The Suitability of Generative Adversarial Training for BERT Natural Language Generation

Benny Longwill

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2020

Committee:

Shane Steinert-Threlkeld

Gina-Anne Levow

Program Authorized to Offer Degree:
Linguistics

University of Washington

Abstract

The Suitability of Generative Adversarial Training for BERT Natural Language Generation

Benny Longwill

Chair of the Supervisory Committee:
Dr. Shane Steinert-Threlkeld
Linguistics

This thesis presents a study that was designed to test the effect of generative adversarial network (GAN) training on the quality of natural language generation (NLG) using a pre-trained language model architecture: Bidirectional Encoder Representations from Transformers (BERT). Perplexity and BLEU scores were used as metrics for evaluation on 1000 samples of generated text. Results indicated that perplexity decreased and BLEU scores comparing the original data distributions increased; thus, there was evidence that quality of NLG was improved by the introduction of GAN training. This alternative training method may also be effective for other more state-of-the-art pre-trained architectures.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to thank the various people who have supported me on this project along the way; Shane Steinert-Threlkeld, for his invaluable guidance and encouragement, even when preliminary results weren't looking so great; Gina-Anne Levow, for her honest reflection and constructive recommendations on the paper; The U.W. Computation Language and Meaning Band of Researchers (CLMBR lab), for giving me the opportunity to present my ideas and receive feedback from such a powerful group of scholars; Last but not least, Alex Wang, for his blessing to build upon his previous work and answering my many questions.

DEDICATION

To those who have yet to master natural language:

01010111 01100101 00100000 01100001 01110010 01100101 00100000 01101111 01101110
01100101 00100000 01110011 01110100 01100101 01110000 00100000 01100011 01101100
01101111 01110011 01100101 01110010 00101110

Chapter 1

INTRODUCTION

Natural language generation (NLG) stands at the forefront of human-computer interaction by transforming non-linguistic representations of data into natural language. With the ease and efficiency of emulating the same pipeline that a human innately uses to communicate with another human, an NLG system would be able to stand in and produce natural language to communicate a response. Given the large body of linguistic and world knowledge that state-of-the-art neural networks demonstrate (Clark et al., 2019; Cui et al., 2020), a functioning NLG system would have many pragmatic applications (e.g., language translation, chatbots, question answering, and summarization (Çelikyilmaz et al., 2020)) and also contribute a vital step toward true artificial intelligence.

Traditionally NLG systems entailed complex frameworks (i.e., template or rule-based systems) or well-understood probabilistic models (i.e. n-gram or loglinear models) (Chen and Goodman, 1996; Koehn et al., 2003). In general these systems are well-received with reports of high interpretability and accuracy; however, traditional systems tend not to scale well with larger datasets because of high manual-engineering labor cost or possible saturation in system performance (Józefowicz et al., 2016). More and more research is taking place on this topic by switching to neural language models.

In many cases, neural language models used for text generation comprise of Recurrent Neural Networks (RNNs), an artificial neural network that generates text sequentially using connections between nodes that form a directed graph along a temporal sequence (Fedus et al., 2018; Graves, 2012). Such models reach state-of-the-art performance on summarization, translation, and speech recognition benchmarks by sampling a word conditioned on the probability of the words that come before it (Bahdanau et al., 2015). Be this as it may,

RNNs perform increasingly poor when conditioned on text not previously encountered in training (Fedus et al., 2018). Because RNN models are typically trained using a technique called "Teacher Forcing" (Williams, 1992), in which only the factual data is fed back into the model, uni-directional text generation poses an obstacle by conditioning on previously generated text. Performance worsens as generated sequences increasingly diverge from training data because small prediction errors multiply and lead to unpredictable outcomes in the hidden state. Indirect attempts have been made to resolve this issue (e.g., Professor Forcing (Goyal et al., 2016) and Scheduled Sampling (Bengio et al., 2015)) but no proposed approach uses a cost function on the output of the RNN improve generation (Fedus et al., 2018)

For this reason, researchers in this area have been examining the effect of different model types and novel training methods to improve the quality and diversity of generated text. Knowing the effect that model type and training methodology have on NLG output may indicate how to improve general NLG performance and also provide insight into the high empirical success but low interpretability of neural language models.

Previous studies have reported high quality NLG from large-scale unsupervised pre-trained transformer-derived architectures. For example, in a study by Wang and Cho (2019) researchers proposed a novel procedure that leveraged a Bidirectional Encoder Representations from Transformers (BERT) architecture (Devlin et al., 2019) as a robust generative model using a Gibbs sampling (Neal, 1993). Researchers found that in comparison to the OpenAI Generative Pre-Training Transformer (GPT) (Radford, 2018), BERT produced more diverse but lesser quality generations. One point of interest, however, is that in this study neither BERT nor GPT were fine-tuned for down-stream tasks; thus, the effect from fine-tuning for generation using this model is unclear.

It is well documented that fine-tuning provides advantages in terms of quicker development, less required data, and better results (Bozinovski, 2020; Conneau et al., 2017; McCann et al., 2017); furthermore, there is research that suggests generative adversarial network (GAN) training may provide a strong candidate to facilitate fine-tuning a pre-trained model. GAN is a prominent machine learning framework in which two neural models are pitted

against each other in a min-max game with the goal to generate new data with the same statistics as the training set (Goodfellow et al., 2014). In a study by Fedus et al. (2018), GAN training was effectively employed to train a Seq2Seq architecture using REINFORCE policy gradients to significantly improve prediction accuracy for text generation on a masked language modeling (MLM) task. Although authors found that policy gradients were effective when accompanied by a learned critic agent, there is also research detailing disadvantages of reinforcement learning methods (e.g., high variance in gradient estimation, slow convergence, and vulnerability to convergence to false optima) (Williams, 1992; Marbach and Tsitsiklis, 2003) that contribute toward poor generated sequence quality.

To side-step these shortcomings, researchers have been looking for efficient ways to train text GANs that rely on discrete nodes rather than policy gradients. One such study by Gao et al. (2019) demonstrated the successful application of GAN fine-tuning by implementing BERT (as both a generator and discriminator) and MLM in order to improve prediction of single mixed-in foreign words in generating code-switching text data. Experiments in this study showed that utilizing GAN training in conjunction with BERT MLM for data generation contributed to a 1.5% reduction in automated speech recognition English word error rate. These significant results indicate that GAN successfully increased similarity between the generated text and training input text when generating only single words. More research is needed in this area to confirm the effect on longer sequence generation.

The purpose of the current study was to investigate the effects of GAN fine-tuning on the pre-trained BERT architecture by developing a text-based NLG system. Based on the results of Wang and Cho (2019), an MLM procedure using Gibbs sampling was utilized as a method for extracting sequences from BERT. In an attempt to improve sample quality of these generations and to clarify the effect of fine-tuning on such a generation method, the proposal by Fedus et al. (2018) for training a text generation model using in-filling with the GAN framework was also incorporated to build upon BERT's knowledge. Finally, in order to avoid some of the shortcomings of policy gradients and to extend the findings of Gao et al. (2019) onto longer generated word sequences, the current system was trained to generate

sequences of length 40 by including BERT as both the generator and discriminator in GAN. I hypothesized that the introduction of GAN fine-tuning would significantly improve the quality of text generation in the current system. Perplexity and BLEU scores were used as metrics for evaluation. Results indicate that the model is more likely to produce text that is reflective of the training data, and consequently more well-formed and coherent.

