How Decoding Strategies Affect the Verifiability of Generated Text

Luca Massarelli[1*†] **Fabio Petroni**[2†] **Aleksandra Piktus**[2†] **Myle Ott**[2]
Tim Rocktäschel[2,3] **Vassilis Plachouras**[2] **Fabrizio Silvestri**[2] **Sebastian Riedel**[2,3]
[1] Sapienza University of Rome
massarelli@diag.uniroma1.it
[2]Facebook AI
{fabiopetroni, piktus, myleott, rockt,
vplachouras, fsilvestri, sriedel}@fb.com
[3]University College London

Abstract

Recent progress in pre-trained language models led to systems that are able to generate text of an increasingly high quality. While several works have investigated the fluency and grammatical correctness of such models, it is still unclear to which extent the generated text is consistent with factual world knowledge. Here, we go beyond fluency and also investigate the verifiability of text generated by state-of-the-art pre-trained language models. A generated sentence is verifiable if it can be corroborated or disproved by Wikipedia, and we find that the verifiability of generated text strongly depends on the decoding strategy. In particular, we discover a tradeoff between factuality (i.e., the ability of generating Wikipedia corroborated text) and repetitiveness. While decoding strategies such as top-k and nucleus sampling lead to less repetitive generations, they also produce less verifiable text. Based on these finding, we introduce a simple and effective decoding strategy which, in comparison to previously used decoding strategies, produces less repetitive and more verifiable text.

1 Introduction

Recent years have led to a considerable surge of interest in and capabilities of pre-trained language models (LMs). Today, they play a critical role in many NLP tasks, such as text classification, machine comprehension and natural language inference (Peters et al., 2018; Devlin et al., 2018; Liu et al., 2019a; Yang et al., 2019), to name just a few. They serve as a pre-training objective for downstream applications and they have been used to showcase and measure the general progress in NLP (Yu et al., 2017; Liu et al., 2019b).

Several works (Radford et al., 2019b; Keskar et al., 2019) show the remarkable fluency and grammatical correctness of text decoded from modern LMs. Additionally, recent works (Petroni et al., 2019; Logan et al., 2019; Broscheit, 2019; Roberts et al., 2020) demonstrate that beyond general linguistic capabilities, language models can also pick up factual knowledge present in the training data. However, it is unclear if LMs are able to convey such knowledge at decoding time when producing long sequences—do they generate fluent, grammatical but "babbler-level" text or can they produce utterances that reflect factual world knowledge?

Understanding this behaviour becomes crucially important as the downstream adoption of automatically generated text increases. Already today LMs face growing scrutiny from the media and the broader society, as well as from the researchers themselves. For example, Radford et al. (2019b) initially argued against releasing their models in order to prevent automatic generation of fake news (Radford et al., 2019a). Several blogs and web resources demonstrate that differentiating between human and machine-generated text has become surprisingly difficult.[1]

With that in mind, we set out to study state-of-the-art auto-regressive transformer-based language models through the lens of their verifiability. Specifically, we use Wikipedia to first create a set of natural language prompts to initiate generation. Next, we use transformer models of various sizes and trained with different corpora to generate sentences off these prompts with varying decoding configurations. Finally, following earlier work in fact checking (Thorne et al., 2018), we use Wikipedia again to verify each sentence as supported, refuted, or unverifiable using both an off-the-shelf automatic fact-checking system and human annotators. We define verifiability metrics on top of the automatic and human fact-checkers'

[*] Work done during internship with Facebook.
[†] Equal contribution.

[1] http://quiz.newsyoucantuse.com/

evaluation outcomes (see Figure 1 for a high-level overview).

The truthfulness of generated text can be traded off with other properties. For example, a decoding algorithm can generate the same true fact over and over again to produce many verifiable utterances, but this would be a poor outcome in terms of repetitiveness. Similarly, a model might generate ungrammatical text that cannot be verified as supported or refuted at all, and hence not as factually wrong either. Our experiments show that the text generated from auto-regressive transformer-based LMs, especially in their large versions (1.4B parameters), is almost always grammatical and fluent regardless of the configuration, but that repetitiveness can vary a lot. We hence focus on this dimension in our analysis and define metrics that combine repetitiveness with verifiability.

One of our main findings is that while sampling methods, such as top-k and nucleus, produce more natural and less repetitive text, they also generate fewer supported and more refuted statements. Beam search, on the other hand, shows much better performance along these dimensions at the cost of producing highly repetitive text. Based on these observations, and inspired by findings in Holtzman et al. (2019), who showed how the probability of human text under language models is varying from token to token, we introduce a simple strategy: Delayed Beam Search (DELAYEDBS). In DELAYEDBS, we iterate between sampling and finding most likely utterances. By simply injecting stochasticity in the beginning of a sentence and then switching to beam search, we generate text that is less repetitive while at the same time scores well in terms of our verifiability metrics. Our main findings hold across several experimental settings, with varying training set size and model size.

To summarize, we make the following contributions: (i) we propose an experimental methodology to assess machine generated text with respect to repetitiveness and verifiability. (ii) we assess a wide range of decoding algorithms with respect to these dimensions, (iii) we introduce a novel decoding strategy that addresses some of the shortcomings of existing solutions, (iv) we carry out an annotation campaign to validate our findings and assess the quality of the automatic fact checking system.

