Challenges with Variational Autoencoders for Text

Ankit Vani
New York University
ankit.vani@nyu.edu

Vighnesh Birodkar
New York University
vighneshbirodkar@nyu.edu

Abstract

We study variational autoencoders for text data to build a generative model that can be used to conditionally generate text. We introduce a mutual information criterion to encourage the model to put semantic information into the latent representation, and compare its efficacy with other tricks explored in literature such as KL divergence cost annealing and word dropout. We compare the log-likelihood lower-bounds on held-out data using variational autoencoders with the log-likelihoods on an unconditional language model. We notice our models quickly learn to exploit the grammatical redundancies in our dataset, but it is more challenging to encode semantic information in the latent representation.

1 Introduction

Recent advances in latent variable models, optimization techniques and data availability have paved the way for better and more complex generative models. Variational autoencoder is one such generative model, which is also theoretically pleasing. Learning latent representations for data enables the use of latent variables for auxiliary supervised tasks, where they can act as simpler surrogates for high-dimensional data. Variational autoencoders have been shown to model image data and exhibit the property of producing realistic images while interpolating in the latent space (Kingma and Welling [2013]). Bowman et al. [2015] showed this interpolation to hold for generating text as well, however they relied on huge and diverse datasets and various tricks to get the model to work.

In this study, we explore additional strategies of parameterizing the variational approximation and encouraging the model to learn semantic latent representations, with the ultimate goal of text generation.

2 Related work

Variational autoencoders Kingma and Welling [2013] have been successful at approximating the generative model for high-dimensional data. Instead of learning a deterministic mapping from the data space to the latent space, variational autoencoders can learn a representation that is more robust to noise by estimating a distribution over the latent representation.

Bowman et al. [2015] proposed a variational autoencoder for text data. They showed that their model can generate grammatically and semantically valid sentences as we interpolate from one sentence to another in the latent space. To force the latent representation to encode semantic information, they used KL cost annealing, going from a standard autoencoder to a variational autoencoder over the first few steps of training. They also use a high word dropout at the decoder, to encourage the decoder to use the latent representation for semantic cues. In our work, we explore another way to encourage the model to encode information in the latent variable, which is to maximize the mutual information between the latent representation and the output of the generative model.

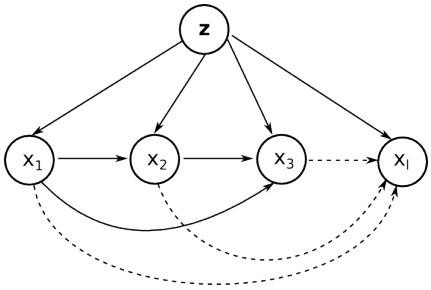


Figure 1: The generative model for the data

Miao et al. [2015] proposed a model for variational inference for text data, where they model text using a bag of words. Unlike this work, we consider the sequential information in text, and implement our generative model in a way that it can be used to generate entire sentences or paragraphs.

3 The generative model

For text, given an idea that a sentence is expressing, there still exist multiple sequences of words which convey the same meaning. Therefore a language model is typically asked to predict the next word given the preceding sequence of words. If $x_1, x_2, x_3 \ldots, x_l$ are the words in a sentence of length l, the distribution of a language model factorizes as

$$p(\mathbf{x}) = \prod_{i=1}^{l} p(x_i \mid x_1, \ldots, x_{i-1}) \tag{1}$$

The generative model we consider uses a latent variable $\mathbf{z}$ to condition on, and models

$$p(\mathbf{x} \mid \mathbf{z}) = \prod_{i=1}^{l} p(x_i \mid x_1, \ldots, x_{i-1}, \mathbf{z}) \tag{2}$$

We assume that our text data $\mathbf{x}$ is generated from latent variables $\mathbf{z}$ by the probability distribution $p(\mathbf{x} \mid \mathbf{z})$ as depicted in the generative model shown in Figure 1. We expect $\mathbf{z}$ to capture semantics such as the context and the sentiment of the text.

