Methods for Countering Posterior Collapse in Sentence Variational Autoencoders

Jonathan Mitnik
10911197
j.mitnik@gmail.com
University of Amsterdam

Pieter de Marez Oyens
10002403
oyenspieter@gmail.com
University of Amsterdam

1 Introduction

Generating believable, human-level language and sentences is a feat that AI, to this day, struggles to accomplish. Sentences generated by model are often clunky, repetitive and are usually void of context or coherence (Knight). The act of generating sentences natural language generation has come a long way since the first language models (Bengio et al.) and recent supervised models such as GTP-2 (Radford et al.) generate stories that resemble humanlike syntactical structures. However still, it remains a challenge to generate language especially if the goal is to apply stylistic conditions on text generation. RNNLMs (Recurrent Neural Network Language Models) (Mikolov et al., 2011) has inspired other authors to push boundaries in this area and have led to the creation of Sentence VAEs (Variational Auto-encoders) with architectures that make it possible to generate text (Bowman et al., 2016). By adding variational inference, these sentences VAEs can construct distributed semantic latent spaces from which one can generate new and unseen texts or interpolate between sentences.

Using VAEs we are able to capture deep latent representations of the text we put into our models. In theory these representation are capable of capturing high-level syntactic and semantic properties such as sentiment, style and subject. This would greatly further the capabilities of AI to convincingly generate text that can be presented as humanlike. These techniques however do have their difficulties and suffer from a problem called *posterior collapse* (Subramanian et al., 2018)(Razavi et al., 2019).

To bring more light to this phenomenon we briefly explain the problem and discuss possible solutions for mitigating this problem. To this end we implement a deep-generative text VAE and three methods to possibly solve or mitigate the posterior collapse problem. These methods are: *word dropout*, *freebits* and μ-forcing.

Our contributions are as follows: we re-implement existing ideas of papers and show their validity. Furthermore, we provide explanation about, and summarize existing methods for preventing posterior collapse. These different implementations are compared to each other and to a baseline RNNLM model. We find that the methods of preventing posterior collapse do indeed improve the quantitative and qualitative performance of the VAE compared to the RNNLM model and the "vanilla" VAE implementation. Where word dropout was the least effective and freebits and μ-forcing the most in terms of prevent posterior collapse.

2 Related Work

We base our research on the seminal paper *'Auto Encoding Variational Bayes'* (Kingma and Welling, 2014) in which Kingma and Welling propose Variational Auto-Encoders. For text generation they are further expanded upon in *'Generating Sentences from a Continuous Space'* (Bowman et al., 2016). However, one of the main challenges of these models is the event of posterior collapse. Multiple methods have been proposed: (Bowman et al., 2016) proposed *word-dropout* and *kl-annealing*, (Kingma et al., 2017) a method called *freebits* while (Liu et al., 2020) have a method called *μ-forcing*. All works hold validity and aside from kl-annealing, these are the methods that will be implemented and explored in this paper.

Finally, other techniques that have been constructed to reduce posterior collapse includes: *Skip-VAE* (Dieng et al., 2019), *Independent δ-VAE* (Razavi et al., 2019) and *minimum desired rate* (Pelsmaeker and Aziz, 2019), among others.

3 Approach

3.1 Task

The goal for sentence VAEs is "to learn global latent representations of sentence content" (Bowman et al., 2016). During training however, these models turn the variational posterior $q(z|x)$ into the prior $p(z)$. This problem is observed when

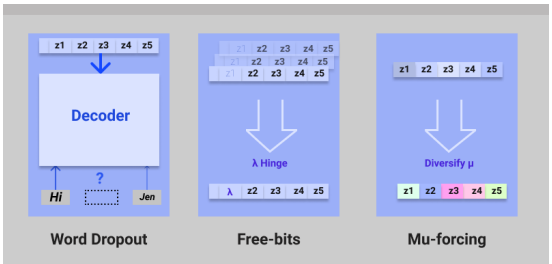

Figure 1: Collapse Prevention Methods Visualised

auto regressive decoders, such as RNNLMs, are to strong. Intuitively this means that it is easier for the decoder to not make use of the latent variable z and optimization of the KL loss in the ELBO term results in a loss which is almost always 0. This problem takes away the possibility to capture any meaningful latent representation of text. Then the task at hand is to implement aforementioned techniques to reduce posterior collapse and enable the model to learn these latent representations.

