BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model

Alex Wang
New York University
alexwang@nyu.edu

Kyunghyun Cho
New York University
Facebook AI Research
CIFAR Azrieli Global Scholar
kyunghyun.cho@nyu.edu

Abstract

We show that BERT (Devlin et al., 2018) is a Markov random field language model. This formulation gives way to a natural procedure to sample sentences from BERT. We generate from BERT and find that it can produce high-quality, fluent generations. Compared to the generations of a traditional left-to-right language model, BERT generates sentences that are more diverse but of slightly worse quality.

1 Introduction

BERT (Devlin et al., 2018) is a recently released sequence model used to achieve state-of-art results on a wide range of natural language understanding tasks, including constituency parsing (Kitaev and Klein, 2018) and machine translation (Lample and Conneau, 2019). Early work probing BERT's linguistic capabilities has found it surprisingly robust (Goldberg, 2019).

BERT is trained on a *masked language modeling* objective. Unlike a traditional language modeling objective of predicting the next word in a sequence given the history, masked language modeling predicts a word given its left and right context. Because the model expects context from both directions, it is not immediately obvious how BERT can be used as a traditional language model (i.e., to evaluate the probability of a text sequence) or how to sample from it.

We attempt to answer these questions by showing that BERT is a combination of a Markov random field language model (MRF-LM, Jernite et al., 2015; Mikolov et al., 2013) with pseudo log-likelihood (Besag, 1977) training. This formulation automatically leads to a sampling procedure based on Gibbs sampling.

2 BERT as a Markov Random Field

Let $X = (x_1, \ldots, x_T)$ be a sequence of random variables x_i, each of which is categorical in that it can take one of M items from a vocabulary $V = \{v_1, \ldots, v_M\}$. These random variables form a fully-connected graph with undirected edges, indicating that each variable x_i is dependent on all the other variables.

Joint Distribution To define a Markov random field (MRF), we start by defining a potential over cliques. Among all possible cliques, we only consider the clique corresponding to the full graph. All other cliques will be assigned a potential of 1 (i.e. $\exp(0)$). The potential for this full-graph clique decomposes into a sum of T log-potential terms:

$$\phi(X) = \prod_{t=1}^{T} \phi_t(X) = \exp\left(\sum_{t=1}^{T} \log \phi_t(X)\right),$$

where we use X to denote the fully-connected graph created from the original sequence. Each log-potential $\phi_t(X)$ is defined as

$$\log \phi_t(X) = \begin{cases} \text{1h}(x_t)^\top f_\theta(X_{\setminus t}), & \text{if } [\text{MASK}] \notin X_{1:t-1} \cup X_{t+1:T} \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where $f_\theta(X_{\setminus t}) \in \mathbb{R}^M$, $\text{1h}(x_t)$ is a one-hot vector with index x_t set to 1, and

$$X_{\setminus t} = (x_1, \ldots, x_{t-1}, [\text{MASK}], x_{t+1}, \ldots, x_T)$$

From this log-potential, we can define a probability of a given sequence X as

$$p_\theta(X) = \frac{1}{Z(\theta)} \prod_{t=1}^{T} \phi_t(X), \quad (2)$$

Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation (NeuralGen), pages 30–36
Minneapolis, Minnesota, USA, June 6, 2019.

where

$$Z(\theta) = \sum_{X'} \prod_{t=1}^{T} \phi_t(X'),$$

for all X'. This normalization constant is unfortunately impractical to compute exactly, rendering exact maximum log-likelihood intractable.

Conditional Distribution Given a fixed $X_{\setminus t}$, the conditional probability of x_t is derived to be

$$p(x_t|X_{\setminus t}) = \frac{1}{Z(X_{\setminus t})} \exp(\mathrm{1h}(x_t)^\top f_\theta(X_{\setminus t})), \tag{3}$$

where

$$Z(X_{\setminus t}) = \sum_{m=1}^{M} \exp(\mathrm{1h}(m)^\top f_\theta(X_{\setminus t})).$$

This derivation follows from the peculiar formulation of the log-potential in Eq. (1). It is relatively straightforward to compute, as it is simply softmax normalization over M terms (Bridle, 1990).

(Stochastic) Pseudo Log-Likelihood Learning One way to avoid the issue of intractability in computing the normalization constant $Z(\theta)$ above[1] is to resort to an approximate learning strategy. BERT uses pseudo log-likelihood learning, where the pseudo log-likelihood is defined as:

$$\mathrm{PLL}(\theta; D) = \frac{1}{|D|} \sum_{X \in D} \sum_{t=1}^{|X|} \log p(x_t|X_{\setminus t}), . \tag{4}$$

where D is a set of training examples. We maximize the predictability of each token in a sequence given all the other tokens, instead of the joint probability of the entire sequence.