We have

$$\begin{aligned}
\log p(\mathbf{x}) &= \log \int_{\mathbf{z}} p(\mathbf{x}, \mathbf{z}) \\
&= \log \int_{\mathbf{z}} q(\mathbf{z}) \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z})} \\
&\geq \int_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{x}, \mathbf{z})}{q(\mathbf{z})} && \text{(Jensen's inequality)} \\
&= \int_{\mathbf{z}} q(\mathbf{z}) \log p(\mathbf{x}, \mathbf{z}) - q(\mathbf{z}) \log q(\mathbf{z}) \\
&= \int_{\mathbf{z}} q(\mathbf{z}) \log p(\mathbf{z}) + q(\mathbf{z}) \log p(\mathbf{x} \mid \mathbf{z}) - q(\mathbf{z}) \log q(\mathbf{z}) \\
&= \mathrm{E}_{\mathbf{z} \sim q(\mathbf{z})} \left[\log p(\mathbf{x} \mid \mathbf{z}) \right] - \mathcal{D}(q(\mathbf{z}) \| p(\mathbf{z}))
\end{aligned}$$

Here, q is the variational approximation, and the above lower bound is tight when it matches the true posterior. This happens when the KL divergence $\mathcal{D}(q(\mathbf{z}) \| p(\mathbf{z} \mid \mathbf{x}))$ becomes zero. Since q can be anything and we are modeling $p(\mathbf{x})$, we can choose to construct q using $\mathbf{x}$. In our implementations, we use the 'encoder' of the variational autoencoder to model the variational approximation, and the 'decoder' implements the generative model.

In summary, we have

$$\log p(\mathbf{x}) - \mathcal{D}[q(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z} \mid \mathbf{x})] = \mathrm{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})}[\log(p(\mathbf{x} \mid \mathbf{z}))] - \mathcal{D}[q(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})]$$
$$\implies \log p(\mathbf{x}) \geq \mathrm{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})}[\log(p(\mathbf{x} \mid \mathbf{z}))] - \mathcal{D}[q(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})] = -\mathcal{L}_{VAE} \quad (3)$$

We fix the prior $p(\mathbf{z})$ to be a Gaussian with zero mean and identity covariance.

Given a finite distribution of samples $\mathbf{X}$, the variational autoencoder optimization problem to maximize the lower bound on the log-likelihood of the data can be phrased as

$$\text{maximize} \quad \mathrm{E}_{\mathbf{x} \sim \mathbf{X}} \Big[\mathrm{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})}[\log(p(\mathbf{x} \mid \mathbf{z}))] - \mathcal{D}[q(\mathbf{z} \mid \mathbf{x}) \| p(\mathbf{z})] \Big] \quad (4)$$

We ask the encoder implementing the variational approximation to predict the sufficient statistics of $q(\mathbf{z} \mid \mathbf{x})$. Since we model this distribution as a Gaussian, these are the mean and the diagonal co-variance. To backpropagate through the $\mathrm{E}_{\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})}[\cdot]$ term, Kingma and Welling [2013] suggested using the reparameterization trick. If $\mu(\mathbf{x})$ and $\mathbf{\Sigma}(\mathbf{x})$ are the predicted mean and covariance for $\mathbf{x}$, the sampled $\mathbf{z}$ is

$$\mathbf{z} = \mu(\mathbf{x}) + \mathbf{\Sigma}(\mathbf{x})^{1/2} * \epsilon \quad (5)$$

where $\epsilon \sim \mathcal{N}(0, I)$.

4 Mutual information criterion

Since the loss surface in deep neural networks is very non-convex, it is possible for the model to get stuck in bad local optima or bad parameter regions. When training the variational autoencoder naively, the model tends to chase the low-hanging fruit of making the KL divergence term zero. Once this happens, the generative model learns to become an unconditional language model, maximizing the likelihood of the training data without considering the latent vector $\mathbf{z}$.