3.2 Models and Baselines

As a baseline for our generative models we implement a RNNLM which is not able to capture latent variables but still can generate non-random sentences. It has a GRU-layer as the recurrent unit, instead of the classical LSTM-structure. This base model will be optimized by minimize the negative log likelihood loss function. This serves as a stepping stone towards a VAE, in which the encoder and decoder will be an RNNLMs. Optimizing the VAE by means of the ELBO, which is defined as: $\mathcal{L}(x) = \mathbb{E}_{q(z|x)}\Big[\log p(x|z)\Big] - KL\Big(q(z|x)||p(z)\Big)$. This model, without any counter measures, will suffers from posterior collapse where the $KL\Big(q(z|x)||p(z)\Big) \approx 0$. Three methods are implemented to prevent this problem, namely: word-dropout, μ-forcing and freebits.

3.3 Posterior Collapse Prevention Methods

A visual representation for each technique can be seen in in Figure 1.

3.3.1 Word Dropout

Originally proposed in (Bowman et al., 2016), word dropout attempts to weaken the decoder. The decoder predicts each word conditioned on the ground-truth previous word. Bowman et al. use this fact and remove the conditioning information during learning. This is done by replacing ground truth words, randomly, with an ⟨UNK⟩ token. Each word has a probability of dropping-out, also called the *dropout rate*(w.d.r.). When w.d.r $= 1$, each word will be replaced, when w.d.r. $= 0$ no words will be replaced.

3.3.2 Freebits

Instead of directly influencing the decoder, Kingma et al. propose to modify the objective function with what they call *freebits*. A constraint is put on the minimum amount of information per group of latent variables. The method involves dividing the latent dimensions into K groups and create a fixed minimum loss for the second term in the objective function. This minimum loss is a hyper parameter λ. The modified objective function looks as follows: $\tilde{\mathcal{L}}(x;\lambda) = \mathbb{E}_{q(z|x;\lambda)}\Big[\log p(x|z)\Big] - \sum_{j=1}^{K} max\Big\{\lambda, KL\Big(q(z|x;\lambda)||p(z)\Big)\Big\}$

3.3.3 μ-forcing

Our final method is called μ-*forcing*, proposed by Liu et al.. It also opts to change the original objective function by adding an additional loss term to the original. It puts extra constraints on the posteriors of latent variables. The intuition is that different data points should have different latent representations, so the authors put an additional constraint on the μ of $q(z|x)$ and forces the VAE to learn discriminated latent representations. Practically, the following additional term is added to the original objective function: $\mathcal{L}_\mu = max\Big\{0, \beta - \frac{1}{2N}\sum_{n=1}^{N}(\mu^{(n)} - \bar{\mu})^\intercal(\mu^{(n)} - \bar{\mu})\Big\}$. The loss is computed from a batch of data points where β is a hyper-parameter and acts as a margin. $\mu^{(n)}$ denotes the μ vector of the n-th sample and $\bar{\mu}$ is the mean of the μ vectors in that batch. The result is that the sample variance of μ to be controlled on the level of β.

4 Experiments and Results

4.1 Data

To run our experiments we use The Penn Treebank (Taylor et al., 2003) dataset as our corpus. It contains annotated text from the Wall Street Journal, Brown Corpus and Switchboard. Words are POS tagged and sentences are structured as trees, with a total 43948 sentences. These attributes are ignored however and only the sentences themselves are used to train the model. A tokenizer is used to tokenize each word. The vocabulary size of

our corpus is set to 10K. Words that are processed and fall don't fit in this range are tokenized to ⟨ UNK ⟩ tokens. Furthermore, we reserve the following special tokens: ⟨BOS⟩, ⟨EOS⟩, ⟨PAD⟩. Finally, the data set is split into the following ratios: 91%/4%/5% for the training-, validation- and test-set respectively.

