Reverse Engineering Configurations of Neural Text Generation Models

Yi Tay
Google Research
Mountain View
yitay@google.com

Dara Bahri
Google Research
Mountain View
dbahri@google.com

Che Zheng
Google Research
Mountain View
chezheng@google.com

Clifford Brunk
Google Research
Mountain View
cliffbrunk@google.com

Donald Metzler
Google Research
Mountain View
metzler@google.com

Andrew Tomkins
Google Research
Mountain View
tomkins@google.com

Abstract

This paper seeks to develop a deeper understanding of the fundamental properties of neural text generations models. The study of artifacts that emerge in machine generated text as a result of modeling choices is a nascent research area. Previously, the extent and degree to which these artifacts surface in generated text has not been well studied. In the spirit of better understanding generative text models and their artifacts, we propose the new task of distinguishing which of several variants of a given model generated a piece of text, and we conduct an extensive suite of diagnostic tests to observe whether modeling choices (e.g., sampling methods, top-k probabilities, model architectures, etc.) leave detectable artifacts in the text they generate. Our key finding, which is backed by a rigorous set of experiments, is that such artifacts are present and that different modeling choices can be inferred by observing the generated text alone. This suggests that neural text generators may be more sensitive to various modeling choices than previously thought.

1 Introduction

The task of generating plausible sounding text from large generative neural networks has garnered significant attention recently (Zellers et al., 2019; Radford et al., 2019; Keskar et al., 2019). The study of these models has been a keen area of interest for many, resulting in research pertaining to the behavior of generation methods (Holtzman et al., 2019; Fan et al., 2018; Gu et al., 2017) as well as modeling techniques (Radford et al., 2019; Welleck et al., 2019; Dai et al., 2019; Radford et al., 2018).

This paper presents a focused empirical study of text generation artifacts, i.e., detectable 'signatures' that originate from certain modeling or decoding choices. There is a growing body of research that has focused on discriminating between human and machine generated texts (Gehrmann et al., 2019; Bakhtin et al., 2019; Ippolito et al., 2019). There is also extensive past research on authorship attribution (Sanderson and Guenter, 2006; Stamatatos, 2009; Stamatatos et al., 2018), for which it was always assumed that the authors were humans. This work takes a much more fine-grained approach by learning to distinguish between text generated by different machine variants. Do certain modeling choices leave more artifacts than others? In short, given a piece of generated text, can we determine the model configuration that generated this text?

The utility of our study manifests in multiple ways. First, the unraveling of artifacts in generated text enables better understanding of neural text generators, revealing potential fundamental weaknesses in modeling or generation schemes. Our study provides *relative* comparisons of the extent to which artifacts emerge from different modeling choices. Second, this research advances tracking the provenance and origination of machine generated texts, which has a range of useful applications pertaining to online trust and safety, thereby helping to mitigate the overall risk of these models in the wild. To the best of our knowledge, this is the first systematic and fine-grained study of detectable artifacts present in neural generated text.

Our contributions The overall contributions of this work can be summarized as follows:

- We present a largescale analysis of generated text with a special focus on studying artifacts produced by large generative models.
- We propose the new task of distinguishing between different fine-grained configurations

based on the generated text alone. The key idea is that classifiers performing better than random can capture configurationspecific artifacts.

- Our findings show that (1) modeling choices can be captured by simple classifiers through artifacts that are present in generated text alone, (2) the ease of prediction varies across different hyperparameter configurations, (3) word order is not that important in unraveling artifacts, i.e., artifacts are probably more related to word choice than syntax and composition and (4) distinguishing between model variants is much harder than predicting between *human-or-machine* only.

2 Related Work

There are many research efforts related to machine generated text. The work in this area can be characterized into two broad categories - (1) learning to generate better text and (2) learning to mitigate against generated text.

In the former, large generative models such as GPT/GPT-2 (Radford et al., 2018, 2019), CTRL (Keskar et al., 2019) and Grover (Welleck et al., 2019) have recently demonstrated the possibility of generating high quality text. The study of sampling methods for auto-regressive models has also been active where sampling methods such as top-k (Fan et al., 2018) and nucleus sampling (Holtzman et al., 2019) have been proposed.