1.1 Road Map

Chapter Two of this thesis provides a comprehensive discussion of literature that introduces the reader to the important topics referred to later in the study: natural language generation, generative adversarial training, and mode collapse. Next, Chapter Three provides a detailed description of the design, implementation, and procedure of the current study. Relevant details about the dataset, system architecture, generation algorithms, training, and evaluation are included in all sections such that results could be re-produced by other investigators or practitioners. Chapter Four describes the variables that were analyzed and presents the results of testing, and Chapter Five continues where this introduction left off, providing a review of the important details mentioned (i.e., hypotheses, results) and discusses relevancy with previous work and possible limitations. Last but not least, Chapter Six provides a conclusion summary that discusses future applications for the results.

Chapter 2
LITERATURE SURVEY

The previous chapter provided a high overview of the topic and introduced the background and motivation for conducting an investigation on the effectiveness of GAN training for BERT NLG. Moving forward with this foundation, the current chapter provides a survey of previous literature to develop familiarity with work related to integral parts of the current system. Special emphasis is given to the description and analysis of prior research in NLG, GAN training, and the issue of mode collapse in order to build an understanding of prior models and algorithms from related attempts.

2.1 Natural Language Generation

2.1.1 MASKGAN: Better Text Generation Via Fill In The ____

Fedus et. al., (2018) seek to answer the question: how can we avoid the issues associated with current neural text generation models? Recurrent Neural Networks (RNNs) are currently the primary model used for text sequence generation and sequence labeling because of high performance on text prediction based bench-mark tasks. However, low perplexity does not necessarily translate to quality in text generation tasks. Training methods involving maximum likelihood estimation or teacher forcing may reduce sampling quality because the model may have to predict based on sequences of words unseen during prior training. Fedus et. al., (2018) propose an actor-critic conditional GAN using a "Seq2Seq" model that fills in missing text conditioned on the surrounding context and attributes error at each time step. REINFORCE policy gradients were used to propagate through the softmax layer to train the generator. Knowing if this proposed method generates higher quality text may indicate better methods for generating natural language text.

To evaluate the proposed model, investigators conducted experiments in which MaskGan, and a separately trained MLE benchmark model, generated samples using the Penn Tree-Bank (Marcus et al., 1993) and Internet Movie Database (Miller et al., 2009). Perplexity was evaluated on these samples and results demonstrated that GAN training does not improve model perplexity in comparison to the MLE benchmark. However, the MaskGAN algorithm did achieve significantly higher sample quality when samples were assessed for grammar, topicality, and overall quality by the human evaluators on Amazon's Mechanical Turk. Finally, the number of unique n-grams produced by the generator that occur in the validation corpus were also calculated and indicate that MaskGan suffers from mode collapse, a common problem in GAN training described in more detail in section 2.3.

2.1.2 BERT has a Mouth, and It Must Speak: BERT as a Non-Equilibrium Language Model

Wang and Cho (2019) seek to answer the question: how can Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) be utilized as a traditional language model? BERT has achieved high success in several other natural language understanding tasks (e.g., constituency parsing and machine translation) but because BERT architecture is trained using a bidirectional masked language modeling objective, it does not resemble traditional unidirectional language generation models and therefore does not appear to inherently be capable of text generation. By developing a strategy that allows BERT to be used as a stand-alone language model would transfer its state-of-the-art knowledge and success toward calculating the probability of a test sentence or sampling from the model as a text generator.

Authors attempt to answer this question by demonstrating that BERT can be defined as a Markov Random Field with pseudo log-likelihood training. Additionally, they use a Gibbs sampling procedure to sample the model for text generation. Through BERT's masked language model pre-training, the model learns a probability distribution over sentences by masking out one or more words of a training sample, and then making a prediction of the mask based on the bidirectional context. In order to sample from this distribution as a stand-alone model, researchers provide a random initial state represented as an all-mask sequence. The

procedure for sampling used is a sequential-parallel generation algorithm in which a single token for a position chosen uniformly at random is generated. Once a pre-defined number of time steps has been surpassed, a token will have been generated for each position, and the process may be terminated or repeated using this newly generated sentence.

Investigators used this sampling procedure in several experiments to evaluate the quality and diversity of sample generations from BERT-large and BERT-base in comparison to another off the shelf language model, OpenAI Generative Pre-Training Transformer (Radford, 2018). As a metric of quality, BLEU scores (Papineni et al., 2002) were used to compare generations with original data distributions. Furthermore, the Gated Convolutional Language Model (Dauphin et al., 2017) was also used to calculate a perplexity score. As an evaluation of diversity, a Self-BLEU score and the percentage of n-grams that are unique, when compared to the original data distribution and within the corpus of generations, were also reported.

Wang and Cho (2019) confirm their hypothesis that BERT could not only be used as a language model, but could also produce highly fluent and diverse text generations. In general the results show that BERT-base performs better than BERT-large in quality and fluency. However, GPT outperformed both BERT models by generating higher quality but less diverse samples.

2.2 Generative Adversarial Training

2.2.1 Generative Adversarial Nets

As shown in Figure 2.1, GAN is a framework by Goodfellow et. al (2014) in which two deep neural models are trained simultaneously: a generator that attempts to learn a given dataset's probability distribution in order to generate new samples, and a discriminator that tries to estimate the probability that a sample originated from the true dataset ('real') rather than from the generator ('fake'). The two networks are trained in a min-max competition in which the discriminator attempts to maximize the probability that it will correctly classify a sample

as from the dataset as 'real', or generated as 'fake'; in contrast, the generator attempts to maximize the probability that the discriminator will make a mistake and incorrectly classify a generated sample as from the dataset. Throughout training, backpropagation allows both networks to update their weights and continuously improve until they reach an equilibrium point.

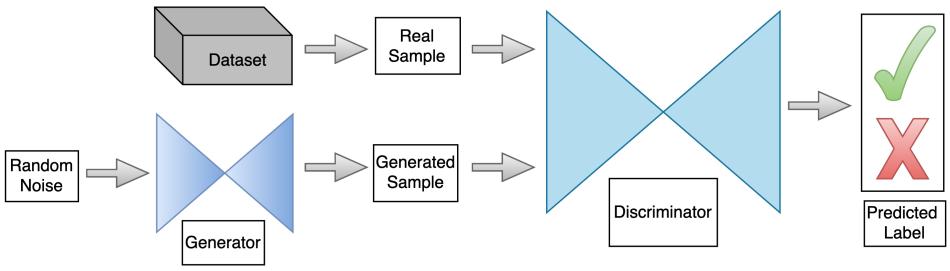

Figure 2.1: GAN framework as depicted by Goodfellow et al. (2014).