4.2 Metrics

The negative log likelihood was measured for the RNNLM and the multi-sample negative log likelihood for the VAEs. From this the perplexity was calculated for all models. Additionally, the KL-, μ- and ELBO-loss were measured for every VAE model.

4.3 Hyper parameters and Training

The following hyper parameters were considered: w.d.r., λ and β. Other hyper parameters like batch size were outside the scope of this project. To find the optimal hyper parameters, grid search was done over the following sets: w.d.r. $\in \{0, 0.5, 1\}$, $\lambda \in \{-1, 0.25, 0.5, 1, 2\}$, $\beta \in \{0, 2, 3, 5\}$. For each combination of hyper parameters, we trained a model on the training set for 5 epochs, with a batch size of 16. It was shown empirically that training beyond 5 epochs resulted in a plateauing effect on the perplexity after which training is halted. Per epoch each model is validated every 50 iterations on the whole validation set. The model with the lowest negative log likelihood loss was considered best and saved[1]. Using this method it was found that the following best hyper parameters when combined are: w.d.r. $= 0$, $\lambda = 2$, $\mu = 2$, this model will be called "Best VAE". When singled out, the the best hyper parameters are w.d.r. $= 0$, $\lambda = 2$, $\mu = 5$. In Figure 3 we can see what effect different training schema has on the posterior and it μ.

4.4 Quantitative Results

Each model was tested once on the test-set with optimal hyper parameters The results are presented in Table 1. Results for VAEs in which word-dropout was active, were omitted. They performed worse then the baseline model (the negative log likelihood and perplexity were in all cases higher), and during hyper parameter selection the model with no word-dropout always got selected.

[1]For complete details and implementation of the code, we refer to our code repository.

	NLLL	Perp	ELBO	KL	μ
RNNLM	108.12	75.21	-	-	-
VAE	107.66	71.89	107.70	0.05	0.43
VAE-λ	101.62	56.49	125.07	23.45	55.95
VAE-μ	107.57	71.90	108.08	0.50	5.21
VAE-Best	101.53	56.44	124.67	23.13	51.98

Table 1: Final Test Results

4.5 Qualitative Results

For we feed each model the beginning of the following sentence: "*the chairman proposed that next fall a new treaty*". This sentence is tokenized and fed to a model; the trained models then finish the sentence using a sampling-approach, where each subsequent token is sampled from a categorical distribution after applying a softmax to a models' output likelihoods, and dividing this vector by a *temperature* of 1. This will be done iteratively, where each token will be used as foundation for the next token until either a maximum of 15 tokens have been generated, or ⟨EOS⟩ has been sampled. These sentences can be found in appendix A.

5 Discussion

Figure 3 shows us the decreasing KL values for training. The figure shows that the KL term is further from zero for both the best posterior methods independently, and used jointly. These values seem to converge very quickly, indicating that The best model is one with no word dropout. These findings are corroborated by Liu et al., in which they found that word dropout did not positively influence results. A possible reason for this is the size of the used data set. In the paper by Bowman et al., the second (*Books* corpus) data set used is orders of magnitude bigger (around 80m sentences) and the w.d.r. is around 75% while the vocabulary sizes are the same. Using a larger data set present the model with more opportunities to learn and more words to be seen as to not always have them disappear, which would also result in a failure to learn. To build support for this hypothesis, we would need to replicate the same experiments while using the same data-set as Bowman used.

