Dissecting Generation Modes for Abstractive Summarization Models via Ablation and Attribution

Jiacheng Xu and **Greg Durrett**
Department of Computer Science
The University of Texas at Austin
{jcxu,gdurrett}@cs.utexas.edu

Abstract

Despite the prominence of neural abstractive summarization models, we know little about how they actually form summaries and how to understand where their decisions come from. We propose a two-step method to interpret summarization model decisions. We first analyze the model's behavior by ablating the full model to categorize each decoder decision into one of several generation modes: roughly, is the model behaving like a language model, is it relying heavily on the input, or is it somewhere in between? After isolating decisions that do depend on the input, we explore interpreting these decisions using several different attribution methods. We compare these techniques based on their ability to select content and reconstruct the model's predicted token from perturbations of the input, thus revealing whether highlighted attributions are truly important for the generation of the next token. While this machinery can be broadly useful even beyond summarization, we specifically demonstrate its capability to identify phrases the summarization model has memorized and determine where in the training pipeline this memorization happened, as well as study complex generation phenomena like sentence fusion on a per-instance basis.

1 Introduction

Transformer-based neural summarization models (Liu and Lapata, 2019; Stiennon et al., 2020; Xu et al., 2020b; Desai et al., 2020), especially pre-trained abstractive models like BART (Lewis et al., 2020) and PEGASUS (Zhang et al., 2020), have made great strides in recent years. These models demonstrate exciting new capabilities in terms of abstraction, but little is known about how these models work. In particular, do token generation decisions leverage the source text, and if so, which parts? Or do these decisions arise based primarily on knowledge from the language model (Jiang et al., 2020; Carlini et al., 2020), learned during pre-training or fine-tuning? Having tools to analyze these models is crucial to identifying and forestalling problems in generation, such as toxicity (Gehman et al., 2020) or factual errors (Kryscinski et al., 2020; Goyal and Durrett, 2020, 2021).

Although interpreting classification models for NLP has been widely studied from perspectives like feature attribution (Ribeiro et al., 2016; Sundararajan et al., 2017) and influence functions (Koh and Liang, 2017; Han et al., 2020), summarization specifically introduces some additional elements that make these techniques hard to apply directly. First, summarization models make sequential decisions from a very large state space. Second, encoder-decoder models have a special structure, featuring a complex interaction of decoder-side and encoder-side computation to select the next word. Third, pre-trained LMs blur the distinction between relying on implicit prior knowledge or explicit instance-dependent input.

This paper aims to more fully interpret the stepwise prediction decisions of neural abstractive summarization models.[1] First, we roughly bucket generation decisions into one of several *modes* of generation. After confirming that the models we use are robust to seeing partial inputs, we can probe the model by predicting next words with various model **ablations**: a basic language model with no input ($\text{LM}_{\emptyset}$), a summarization model with no input ($\text{S}_{\emptyset}$), with part of the document as input (S_{part}), and with the full document as input (S_{full}). These ablations tell us when the decision is context-independent (generated in an LM-like way), when it is heavily context-dependent (generated from the context), and more. We *map* these regions in Figure 2 and can use these maps to coarsely analyze model behavior. For example, 17.6% of the decisions on

[1] Code and visualization are available at https://github.com/jiacheng-xu/sum-interpret

*Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics
and the 11th International Joint Conference on Natural Language Processing*, pages 6925–6940
August 1–6, 2021. ©2021 Association for Computational Linguistics