It is still expensive to compute the pseudo log-likelihood in Eq. (4) for even one example, especially when f_θ is not linear. This is because we must compute $|X|$ forward passes of f_θ for each sequence, when $|X|$ can be long and f_θ be computationally heavy. Instead we could stochastically estimate it by

$$\begin{aligned}\frac{1}{|X|} \sum_{t=1}^{|X|} &\log p(x_t|X_{\setminus t}) \\ &= \mathbb{E}_{t \sim \mathcal{U}(\{1,\ldots,|X|\})} \left[\log p(x_t|X_{\setminus t})\right] \\ &\approx \frac{1}{K} \sum_{k=1}^{K} \log p(x_{\tilde{t}_k}|X_{\setminus \tilde{t}_k}),\end{aligned}$$

where $\tilde{t}_k \sim \mathcal{U}(\{1, \ldots, |X|\}$. Let us refer to this as stochastic pseudo log-likelihood learning.

In Reality The stochastic pseudo log-likelihood learning above states that we "mask out" one token in a sequence at a time and let f_θ predict it based on all the other "observed" tokens in the sequence. Devlin et al. (2018) however proposed to "mask out" multiple tokens at a time and predict all of them given both all "observed" and "masked out" tokens in the sequence. This brings the original BERT closer to a denoising autoencoder (Vincent et al., 2010), which could still be considered as training a Markov random field with (approximate) score matching (Vincent, 2011).

3 Using BERT as an MRF-LM

The discussion so far implies that BERT is a Markov random field language model (MRF-LM) and that it learns a distribution over sentences (of some given length). This framing suggests that we can use BERT not only as parameter initialization for finetuning but as a generative model of sentences to either score a sentence or sample a sentence.

Ranking Let us fix the length T. Then, we can use BERT to rank a set of sentences. We cannot compute the exact probabilities of these sentences, but we can compute their unnormalized log-probabilities according to Eq. (2):

$$\sum_{t=1}^{T} \log \phi_t(X).$$

These unnormalized probabilities can be used to find the most likely sentence within the set or to sort the sentences according to their probabilities.

Sampling Sampling from a Markov random field is less trivial than is from a directed graphical model which naturally admits ancestral sampling. One of the most widely used approaches

[1] In BERT it is not intractable in the strictest sense, since the amount of computation is bounded (by $T = 500$) each iteration. It however requires computation up to $\exp(500)$ which is in practice impossible to compute exactly.

the nearest regional centre is alemanno , with another connection to potenza and maradona , and the nearest railway station is in bergamo , where the line terminates on its northern end	for all of thirty seconds , she was n't going to speak . maybe this time , she 'd actually agree to go . thirty seconds later , she 'd been speaking to him in her head every
' let him get away , mrs . nightingale . you could do it again . ' ' he - ' ' no , please . i have to touch him . and when you do , you run .	" oh , i 'm sure they would be of a good service , " she assured me . " how are things going in the morning ? is your husband well ? " " yes , very well
he also " turned the tale [of] the marriage into a book " as he wanted it to " be elegiac " . both sagas contain stories of both couple and their wedding night ;	" i know . " she paused ." did he touch you ? " " no . " " ah . " " oh , no , " i said , confused , not sure why
" i had a bad dream . " " about an alien ship ? who was it ? " i check the text message that 's been only partially restored yet, the one that says love .	i watched him through the glass , wondering if he was going to attempt to break in on our meeting . but he did n't seem to even bother to knock when he entered the room . i was n't
replaced chris hall (st . louis area manager) . june 9 : mike howard (syndicated " good morning " , replaced steve koval , replaced dan nickolas , and replaced phil smith) ;	" how long has it been since you have made yourself an offer like that ? " asked planner . " oh " was the reply . planner had heard of some of his other business associates who had

Table 1: Random sample generations from BERT base (left) and GPT (right).

is Markov-chain Monte-Carlo (MCMC) sampling (Neal, 1993; Swendsen and Wang, 1986; Salakhutdinov, 2009; Desjardins et al., 2010; Cho et al., 2010). In this report, we only consider Gibbs sampling which fits naturally with (stochastic) pseudo log-likelihood learning.