The other methods fared better, both the mu-forcing method and the Free-bits approach increased KL Loss while decreasing perplexity. Furthermore, during hyper-parameter selection, the model with the highest available freebits-parameter was selected. The performance improvement of adding the β parameter from the mu-forcing

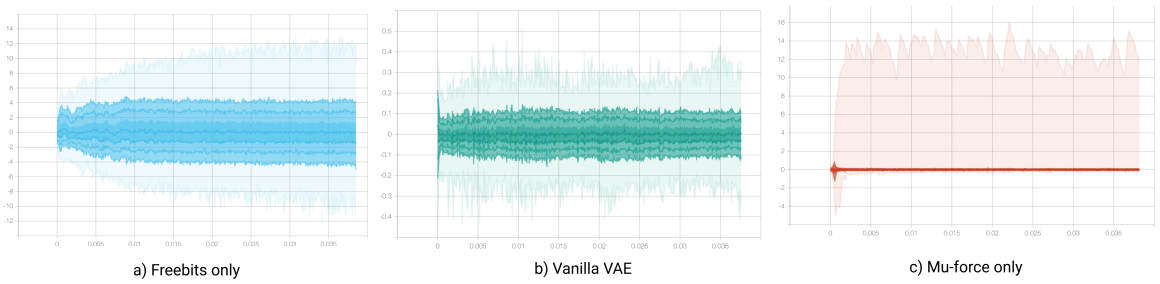

Figure 2: Distribution of posterior mean values as time passes. Note that in this figure, the three figures have a different scale, due to the effects these methods implicitly have on the magnitude of the means. These values seem to remain constant across the training, however, even for mu-force.

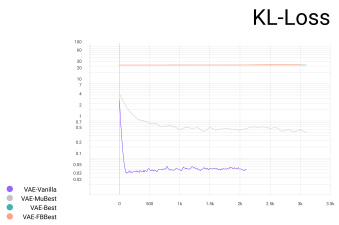

Figure 3: Posterior Collapse Visualised, projected on a log scale.

approach improved the performance marginally, which is reflected in the test results in Table 1. The KL-loss is higher for models using the mu-force approach but perplexity does not decrease in general: thus, only freebits seems to increase model performance significantly while also increasing the KL Loss at the same time. Similarily to the word-dropout approach, mu-force might show better values if trained on the same dataset as the authors Liu et al. reportedly used, which consists of two datasets. However, due to the stark difference in KL measure between the KL values achieved in this paper once the model plateaus, the most likely cause points to a flawed implementation. Furthermore it would be expected in Figure 2, to see the mu of at least the third model eventually spread out more, yet these values remain constant. Regardless, all the models beat out the baseline in perplexity.

Notably, we analyze that for both the RNNLM and the regular VAE the perplexity is lower than similar model results presented by Bowman et al., whereas the negative log likelihood is marginally worse. A naive explanation is that the splitting of data is done differently. In the paper authors mention that they use the standard train-test split for the data set. It may be that their training set is smaller and on which the model fails to generalize, and thus performing worse during testing. Another explanation may be that the specific structure of our models vary. Elements such as hidden layer and embedding size all play a roll and without further knowledge of their specific model and training setup, we can only speculate.

When looking at the qualitative results, one of the most noticeable details is the obvious improvement of semantic meaning of these models. Where the collapsed vanilla model generally produced topics that seemed incoherent, the counter-balanced methods contained notions like 'treaty manager', legality, and even interesting clauses which in context can make sense, such as a 'treaty being struck'. It is not a simple task to distinguish between the freebits and mu-forcing based predictions, where both produce realistic-seeming sentences. Empirically, the freebits sentences have a more syntactically correct structure, while the mu-forcing contains just as much semantic structure but prove less readable. Regardless, these techniques seem to support a more coherent language generation model.

Finally we can conclude that methods for preventing posterior collapse have not only quantitatively verifiable results, but also qualitatively. We observe the KL-loss not dropping to 0 and can say that these methods prevent, in general, the posterior likelihood from collapsing. Qualitatively, sentences generated by VAEs without collapsed posteriors, using simple sampling methods, produce sentences that convey subject and style. Recommendations for future research include testing the models on different language generation tasks and observe metrics when the model is trained on a larger, more diverse dataset.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A Neural Probabilistic Language Model. page 19.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating Sentences from a Continuous Space. *arXiv:1511.06349 [cs]*. ArXiv: 1511.06349.

