEU scores(Papineni et al., 2002), a common scoring method for NLG tasks, generated using the SacreBLEU framework (Post, 2018). We analyse the relationship between the entropy of the model and the corresponding BLEU scores. We also calculate the normalized entropy of the model, and compare that with the corresponding perplexity to better aid our understanding on the relationship between them.

3 Experiments

3.1 Setup

Dataset We use story generation data[1] from Reddit's /rWritingPrompts thread, where users reply to prompts by stories they have written. We then study the effect of various Entropy-based Regularizers on pre-trained hierarchical sequence models for story generation. We use the code released with the original work of (Meister et al., 2020), which is built on top of fairseq (Ott et al., 2019), a sequence modeling toolkit.This plug-in provides a way to compute the loss given a model and a batch

[1] https://github.com/pytorch/fairseq/tree/main/examples/stories

of data which will allows us to add entropy regularization to the loss calculation. We study how model perplexity changes with varying values of α, the weighted density and β, the regularization coefficient.

Baseline In order to have a comprehensive evaluation about how model performance changes with different values of α, and β, we use a **non-regularized baseline**, by setting β to be 0. This allows us to observe how the entropy affects the perplexity of the model, as well as whether it results in improved BLEU scores.

Experimental Environment and Hyperparameters Experiments are run on NYU's Greene HPC Cluster, modifying the code [2] released with the original paper (Meister et al., 2020). We use a convolutional neural network with self attention, and follow (Fan et al., 2018)'s method of hierarchical sequence-to-sequence model usage. We set the learning rate to 0.25 and the gradient clip threshold to 0.1, and use a plateau LR Scheduler. We vary the value of α between [0,0.25,0.50,0.75, 1] and β as [0.35, 0.7]. The model is trained on a training set using early stopping where model entropy and perplexity is calculated, using these hyperparameters. The trained model is then tested on an unseen test set and the corresponding BLEU scores are calculated using the SacreBleu framework (Post, 2018), where decoding is performed with length-normalized beam search with a beam size of 1. Stories are generated from our models using a top-k random sampling scheme, to overcome repetitive text/phrase generation (Shao et al., 2017). We randomly sample from the k = 10 most likely next words. Then future words are generated based on previously generated words.

3.2 Results

We calculate the Normalized Entropy $\hat{H}$, and the BLEU scores of the model with different values of α and β. For brevity, we refer to the regularized function at value α using Entropy as the generator function as R_α. The results are depicted in Table 1. We also calculate the Perplexity based on the Negative Log Loss NLL (2^{NLL}) of the model with different values of α and β, and compare it to the model entropy in Figure 1

[2] https://github.com/rycolab/entropyRegularization

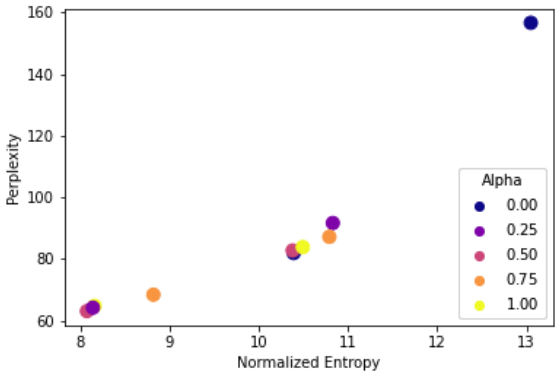

Figure 1: Relationship between Model Entropy and Perplexity. Perplexity increases with Entropy, discouraging overfitting. No obvious relationship between α and perplexity is seen.

3.3 Analysis

Perplexity of a model is directly proportional to entropy, so we expect to see a rise in model perplexity with entropy, as seen in Figure 1. While high entropy models thus produce models with large perplexities, this helps prevent overfitting, by allowing some probability mass distribution to be reserved for unseen word sequences. However, no obvious relationship between α and model perplexity is observed.

We also see an increase in BLEU scores with model entropy for most values of α, which align with the observations of (Meister et al., 2020). However, we note that the increase in BLEU scores is not proportional to the increase in entropy. Also, we see a decrease in values for α = 0.75. Further, we see that the least increase compared to baseline performance is when α =1 (label smoothing), which agrees with evidence label smoothing adds undesirable constraints to the model(Meister et al., 2020).