Likewise, there have also been recent ongoing efforts that are targeted at distinguishing human text from machine generated text. (Gehrmann et al., 2019) proposed GLTR, a visual and statistical tool for aiding the detection of machine generated text. In a similar vein, (Bakhtin et al., 2019) proposed energy-based models. Statistical detection of machine generated text is possible largely due to the the presence of artifacts. To this end, the race between generators and discriminators is not entirely de-coupled. (Welleck et al., 2019) showed that a good generator is also a good discriminator.

Concurrent work (Ippolito et al., 2019) investigates the performance of human raters on the task of detecting machine generated text. Similarly, they also investigate the effect of model hyperparameters with respect to the ease of being detected by human raters.

Our work is also related to the field of authorship attribution (Stamatatos, 2009) which tries to identify the author behind a piece of text. A series of shared tasks have been proposed over the years (Stamatatos et al., 2018; Tschuggnall et al., 2017). The tasks have primarily focused on stylometry and text-based forensics. A key assumption is that authors leave behind distinguishable signatures (or artifacts) in their writings. Along a similar vein, our work re-imagines this task by considering different instances of generative models as authors.

The emergence of artifacts left behind by machine generated text is a peculiar and interesting phenomena. This work takes this direction further by studying the fine-grained artifacts produced by different modeling choices in hopes of better understanding machine generation in general.

3 Methodology

In this section, we introduce our experimental settings and setup.

3.1 Generative Model Configuration

Our experiments employ Grover (Zellers et al., 2019) as the text generator. We consider three generation configurations in our experiments. They are described as follows:

- **Model Sizes** - Generative models often come with pre-defined sizes that refer to the layer widths and parameterization. For Grover, the model size options include *Base*, *Large*, and *Mega*.

- **Sampling Method** - The sampling function controls the decoding process used to generate text. We explore variants of top-k (Fan et al., 2018), top-p nucleus sampling (Holtzman et al., 2019), and associated p/k values.

- **Conditioning** - Length of initial article conditioning. We define ℓ which is the amount of text given to the model. The initial ℓ tokens is concatenated at the end of the title sequence for the model to start generating.

In the design of our experiments, while there are countless possibilities to search for, we deliberately sought out settings that are most general and/or are considered fine-grained subtle changes. Such subtle changes are likely to be more challenging to detect compared to larger changes. For example, predicting Grover parameterization subsumes the task of distinguishing Grover versus GPT-2. We assume that if a model is able to solve the former, the latter becomes relatively trivial.

3.2 Classifier Models

We train a classifier model to discriminate between different model configurations. Generally, the task is framed as a multi-class classification problem where each model configuration is a class that is predicted. Models accept a sequence of tokens as an input. Sequences pass through a parameterized or non-parameterized encoder which are finally passed as input to a softmax classification layer.

In this work, we explore and benchmark the effectiveness of various encoding inductive biases such as recurrent, convolutional, and self-attention based models. This is primarily motivated as a probe into the problem domain, i.e., by witnessing the behaviour of different encoder architectures, we may learn more about the nature of these tasks/datasets.

Inductive Biases We consider the following encoding architectures (1) **BoW (Linear)** - a simple bag-of-words (BoW) baseline that averages the word embeddings and passes the average representation into a single linear classifier. $Y = Softmax(W(X))$. (2) **BoW (MLP)** - another simple baseline that builds on top of the Linear baseline. We add a single nonlinear layer with ReLU activation function, i.e., $Y = Softmax(W_2\sigma_r(W_1(X)))$. (3) **ConvNet** - We consider a 1D Convolution layer of filter width 3. We convolve over the input embeddings and pass the average (representation) into a linear Softmax classification layer. (4) **LSTM** - Similar to the CNN model, we encode the input sequence with an LSTM layer and pass the mean-pooled representation into a Softmax layer. (4) **Transformer Encoders** - We use 4-layered multi-headed Transformer (Vaswani et al., 2017) encoders with multi-head self-attention.