The equilibrium point, known as "Nash equilibrium", is when the generator creates 'fake' samples that are so well crafted that they appear to be directly from the dataset in such a way that the discriminator is unable to distinguish and its accuracy drops to random chance. Through qualitative and quantitative evaluation, generated samples are found to be at least competitive with the better generative models in literature and hold high potential for future research.

2.2.2 Code-Switching Sentence Generation by Bert and Generative Adversarial Networks

Gao et al. (2019) attempt to solve the problem of data availability by code-switching text used in language model training. Because people tend to mix foreign words in spoken language more often than in written language, it is difficult to collect enough data to effectively train a language model in this domain. Previous statistical-based approaches have achieved high automated speech recognition accuracy; however, these approaches still are unable to obtain the same accuracy found in mono-lingual tasks. In the current study, investigators

reflect on previous Mandarin-English code-switching datasets. Based off their observations of complexity and other shortcomings (e.g., vocabulary limitations, phrase or sentence transformations, and spoken citations), investigators propose a methodology that replaces the 4-layer neural network LSTM model from prior literature with BERT trained on Chinese data (BERT-C) for generating code-switching derived from Chinese text resources. Unlike the work of Wang and Cho (2019), only English words and randomly selected Chinese words were masked and predicted by BERT using a single forward pass. Furthermore, to assure similarity between generated samples and real sentences, investigators also experimented with GAN training with BERT as both the discriminator and generator. A more robust artificial dataset generation method and architecture could greatly increase the accuracy of ASR systems for code-switching as a downstream task.

Investigators defined model effectiveness as high English word prediction accuracy, low language model perplexity, and high ASR word error rate reduction across several experiments. In the first experiment, model type was varied to evaluate how training affects prediction accuracy of English words using the following conditions: BERT-C, BERT-C with random masking for all words, BERT-C with masking only English words, BERT-C with GAN fine-tuning, and Bert-C with GAN fine-tuning selecting only the generated sentences that predicted English tokens. For each condition, a similar algorithm to the original BERT pre-training was used, with the exception that batch size is reduced to 32, the maximum sequence length is set to 128 and the learning rate is fixed as 2e-5. The BERT-C model with GAN fine-tuning and English selection module performed highest with a prediction accuracy of 60% and thus was the model used in the subsequent perplexity experiments.

In a second experiment aimed to evaluate the language models (LM) perplexity (PPL), code-switching data was generated using the proposed generator and monolingual Chinese texts from the AISHELL corpus (Bu et al., 2017). A selection threshold was manipulated across three levels (e.g., no threshold, $log(P) \geq 2$, and $log(P) \geq 1$); the AISHELL corpus was also included as a control. For all levels, the original LM was used to evaluate perplexity scores. Mixed-lingual texts generated from the proposed model demonstrate lower perplexity.

Among the generated texts, those with more strict thresholds also gave lower perplexity, indicating higher similarity with real code-switching texts.

Additionally, Gao et al. (2019) introduced the previously generated text to the OC16-CE80 corpus training data (Wang et al., 2016) in order to train new LMs. After training, these expanded LMs were used to calculate the perplexity of the OC16-CE80 corpus test data. Under these circumstances there was an increase to PPL in the control condition due to dissimilarity in corpora. However, the expanded generator with a selection threshold $log(P) \geq -1$ demonstrated the lowest perplexity score.

In a third experiment, the expanded language models were applied to an ASR system. Word error rate (WER) was calculated using the test set from the OC16-CE80 corpus. Results show that LMs created using the additional generated data contribute lower WER. The LM with the selection threshold $log(P) \geq 2$ resulted in the lowest WER for English words.

2.3 Mode Collapse

2.3.1 Improved Techniques for Training GANs

Salimans et al., (2016) describes the issue of mode collapse for GAN as the point when the generator produces a limited group of identical outputs, or even just a single repeated output, regardless of variation from the noise vector input. Because of stochastic gradient descent, the generator continuously optimizes toward generating text that contains the exact attributes for which the discriminator deems to be realistic. However, this poses a problem if each of the generator's outputs are processed independently by the discriminator, or if the generator has been trained more extensively than the discriminator because there may not be coordination between discriminator gradients. As a result, the gradients may point in the same direction for different generated input samples.

Under these circumstances, there is no means for the discriminator to signal to the generator to vary its output and the generator will perpetually fail to produce varied and unique

output. In the case that the discriminator does correctly predict the text to be artificially generated, gradients will only relocate the generator's target for producing realistic output to a new collapsed point in the output space. That is to say, assigning lower probability weights to an already collapsed mode fails to incite diversity in the generator output because the generator will simply re-allocate another collapsed distribution to comply.

One solution that enables the discriminator to encourage the generator to re-calibrate while a mode is in the process of collapse is called Mini-Batch Discrimination. This technique allows for the discriminator to review several samples of generated text at once, detect mode collapse by evaluating similarity among the batch of samples. When the mode starts to collapse the similarity of the batch rises and triggers the discriminator to penalize the generator and effectively lower the assigned probabilities at the collapsing mode.

Chapter 3

METHOD

The current study recreates the GAN framework as described by Goodfellow et al., (2014). This work diverges from its predecessor, however, by incorporating a massively pre-trained neural model, BERT, as both the generator and discriminator modules' central architecture. More recently, Gao et al. (2019) conducted a related study in which they developed a code-switching data generator based on BERT and GAN, but this work focused on only predicting single mixed-in foreign words considering the bidirectional context. Thus, the current study aims to improve generatation of longer sequences of words or phrases. Although BERT is not traditionally known to be effective at text generation, previous work by Wang and Cho (2019) presented a novel sampling procedure that has been modified and applied in the current study to avoid reaching memory limits while training.

3.1 Dataset

To build upon the previous work of Wang and Cho (2019), 5000 sentences from the test set of WikiText-103 (Merity et al., 2016) and 5000 sentences from The BookCorpus (Kiros et al., 2015) were used for training and evaluation.

3.1.1 WikiText-103

WikiText-103 (WT103) is a long term dependency language modeling dataset that compiles over 100 million tokens extracted from the set of verified 'Good' and 'Featured' articles on Wikipedia. WT103 is derived from full articles and retains the original case, punctuation and numbers. WT103 is also 110 times larger than the Penn TreeBank (Marcus et al., 1993).

3.1.2 The BookCorpus Dataset

The BookCorpus Dataset (TBC) is a corpus of 11,038 books, free, unpublished books from 16 genres gathered from the web. Together they compile 74,004,228 sentences. Books of less than 20k words were filtered in order to reduce noise.