4 Related Work

Entropy Regularization NLG models often produce peaky (low entropy) distributions that decrease model generalization abilities (Chorowski and Jaitly, 2016). An intuitive way to combat this behaviour is to use regularization techniques that reward high entropy models which generalize better.(Meister et al., 2020) use two common regularizers to introduce such entropy based regularizers-label smoothing and confidence penalties. Label smoothing is a technique to smooth hard target distributions by using a weighted average between

	α	β	$\hat{H}$	BLEU	
No Regularization	-	-	5.567	16.0	
Confidence Penalty R_0	0	0.35	10.832	16.2	**+0.2**
	0	0.70	13.05	16.3	**+0.3**
GER $R_{0.25}$	0.25	0.35	8.077	16.7	**+0.7**
	0.25	0.70	10.793	16.4	**+0.4**
GER $R_{0.50}$	0.50	0.35	8.163	16.5	**+0.5**
	0.50	0.70	10.393	16.4	**+0.4**
GER $R_{0.75}$	0.75	0.35	8.144	15.5	**-0.5**
	0.75	0.70	10.832	15.4	**-0.6**
Label Smoothing R_1	1	0.35	8.163	16.1	**+0.1**
	1	0.70	10.495	16.0	**0**

Table 1: Experimental Results. $\hat{H}$ and the BLEU scores of the model with different values of α and β. For brevity, we refer to the regularized model at value α using Entropy as the generator function as R_α. We see an improvement of scores for all values of α except 0.75, with the least improvement from baseline observed for label smoothing.

hard targets of a data set and uniform distribution over labels (Szegedy et al., 2015). What this effectively does is "reserve" some probability mass for unseen words, improving the generalization power of a model. By bringing the approximate empirical distribution of a model close to the uniform distribution (the distribution of maximum entropy), label smoothing can be thought of as a method of increasing entropy. In a similar manner, Confidence penalties also aim to penalize low entropy models, and both these methods can be thought of as regularization techniques. GER is based on the skew Jensen family of divergences $J_{\alpha,G}$ (Nielsen and Boltz, 2011)and thus may be generalized through the choice of generator function G and weighted density α. (Meister et al., 2020) find that entropy regularization leads to improvements over baseline systems for all values of α, for Neural Machine Translation and Abstractive Summarization tasks.

Hierarchical Story Generation Hierarchical Story Generation generates stories by first generating a *prompt*- a small description describing the content of the story, and then generating a *story* by conditioning on this prompt (Fan et al., 2018). Prompts are generated using convolutional language models (Dauphin et al., 2016), then sequence-to-sequence models (Gehring et al., 2017) are used to generate the story from the prompt. Story generation is different from other NLG tasks because of their requirement for modelling very large range dependencies.This is because stories are required to be consistent, as well as a require a high level plot, implying that simple word-to-word generation as done by other sequence-to-sequence models may prove inefficient. By conditioning the story based on the generated prompt, hierarchical story generation models can allow a more grounded, consistent plot (Fan et al., 2018).

5 Conclusion

Generalized Entropy Regularizers are a family of regularizers based on the skew-Jensen family of divergences, which are used to combat overfitting by rewarding high entropy models. They are motivated by the fact that Natural Language Generation tasks often form low-entropy models (having "peaky" distributions) that overfit and generalize poorly.

This project empirically studies the relationship between Entropy Regularization and Model performance for Hierarchical Story Generation, where stories are generated from prompts. We find that entropy regularization leads to improvements over baseline systems on evaluation metrics for most values of the parameter α with the regularizer R_α, where α is the weight term for the regularizer. We empirically prove that perplexity increases with model entropy, however, no direct relationship between α and model perplexity is observed. Further, we also observe that the least increase compared to baseline performance is when α =1 (label smoothing), which agrees with evidence label smoothing adds undesirable constraints to the model(Meister et al., 2020). Thus, our findings advocate for the use of entropy regularization in Natural Language Generation tasks.

References

Jan Chorowski and Navdeep Jaitly. 2016. Towards better decoding and language model integration in sequence to sequence models. *CoRR*, abs/1612.02695.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *CoRR*, abs/1612.08083.

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. *CoRR*, abs/1805.04833.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. *CoRR*, abs/1705.03122.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Clara Meister, Elizabeth Salesky, and Ryan Cotterell. 2020. Generalized entropy regularization or: There's nothing special about label smoothing. *CoRR*, abs/2005.00820.

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

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. *CoRR*, abs/1904.01038.

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 of the Association for Computational Linguistics*, pages 311–318.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *CoRR*, abs/1701.06548.

Matt Post. 2018. A call for clarity in reporting BLEU scores. *CoRR*, abs/1804.08771.

Louis Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating long and diverse responses with neural conversation models. *CoRR*, abs/1701.03185.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567.