To encourage the model to encode information in $\mathbf{z}$, we consider the objective of maximizing the mutual information between the latent variable $\mathbf{z}$ and the generated text $G(\mathbf{x} \mid \mathbf{z})$. We can consider $G(\mathbf{x} \mid \mathbf{z})$ to be a random variable drawn from $p(\mathbf{x} \mid \mathbf{z})$. The mutual information $\mathcal{I}(\mathbf{z}; G(\mathbf{x} \mid \mathbf{z}))$ is given by

$$\mathcal{I}(\mathbf{z}; G(\mathbf{x} \mid \mathbf{z})) = \mathcal{H}(G(\mathbf{x} \mid \mathbf{z})) - \mathcal{H}(G(\mathbf{x} \mid \mathbf{z}) \mid \mathbf{z}) = \mathcal{H}(\mathbf{z}) - \mathcal{H}(\mathbf{z} \mid G(\mathbf{x} \mid \mathbf{z}))$$

which is the reduction in the uncertainty of $G(\mathbf{x} \mid \mathbf{z})$ when $\mathbf{z}$ is observed, or vice versa.

Thus, we can say

$$\begin{aligned}
\mathcal{I}(\mathbf{z}; G(\mathbf{x} \mid \mathbf{z})) &= \mathcal{H}(\mathbf{z}) - \mathcal{H}(\mathbf{z} \mid G(\mathbf{x} \mid \mathbf{z})) \\
&= \mathcal{H}(\mathbf{z}) - \mathrm{E}_{\mathbf{x}' \sim p(\mathbf{x}|\mathbf{z})} \left[\mathcal{H}(\mathbf{z} \mid \mathbf{x}') \right] \\
&= \mathcal{H}(\mathbf{z}) + \mathrm{E}_{\mathbf{x}' \sim p(\mathbf{x}|\mathbf{z})} \left[\mathrm{E}_{\mathbf{z}' \sim p(\mathbf{z}|\mathbf{x}')} \left[\log p(\mathbf{z}' \mid \mathbf{x}') \right] \right] \\
&= \mathcal{H}(\mathbf{z}) + \mathrm{E}_{\mathbf{x}' \sim p(\mathbf{x}|\mathbf{z})} \left[\mathrm{E}_{\mathbf{z}' \sim p(\mathbf{z}|\mathbf{x}')} \left[\log \frac{p(\mathbf{z}' \mid \mathbf{x}') q_{MI}(\mathbf{z}' \mid \mathbf{x}')}{q_{MI}(\mathbf{z}' \mid \mathbf{x}')} \right] \right] \\
&= \mathcal{H}(\mathbf{z}) + \mathrm{E}_{\mathbf{x}' \sim p(\mathbf{x}|\mathbf{z})} \left[\mathcal{D}(p(\mathbf{z}' \mid \mathbf{x}') || q_{MI}(\mathbf{z}' \mid \mathbf{x}')) + \mathrm{E}_{\mathbf{z}' \sim p(\mathbf{z}|\mathbf{x}')} \left[\log q_{MI}(\mathbf{z}' \mid \mathbf{x}') \right] \right] \\
&\geq \mathcal{H}(\mathbf{z}) + \mathrm{E}_{\mathbf{x}' \sim p(\mathbf{x}|\mathbf{z})} \left[\mathrm{E}_{\mathbf{z}' \sim p(\mathbf{z}|\mathbf{x}')} \left[\log q_{MI}(\mathbf{z}' \mid \mathbf{x}') \right] \right] \quad (6)
\end{aligned}$$

Here, we introduced a variational approximation $q_{MI}(\mathbf{z}' \mid \mathbf{x}')$, that approximates the posterior distribution of $\mathbf{z}'$, given a generated $\mathbf{x}'$. The variational mutual information bound is tight when the KL divergence $\mathcal{D}(p(\mathbf{z}' \mid \mathbf{x}') || q_{MI}(\mathbf{z}' \mid \mathbf{x}'))$ is zero. Note that this is different from the variational approximation q implemented by the encoder of the variational autoencoder, which acts on the discrete input sentence. Since gradients cannot be backpropagated through discrete inputs, this variational approximation takes as input the hidden states of the decoder.

