Variational Autoencoders for Text Generation

Rosemary Fortanely
Department of Computer Science
The University of Texas at Austin
Austin, TX 78712
rosemary.fortanely@utexas.edu

December 9, 2020

Abstract

Recent work on generative text modeling has found that variational autoencoders (VAE) with dilated CNN decoders can outperform LSTM language models (Yang et al., 2017). In this paper, I experiment with a VAE with an LSTM encoder and dilated CNN decoder. I will conduct an investigation of the use of VAE for language modeling with a dataset labeled with sentiment. Further, I will use the VAE to generate novel sentences conditioned on sentiment.

1 Introduction

VAEs are suitable for generation tasks with various levels of expression. They function by sampling a continuous latent vector and using this to parameterize a decoder network. During training, an encoder network creates latent vectors from the input. It is then possible to sample from this continuous latent space to create output with a similar style of the input that maps to that latent space.

LSTMs are a popular model for language modeling. However, it's been found that existing VAE models consisting of LSTM encoder and decoders (LSTM-VAE) underperform against regular LSTM models due to their inability to model long-range dependencies (Bowman et al., 2016).

I propose the use of a dilated CNN as a decoder for the VAE, inspired by previous success of VAE with a dilated CNN decoder for language modeling (Yang et al., 2017). Despite this, I found that the VAE performed poorly as measured by its ability for language modeling when trained on a sentiment labelled dataset.

2 Method

2.1 Model overview

In this section, I will describe the dilated CNN architecture that is used for the decoder, as well as give background on the use of VAE for language modeling and details of the VAE that will be used in the experiments.

2.2 Dilated convolutional decoder

CNNs as used for text generation are similar to CNN models used for image-related tasks, but perform the convolutions in one dimension.

To prevent conditioning on future tokens in the sentence, which would lead to nonsensical output, the CNN must be causal, meaning it only conditions on past tokens $x_{<t}$. Traditional convolutional layers use all tokens. I avoid this by shifting the input. The effective filter size with a kernel size of k and n layers is $(k-1) \cdot n + 1$.

I use residual connections in the decoder to allow for the training of a deeper model. Each block consists of two convolutional layers with kernel size $1 \cdot k$ and Leaky ReLU activations between convolutional layers.

Dilated convolutions allow for the receptive field to be increased without increasing computational cost. With dilations, the convolution is then applied so that every $d-1$ input is skipped. The dilations are set to double with every residual block, increasing the receptive field exponentially.

2.3 Variational autoencoder

Typical language models that generate tokens conditioned on the history of previously generated tokens can have difficulty learning high-level properties, such as topic or style. Bowman et al. (2016) proposed a method to generative text modeling, defining $p(x)$ as a marginal distribution. The continuous latent vector z is generated using a decoder network, and then the text sequence x is generated from a conditional distribution $p_\theta(x|z)$ which is parameterized by the encoder network:

$$p_\theta(x|z) = \prod_t p_\theta(x_t|x_1, x_2, ..., x_{t-1}, z). \tag{1}$$

This latent variable z allows for the network to better capture high-level properties.

The LSTM encoder outputs the mean and log variance. These are used to form the posterior probability $q_\theta(z|x)$, which is assumed to be a Gaussian so a reparameterization trick can be used (Kingma & Welling, 2013). Then, z is sampled from $q_\theta(z|x)$. The decoder is conditioned on z by concatenating z with every word embedding of the decoder input.

2.4 Data set

I want to investigate VAEs for the use of language modeling and modify output as conditioned on sentiment, so the dataset should contain samples with sentiment labels. I am using the Sentiment Labelled Sentences Data Set from the UCI Machine Learning Repository (Kotzias et al., 2015). From it, I use 1000 samples of Amazon reviews as training data, and 10 samples of Yelp reviews as testing data. For each of the partitions of the dataset, they contain a sentence with an integer label representing a negative or positive sentiment.

2.5 Model and training details

I use an LSTM as an encoder and a CNN as a decoder. For both models, I use a vocabulary size of 5020 and a word embedding dimension of 300. These word embeddings are created from relativized GloVe vectors.

The hidden dimension of the LSTM is 150 and a dropout of 0.2 is used. I use the output of the encoder LSTM and feed it through a linear layer and log softmax layer to get the mean and log variance of $q_\theta(z|x)$, from which z is sampled and used as the starting state of the CNN decoder.

For the CNN, the starting state is concatenated with the word embedding input. I set the kernel size to 3 and initialize the dilation to 1. The dilation is increased by a factor of 2 over two residual blocks. The effective filter size is then 5. The number of channels for the internal convolutions of the CNN is 600 for all of the convolutional layers in the residual blocks. A dropout layer with a ratio of 0.1 is used between each convolutional layer.

I use Adam to optimize the model with learning rate 1e-3. Batching is not used and the model is trained for 5 epochs.

3 Results

3.1 Language modeling results

The main measures of results are the log probability and perplexity of the test set. The model achieved a log probability of -3752 and a perplexity of 277944767.

These results are interesting, given that the training set achieved a final average loss of 0.95. I suspect there may be an error in the way the input was created for the decoder using the sample from the poster probability $q_\theta(z|x)$. Additionally, the effective filter size isn't very large, which makes the decoder less effective. A couple of options to increase this could have been increasing the dilation between every layer instead of every residual block, as well increasing the total number of layers. Finally, the number of epochs could have been increased. On the machine I used, each epoch took almost 30 minutes to run, so training was capped after 5 epochs. Training for a longer number of epochs could have helped the model generalize better.

3.2 Latent representation visualization

To visualize the latent representation, I ran PCA on the mean of the posterior probability $q_\theta(z|x)$, reducing the dimension of the mean to 2, as shown in Figure 1.

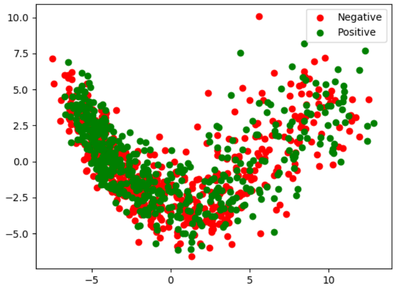

Figure 1: Visualization of learned latent representations

I was unable to create an algorithm that reduced the dimensionality of the data while maintaining a separating hyperplane, so it's inconclusive whether the VAE maps negative and positive sentiment samples into separate clusters.

3.3 Conditional text generation

Text can be generated conditionally by label. To do this, I condition the VAE on the average mean of the posterior probability $q_\theta(z|x)$ for the negative and positive examples from the training data, respectively. For each group of generated text, I fix the length to 5 and pick x via a random sample on the logits.

Random samples	itemphoneechopainfulsweetest worksitemdisappointedlousybeautiful adaptercoolfonduebeautifulinexperience
Negatively conditioned samples	disappointedbuttonscoolkindcool sweetestadapterpoorsweetestecho sweetestlovelousydisappointedpoor
Positively conditioned samples	crispechocoolwouldbeautiful soggyawesomeechoadapterphone sweetestechohavingpainfulearbud

Table 1: Text generated by conditioning on sentiment label.

References

Yang, Z., Hu, Z., Salakhutdinov, R., & Berg-Kirkpatrick, T. (2017). Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. *ArXiv, 1702.08139.*

Bowman, S., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., & Bengio, S. (2016). Generating Sentences from a Continuous Space. *ArXiv, 1511.06349.*

Kingma, Diederik, and Max Welling. (2013) Auto-Encoding Variational Bayes. *ArXiv Preprint, 1312.6114.*

Kotzias et. al. (2015) From Group to Individual Labels using Deep Features. *KDD*