3.2 Data structure and Algorithms

3.2.1 System Architecture

As seen in Figure 3.1, the current study recreates the GAN architecture as described by Goodfellow et al., (2014), and with using HuggingFace's Transformers library with Pytorch (Wolf et al., 2019; Paszke et al., 2019). The *bert-base-uncased* model with a *BertForSequenceClassification* head was used for the generator module, and a separate *bert-base-uncased* model with the *BertForSequenceClassification* head was used for the discriminator module. As an optimizer, both the generator and discriminator implemented *AdamW*, but with different hyperparameters. More specifically, the learning rate for the generator was 2e-6 with epsilon value of 10e-4; whereas, the learning rate for the discriminator was 2e-5 with epsilon value of 1e-8. Finally, both modules utilized the binary cross entropy loss function *BCEWithLogitsLoss*.

3.2.2 Text Generation

In following suit with Wang and Cho (2019), BERT can be defined as a non-equilibrium language model. As such, the generation algorithm used in the current study utilizes a Markov Chain Monte Carlo sampling procedure, known as Gibbs sampling, and BERT's Masked Language Modeling head in order to sample from the probability distribution. First, a random state vector of a specified length is initialized (eg., [[CLS], [Mask], [Mask], [Mask], [Mask], [Mask], [Mask], [SEP]]). At each iteration i, the position i_t is sampled uniformly at random from *[1, . . . , T]* and the selected index is replaced with a masked token, *[MASK]*. Next, a forward pass is performed through BERT and a probability distribution is generated.

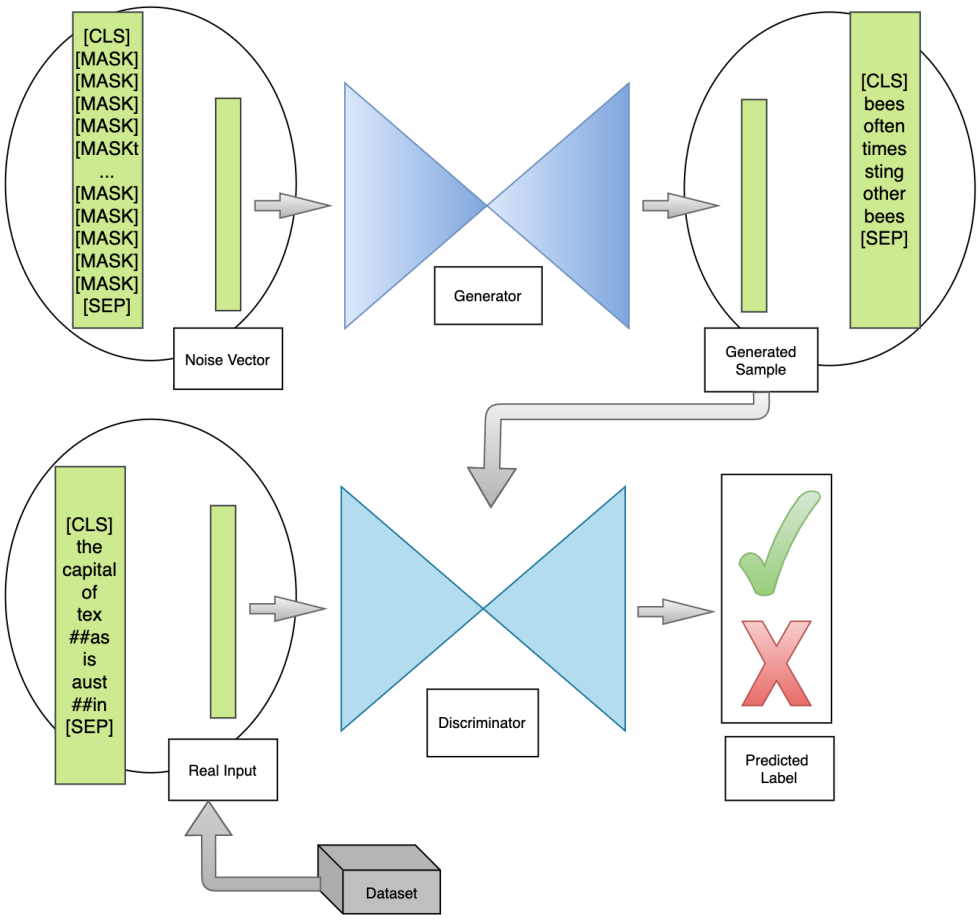

Figure 3.1: The current system architecture.

A smoothing parameter value, *temperature*, is applied to the probability distribution, and a value is sampled from the *top-k* probabilities. The element at index i is replaced with the newly selected value, and the process repeats until the specified maximum number of time steps is reached. A *burn-in* hyperparameter also exists such that for the specified number of steps *top-k* is ignored and any value within the distribution can be sampled.

Wang and Cho (2019) suggests using a maximum of 500 time steps in their evaluation; however, training requires gradients to flow through the model at each pass. Because this hyperparameter entails accumulating gradients for 500 passes through BERT, a change had to be made because of GPU memory constraints. The training procedure was modified to

limit the number of passes through BERT by re-using the same probability distribution from a single forward pass for all indices of the input vector. Therefore when training, a single step corresponds to replacement of all indices of the input vector with the newly selected values in parallel, not just single indices like when generating for evaluation.

3.2.3 Reparameterization Tricks for Backpropagation

As discussed in Jang et. al., (2017), using discrete representations for text has many benefits (e.g., discrete representations are more interpretable (Chen et al., 2016), and more computationally efficient (Rae et al., 2016)); however, the computational graph for re-visiting the chain of operations that allow for the backward propagation of errors breaks due to the usage of non-differentiable functions. Past literature has put forth several methods to side-step the problem of backpropagating through discrete nodes (e.g., REINFORCE policy gradients (Williams, 1992), and avoiding discrete space using an encoder (Subramanian et al., 2017)); however, because of known issues with training stability and convergence in REINFORCE and the lack of easily obtained continuous representations of real dataset sentences, the Straight-Through Gumbel Softmax Reparameterization Trick (Jang et al., 2016) was chosen as a solution in the current study.

The Straight-Through Gumbel Softmax Reparameterization Trick is employed to relocate non-differentiable nodes to the outside or start of a model in order to allow the gradients to flow unrestricted through the remaining nodes. In the current study's generation algorithm, discrete representations require the use of a probability sampling function (i.e., a non-differentiable function containing a random node) to select a token value from a list corresponding to the top-k predicted probabilities. For this reason, the source of randomness for sampling must be shifted to another variable in order for backpropagation to take place. By taking the probability distribution and adding "gumbel noise" (i.e., G = -log(-log(Uniform(0,1)))), the origin of randomness is moved to outside of the computational model and a differentiable quasi-categorical continuous sample is obtained by performing a softmax

parameratized by temperature, τ:

$$y_i = \frac{exp((log(x_i) + G_i)/\tau)}{\sum_{j=1}^{k} exp((log(x_j) + G_j)/\tau)} \text{ for } i = 1, ..., \text{k}$$

In order to obtain a 'hard' one-hot vector, this soft representation is discretized using *argmax*, and with the use of clever gradient arithmetic, the soft gradients are preserved making it possible to sample from the generator and also backpropagate for gradient descent.