In Gibbs sampling, we start with a random initial state X^0, which we initialize to be an all-mask sequence, i.e., $([\text{MASK}], \ldots, [\text{MASK}])$, though we could with a sentence consisting of randomly sampled words or by retrieving a sentence from data. At each iteration i, we sample the position t^i uniformly at random from $\{1, \ldots, T\}$ and mask out the selected location, i.e., $x^i_{t^i} = [\text{MASK}]$, resulting in $X^i_{\setminus t^i}$. We now compute $p(x_{t^i}|X^i_{\setminus t^i})$ according to Eq. (3), sample $\tilde{x}_{t^i}$ from it[2], and construct the next sequence by

$$X^{i+1} = (x^i_1, \ldots, x^i_{t^i-1}, \tilde{x}_{t^i}, x^i_{t^i+1}, \ldots, x^i_T).$$

We repeat this procedure many times, preferably with thinning.[3] Because Gibbs sampling, as well as any MCMC sampler with a local proposal distribution, tends to get stuck in a mode of the distribution, we advise running multiple chains of Gibbs sampling or using different sentence initializations.

Sequential Sampling The undirectedness of the MRF-LM and the bidirectional nature of BERT do not naturally admit sequential sampling, but given that the dominant approach to text generation is left-to-right, we experiment with generating from BERT in such a manner.

As with our non-sequential sampling scheme, we can begin with a seed sentence of either all masks or a random sentence. Whereas previously we sampled a position $t \in \{1, \ldots, T\}$ to mask out and generate for at each time step, in the sequential setting, at each time step t, we mask out x^t_t, generate a word for that position, and substitute it into the sequence. After T timesteps, we have a sampled a token at each position, at which we point we can terminate or repeat the process from the current sentence.

4 Experiments

Our experiments demonstrate the potential of using BERT as a *standalone* language model rather than as a parameter initializer for transfer learning (Devlin et al., 2018; Lample and Conneau, 2019; Nogueira and Cho, 2019). We show that sentences sampled from BERT are well-formed and are assigned high probabilities by an off-the-shelf language model. We take pretrained BERT models trained on a mix of Toronto Book Corpus (TBC, Zhu et al., 2015) and Wikipedia provided by Devlin et al. (2018) and its PyTorch implementation[4] provided by HuggingFace. We experiment with both the base and large BERT configuations.

4.1 Evaluation

We consider several evaluation metrics to estimate the quality and diversity of the generations.

[2] In practice, one can imagine sampling from the k-most probable words (Fan et al., 2018). We find $k = 100$ to be effective in early experiments.

[3] Thinning refers to the procedure of selecting a sample only once a while during MCMC sampling.

[4] https://github.com/huggingface/pytorch-pretrained-BERT

Model	Self-BLEU (↓)	% Unique n-grams (↑)								
		Self			WT103			TBC		
		n=2	n=3	n=4	n=2	n=3	n=4	n=2	n=3	n=4
BERT (large)	9.43	63.15	92.38	98.01	59.91	91.86	98.43	64.59	93.27	98.59
BERT (base)	10.06	60.76	91.76	98.14	57.90	91.72	98.55	60.94	92.04	98.56
GPT	40.02	31.13	67.01	87.28	33.71	72.86	91.12	25.74	65.04	88.42
WT103	9.80	70.29	94.36	99.05	56.19	88.05	97.44	68.35	94.20	99.23
TBC	12.51	62.19	92.70	98.73	55.30	91.08	98.81	44.75	82.06	96.31

Table 2: Self-BLEU and percent of generated n-grams that are unique relative to own generations (left) WikiText-103 test set (middle) a sample of 5000 sentences from Toronto Book Corpus (right). For the WT103 and TBC rows, we sample 1000 sentences from the respective datasets.

Quality To automatically measure the quality of the generations, we follow Yu et al. (2017) by computing BLEU (Papineni et al., 2002) between the generations and the original data distributions to measure how similar the generations are. We use a random sample of 5000 sentences from the test set of WikiText-103 (WT103, Merity et al., 2016) and a random sample of 5000 sentences from TBC as references.

We also use the perplexity of a trained language model evaluated on the generations as a rough proxy for fluency. Specifically, we use the Gated Convolutional Language Model (Dauphin et al., 2016) pretrained on WikiText-103[5].

Diversity To measure the diversity of each model's generations, we compute self-BLEU (Zhu et al., 2018): for each generated sentence, we compute BLEU treating the rest of the sentences as references, and average across sentences. Self-BLEU measures how similar each generated sentence is to the other generations; high self-BLEU indicates that the model has low sample diversity.

We also evaluate the percentage of n-grams that are unique, when compared to the original data distribution and within the corpus of generations. We note that this metric is somewhat in opposition to BLEU between generations and data, as fewer unique n-grams implies higher BLEU.