Task Name	Classes
p-Samp (P1)	$p \in [0.95, 0.90, 0.85]$
p-Samp (P2)	$p \in [0.95, 0.85, 0.75]$
p-Samp (P3)	$p \in [0.95, 0.90, 0.85, 0.80, 0.75]$
k-Samp (K1)	$k \in [10, 20, 30]$
k-Samp (K2)	$k \in [10, 30, 50]$
k-Samp (K3)	$k \in [10, 20, 30, 40, 50]$
Cond (C1)	$\ell \in [10, 50, 100]$
Cond (C2)	$\ell \in [10, 20, 30]$
Cond (C3)	$\ell \in [10, 20, 30, 40, 50]$
Size (S1)	$S \in \{Base, Large, Mega\}$

Table 1: List of proposed Machine Configuration Discrimination (MCD) tasks.

3.3 Experimental Setup

This section outlines our experimental setup.

News Corpora As a seed corpus, we use the CNN/Dailymail news corpus. This corpus is widely used in other NLP tasks (Hermann et al., 2015) such as question answering and summarization. The CNN/Dailymail corpus comprises approximately $90K$ news articles. Given an initial seed corpora of N news articles, we generate an additional collection of N machine generated articles for each configuration.

Tasks We define **ten** tasks as described in Table 1. These tasks aim at predicting the correct model configuration given the generated text. For all tasks, we use a maximum sequence length of 500 and split the dataset into 80%/10%/10% train, development, and testing splits. We include an additional variant $+h$ which denotes that we add the humanwritten article as an additional class to the mix.

Model Training For all models, we fix the word embeddings to $d = 64$. Embeddings are trained from scratch. All encoder hidden unit size is also set to 64. We tuned the dimensions of models in the range of $d \in \{16, 32, 64, 128, 256\}$ and found no noticable improvement beyond $d = 64$. We train all models for 50 epochs with a batch size of 64. We employ early stopping with patience 3 if validation accuracy does not improve. Final test accuracy is reported based on the best results on the validation set.

4 Insights and Findings

This section presents the insights and findings uncovered by our experiments. Table 2 and Table 3 present the core of our experimental results.

(1) Artifacts are found. Our experiments show that simple classifiers are able to distinguish fine-grained and subtle differences between modeling choices (e.g., top-p probabilities or condition length ℓ) in generated texts. In Table 2, we observe that all classifiers have an accuracy much higher than random chance (almost **double** in some cases), which suggests that distinguishing between different classes is relatively straightforward. In short, we are able to empirically conclude that all modeling choices leave behind some form of detectable artifacts.

Model	P1	P2	P3	K1	K2	K3	C1	C2	C3	S1	AVG
Chance	33.3	33.3	20.0	33.3	33.3	20.0	33.3	33.3	20.0	33.3	29.3
Bow-L	55.2	69.2	55.9	54.5	62.8	38.4	42.3	34.7	22.0	43.7	47.9
Bow-M	55.2	69.7	56.9	56.1	62.7	40.0	42.9	34.6	22.7	43.2	48.4
Cnn	55.4	69.6	57.5	55.5	63.9	40.3	43.0	35.1	23.1	43.7	48.7
Lstm	54.9	68.9	54.5	55.0	62.7	40.2	45.7	34.0	23.8	43.5	48.3
Trans.	53.7	70.2	59.7	55.2	63.4	40.5	43.9	34.4	24.0	42.2	48.7
% Gain	+66%	+111%	+199%	+68%	+92%	+21%	+37%	+5%	+20%	+31%	+66%

Table 2: Results on machine configuration detection. $\%$ gain provides a general sense of how prevalent artifacts are for a given configuration.

Model	P1	P2	P3	K1	K	K3	C1	C2	C3	S1	AVG
Chance	25.0	25.0	16.7	25.0	25.0	16.7	25.0	25.0	33.3	25.0	24.2
Bow-L	67.5	76.6	63.8	73.27	78.9	57.5	47.3	46.10	33.2	58.6	60.3
Bow-M	68.0	76.7	65.6	74.1	78.9	57.2	49.2	47.5	33.9	58.2	60.9
Cnn	68.4	75.6	64.8	73.3	78.8	57.2	49.4	47.5	33.9	58.6	60.7
Lstm	69.0	77.0	68.7	74.4	78.6	57.9	50.5	48.4	34.3	58.1	61.7
Trans.	69.0	78.6	68.6	74.6	79.3	57.2	50.9	48.7	35.2	59.6	62.2
% Gain	+176%	+215%	+312%	+198%	+217%	+247%	+104%	+95%	+6%	+139%	+157%

Table 3: Results on the machine configuration detection tasks with human articles as an additional class.