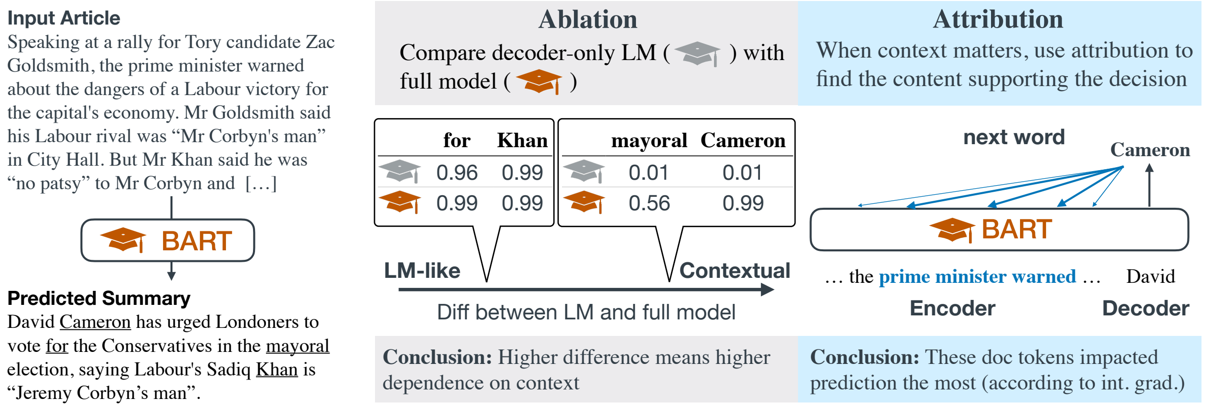
Input Article
Speaking at a rally for Tory candidate Zac Goldsmith, the prime minister warned about the dangers of a Labour victory for the capital's economy. Mr Goldsmith said his Labour rival was "Mr Corbyn's man" in City Hall. But Mr Khan said he was "no patsy" to Mr Corbyn and [...]
BART
Predicted Summary
David Cameron has urged Londoners to vote for the Conservatives in the mayoral election, saying Labour's Sadiq Khan is "Jeremy Corbyn's man".
Ablation
Compare decoder-only LM ( ) with full model ( )
for Khan
0.96 0.99
0.99 0.99
mayoral Cameron
0.01 0.01
0.56 0.99
LM-like
Contextual
Diff between LM and full model
Conclusion: Higher difference means higher dependence on context
Attribution
When context matters, use attribution to find the content supporting the decision
next word
Cameron
BART
... the prime minister warned ...
David
Encoder
Decoder
Conclusion: These doc tokens impacted prediction the most (according to int. grad.)

Figure 1: Our two-stage ablation-attribution framework. First, we compare a decoder-only language model (not fine-tuned on summarization task, and not conditioned on the input article) and a full summarization model. They are colored in gray and orange respectively. the The higher the difference, the more heavily model depends on the input context. For those context-dependent decisions, we conduct content attribution to find the relevant supporting content with methods like Integrated Gradient or Occlusion.

XSum are in the lower-left corner (LM-like), which means they do not rely much on the input context.

Second, we focus on more fine-grained **attribution** of decisions that arise when the model *does* rely heavily on the source document. We carefully examine interpretations based on several prior techniques, including occlusion (Zeiler and Fergus, 2014), attention, integrated gradients (Sundararajan et al., 2017), and input gradients (Hechtlinger, 2016). In order to evaluate and compare these methods, we propose a comprehensive evaluation based on presenting counterfactual, partial inputs to quantitatively assess these models' performance with different subsets of the input data.

Our two-stage analysis framework allows us to (1) understand how each individual decision depends on context and prior knowledge (Sec 3), (2) find suspicious cases of memorization and bias (Sec 4), (3) locate the source evidence for context dependent generation (Sec 5). The framework can be used to understand more complex decisions like sentence fusion (Sec 6).

2 Background & Setup

A seq2seq neural abstractive model first encodes an input document with m sentences $(s_1, \cdots, s_m)$ and n tokens $(w_1, w_2, \cdots, w_n)$, then generates a sequence of tokens $(y_1, \cdots, y_T)$ as the summary. At each time step t in the generation phase, the model encodes the input document and the decoded summary prefix and predicts the distribution over tokens as $p(y_t \mid w_1, w_2, \ldots, w_m, y_{<t})$.

2.1 Target Models & Datasets

We investigate the English-language CNN/DM (Hermann et al., 2015) and XSum (Narayan et al., 2018) datasets, which are commonly used to fine tune pre-trained language models like BART, PEGASUS and T5. As shown in past work (Narayan et al., 2018; Chen et al., 2020b; Xu et al., 2020a), XSum has significantly different properties from CNN/DM, so these datasets will show a range of model behaviors. We will primarily use the development sets for our analysis.