2 Related Work

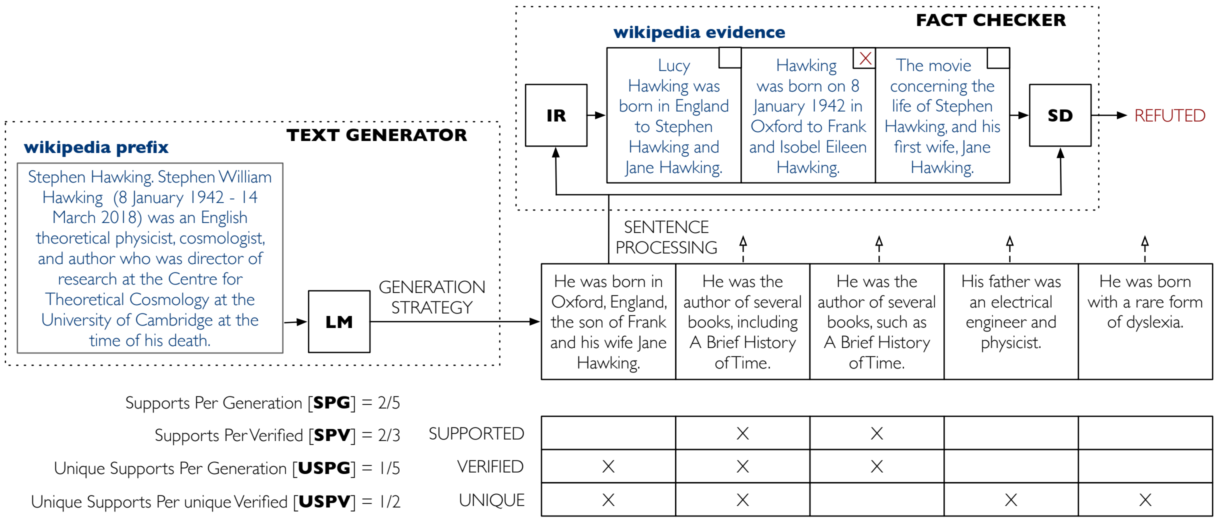
Keskar et al. (2019) trained CTRL, a large (1.63B parameters) pretrained language model that can be conditioned on style or content for controlling generated text. Users can, for example, specify the domain, entities, as well as relationships between entities, to control the generated text. While impressive, their work does not provide insights into the verifiability of the generated text.

Multiple efforts focus on improving text decoding with respect to different criteria. Vijayakumar et al. (2016) and Li et al. (2016) introduce alternative scoring strategies to diversify the hypothesis tree explored by beam search. Fan et al. (2018) propose *top-k sampling*, *i.e.*, sampling from the top k tokens with the highest probability to generate stories. Holtzman et al. (2019) find that for the same neural language model, the choice of the decoding strategy can have a dramatic effect on the fluency and repetitiveness of the generation. They propose *nucleus sampling* as a way to increase diversity of the generated text while improving fluency. In our work, we find that while this strategy does create more fluent and less repetitive text, it does also result in a less factually true generation. Cho et al. (2019) choose to separate the generation and diversification steps altogether, and focus on leveraging content selection to map the input to diverse sequences. We describe various generation strategies in more detail in section 3.

Welleck et al. (2019) note that with nucleus sampling, per-token probabilities can be very low which they attribute to the likelihood training objective. They propose a novel *unlikelihood* training objective which lowers the probability of tokens in the context of the model. Their approach is orthogonal to the decoding strategy and testing alternative training objectives is out of the scope of our paper.

A recent approach by Bakhtin et al. (2019) learns to distinguish human from machine generated text. Zellers et al. (2019) investigate generating and detecting fake news using neural language models. Niewinski et al. (2019) propose a variation of the GPT-2 language model to explicitly generate malicious claims. Instead of directly optimizing for generating fake or factual news, we are interested in investigating the relationship between the verifiability of the existing language models and different decoding strategies they are coupled with.

Several metrics have been proposed to evaluate natural language generations in the past (Novikova


FACT CHECKER
wikipedia evidence
IR
Lucy Hawking was born in England to Stephen Hawking and Jane Hawking.
Hawking was born on 8 January 1942 in Oxford to Frank and Isobel Eileen Hawking.
The movie concerning the life of Stephen Hawking, and his first wife, Jane Hawking.
SD
REFUTED
TEXT GENERATOR
wikipedia prefix
Stephen Hawking. Stephen William Hawking (8 January 1942 - 14 March 2018) was an English theoretical physicist, cosmologist, and author who was director of research at the Centre for Theoretical Cosmology at the University of Cambridge at the time of his death.
LM
GENERATION STRATEGY
SENTENCE PROCESSING
Supports Per Generation [SPG] = 2/5
Supports Per Verified [SPV] = 2/3
Unique Supports Per Generation [USPG] = 1/5
Unique Supports Per unique Verified [USPV] = 1/2

	He was born in Oxford, England, the son of Frank and his wife Jane Hawking.	He was the author of several books, including A Brief History of Time.	He was the author of several books, such as A Brief History of Time.	His father was an electrical engineer and physicist.	He was born with a rare form of dyslexia.
SUPPORTED		×	×		
VERIFIED	×	×	×		
UNIQUE	×	×		×	×

Figure 1: High level description of our experimental methodology that combines a language model (LM) with a fact checker, usually implemented combining an information retrieval (IR) and a stance detector (SD) component.

et al., 2017). Given that recent studies (Fan et al., 2018; Holtzman et al., 2019; Welleck et al., 2019) point to repetitiveness as one of the main problems affecting the generation of state-of-the-art models, we mainly consider this dimension in our analysis.

3 Background

Language models (LMs) assign probabilities to sequences of tokens. Given a context, that is, a sequence of tokens $\mathbf{c_t} = [w_1, w_2, \ldots, w_{t-1}]$, autoregressive LMs commonly estimate the probability distribution of the next target using neural models (Mikolov and Zweig, 2012; Melis et al., 2017; Bengio et al., 2003) with:

$$p(w_t \mid c_t) = \mathrm{softmax}(\mathbf{W}\mathbf{h}_t + \mathbf{b}) \qquad (1)$$

where $\mathbf{h}_t \in \mathbb{R}^k$ is the output vector of a neural network at position t and $\mathbf{W} \in \mathbb{R}^{|\mathcal{V}| \times k}$ is a learned parameter matrix that maps $\mathbf{h}_t$ to unnormalized scores for every word in the vocabulary $\mathcal{V}$. In this work, we consider self-attention mechanisms (Radford et al., 2018; Dai et al., 2019; Radford et al., 2019b) to compute $\mathbf{h}_t$ given the word history.