Methodology We use the non-sequential sampling scheme with sampling from the top $k = 100$ most frequent words at each time step, as empirically this led to the most coherent generations. We show generations from the sequential sampler in Table 4 in the appendix. We compare against generations from a high-quality neural language model, the OpenAI Generative Pre-Training Transformer (Radford et al., 2018, GPT), which was trained on TBC and has approximately the same number of parameters as the base configuration of BERT. For BERT, we pad each input with special symbols [CLS] and [SEP]. For GPT, we start with a start of sentence token and generate left to right. For all models, we generate 1000 uncased sequences of length 40. Finally, as a trivial baseline, we sample 1000 sentences from TBC and the training split of WT103 and compute all automatic metrics against these samples.

[5] https://github.com/pytorch/fairseq/tree/master/examples/conv_lm

Model	Corpus-BLEU (↑)		PPL (↓)
	WT103	TBC	
BERT (large)	5.05	7.60	331.47
BERT (base)	7.80	7.06	279.10
GPT	10.81	30.75	154.29
WT103	17.48	6.57	54.00
TBC	10.05	23.05	314.28

Table 3: Quality metrics of model generations. Perplexity (PPL) is measured using an additional language model (Dauphin et al., 2016). For the WT103 and TBC rows, we sample 1000 sentences from the respective datasets.

5 Results

We present sample generations, quality results, and diversity results respectively in Tables 1, 2, 3.

We find that, compared to GPT, the BERT generations are of worse quality, but are more diverse. Surprisingly, the outside language model, which was trained on Wikipedia, is less perplexed by the GPT generations than the BERT generations, even though GPT was only trained on romance novels and BERT was trained on romance novels and Wikipedia. On actual data from TBC, the outside language model is about as perplexed as on the BERT generations, which suggests that domain shift is an issue in using a trained language

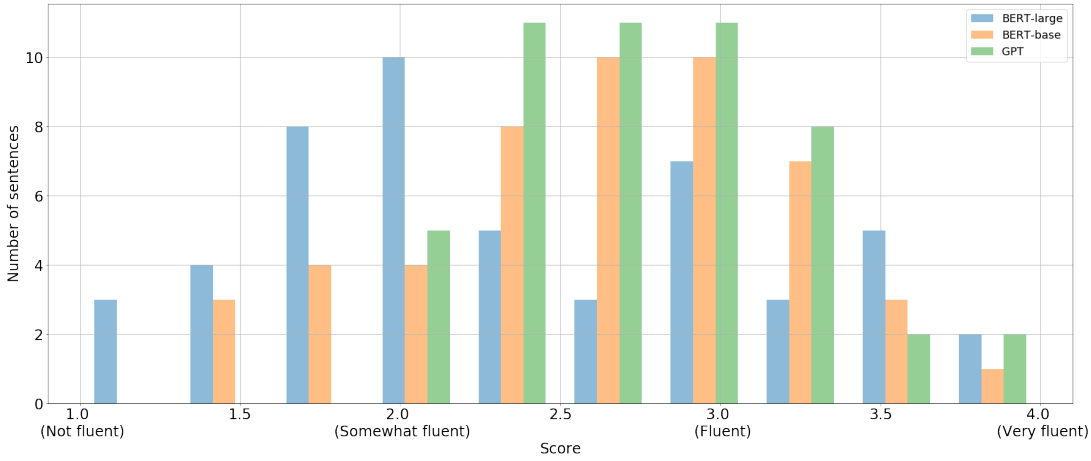

Figure 1: Fluency scores for 100 sentences samples from each of BERT large, BERT base, and GPT, as judged by human annotators according to a four-point Likert scale.

model for evaluating generations and that the GPT generations might have collapsed to fairly generic and simple sentences. This observation is further bolstered by the fact that the GPT generations have a higher corpus-BLEU with TBC than TBC has with itself. The perplexity on BERT samples is not absurdly high, and in reading the samples, we find that many are fairly coherent. The corpus-BLEU between BERT models and the datasets is low, particularly with WT103.

We find that BERT generations are more diverse than GPT generations. GPT has high n-gram overlap (smaller percent of unique n-grams) with TBC, but surprisingly also with WikiText-103, despite being trained on different data. Furthermore, GPT generations have greater n-gram overlap with these datasets than these datasets have with themselves, further suggesting that GPT is relying significantly on generic sentences. BERT has lower n-gram overlap with both corpora, with similar degrees of n-gram overlap as the samples of the data.