Another key issue with applying GAN with text is that the gradient of a discrete function is either infinite or undefined, and therefore the computational graph for backpropagation is also broken where discrete labels (i.e., words represented as discrete integer values) are passed through an embedding layer and converted to a higher dimensional floating point representation. To be more specific, the computational graph in the current study is broken when a generated text sample is passed from the generator to the discriminator for classification. When loss from the classification is calculated, the gradients are unable to backpropagate through the discriminator into the generator to update the generator's weights.

In the current study's GAN training algorithm, the generator and discriminator's weights are not trained simultaneously; therefore, in training the generator alone, it is not necessary for gradients to move backward through the discriminator and a straight-through reparameterization trick can be used to remove the discriminator's nodes from the computation graph. As shown in Figure 3.2, gradients can be transferred from the generator's loss object to the sample object that was initially passed into the discriminator, allowing for the gradients to skip the discriminator and continue calculating gradients for the generator.

3.2.4 Double-Sided Label Smoothing

Label Smoothing is a technique that has been shown to reduce the vulnerability of neural networks to adversarial examples (Szegedy et al., 2016). In order to prevent network overconfidence in the current study, the GAN architecture smooths the labels used in calculating loss. The commonly used 'fake' and 'real' labels (i.e., '0' and '1' respectively) are represented

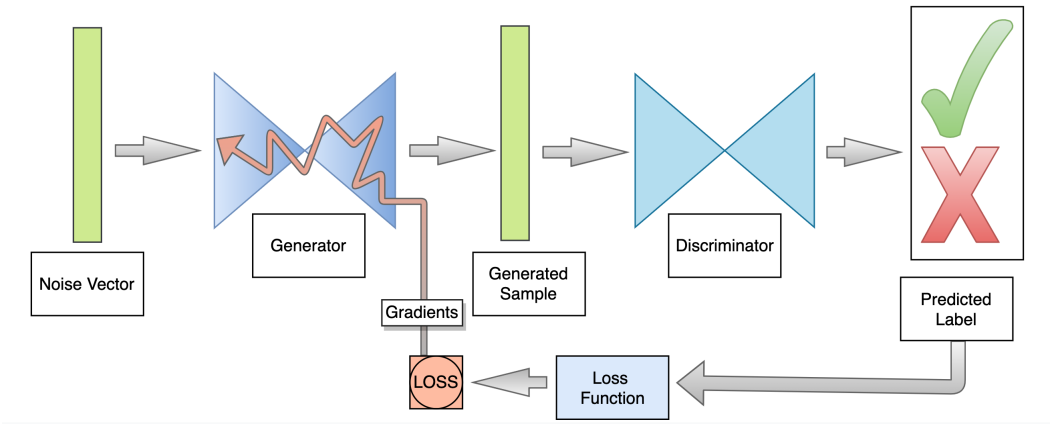

Figure 3.2: Illustration of a backward pass performed on the generator. Gradients are shown to bypass the discriminator because of a reparameterization trick.

in the current study with random floating point numbers within the ranges [0,.1] for 'fake' and [.9,1] for 'real' labels.

3.3 Procedure

In the current study, twenty trials were run individually using unique random seeds and averages were taken for all evaluation measures. Each trial was carried out in the following order: data collection and processing, preliminary evaluation of GPT and the BERT generator prior-to-training, training, and evaluation of the BERT generator post-training.

3.3.1 Data Collection and Processing

For use in evaluation, 10,000 sentences from the datasets were read-in from file and processed line by line. Line-initial and line-final white space, if any, was removed and lines were split on the remaining white space to create lists of word tokens. Word tokens were lower cased and some were replaced according to the parent corpus: WT103 replaced '@@unknown@@' with the special token '[UNK]' and TBC replaced backtick characters (i.e., ' ` ') and escaped apostrophe (i.e., ' \' ') to an escaped quote character (i.e., ' \" ').

Additionally to be used in training, lists of tokens required further preparation for use with BERT. After replacement, word tokens were then converted to BERT Tokenizer tokens, and truncated to the length of 40; however, the BERT class (i.e.,'[CLS]') and separator (i.e., '[SEP]') special tokens were inserted respectively at the initial and final index of each list; thus the maximum length increased to 42. Sequences of words shorter than length 10, and sequences that contained an '[UNK]' special token were removed. Word sequences were sorted by length into 40 batches of size 32, such that all sequences within a batch contained the same number of BERT Tokens and all lengths were equally represented while quantity permitted (McCormick, 2020); any remainder was discarded. Finally, the *train_test_split* module from sklearn was utilized to randomly divide 10% of the data (i.e., 128 lines) into a validation subset and the remaining 90% (i.e., 1152 lines) into a training set.

3.3.2 Training and Validation

In following a similar algorithm to the Deep Convolutional GAN (Radford et al., 2016), the initial step of the training loop is to train the discriminator as a classifier that distinguishes real data from generated data. The goal is "to update the discriminator by ascending its stochastic gradient" (Goodfellow et al., 2014). First, 1152 samples from the training dataset are labeled as 'real' according to the notation described in section 3.2.4 and passed to the discriminator in mini-batches of size 32. Loss is calculated using the discriminator loss function and the gradients are accumulated then clipped to 1.0. Subsequently, a batch is generated from the model's generator using hyperparameters identical to the batch of 'real' samples (i.e., *number of samples*, *batch size*, and *sample length*) and with a *temperature* of 1.0, *top-k* of 10, *burn-in* of 0, and by only sampling each index of the vector a single time. Generated samples are labeled as 'fake' and passed through the discriminator. Loss and gradients are accumulated and clipped exactly as before. After having accumulated gradients from both the 'real' and 'fake' batches separately, a step of the discriminator's optimizer is called and the discriminator's weights are updated; thus, completing the discriminator's training for the mini-batch.

After completing the discriminator's training, the second step is to train the generator. The generated 'fake' batch from the previous step is used again to perform another forward pass through the discriminator; however, instead of the discriminator loss function calculating the loss itself, the discriminator's output predictions are labeled with 'real' labels, passed to the loss function of the generator, and gradients are accumulated and clipped to 1.0. After having calculated the loss and accumulated gradients for the generator, a step of the generator's optimizer is called and the generator's weights are updated.

A validation process follows each training epoch. During this process, gradient accumulation is frozen in order to evaluate the discriminator accuracy without affecting model weights. A mini-batch of 32 samples from the validation dataset is paired with a batch of newly generated validation samples and shuffled. A generated validation batch matches the paired batch of samples from the dataset by *number of samples*, *batch size*, and *sample length*, and otherwise take to the hyperparameter values described in 3.3.3. As such, 64 samples are evaluated with each epoch.

3.3.3 Evaluation

Identical to Wang and Cho (2019), 1000 text samples of length 40 tokens were generated using *top-k* of 100, *temperature* of 1.0, and indices were sampled uniformly at random 500 times with a *burn-in* of 250 in order to examine the generator output for quality and diversity, before and after training. As a baseline, GPT text was also generated and evaluated under identical conditions. In following Wang and Cho (2019) and Yu et al., (2017), perplexity and BLEU were used as metrics of quality; whereas, Self-BLEU was used as a measure diversity.