Open-Ended Text Generation As described in Holtzman et al. (2019), the task of open-ended text generation involves producing a coherent completion of the provided context. We consider the common left-to-right generation, where a token at position t in the sequence is generated by considering the probability distribution over the vocabulary defined in equation 1. Once a decision is made for w_t according to a decoding strategy, it is incorporated into the context and the process is iterated - *i.e.*, the token at position $t+1$ is generated by considering $p(w_{t+1} \mid c_{t+1} = [w_1, \ldots, w_t])$. In this work, we consider different decoding strategies of selecting w_t given $p(w_t \mid c_t)$.

3.1 Decoding Strategies

The decoding strategies we consider in our analysis can be broadly divided in two families: *sampling-based* and *likelihood-based*.

Sampling-based This family of techniques aims at increasing the diversity of the output and avoiding repetitions by introducing stochastic decisions during the generation process.
Top-k sampling (Fan et al., 2018) selects w_t by sampling from the k tokens with the highest probability in $p(w_t \mid c_t)$.
Top-p sampling, also referred to as nucleus sampling (Holtzman et al., 2019), selects w_t from the smallest set of tokens whose cumulative probability (given by $p(w_t \mid c_t)$) is above a threshold p.

Likelihood-based These strategies navigate the solution space by selecting sequences of tokens that maximize the overall likelihood. Given that the number of possible sequences is typically very large, it is a common practice to define heuristics to make the generation practical.
Beam Search (BS). This strategy approximately maximizes the likelihood of the whole sequence. Throughout the generation, we hold a *beam* of β prefixes which are iteratively extended. At each time-step, β tokens are generated to complete each of the prefixes in the beam and we retain β hypothe-

ses with the highest score out of the β^2 candidates for the next step. β is referred to as the beam size. *Greedy* decoding, where at each step the most likely token is selected, is a special case of beam search with beam size 1.

Group diverse Beam Search (GROUPBS). To favor the diversity of the exploration, Vijayakumar et al. (2016) propose to divide the beam into groups. The diversity between groups is imposed by introducing a group dissimilarity penalty into the search objective.

Sibling diverse Beam Search (SIBLINGBS). With the same aim of diversifying the exploration, Li et al. (2016) propose a variant of beam search which introduces a penalty proportional to the rank of a candidate token with respect to its source in the beam. The goal is to encourage preserving hypotheses from diverse sources within the beam.

A simple trick to reduce repetitiveness is to explicitly prevent the generation of already observed n-grams (Paulus et al., 2017). We refer to this approach as *n-gram blocking*.

Delayed Beam Search (DELAYEDBS). We propose a new hybrid strategy that uses sampling to generate the first L tokens of a sentence and then it finishes the sentence using beam search. The smaller the L, the closer the behaviour is to beam search. Conversely, the larger the L, the closer we are to sampling strategies. Consequently, by tuning L, it is possible to combine the advantages of both sampling and likelihood-based strategies.

4 Evaluating Verifiability

In this section we first describe the tools used to evaluate the verifiability of the generated text. We then formally introduce our repetitiveness and verifiability metrics.

The high level overview of our evaluation setup is shown in Figure 1. For the purpose of this analysis, we consider both the *text generator* and the *fact checker* as black boxes which produce and assess text respectively. More specifically, the text generator gets in input a prefix p and produces a sequence of tokens that can be interpreted as a completion of p. We segment the generated completion into sentences and consider the first k sentences. The fact checker gets in input a sentence and outputs a positive (SUPPORTED), negative (REFUTED) or unverifiable (NOT ENOUGH INFO) response as well as textual evidence used for the judgment. We consider a sentence as *verified* if the output label is either SUPPORTED or REFUTED.

Our metrics assess the generation process given a set of prefixes P. The set P can be seen as the data source for our verifiability probe. Let $G^p = [s_1^p, ..., s_k^p]$ be the sequence of sentences generated by the LM from prefix $p \in P$. We indicate with $V^p \in G^p$ the set of sentences that are verified by the fact checker, while with $S^p \in V^p$ we denote the subset of sentences labeled as SUPPORTED. To assess the verifiability of the generated text we introduce the following two metrics:

Supports Per Generation (SPG): is the fraction of supported sentences among the generated ones:

$$\text{SPG} = \frac{1}{|P|} \sum_{p \in P} \frac{|S^p|}{k} \tag{2}$$

Supports Per Verified (SPV): is the fraction of supported sentences among the verified ones:

$$\text{SPV} = \frac{1}{|P|} \sum_{p \in P} \frac{|S^p|}{|V^p|} \tag{3}$$

SPG can be interpreted as a sort of a recall metric while SPV as a precision one.

Note that a generation could achieve a high score in terms of SPG and SPV by repeating the same supported sentence over and over again. To capture this behaviour, we define the *unique* variants of our metrics. We consider two sentences as equivalent if they have the same factuality label (*i.e.*, SUPPORTED or REFUTED) and the decision is justified by the same evidence. For a set of equivalent sentences, we consider only the one which appeared first in the generation as unique. We denote the set of unique sentences as $S_u^p \in S^p$, $V_u^p \in V^p$ is a set of unique verified sentences. We introduce:

Unique Supports Per Generation (USPG): the fraction of unique supported sentences among the generated ones:

$$\text{USPG} = \frac{1}{|P|} \sum_{p \in P} \frac{|S_u^p|}{k} \tag{4}$$

Unique Supports Per unique Verified (USPV): the fraction of unique supported sentences among unique verified sentences:

$$\text{USPV} = \frac{1}{|P|} \sum_{p \in P} \frac{|S_u^p|}{|V_u^p|} \tag{5}$$

5 Methodology

In this section we describe in detail the implementational choices for all components in Figure 1.