Adji B. Dieng, Yoon Kim, Alexander M. Rush, and David M. Blei. 2019. Avoiding Latent Variable Collapse With Generative Skip Models. *arXiv:1807.04863 [cs, stat]*. ArXiv: 1807.04863.

Diederik P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. 2017. Improved Variational Inference with Inverse Autoregressive Flow. *arXiv:1606.04934 [cs, stat]*. ArXiv: 1606.04934.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*. ArXiv: 1312.6114.

Will Knight. AI's Language Problem. Library Catalog: www.technologyreview.com.

Dayiheng Liu, Xu Yang, Feng He, Yuanyuan Chen, and Jiancheng Lv. 2020. mu-Forcing: Training Variational Recurrent Autoencoders for Text Generation. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 19(1):1–17. ArXiv: 1905.10072.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, J.H. Cernocky, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. pages 5528–5531.

Tom Pelsmaeker and Wilker Aziz. 2019. Effective Estimation of Deep Generative Language Models. *arXiv:1904.08194 [cs]*. ArXiv: 1904.08194.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. page 24.

Ali Razavi, Aäron van den Oord, Ben Poole, and Oriol Vinyals. 2019. Preventing Posterior Collapse with delta-VAEs. *arXiv:1901.03416 [cs, stat]*. ArXiv: 1901.03416.

Sandeep Subramanian, Sai Rajeswar Mudumba, Alessandro Sordoni, Adam Trischler, Aaron C Courville, and Chris Pal. 2018. Towards Text Generation with Adversarially Learned Neural Outlines. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7551–7563. Curran Associates, Inc.

Ann Taylor, Mitchell Marcus, and Beatrice Santorini. 2003. The Penn Treebank: An Overview. In Nancy Ide, Jean Véronis, and Anne Abeillé, editors, *Treebanks*, volume 20, pages 5–22. Springer Netherlands, Dordrecht. Series Title: Text, Speech and Language Technology.

A Generated sentences

In this section, a number of samples of sentences are generated for different models based on how they would complete the sentence *the chairman proposed that next fall a new treaty*".

A.1 Vanilla VAE

- "the chairman proposed that next fall a new treaty **, for the french electronics , loan rows of** "
- "the chairman proposed that next fall a new treaty **" do nonetheless around risk for that other network**"
- "the chairman proposed that next fall a new treaty **on small part of a country where " call**"
- "the chairman proposed that next fall a new treaty **board went on a fairly period used and then they**"

A.2 Best VAE

- "the chairman proposed that next fall a new treaty **last week it will concentrate on legal and homeowners together**"
- "the chairman proposed that next fall a new treaty **entered the deal with its legal**"
- "the chairman proposed that next fall a new treaty , **which struck will include a serious supervision over**"
- "the chairman proposed that next fall a new treaty **that led about the collapse of the** "

A.3 Freebits only

- " the chairman proposed that next fall a new treaty **forecast the increase in the lower** ."
- the chairman proposed that next fall a new treaty **through feb. 1 to boeing now widened more than a**"
- "the chairman proposed that next fall a new treaty **that he marks in improperly out that fiscal 1990 social**"

- ”the chairman proposed that next fall a new treaty **world accounts for mr. lawson ’s resignation ’s ghost who**”

A.4 Mu-forcing only

- ” the chairman proposed that next fall a new treaty **to require restrictions on the existing real treaty of the**.”
- ” the chairman proposed that next fall a new treaty **by failure to a new quota owners of the** ”
- ”the chairman proposed that next fall a new treaty **manager at between midnight and 1994 .**”
- ”the chairman proposed that next fall a new treaty , **improving the business disputes , which links to**”