For a more rigorous evaluation of generation quality, we collect human judgments on sentence fluency for 100 samples from BERT large, BERT base, and GPT using a four point Likert scale. For each sample we ask three annotators to rate the sentence on its fluency and take the average of the three judgments as the sentence's fluency score. We present a histogram of the results in Figure 1. For BERT large, BERT base, and GPT we respectively get mean scores over the samples of 2.37 ($\sigma = 0.83$), 2.65 ($\sigma = 0.65$), and 2.80 ($\sigma = 0.51$). All means are within a standard deviation of each other. BERT base and GPT have similar unimodal distributions with BERT base having a slightly more non-fluent samples. BERT large has a bimodal distribution.

6 Conclusion

We show that BERT is a Markov random field language model. Formulating BERT in this way gives rise to a practical algorithm for generating from BERT based on Gibbs sampling that does not require any additional parameters or training. We verify in experiments that the algorithm produces diverse and fairly fluent generations. The power of this framework is in allowing the principled application of Gibbs sampling, and potentially other MCMC algorithms, for generating from BERT.

Future work might explore these improved sampling methods, especially those that do not need to run the model over the entire sequence each iteration and that more robustly handle variable-length sequences. To facilitate such investigation, we release our code on GitHub at https://github.com/nyu-dl/bert-gen and a demo as a Colab notebook at https://colab.research.google.com/drive/1MxKZGtQ9SSBjTK5ArsZ5LKhkztzg52RV.

Acknowledgements

We thank Ilya Kulikov and Nikita Nangia for their help, as well as reviewers for insightful comments. AW is supported by an NSF Fellowship. KC is partly supported by Samsung Advanced Institute of Technology (Next Generation Deep Learning: from Pattern Recognition to AI) and Samsung Electronics (Improving Deep Learning using Latent Structure).

References

Julian Besag. 1977. Efficiency of pseudolikelihood estimation for simple gaussian fields. *Biometrika*, pages 616–618.

John S Bridle. 1990. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pages 227–236. Springer.

KyungHyun Cho, Tapani Raiko, and Alexander Ilin. 2010. Parallel tempering is efficient for learning restricted boltzmann machines. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–8. IEEE.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *arXiv preprint:1612.08083*.

Guillaume Desjardins, Aaron Courville, Yoshua Bengio, Pascal Vincent, and Olivier Delalleau. 2010. Tempered markov chain monte carlo for training of restricted boltzmann machines. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 145–152.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint:1810.04805*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint:1805.04833*.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities.

Yacine Jernite, Alexander Rush, and David Sontag. 2015. A fast variational approach for learning markov random field language models. In *International Conference on Machine Learning*, pages 2209–2217.

Nikita Kitaev and Dan Klein. 2018. Multilingual constituency parsing with self-attention and pre-training.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual Language Model Pretraining. *arXiv e-prints*, page arXiv:1901.07291.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint:1301.3781*.

Radford M Neal. 1993. Probabilistic inference using markov chain monte carlo methods.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint:1901.04085*.

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. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/languageunsupervised/language understanding paper. pdf*.

Ruslan R Salakhutdinov. 2009. Learning in markov random fields using tempered transitions. In *Advances in neural information processing systems*, pages 1598–1606.

Robert H Swendsen and Jian-Sheng Wang. 1986. Replica monte carlo simulation of spin-glasses. *Physical review letters*, 57(21):2607.

Pascal Vincent. 2011. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *arXiv preprint:1802.01886*.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *arXiv preprint:1506.06724*.

A Other Sampling Strategies

We explored two other sampling strategies: left-to-right and generating for all positions at each time step. See Section 3 for an explanation of the former. For the latter, we start with an initial sequence of all masks, and at each time step, we would not mask any positions but would generate for all positions. This strategy is designed to save on computation. However, we found that this tended to get stuck in non-fluent sentences that could not be recovered from. We present sample generations for the left-to-right strategy in Table 4.

all the good people , no more , no less . no more . for ... the kind of better people ... for ... for ... for ... for ... for ... for ... as they must become again .
sometimes in these rooms , here , back in the castle . but then : and then , again , as if they were turning , and then slowly , and and then and then , and then suddenly .
other available songs for example are the second and final two complete music albums among the highest played artists , including : the one the greatest ... and the last recorded album , " this sad heart " respectively .
6 that is i ? ? and the house is not of the lord . i am well ... the lord is ... ? , which perhaps i should be addressing : ya is then , of ye ? ?
four - cornered rap . big screen with huge screen two of his friend of old age . from happy , happy , happy . left ? left ? left ? right . left ? right . right ? ?

Table 4: Random sample generations from BERT base using a sequential, left-to-right sampling strategy.