Prefix Dataset We retrieve title and description of the top-1000 most visited Wikipedia pages of 2017 and 2018. For each page, we concatenate the title and the first sentence in the description to create a string prefix for the language model. We use 2018 data as validation set and run parameter sweeps over it. We tested the best configuration of every decoding strategy on 2017 data (test set). We ensure no overlap between 2017 and 2018 prefixes.

Language Model We consider three sizes of language models (small, medium, large) based on the Transformer architecture (Vaswani et al., 2017; Radford et al., 2019b), with 124M, 354M and 1.4B parameters respectively. We train models on four corpora: (i) WIKIPEDIA, an English Wikipedia dump consisting of roughly 2 Billion Words; (ii) BOOKS, the Toronto books corpus (Zhu et al., 2015; Kiros et al., 2015), which consists of fiction books totaling about half a billion words; (iii) OPENWEBTEXT, a reconstruction of the WebText corpus (Radford et al., 2019b) consisting of roughly 3 Billion Words; (iv) CCNEWS, a de-duplicated subset of the English portion of the CommonCrawl news dataset (Nagel, 2016; Bakhtin et al., 2019; Liu et al., 2019a), which totals around 16 Billion words. We train models using the FAIRSEQ toolkit (Ott et al., 2019).

Generation Strategy We consider the generation strategies discussed in Section 3.1, namely top-k, top-p, greedy, Beam Search (BS), Group-Diverse Beam Search (GROUPBS), Sibling-Diverse Beam Search (SIBLINGBS) and Delayed Beam Search (DELAYEDBS). Additionally, we experiment with n-gram blocking and indicate that a model is equipped with blocking with a subscript b, *e.g.*, BS_b. We fix the generation length to 256 tokens. We perform three generations per prefix with different seeds for all strategies that make stochastic decisions, and report average values.

Sentence Processing Given that our fact checker expects a single sentence as input, we segment the generated text into sentences. We consider the first $k = 5$ sentences. We perform coreference resolution to replace pronouns with the corresponding referring entity in order to give the complete information to the fact checker. For the same reason, we apply a simple heuristic that replaces each determiner (*i.e.*, *"The"*) at the beginning of a sentence and the subsequent noun with the original entity (*i.e.*, the title of the Wikipedia page). For all these steps we use spaCy.[2] We consider sentences longer than 50 tokens as not verifiable, since long sentences are likely to contain multiple claims and can be misclassified by the automatic fact-checking system, we consider that has been trained on short single claim statements.

Fact Checker We consider an off-the-shelf fact checker[3] trained on the FEVER dataset (Thorne et al., 2018) which achieves the highest FEVER score of 68.46% in the second FEVER shared task (Thorne et al., 2019). This solution takes inspiration from Hanselowski et al. (2018) and consists of three main stages: (i) identify relevant Wikipedia pages, as in Hanselowski et al. (2018); (ii) retrieve relevant sentences from such pages; (iii) recognize textual entailment between input and retrieved text. The system uses a hierarchical sentence retrieval approach in order to verify claims that require multiple statements as evidence. It uses BERT (Devlin et al., 2018) for both retrieval and entailment.

Metrics We use all the metrics introduced in Section 4. We also consider the following metrics to capture the repetitiveness of the generation:
Distinct 4-gram: is the average number of distinct 4-grams present in the generated text (Vijayakumar et al., 2016).
4-gram proportion: is the average ratio between distinct 4-grams in machine and human generated text (Holtzman et al., 2019). For the latter, we consider the 256 tokens after the first sentence in the description for each Wikipedia page.

6 Results

We summarize the main results in Table 1. It shows the performance of the different generation strategies on the considered metrics on the test set of prefixes, considering the large transformer model trained on CCNEWS (this corpus led to the best performance according to our ablation, see Figure 2a). We performed an exhaustive grid search over the parameters for all considered generation strategies using the small model on the validation set, and consider the configuration that led to the highest

[2]`https://spacy.io`
[3]`https://github.com/dominiksinsaarland/domlin_fever`

strategies \ metrics		*repetitiveness*		*verifiability*		*diverse verifiability*	
		distinct 4-grams	**4-grams proportion**	**SPG**	**SPV**	**USPG**	**USPV**
human - Wikipedia		222.48	100.00	36.56	93.03	36.56	93.03
sampling	**top-k**	**143.52**	**64.51**	13.02	70.15	11.06	69.39
	top-p	136.66	61.43	13.94	70.76	11.36	68.93
likelihood	**greedy**	67.42	30.31	19.62	78.67	12.06	77.21
	BS	59.53	26.76	**25.50**	**84.49**	11.88	**81.59**
	GROUPBS	66.06	29.69	20.56	78.29	11.54	76.53
	SIBLINGBS	67.11	30.16	22.32	80.11	11.36	76.76
hybrid	**DELAYEDBS**	112.12	50.40	17.52	78.99	12.74	77.59
blocking	**BS$_b$**	92.00	41.35	23.62	83.35	**15.28**	80.76

Table 1: Performance of the different generation strategies on the considered metrics. We report percentage values for the large transformer model on the test set. The first row shows human performance computed on Wikipedia.