Chen et al. [2016] show that under suitable regularity conditions,

$$\mathrm{E}_{x \sim X, y \sim Y|x, x' \sim X|y} \left[f(x', y) \right] = \mathrm{E}_{x \sim X, y \sim Y|x} \left[f(x, y) \right] \quad (7)$$

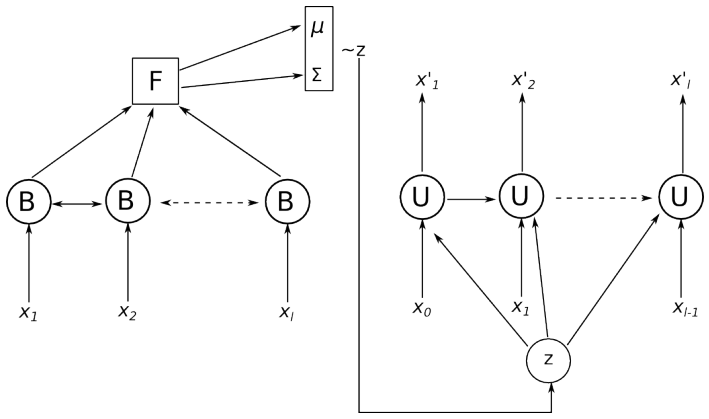


Figure 2: The encoder-decoder architecture. B represents the GRU cell(s) at a timestep in a bidirectional RNN, whereas U represents a GRU cell(s) at a timestep in a unidirectional RNN. x'_i denotes the predicted value for word x_i.

We can use (7) to rewrite (6) as

$$\begin{aligned}\mathcal{I}(\mathbf{z}; G(\mathbf{x} \mid \mathbf{z})) &\geq \mathcal{H}(\mathbf{z}) + \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z}),\mathbf{x}\sim p(\mathbf{x}|\mathbf{z})}\left[\log q_{MI}(\mathbf{z} \mid \mathbf{x})\right] \\ &= \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[-\log p(\mathbf{z}) + \mathrm{E}_{\mathbf{x}\sim p(\mathbf{x}|\mathbf{z})}\left[\log q_{MI}(\mathbf{z} \mid \mathbf{x})\right]\right] \\ &= \mathrm{E}_{\mathbf{z}\sim p(\mathbf{z}),\mathbf{x}\sim p(\mathbf{x}|\mathbf{z})}\left[\log q_{MI}(\mathbf{z} \mid \mathbf{x}) - \log p(\mathbf{z})\right] = -\mathcal{L}_{MI} \end{aligned} \tag{8}$$

This expectation can be approximated by maximizing the inner terms using stochastic gradient descent along with the variational autoencoder objective. $p(\mathbf{z})$ in (8) is the Gaussian prior we considered when setting up the variational autoencoder objective. It is important to ensure that we do not backpropagate gradients to $\mathbf{z}$, and thus to the encoder, otherwise the encoder starts predicting $q(\mathbf{z} \mid \mathbf{x})$ to be very far from $p(\mathbf{z})$ to maximize the mutual information objective, and the loss quickly implodes.

Intuitively, the mutual information loss ensures that there is enough information in the conditionally generated text from $\mathbf{z}$ such that it can estimate a spiky Gaussian under which the $\mathbf{z}$ has high likelihood.

5 Model architecture

The variational autoencoder implementation architecture with a bidirectional encoder is illustrated in Figure 2. From (3) and (8), we can define the loss function we want to minimize as

$$\mathcal{L} = \mathcal{L}_{VAE} + \lambda \mathcal{L}_{MI} \tag{9}$$

where λ is the fixed weight for the mutual information cost.