Perplexity

In natural language processing (NLP), perplexity is an efficient metric from information theory for comparing the ability of statistical models to predict testing data (Jurafsky and Martin, 2009; Cover and Thomas, 2006). It can be defined as $PP(p) := 2^{H(p)} = 2^{-\sum_x p(x) \log_2 p(x)}$, where H(p) is the entropy of the distribution and x ranges over events. In the current study,

perplexity is evaluated on generated samples through use of an outside language model, *transformer_lm.wiki103.adaptive*, pre-trained on one of the original training data distributions (i.e., WT-103). Because this model was trained using the identical dataset that the generator was trained on, perplexity is calculated on a closed-vocabulary, avoiding the use of the unknown vocabulary token, '[UNK]'.

BLEU

Initially a method developed for automatic machine translation by Papineni et al. (2002), BLEU is a score within the range of [0,1] that compares a candidate text sample to reference text using n-gram overlap as a measure of similarity. The rationale in the original usage was that a good machine translation would compare to a professional human translation with higher similarity receiving a score closer to 1.0. BLEU was also one of the first metrics to correlate well with human judgement, therefore it has remained popular as an inexpensive tool for also evaluating NLG. In the current study, BLEU analyzes n-grams of $n \leq 4$ to compare generated text samples with the original data distributions that BERT was pre-trained on (i.e., WT-103 and TBC). Because these reference are human-derived and assumed to be generally well formed, higher BLEU scores would indicate higher coherence or fluency in candidate generated text.

Self-BLEU

Self-BLEU is a metric proposed by Zhu et al. (2018) to evaluate the ability for a model to generate unique samples. The same BLEU metric that assess similarity between text samples in the previous subsection is carried out, however, generated samples are instead evaluated in reference to all remaining samples within a batch. In contrast to BLEU, higher Self-BLEU scores are an indication of low diversity and one of the identifying characteristics of mode collapse. For this reason, Self-BLEU is expected to be low to signify greater dissimilarity among samples in a generated batch.

Chapter 4

RESULTS

The effect of GAN fine-tuning on BERT generation was tested. Preliminary GPT scores were also tested as a baseline. Table 4.1 presents a small uncurated sample of the generated sentences on which evaluation was carried out. Author's judgement from reading these sentences suggests that those generated after fine-tuning are more coherent.

Table 4.2 presents the average BLEU scores and standard deviation of text generated before GAN fine-tuning and afterwards. Looking at the pattern of results in Table 4.2, it appears that BLEU scores for BERT, in general, were significantly higher after fine-tuning rather than prior. The degree to which BLEU was affected by training, however, depended on the reference corpus used in evaluation. More specifically, when generated text was evaluated against the WT103 corpus, BLEU scores after training (M=9.66, SD=0.25) were 6.41% higher than before training (M=9.06, SD=0.23); whereas when generated text was evaluated against the TBC corpus, there was a much larger effect in which BLEU scores after training (M=11.45, SD=0.42) were 35.96% higher than before training (M=7.96, SD=0.25). Average BLEU scores comparing a single sentence with all other sentences within a sample were 24.37% higher after training (M=11.51, SD=0.72) than before training (M= 9.01, SD=0.27). The GPT baseline was higher than BERT across all conditions.

Furthermore, Table 4.3 presents the average percentage of generated n-grams and standard deviation that are unique relative to each of the other samples within a 1000 sample batch (Self-BLEU), 5000 randomly selected sentences from WikiText103 test set, and 5000 sentences from The BookCorpus. Across all references, the percentage of BERT generated n-grams decreased after fine-tuning. Bigrams relative to the TBC varied the most by decreasing roughly 7% after training. The percentage of unique n-grams generated from GPT

7 a . m . ... where am i now ? is it my imagination (you know almost nothing about michael spelman) ? you can honestly say not whether i tell you how long i know .	and the creature blinked again and again and then everything else was gone . and a memory struck me : of the sign on the wall and the text from doom12 and the picture of spenser .
i was so glad i had walked in here in my get - up on top - no indeed - ensemble . dark hair and sharp cologne fumed up a just - opened , black mini card table .	i noticed that people are walking in the park . it is a different place entirely . slowly but quickly outside in the distance a brief night fog passed by revealing the black silhouettes of something moving somewhere .
the collective also included walker and white as african - american artists ; and sculptor paul wright . his obituary in the natchez chronicle of napa valley claimed that he had worked at the artist colony .	all that was normal his whole life ; all that was making all his friends remember the joys of their time . neal and georgie were making commercials , showing men and women in suits with funny faces .
his brother provided they could get a low - budget comedy film in their own way - but at the same time acting and play over movies were a priority for the son (or for the actor) .	the allen marcus ' songs east meets west , monkey house , " hate me , " and others , both in the " friends " form , were also featured in viral videos directed by daniel del toro .
our loved ones are lost or in danger . but must we leave yet , then ? someone called , and everyone shows with what they want (bob calls) . we must look for our loved ones .	people with supernatural powers are undead ! you know they sweep through the building main parking lot saving the owners , the crew , from the building , anywhere ! " " you can talk about murder , right ?

Table 4.1: Uncurated generations from BERT Preliminary (left) and BERT Fine-tuned (right).

were also significantly lower than fine-tuned BERT on the same condition with a decrease of nearly 29% unique bigrams in reference to TBC.

Finally in addition to BLEU, the perplexity of generated text samples given by an outside model were also analyzed. Table 4.4 presents the average perplexity and standard deviation of text when generated before GAN fine-tuning and afterwards. This method of training also revealed a significant main effect such that text generated after the model was fine-tuned (M=236.17, SD=9.75) showed a 19.99% lower perplexity than when generated prior to fine-tuning (M=288.6, SD=10.23).

Model	Corpus-BLEU		Self-BLEU
	WT103	TBC	
BERT - Preliminary	9.06 ±0.23	7.96 ±0.25	9.01 ±0.27
BERT - Fined-tuned	9.66 ±0.25	11.45 ±0.42	11.51 ±0.72
GPT	10.77±0.33	30.43 ±0.40	39.09 ±0.55

Table 4.2: BLEU score comparison between 1000 generated text samples to 5000 randomly selected sentences from WikiText103 test set, 5000 sentences from The BookCorpus, and to all other samples within the same generated batch (Self-BLEU).

Model	% 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 - Preliminary	62.70±0.42	92.61±0.27	98.41±0.19	58.48±0.37	91.75±0.16	98.62±0.12	61.83±0.51	92.25±0.20	98.66±0.13
BERT - Fined-tuned	57.18±0.89	90.23±0.56	97.67±0.29	54.11±0.66	90.02±0.40	98.13±0.19	54.07±0.85	88.98±0.55	97.89±0.22
GPT	31.49±0.38	67.99±0.49	87.86±0.44	33.81±0.31	73.52±0.43	91.48±0.33	25.87±0.29	65.65±0.44	88.67±0.37

Table 4.3: Percentage of generated n-grams that are unique in comparison to other samples within a batch (Self-BLEU), 5000 randomly selected sentences from WikiText103 test set, and 5000 sentences from The BookCorpus.