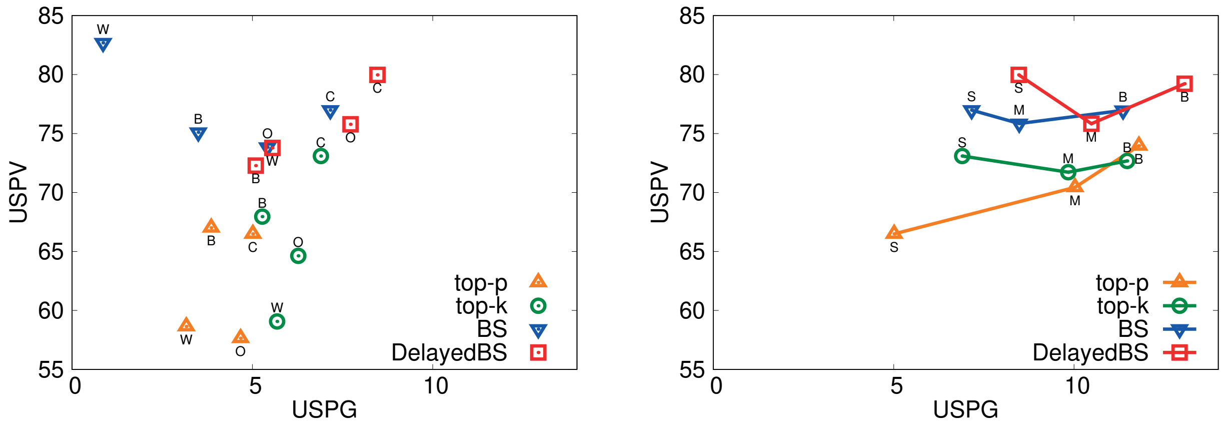
USPG value (see the Appendix for details). We report as reference human performance computed on Wikipedia considering at most the first 5 sentences of the prefix article.

Sampling strategies (*i.e.*, top-p and top-k) outperform the other strategies in terms of repetitiveness metrics, that is, they are able to generate text with a higher degree of diversity, consistently with previous works (Fan et al., 2018; Holtzman et al., 2019). However, diversity comes at a price, as the verifiability metrics are low (in particular, precision values - they generate more refuted sentences). Intuitively, random choices might hamper verifiability when sampling a token in specific positions of the sentence, for instance, in a named entity, potentially making the overall sentence non factual. We notice that this problem gets even worse by increasing k or p. Following a generation path that maximizes likelihood is a better approach for verifiability. In particular, BS achieves the highest performance in terms of SPG and SPV. Nevertheless, generation diversity drops, consistently with previous works (Vijayakumar et al., 2016; Li et al., 2016; Welleck et al., 2019; Holtzman et al., 2019). Solutions such as GROUPBS and SIBLINGBS have been proposed to mitigate this problem, and their numbers actually look slightly better than BS in terms of repetitiveness metrics.

When we assess diverse verifiability (that is, we consider distinct supported/refuted sentences), likelihood and sampling based strategies are similar in terms of recall (*i.e.*, USPG), while likelihood-based solutions outperform both top-k and top-p in terms of precision (*i.e.*, USPV) by a large margin - they generate less sentences refuted by the fact checker.

DELAYEDBS tries to combine the best of these two approaches, by defining a hybrid strategy that starts a sentence by sampling tokens and ends it by following a max-likelihood path. It achieves results comparable to likelihood-based solutions in terms of precision and recall for diverse verifiability while being much less repetitive (it almost doubles the number of distinct 4-grams). Interestingly, it is sufficient to sample just the first token with high uncertainty (top-100) and finish the sentence with beam search to trigger this behaviour (Figure 5 in the Appendix Section reports a detailed ablation study for the delay length).

Another way of mitigating repetitiveness is through n-gram blocking. We combine it with BS, sweeping over the values of n between 3 and 20. In line with our expectations, low n values score low in verifiability metrics, as the model is forced to explore less likely parts of the solution space in order to avoid generating previously observed n-grams. Unsurprisingly, the diversity of the solution drops as n increases. In this sense, BS$_b$ and DELAYEDBS attempt to strike a similar balance between diversity (introduced via n-gram blocking in BS$_b$ and via sampling in DELAYEDBS) and verifiability (achieved by incorporating BS). Figure 3 highlights this analogy further. Overall, we achieve the best USPG performance by combining 20-gram blocking and BS - we believe it is due to the fact that n-gram blocking prevents BS from repeating the same phrases multiple times, while remaining



(a) Performance of the small transformer model trained on different corpora, i.e., WIKIPEDIA (W), BOOKS (B), OPENWEBTEXT (O) and CCNEWS (C).

(b) Ablation study on our transformer model trained on CC-NEWS with increasing number of parameters, i.e., 124M (S), 354M (M) and 1.4B (B).

Figure 2: USPV vs USPG, inspired by precision-recall curve.

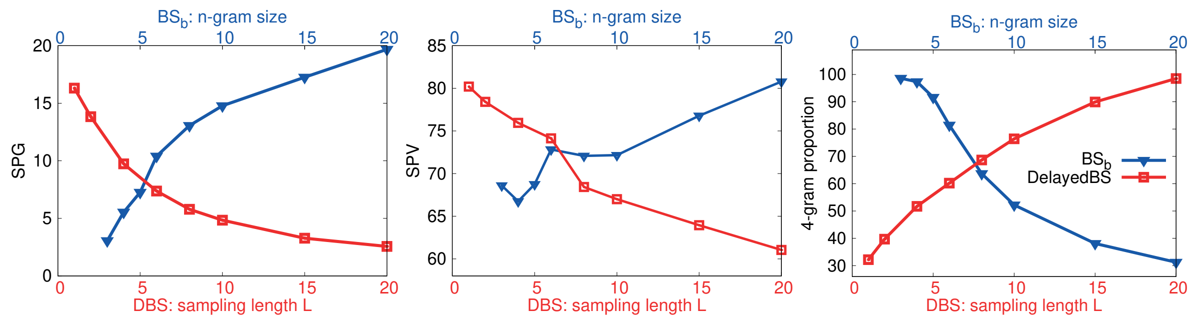


Figure 3: SPG, SPV and 4-gram proportion values for BS$_b$ and DELAYEDBS, by varying the sampling length L for DELAYEDBS (bottom axis) and the n-gram blocking size for BS$_b$ (top axis).

relaxed enough to allow the generation to produce a high-likelihood solution. However, even though BS$_b$ archives the best results in terms of diverse verifiability metrics, DELAYEDBS still produces less repetitions, hence constituting a viable alternative.