The discrete word inputs at each step are used to look up their corresponding continuous word embeddings, which are fed into the recurrent neural networks (RNNs). We share the same word embedding matrix for both the encoder and the decoder.

5.1 Encoder

We use a bidirectional recurrent neural network using Gated Recurrent Units (GRUs) (Cho et al. [2014]) to implement the variational approximation $q(\mathbf{z} \mid \mathbf{x})$. We also experimented with unidirectional RNN and convolutional approaches for the variational approximation, but did not find any improvement over the bidirectional encoder. The GRU outputs at each time step of the encoder are summarized with a function F to produce the distribution over the latent variable $\mathbf{z}$.

5.2 Decoder

We parameterize the generative model $p(\mathbf{x} \mid \mathbf{z})$ with a unidirectional recurrent neural network with GRUs, which we also refer to as the decoder. To additionally condition on $\mathbf{z}$, we concatenate $\mathbf{z}$ after passing it through a highway network to the input word embeddings at each timestep.

Since we train the decoder as a conditional language model, we give it the true inputs at every timestep, and ask it to predict the next word. Because of this very strong guidance during training, the model has less incentive to encode information in the latent variable. Using the mutual information loss is one way to alleviate this problem. We also try KL divergence annealing and word dropout as used by Bowman et al. [2015].

5.3 Summarization function

The summarization F can be done by picking the final states in both directions, considering the mean of the concatenated hidden states, or by considering the weighted mean (or attention) of the concatenated hidden states.

We found that using the mean and attention summarization strategies improved the model performance, possibly because this provides a direct path for gradients to flow to each timestep. We hypothesized that using an attention mechanism would allow the model to pay more attention to certain words in the input which played a stronger role in determining the latent representation.

5.3.1 KL divergence cost annealing

If the model quickly learns to minimize the KL divergence term in (4) to zero, the latent variable $\mathbf{z}$ will live in a Gaussian with zero mean and identity covariance for any sample $\mathbf{x}$, consequently encoding no useful information.

Therefore, we experimented with annealing the weight to the KL-divergence term from 0 and 1 as the training progresses according to a logistic curve.

5.4 Word-dropout annealing

Our generative model generates a word given the sequence of words preceding it. Due to redundancies in natural language, the model can learn to generate words from the rules of grammar alone and not take the latent representation into account at all. To prevent the model from doing so, we experimented with dropping out words which are presented to the generative model during training so that it is forced to derive information from the latent variable $\mathbf{z}$. Dropped out words are replaced by '<unk>' token. Unlike Bowman et al. [2015], we annealed the word dropout rate from probability 1.0 at step 2000 to 0.0 at step 20,000.

6 Experiments and results

6.1 Data

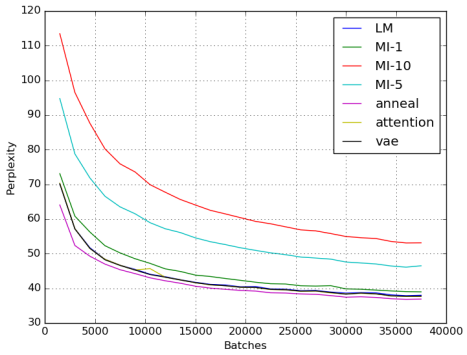
We use the Yelp review dataset (Yelp Inc [2016]) for our experiments. Each sample is a review for a business listing written by users of Yelp. The reviews also contain other labels which we have ignored for the purpose of this study. Our training set consisted of 2,443,410 samples where as our validation set contained 3,000 samples. As a pre-processing step we replace all numbers with the '#' token. Since the estimation of probabilities over the vocabulary involves a softmax operation, having a large vocabulary can prove to be a severe bottleneck. To reduce the vocabulary size we keep words which explain 97% of the data and replace the remaining words with an '<unk>' token. The final vocabulary contains 8595 words.