Perplexity	
Model	Mean
BERT - Preliminary	288.62 ±10.23
BERT - Fined-tuned	236.17 ±9.75
GPT	176.65 ±4.50

Table 4.4: Average perplexity obtained from 1000 generated text samples using the fairseq transformer_lm.wiki103.adaptive model (Baevski and Auli, 2019).

Chapter 5

DISCUSSION

The current study was designed to examine how the introduction of GAN fine-tuning to the BERT pre-trained architecture affects NLG. The hypothesis was that the quality of NLG would significantly increase after fine-tuning. Metrics used to evaluate quality were BLEU scores that compared generated text to the reference training corpora, and perplexity scores obtained through an outside model. The results of the current study indicated that the quality of BERT NLG was significantly increased by this method of fine-tuning. On average, perplexity of generated test samples was significantly lower after fine-tuning, indicating a greater sense of coherence or fluency; analogously, mean corpus BLEU scores were found to be significantly higher after training than prior, indicating a stronger ability for the model to sample text similar to a well formed dataset.

These results were consistent with previous studies in that there was a significant increase in quality due to using generative adversarial network training. One such example of this previous work is that of Gao et al. (2019), in which a significant effect was found when GAN training was applied to BERT in order to generate single tokens in a code switching experiment. This consistency may be due to a non-apparent advantage in utilizing identical pre-trained models in both the discriminator and generator, contributing to more balanced adversarial training.

Despite the consistency in results with previous work, evaluation methods are known to be flawed as measures of quality or fluency (Hardcastle and Scott, 2008) (Çelikyilmaz et al., 2020) and may have affected the external validity of the current study. Pilot experiments demonstrated much higher improvement than what is reported in the current study, but visual inspection indicated that those samples may have suffered from semi-mode collapse.

This observation is consistent with Wang and Cho (2019) where it was noted that the higher scoring generations, those generated by GPT, "might have also collapsed to fairly generic and simple sentences". In the current study, Self-BLEU was high for GPT at around 39 and BERT Self-BLEU also increased after fine-tuning. These results may provide evidence that BLEU and perplexity artificially become skewed as a function of decreasing diversity if the generator begins to produce less diverse or repetitive output (i.e., mode collapse). Further investigation is needed to clarify the parallel between text diversity and quality in order to develop more holistic strategies for automatic evaluation of NLG. Until progress is made in this area, a more thorough human evaluation of word sequences would greatly benefit any future extension of the current study, despite the high cost (Papineni et al., 2002).

Another constraint of the BERT model, first mentioned in Wang and Cho (2019), remains unresolved in the current study and is still open for further investigation. In BERT generation, the length of the generated output hinges on a max length hyperparameter that must be specified prior to run-time. More specifically, the user must declare how many '[MASK]' tokens to add to the vector to later be un-masked and converted to the predicted tokens. This parameter poses a problem because for an NLG application to be practical a user should not need to dictate the exact length of the BERT response.

A proposed solution in piloting the current study was to remove the input vector's final '[SEP]' token. '[SEP]' is used to signal the separation of two sentences in the input to BERT for several NLP tasks (e.g., Question Answering, Natural Language Inference, and Next Sentence Prediction) (Devlin et al., 2019); therefore, by not including this token and by providing sufficient masked space, the generator should predict '[SEP]' independently when carrying out the same unmasking procedure used to generate tokens based on context. For example in the context of Question Answering, if provided with the token embedding, ['[CLS]', 'when', 'is', 'lunch', '?' '[SEP]', '[MASK]', '[MASK]', '[MASK]', '[MASK]', '[MASK]', '[MASK]', '[MASK]'], we can expect that after un-masking a sufficient response given this context, the model will also unmask a separator token when initializing another sentence due to what it has previously seen. For example, the model could predict, ['[CLS]', 'when', 'is', 'lunch',

‘?’, ‘[SEP]’, ‘lunch’, ‘is’, ‘at’, ‘noon’, ‘[SEP]’, ‘i’ ,‘eat’], and anything after the final predicted separator would be extraneous and could be trimmed. The problem with this solution lies in that BERT never predicted ‘[SEP]’ with enough probability for it to ever surface in preliminary experiments. Be this as it may, future work may be in investigating another method to allow BERT’s generation to independently sample sentences of variable length for use in an end-to-end question answering system.

Chapter 6

CONCLUSION

This study examined the effects of GAN fine-tuning on a pre-trained language model architecture in order to investigate its value in improving the quality of text generation. The results of this study indicated a significant increase in quality of generated text; thus, providing evidence that the introduction of GAN training is effective tool for improving NLG. This is an exciting frontier of research because text generation is a key component of language translation, chatbots, question answering, summarization, and several other applications that people interact with everyday (Çelikyilmaz et al., 2020). Further research in this domain could cause a cascading effect for research into any of these applications. For this reason, code has been released at `https://github.com/blongwill/bert-gan-thesis`. Future work should continue to explore the utility of GAN fine-tuning on other pre-trained models in order to expand the boundaries of state-of-the-art NLG.

BIBLIOGRAPHY

Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ByxZX20qFQ.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1171–1179, 2015. URL http://papers.nips.cc/paper/5956-scheduled-sampling-for-sequence-prediction\-with-recurrent-neural-networks.

Stevo Bozinovski. Reminder of the first paper on transfer learning in neural networks, 1976. *Informatica (Slovenia)*, 44(3), 2020. URL http://www.informatica.si/index.php/informatica/article/view/2828.

Hui Bu, Jiayu Du, Xingyu Na, Bengu Wu, and Hao Zheng. AISHELL-1: an open-source mandarin speech corpus and a speech recognition baseline. In *20th Conference of the*

Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment, O-COCOSDA 2017, Seoul, South Korea, November 1-3, 2017, pages 1–5. IEEE, 2017. doi: 10.1109/ICSDA.2017.8384449. URL https://doi.org/10.1109/ICSDA.2017.8384449.

Asli Çelikyilmaz, Elizabeth Clark, and Jianfeng Gao. Evaluation of text generation:t A survey. *CoRR*, abs/2006.14799, 2020. URL https://arxiv.org/abs/2006.14799.

Ching-Ting Chang, Shun-Po Chuang, and Hung-yi Lee. Code-switching sentence generation by generative adversarial networks and its application to data augmentation. *CoRR*, abs/1811.02356, 2018. URL http://arxiv.org/abs/1811.02356.

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In Aravind K. Joshi and Martha Palmer, editors, *34th Annual Meeting of the Association for Computational Linguistics, 24-27 June 1996, University of California, Santa Cruz, California, USA, Proceedings*, pages 310–318. Morgan Kaufmann Publishers / ACL, 1996. doi: 10.3115/981863.981904. URL https://www.aclweb.org/anthology/P96-1041/.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2172–2180, 2016. URL http://papers.nips.cc/paper/6399-infogan-interpretable-representation-\learning-by-information-maximizing-generative-adversarial-nets.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341, 2019. URL http://arxiv.org/abs/1906.04341.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 670–680. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1070. URL https://doi.org/10.18653/v1/d17-1070.

Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006. ISBN 978-0-471-24195-9. URL http://www.elementsofinformationtheory.com/.

Leyang Cui, Sijie Cheng, Yu Wu, and Yue Zhang. Does BERT solve commonsense task via commonsense knowledge? *CoRR*, abs/2008.03945, 2020. URL https://arxiv.org/abs/2008.03945.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941. PMLR, 2017. URL http://proceedings.mlr.press/v70/dauphin17a.html.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

William Fedus, Ian J. Goodfellow, and Andrew M. Dai. Maskgan: Better text generation via

filling in the _____. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=ByOExmWAb.

Yingying Gao, Junlan Feng, Ying Liu, Leijing Hou, Xin Pan, and Yong Ma. Code-switching sentence generation by bert and generative adversarial networks. In Gernot Kubin and Zdravko Kacic, editors, *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pages 3525–3529. ISCA, 2019. doi: 10.21437/Interspeech.2019-2501. URL https://doi.org/10.21437/Interspeech.2019-2501.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.

Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 4601–4609, 2016. URL http://papers.nips.cc/paper/6099-professor-forcing-a-new-algorithm-for\-training-recurrent-networks.

Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385 of *Studies in Computational Intelligence*. Springer, 2012. ISBN 978-3-642-24796-5. doi: 10.1007/978-3-642-24797-2. URL https://doi.org/10.1007/978-3-642-24797-2.

David Hardcastle and Donia Scott. Can we evaluate the quality of generated text? In *Proceedings of the International Conference on Language Resources and Evaluation, LREC 2008, 26 May - 1 June 2008, Marrakech, Morocco*. European Language Resources Association, 2008. URL http://www.lrec-conf.org/proceedings/lrec2008/summaries/797.html.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *CoRR*, abs/1611.01144, 2016. URL http://arxiv.org/abs/1611.01144.

Rafal Józefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. Exploring the limits of language modeling. *CoRR*, abs/1602.02410, 2016. URL http://arxiv.org/abs/1602.02410.

Dan Jurafsky and James H. Martin. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*. Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, 2009. ISBN 9780135041963. URL https://www.worldcat.org/oclc/315913020.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015. URL http://arxiv.org/abs/1506.06726.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In Marti A. Hearst and Mari Ostendorf, editors, *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, HLT-*

NAACL 2003, Edmonton, Canada, May 27 - June 1, 2003. The Association for Computational Linguistics, 2003. URL https://www.aclweb.org/anthology/N03-1017/.

Peter Marbach and John N. Tsitsiklis. Approximate gradient methods in policy-space optimization of markov reward processes. *Discret. Event Dyn. Syst.*, 13(1-2):111–148, 2003. doi: 10.1023/A:1022145020786. URL https://doi.org/10.1023/A:1022145020786.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. *Comput. Linguistics*, 19(2):313–330, 1993.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6294–6305, 2017. URL http://papers.nips.cc/paper/7209-learned-in-translation-contextualized\-word-vectors.

Chris McCormick. Smart batching tutorial - speed up bert training, Jul 2020. URL http://mccormickml.com/2020/07/29/smart-batching-tutorial/.

Chris McCormick and Nick Ryan. Bert fine-tuning tutorial with pytorch, Jul 2019. URL https://mccormickml.com/2019/07/22/BERT-fine-tuning.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *CoRR*, abs/1609.07843, 2016. URL http://arxiv.org/abs/1609.07843.

Frederic P. Miller, Agnes F. Vandome, and John McBrewster. *Internet Movie Database*. Alpha Press, 2009. ISBN 6130099681.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL, 2002. doi: 10.3115/1073083.1073135. URL `https://www.aclweb.org/anthology/P02-1040/`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019. URL `http://papers.nips.cc/paper/9015-pytorch-an-imperative-style-high\-performance-deep-learning-library`.

A. Radford. Improving language understanding by generative pre-training. 2018.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL `http://arxiv.org/abs/1511.06434`.

Jack W. Rae, Jonathan J. Hunt, Ivo Danihelka, Timothy Harley, Andrew W. Senior, Gregory Wayne, Alex Graves, and Tim Lillicrap. Scaling memory-augmented neural networks with sparse reads and writes. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*

2016, December 5-10, 2016, Barcelona, Spain, pages 3621–3629, 2016. URL http://papers.nips.cc/paper/6298-scaling-memory-augmented-neural-networks\-with-sparse-reads-and-writes.

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2226–2234, 2016. URL http://papers.nips.cc/paper/6125-improved-techniques-for-training-gans.

Sandeep Subramanian, Sai Rajeswar, Francis Dutil, Chris Pal, and Aaron C. Courville. Adversarial generation of natural language. In Phil Blunsom, Antoine Bordes, Kyunghyun Cho, Shay B. Cohen, Chris Dyer, Edward Grefenstette, Karl Moritz Hermann, Laura Rimell, Jason Weston, and Scott Yih, editors, *Proceedings of the 2nd Workshop on Representation Learning for NLP, Rep4NLP@ACL 2017, Vancouver, Canada, August 3, 2017*, pages 241–251. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-2629. URL https://doi.org/10.18653/v1/w17-2629.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.308. URL https://doi.org/10.1109/CVPR.2016.308.

Alex Wang and Kyunghyun Cho. BERT has a mouth, and it must speak: BERT as a Markov random field language model. In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 30–36, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-2304. URL https://www.aclweb.org/anthology/W19-2304.

Dong Wang, Zhiyuan Tang, Difei Tang, and Qing Chen. OC16-CE80: A chinese-english mixlingual database and A speech recognition baseline. *CoRR*, abs/1609.08412, 2016. URL http://arxiv.org/abs/1609.08412.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8:229–256, 1992. doi: 10.1007/BF00992696. URL https://doi.org/10.1007/BF00992696.

Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Comput.*, 1(2):270–280, 1989. doi: 10.1162/neco.1989.1.2.270. URL https://doi.org/10.1162/neco.1989.1.2.270.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019. URL http://arxiv.org/abs/1910.03771.

Jinxing Yu, Xun Jian, Hao Xin, and Yangqiu Song. Joint embeddings of chinese words, characters, and fine-grained subcharacter components. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 286–291. Association for Computational Linguistics, 2017a. doi: 10.18653/v1/d17-1027. URL https://doi.org/10.18653/v1/d17-1027.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2852–2858. AAAI Press, 2017b. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14344.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong

Yu. Texygen: A benchmarking platform for text generation models. In Kevyn Collins-Thompson, Qiaozhu Mei, Brian D. Davison, Yiqun Liu, and Emine Yilmaz, editors, *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM, 2018. doi: 10.1145/3209978.3210080. URL https://doi.org/10.1145/3209978.3210080.