Ablation studies We experiment with different training corpora (Figure 2a) and different sizes of the transformer model (Figure 2b), using the validation set. We report USPV vs USPG values, taking inspiration from the popular precision-recall curve. The average perplexity of the small transformer model is the lowest for WIKIPEDIA (8.31) compared to BOOKS (53.08), OPENWEBTEXT (11.14) and CCNEWS (12.23). Even though all prefixes are likely to be in the corpus, WIKIPEDIA performance in terms of USPG is low regardless of the decoding strategy. This counter-intuitive behaviour seems to occur mainly due to the tendency of the small model trained on WIKIPEDIA to generate endless, unverifiable entity lists, mimicking Wikipedia lists. CCNEWS leads to the best performance in terms of recall (USPG) for all decoding strategies, but also in terms of precision (USPV) for top-k and DELAYEDBS.

We did explore several other dimensions, including grammaticality (through a syntactic parser) and relevance (i.e., tf-idf score with the prefix Wikipedia page) during our experiments (see Table 4 in the Appendix). Figure 4 reports the Pearson correlation coefficient between supported and verified sentences and these set of metrics. We consider the four runs of the large transformer model reported in Figure 2b. We notice, for instance, that the average log probability of a sentence is positively correlated with verifiability, suggesting that max-likelihood strategies are better suited in this regards. Furthermore, the tf-idf score with the prefix Wikipedia page content is positively correlated with supported sentences. This behaviour is related to the implementation of the fact checker we use, which, by considering exclusively Wikipedia as knowledge source, favours text with a high overlap

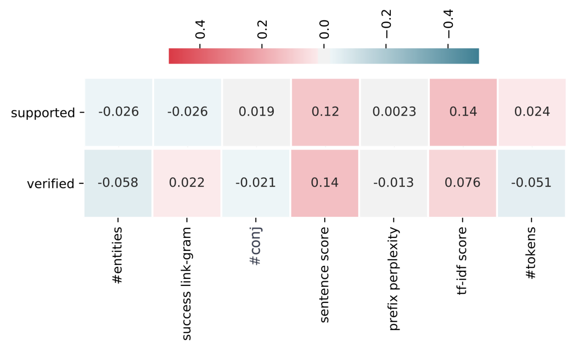


Figure 4: Pearson correlation coefficient for supported/verified sentences (large model) and a set of metrics per sentence: number of entities, if successfully parsed by the Link-Grammar syntactic parser,[4]number of conjunctions in the dependency tree, average token log probability, prefix perplexity, tf-idf score with the prefix Wikipedia page, number of tokens.

with the latter. Note, however, that the model was not explicitly exposed to Wikipedia during training (*i.e.*, CCNEWS does not explicitely include it).

We report examples of text generated by the large transformer model using different decoding strategies in the Appendix section (Table 5).

Human Evaluation We carry out an annotations campaign, where we ask human annotators to fact check generated text. We base the evaluation on a set of 200 prefixes randomly selected from the test set. We consider completions produced by 5 of the generation strategies studied in this paper. We collect 5 annotations per generation. Results, reported in Table 2, confirm our findings: sampling strategies generate text which is less repetitive but also with less supported sentences than in the case of beam search. DELAYEDBS emerges as a reasonable trade-off between the two, being less repetitive than BS and producing more supported sentences than top-k. The analysis also highlights how blocking n-grams does not really address the repetitive nature of BS. Looking at some examples (see Table 5) we notice that BS_b avoids repeating n-grams by introducing superficial, token-level modifications which, most of the time, fail to alternate the underline meaning of the sentence. In terms of absolute values, precision metrics (*i.e.*, USPV and SPV) are lower than those computed with the automatic fact checker, and recall metrics (*i.e.*, SPG and USPG) higher. This is due to the poor recall performance of the fact checking system - 45.66% for SUPPORTED and 5.78% for REFUTED. Precision

[4]abisource.com/projects/link-grammar

	REP	*NAC*	*UNG*	**SPG**	**SPV**	**USPG**	**USPV**
top-k	**16.0**	3.4	**1.4**	27.2	41.36	20.1	41.16
greedy	35.6	**1.7**	2.0	32.5	42.23	17.6	41.75
BS	38.7	8.2	1.8	**44.6**	**64.62**	20.1	**65.1**
DELAYEDBS	25.0	6.2	3.0	35.4	50.22	**23.3**	50.68
BS_b	31.6	9.8	4.3	38.2	56.92	19.7	58.12

Table 2: Results based on human fact checkers, 5 annotations per sentence. Average inter-annotator agreement is 0.66 Cohen's kappa (average majority of 81% for SUPPORTED, 78% for REFUTED and 65% for NOT ENOUGH INFO). We report the percentage of sentences annotated as repetitions (*REP*), not a claim (*NAC*), ungrammatical (*UNG*), and our verifiability metrics.

values are 80.89% for SUPPORTED and 52.69% for REFUTED. In sum we find that while off-the-shelf, state-of-the-art fact checker systems still leave ample room for improvement, they already serve as a good proxy for ranking pre-trained language models and decoding strategies with respect to the verifiability of the text they generate.

7 Conclusion and Discussion

We presented a systematic analysis of the verifiability of text generated by a wide range of decoding strategies from large autoregressive language models. We assessed generated sentences with an off-the-shelf automatic fact-checker as well as through human annotations. We found that sampling decoding strategies produce text that is less verifiable, but also less repetitive when compared to strategies that consider most likely sequences according to the model distribution. We proposed a hybrid decoding strategy, combining the non-repetitive nature of sampling solutions with the verifiable generation of likelihood-based approaches.

In our analysis, we considered the most viewed Wikipedia pages in 2017 and 2018. Our rationale was that such pages would represent topics that are likely to be highly covered in a random web crawl (e.g., OPENWEBTEXT and CCNEWS). Results (not reported in the paper) with a random set of Wikipedia pages showed lower values in terms of SPG and USPG (*i.e.*, recall metrics). A potential line of future work could be to investigate relationships among training corpora and generation.