(2) Different generating choices leave behind different amounts of artifacts. From Table 2, the difficulty of each task generally depends on the specific modeling choice. For example, distinguishing between model size (S1) is much harder than the top-p value. Overall, we observe that methods that directly operate at the generation level (sampling p or k values) are much easier to predict (i.e., leave more artifacts) than condition length ($C1, C2$) or model size ($S1$). It is a somewhat surprising result that varying the initial condition length leaves artifacts in the generated text.

A secondary finding is that discriminating p or k values that are close together is a significantly more challenging task than those that are far apart (i.e., task P1 vs P2). This empirically shows that generated text moves along some form of ordering and magnitude, i.e., $s(a,b) \leq s(b,c)$ if $a-b > b-c$ where $a,b,c \in \mathbb{R}$ and $s(x,y)$ is the accuracy score obtained by classifying between configurations x, y.

(3) Word order does not matter too much. The key observation when pitting various sequence encoding inductive biases against each other is to observe if modeling sequential interactions (short-term or long-range dependencies) and/or word order helps in any of the MCD tasks. The observation is that most complex encoders that takes into account word order do not outperform simple BoW (bag of words) with linear classifiers. This suggests that artifacts found in the text are mostly related to style (e.g., word choices), as opposed to compositional dependencies (e.g., word order). Occasionally, we observe some marginal gains when utilizing ConvNet or Transformers. We hypothesize that considering some amount of token interaction is indeed useful, albeit very marginally. Moreover, the recurrent model (LSTM) performs worse in most cases, suggesting that complex compositional relations are not necessary to capture artifacts.

(4) Discriminating between machines is harder than human and machine. Table 3 report the results of MCD tasks with an additional human article class. By adding human generated articles into the mix, the classification accuracy increases ($\approx 10\%$) across all tasks. Upon inspection, we find that the model separates the human written articles at beyond 90% accuracy, which leads to an overall increase in performance. Hence, the task of distinguishing between machine-machine text is much harder than distinguishing between human-machine text.

5 Discussion

This section discusses the implications of our results and findings.

(1) The sensitivity of neural text generation models emerge as artifacts in the generated text. Our results show that a state-of-the-art text generation model produces significant amounts of artifacts even when making small hyperparameter changes (such as sampling probabilities). It is also relatively surprising that the amount of article conditioning and model size can also be predicted to a certain degree. We feel that this might arise from limitations in the design of neural generation models which may warrant further study.

(2) Tracing the provenance and origination of text generation models is easier than expected. Given that minor changes to decoding settings leave distinguishable signatures, we hypothesize that it is relatively easy to trace and cluster content produced by specific generative models.

6 Conclusion

We studied machine generated text and found that modeling choices leave artifacts, i.e., it is possible to predict modeling choices such as parameterization/sampling choices by looking at generated text alone. We proposed the novel task of machine configuration detection (MCD) which aided in the discovery of these artifacts. We believe our work paves the way for better understanding of neural text generation models and understanding that modeling choices reveals the model configurations is a first crucial step.

References

Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. *arXiv preprint arXiv:1906.03351*.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

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

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043*.

Jiatao Gu, Kyunghyun Cho, and Victor OK Li. 2017. Trainable greedy decoding for neural machine translation. *arXiv preprint arXiv:1702.02429*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Human and automatic detection of generated text. *arXiv preprint arXiv:1911.00650*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Conrad Sanderson and Simon Guenter. 2006. Short text authorship attribution via sequence kernels, markov chains and author unmasking: An investigation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 482–491.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556.

Efstathios Stamatatos, Francisco Rangel, Michael Tschuggnall, Benno Stein, Mike Kestemont, Paolo Rosso, and Martin Potthast. 2018. Overview of pan 2018. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 267–285. Springer.

Michael Tschuggnall, Efstthios Stamatatos, Ben Verhoeven, Walter Daelemans, Günther Specht, Benno Stein, and Martin Potthast. 2017. Overview of the author identification task at pan-2017: style breach detection and author clustering. In *Working Notes Papers of the CLEF 2017 Evaluation Labs/Cappellato, Linda [edit.]; et al.*, pages 1–22.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *arXiv preprint arXiv:1905.12616*.