6.2 Methodology

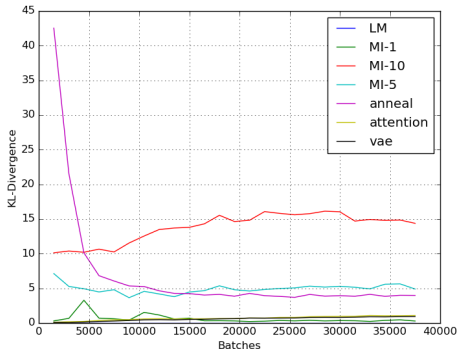
To estimate the variational lower bound (3) of the log-likelihood $\log p(\mathbf{x})$ for the validation set, we replicated the validation set 10 times to essentially estimate the expectation with 10 trials for each sample in the validation set. We compare the lower bounds to the log-likelihoods found with the log-likelihood of the data under an unconditional language model.

6.3 Results

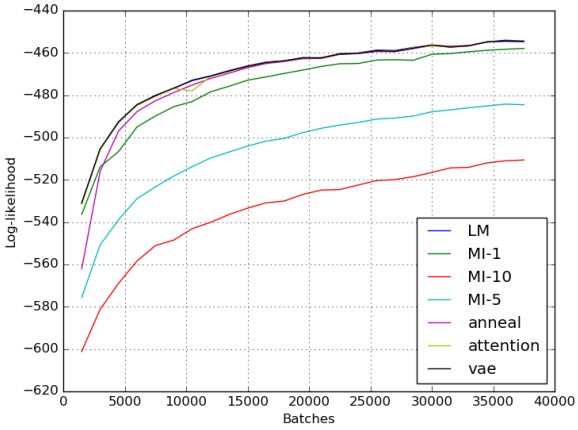
The conditional perplexities and log-likelihood of the validation data under the learnt models can be seen in Figure 3a and Figure 3c. It can be seen that the language model (LM) achieves higher or



(a) Perplexity of over time.



(b) KL Divergence over time.



(c) Log-likelihood of validation data over time.

Figure 3: Conditional perplexities, KL divergence and the data log-likelihoods on validation data with various experiments.

comparable log-likelihoods against the variational autoencoders. However, it is important to note that the log-likelihoods for the variational autoencoders are lower bounds to the true log-likelihoods, whereas the log-likelihood is computed exactly for the language model.

The language model achieves lower perplexity than the mutual information models with $\lambda = 1, 5, 10$ denoted by MI-1, MI-5 and MI-10. Without a mutual information objective, the model could choose to ignore the latent variable and become a language model. However, we are explicitly preventing the model from doing so with a mutual information objective, and forcing it to condition on the noisy latent representations. We are also asking the model to use up some of the model capacity to satisfy the mutual information criteria, essentially transferring some information from $\mathbf{z}$ to the decoder hidden states, which can be used to predict a distribution under which $\mathbf{z}$ has a high likelihood. This is one possible explanation to why the perplexity goes up and the data log-likelihood goes down by increasing the mutual information weight.

Variational autoencoders with mean (vae) and attention mechanism (attention) for summarizing GRU states have almost identical performance with respect to all 3 metrics. The lowest perplexity was achieved by annealing KL Divergence weight and summarizing GRU outputs using mean. The word-dropout annealing result has been omitted from this plot because it has approximately 60 times higher perplexity than the other results.

The more information there is in $\mathbf{z}$, the higher should be the KL divergence (assuming the KL divergence cost weight to be 1). Looking at Figure 3b, we can see that we are able to encode the most information in the model with the mutual information weight of 10, but it gives a poorer

True Sentences	Generated Sentences
my first hot air balloon experience and it was really great . our pilot was patrick and he did a great job . i would fly again with him anytime .	i had a great experience with dr . <unk> . the staff was very friendly and informative . i would recommend dr . <unk> to anyone looking for a great doctor . i would recommend him to anyone looking for a good ple to have a good experience .
there is no flavor in the food . the noodle tastes so weird and even worse than instant noodle . it has the worst wonton noodle i ve ever tasted .	if you re looking for a place to grab a quick bite , this is the place to go . the food is always fresh , and the service is fast . the only reason i m not giving # stars is because it s a little pricey .
jack and mary are the best ... they are always <unk> to their customers ! ! ! ! calm under pressure . <unk> is an excellent chef and very entertaining	the food was good , the service was excellent , and the prices were very reasonable . i would recommend this place to anyone looking for a great place to eat in vegas . i would recommend this place to anyone looking for a great place to eat .