We considered each sentence as a single claim to keep our experimental setting clean and avoid noise from an automatic claim extractor. However, some generations contain multiple claims that could be independently assessed. Studying such phenomena is an interesting future direction.

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*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Samuel Broscheit. 2019. Investigating entity knowledge in bert with simple neural end-to-end entity linking. *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*.

Jaemin Cho, Minjoon Seo, and Hannaneh Hajishirzi. 2019. Mixture content selection for diverse sequence generation. *arXiv preprint arXiv:1909.01953*.

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*.

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

Angela Fan, Mike Lewis, and Yann N. Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL*, pages 889–898.

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. Ukp-athene: Multi-sentence textual entailment for claim verification. *arXiv preprint arXiv:1809.01479*.

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

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*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems 28*, pages 3294–3302.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019a. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Zihan Liu, Yan Xu, Genta Indra Winata, and Pascale Fung. 2019b. Incorporating word and subword units in unsupervised machine translation using language model rescoring. *arXiv preprint arXiv:1908.05925*.

IV Logan, L Robert, F Nelson, E Matthew, et al. 2019. Barack's wife hillary: Using knowledge-graphs for fact-aware language modeling. *arXiv preprint arXiv:1906.07241*.

Gábor Melis, Chris Dyer, and Phil Blunsom. 2017. On the state of the art of evaluation in neural language models. *arXiv preprint arXiv:1707.05589*.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *Proceedings of the fourth IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239.

Sebastian Nagel. `http://web.archive.org/save/http://commoncrawl.org/2016/10/news-dataset-available/` [online]. 2016. Accessed: 2019-11-08.

Piotr Niewinski, Maria Pszona, and Maria Janicka. 2019. Tmlab: Generative enhanced model (gem) for adversarial attacks. *arXiv preprint arXiv:1910.00337*.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for nlg. *arXiv preprint arXiv:1707.06875*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 16th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 2227–2237.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

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

Alec Radford, Jeff Wu, Dario Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. 2019a. Better language models and their implications. https://blog.openai.com/better-language-models/. Accessed: 2019-11-08.

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

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *Proceedings of the 16th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 809–819.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2019. The FEVER2.0 shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 1–6, Hong Kong, China. Association for Computational Linguistics.

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 30*, pages 5998–6008.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

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*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*.

Lei Yu, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Tomás Kociský. 2017. The neural noisy channel. In *Proceedings of the 5th International Conference on Learning Representations, ICLR*.

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*.

Yukun Zhu, Ryan Kiros, Rich 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 *Proceedings of the 15th IEEE international conference on computer vision (ICCV)*, pages 19–27.

strategy	best parameters
top-k	k= 2
top-p	p= 0.4
BS	beam size= 15

GROUPBS	groups= 2
	penalty= 0.2

SIBLINGBS	penalty= 0.1

DELAYEDBS	top-k= 100
	beam size= 6;
	L= 1

BS_b	beam size= 15 blocking order= 20

Table 3: Best parameters per decoding strategy.

8 Appendix

8.1 Hyperparameters

We conduct a parameter sweep on the small transformer model on the validation set. The following table shows the configuration for each decoding strategy that leds to the highest USPG score.

8.2 Generation Examples

We reported some examples generated with different strategies in table 5.

8.3 Other metrics

We explored how decoding strategy affects other dimensions of the generated text. Results are reported in table 4. We measure several statistics ovtaer the generated text:

- The average number of distinct sentences for each generated text;
- The average number of named entities in each sentence;
- The average number of tokens in each sentence;
- The average number of conjunctions in the dependency tree of each sentence;

To compute the above metrics, we used *spaCy*. In particular we used its tokenizer to split tokens and sentences, its named entity recognition capability to identify named entities and its dependency parser to count the number of conjunctions.

Furthermore, we analyzed the grammatical correctness of the generated text, counting the success rate of the link-gram parser [5] over the sentences in the generated text.

We also measure the relevance of the generated text against the Wikipedia page that contains the prefix used for the generation. For this purposes, we compute the *tf-idf* score of the generated text and the related Wikipedia page.

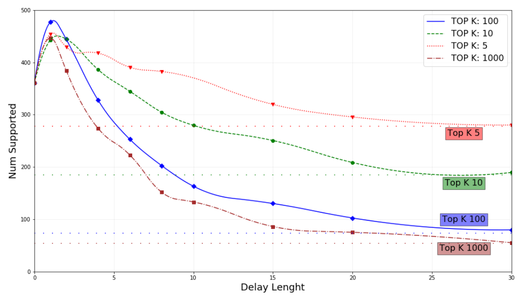


Figure 5: Ablation study over delay length. We report on the x-axis the delay length and on the y-axis the number of distinct supported sentences obtained for each delay length. Horizontal lines represent the value obtained on the validation set using top-k decoding strategy. All the generations were performed on the validation set using the small transformer trained on CCNEWS.

8.4 Ablation study over delay length

We perform an ablation study to measure how the number of supported sentences generated with DELAYEDBS is affected by the delay length. We generated text using the prefixes in the validation set using DELAYEDBS with top-k as sampling strategy and with different delay length. Our hypothesis is that using larger delay length the number of supported sentences in the generated text will become close to the one obtained for top-k. We report the results in figure 5. From the figure it is clear that with larger delay length the number of supported sentences is very close to the one obtained with top-k. Moreover, as expected, a short delay length seems to produce a larger number of supported sentences.

[5]https://github.com/opencog/link-grammar

strategy	param	distinct sentences	#entities	#tokens	#conj	% success link-gram	tf-idf score
greedy	1	2.44	2.02	18.68	1.18	81.30	255.04
beam search	6	2.20	2.79	22.71	1.32	74.76	510.17
	12	2.13	3.13	23.72	1.39	71.92	565.97
	15	2.14	3.13	23.51	1.40	72.41	568.96
top-k	2	4.63	2.43	21.92	1.26	83.58	259.75
	10	4.95	2.70	25.22	1.33	78.73	246.18
	100	4.98	2.72	27.29	1.36	74.91	203.19
top-p	0.1	2.57	2.02	18.93	1.17	81.10	251.27
	0.3	3.88	2.19	19.41	1.19	85.06	238.72
	0.7	4.90	2.59	23.76	1.27	79.90	215.68
	1	4.97	2.81	28.55	1.38	70.40	162.95
delayed beam search	5-6-1	3.74	3.01	23.39	1.22	77.23	401.76
	10-6-1	3.95	3.04	23.95	1.23	77.00	406.03
	100-6-1	4.22	3.04	24.10	1.22	76.36	399.91
	1000-6-1	4.30	3.03	24.62	1.21	76.27	397.27

Table 4: Analysis of the generated text among different dimensions. We used the prefix in the validation set and the small transformer model trained on CCNEWS to generate the text. Under the column *params* we report hyper parameters for the decoding strategy considered. For DELAYEDBS we we have varied only the parameter for top-k sampling and we used beam size equal to 6 and delay length of 1.

prefix	**Princess Margaret, Countess of Snowdon**. *Princess Margaret, Countess of Snowdon, (Margaret Rose 21 August 1930 - 9 February 2002) was the younger daughter of King George VI and Queen Elizabeth The Queen Mother and the only sibling of Queen Elizabeth II.*	**Whitey Bulger**. *James Joseph Whitey Bulger Jr. (September 3, 1929 - October 30, 2018) was an Irish-American organized crime boss and FBI informant who led the Winter Hill Gang in the Winter Hill neighborhood of Somerville, Massachusetts, a city directly northwest of Boston.*
top-k	She is the only surviving member from the Royal Family who was born in the same year as the Queen Mother. Princess Margaret, Countess of Snowdon, (Margaret Rose 21 August 1930 - 9 February 2002) was a British princess who married into the British Royal Family in 1939. She is a cousin to the Queen, Prince Philip, Prince Charles, Prince William, Prince Harry and Prince Edward. Princess Margaret, Countess of Snowdon, (Margaret Rose 21 August 1930 - 9 February 2002) is the younger daughter of King George VI and Queen Elizabeth. She is the only survivor from the Royal Family who was born in the same year as Queen Elizabeth.	He was a fugitive for 16 years before his arrest in 2011. He was sentenced in 2013 to two life terms plus five years for his role in 11 murders and was released in 2014. Bulger was found dead in his cell at the U.S. penitentiary in West Virginia. He was 89. He was serving the life sentences for his role in the infamous Boston crime family.
BS	Princess Margaret, Countess of Snowdon, (Margaret Rose 21 August 1930 - 9 February 2002) was the younger daughter of King George VI and Queen Elizabeth The Queen Mother and the only sibling of Queen Elizabeth II. Princess Margaret, Countess of Snowdon, (Margaret Rose 21 August 1930 - 9 February 2002) was the younger daughter of King George VI and Queen Elizabeth The Queen Mother and the only sibling of Queen Elizabeth II. Princess Margaret, Countess of Snowdon, (Margaret Rose 21 August 1930 - 9 February 2002) was the younger daughter of King George VI and Queen Elizabeth The Queen Mother and the only sibling of Queen Elizabeth II. ...	Bulger was one of the FBI's most wanted fugitives for 16 years until he was captured in Santa Monica, California, in 2011. He was convicted in 2013 of a litany of crimes, including racketeering, extortion, money-laundering, and murder. He was sentenced to two consecutive life sentences plus five years. He died in federal prison in West Virginia on Tuesday at the age of 89. Bulger was one of the FBI's most wanted fugitives for 16 years until he was captured in Santa Monica, California, in 2011.
DBS	**She** married Antony Armstrong-Jones, a photographer, in 1960. **It** was the first marriage for the Queen and the first for Prince Philip, Duke of Edinburgh. **After** divorcing Armstrong-Jones in 1978, she married Group Captain Peter Townsend in June that same year. **She** died at the age of 71 on 9 February 2002. **Why** did Princess Margaret marry Antony Armstrong-Jones?	**He** was one of the FBI's most wanted fugitives for 16 years until his capture in 2011. **Born** in Boston, Whitey Bulger was the son of Irish immigrants. **After** serving in the U.S. Navy during World War II, Whitey Bulger joined the Irish-American mafia, the Winter Hill Gang, in the early 1950s. **He** quickly rose through the ranks of the gang, eventually becoming its leader. **He** was known as "Whitey" because of his light brown hair and blue eyes.
BS$_b$	Princess Margaret, Countess of Snowdon (Margaret Rose 21 August 1930 - 9 February 2002) was the eldest daughter of King George VI and Queen Elizabeth The Queen Mother. Princess Margaret (Margaret Rose 21 August 1930 - 9 February 2002) was the eldest child of King George VI and Queen Elizabeth The Queen Mother. Princess Margaret, Countess of Snowdon. (Margaret Rose 21 August 1930 - 9 February 2002) was the eldest daughter of Queen Elizabeth The Queen Mother. (Margaret Rose 21 August 1930 - 9 February 2002) was the oldest child of King George VI and Queen Elizabeth The Queen Mother.	Bulger was one of the FBI's most wanted fugitives for 16 years **until** he was captured in Santa Monica, California, in 2011. He was convicted in 2013 of a litany of crimes, including racketeering, extortion, money-laundering, and murder. He was sentenced to two consecutive life sentences plus five years. He died in federal prison in West Virginia on Tuesday at the age of 89. Bulger was one of the FBI's most wanted fugitives for 16 years **before** he was captured in Santa Monica, California, in 2011.

Table 5: Two examples of text generated with different strategies by the large transformer model. One the left a cherry picked example (in terms of repetitive generation for BS) while on the right a random one. Sentence refuted by the fact checker are highlighted in red, supported in green.