Table 1: Sentences generated from Variational autoencoder with KL-Divergence annealing.

True Sentences	Generated Sentences
went for lunch , the waiter gave us his recommendations which were excellent . had the spicy thai soup and crab vermicelli . would definitely go back !	this place is amazing ! i ve been here twice now and have never been disappointed . i ve been coming here for over # years and have never been disappointed . the food is delicious and the service is always friendly . i will definitely be back !
staff were great here . they helped me use the computer which was running on windows # so old and <unk> some <unk> for me . thank you guys .	i love this place ! ! ! ! ! thank you ! ! ! ! ! ! !
food is excellent , the bread and dipping oil is to die for , service is excellent very attentive . love the patio	my husband and i decided to give this place a try . i have been here a few times and have never been disappointed . the food is good and the service is good . the food is great , the food is great and the service is great

Table 2: Sentences generated from Variational autoencoder with mutual information weight $\lambda = 10$.

log-likelihood lower bound for the validation data. This indicates that the encoded information may be specifically tailored to satisfy the mutual information criteria, rather than help with maximizing the data likelihood.

6.4 Sentence generation

We used beam search with a beam of of 16 to generate sentences. The sentences generated from the model with KL divergence cost annealing are shown in Table 1. Although the generated reviews are grammatically correct and appear like reviews from the dataset, they seem to have no relation to the true input sentences the latent representation is supposed to encode. This suggests that although the model is encoding some information as seen from Figure 3b, it is not enough.

Some example sentences from the model with a mutual information weight of 10 are shown in Table 2. Again, we see little relation of the input sentences with the generated sentences, however, we see words such as 'thank you' or 'will come back' overlapping between the input and generated reviews. This can explain some of the encoded information that is increasing the KL divergence.

Across all models, the generations are limited in the kinds of sentences they produce. Most of them reuse phrases such as 'i love this place', 'the service was great' and so on. Another avenue of experiments would be to explore diversity-encouraging objectives in beam search.

7 Conclusion

We find that building latent variable models for text is challenging, especially under certain kinds of datasets that have a high redundancy in them. We experimented with different variational approximations for $p(\mathbf{z} \mid \mathbf{x})$, and tried various tricks to encourage the model to put semantic information in the latent variable $\mathbf{z}$.

Our future work can involve further investigation into the mutual information objective, to understand what the model is actually encoding and why it does not benefit the data log-likelihood.

The source code for this project can be found at `https://github.com/ankitkv/SentimentVAE`.

Acknowledgments

We would like to thank Prof. David Sontag on his constant guidance throughout the project, and for his suggestion on considering a mutual information loss to encourage encoding meaningful information in the latent representations.

References

S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Józefowicz, and S. Bengio. Generating sentences from a continuous space. *CoRR*, abs/1511.06349, 2015. URL `http://arxiv.org/abs/1511.06349`.

X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *CoRR*, abs/1606.03657, 2016. URL `http://arxiv.org/abs/1606.03657`.

K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. URL `http://arxiv.org/abs/1406.1078`.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, number 2014, 2013.

Y. Miao, L. Yu, and P. Blunsom. Neural variational inference for text processing. *CoRR*, abs/1511.06038, 2015. URL `http://arxiv.org/abs/1511.06038`.

Yelp Inc. Yelp dataset challenge, 2016. URL `https://www.yelp.com/dataset_challenge`.