We focus on BART (Lewis et al., 2020), a state-of-the-art pre-trained model for language modeling and text summarization. Specifically, we adopt 'bart-large' as the language model $\mathrm{M_{LM}}$, 'bart-large-xsum' as the summarization model $\mathrm{M_{SUM}}$ for XSum, and 'bart-large-cnn' for CNN/DM, made available by Wolf et al. (2019). BART features separate LM and summarization model sharing the same subword tokenization method.[2]

Our approach focuses on teasing apart these different modes of decisions. We first run the full model to get the predicted summary $(y_1, \cdots, y_T)$. We then analyze the distribution placed by the full model $\mathbf{S_{full}}$ to figure out what contributes towards the generation of the next token.

2.2 Overview of Ablation and Attribution

Figure 1 shows our framework with an example of our analysis of four generation decisions. In

[2] Our analysis can generalize to other pre-trained models, but past work has shown BART and PEGASUS to be roughly similar in terms of behavior (Xu et al., 2020a), so we do not focus on this here.

Config	LM$_\emptyset$	S$_\emptyset$	S$_{\text{part}}$	S$_{\text{full}}$
Decoder prefix	✓	✓	✓	✓
Input document	✗	✗	partial	full
Model parameters	M$_{\text{LM}}$	M$_{\text{SUM}}$	M$_{\text{SUM}}$	M$_{\text{SUM}}$

Table 1: Model configurations with different amount of input document and back-end model. M$_{\text{LM}}$ and M$_{\text{SUM}}$ are the BART language model and summarization model respectively. S$_\emptyset$ is the summarization model without any source document (encoder) input.

the **ablation** stage, we compare the predictions of different model and input configurations. The goal of this stage is to coarsely determine the mode of generation. Here, *for* and *Khan* are generated in an LM-like way: the model already has a strong prior that *Sadiq* should be *Sadiq Khan* and the source article has little impact on this decision. *Cameron*, by contrast, does require the source in order to be generated. And *mayoral* is a complex case, where the model is not strictly copying this word from anywhere in the source, but instead using a nebulous combination of information to generate it. In the **attribution** stage, we interpret such decisions which require more context using a more fine-grained approach. Given the predicted prefix (like *David*), target prediction (like *Cameron*), and the model, we use attribution techniques like integrated gradients (Sundararajan et al., 2017) or LIME (Ribeiro et al., 2016) to track the input which contributes to this prediction.

2.3 Ablation Models and Assumptions

The configurations we use are listed in Table 1 and defined as follows:

LM$_\emptyset$ is a pre-trained language model only taking the decoded summary prefix as input. We use this model to estimate what a pure language model will predict given the prefix. We denote the prediction distribution as $P_{\text{LM}_\emptyset} = P(y_t \mid y_{<t}; M_{\text{LM}})$.

S$_\emptyset$ is the same BART summarization model as S$_{\text{full}}$, but without the input document as the input. That is, it uses the same parameters as the full model, but with no input document fed in. We use the prediction of this model to estimate how strong an effect the in-domain training data has, but still treating the model as a decoder-only language model. It is denoted as $P_\emptyset = P(y_t \mid y_{<t}; M_{\text{SUM}})$. Figure 1 shows how this can effectively identify cases like *Khan* that surprisingly do not rely on the input document.

S$_{\text{part}}$ is a further step closer to the full model: this is the BART summarization model conditioned on the decoder prefix and *part of* the input document, denoted as $P_{\text{part}} = P(y_t \mid y_{<t}, \{s_i\}; M_{\text{SUM}})$ where $\{w_i\}$ is a subset of tokens of the input document. The selected content could be a continuous span, or a sentence, or a concatenation of several spans or sentences.

Although M$_{\text{SUM}}$ is designed and trained to condition on input document, we find that the model also works well with no input, little input and incomplete sentences. As we will show later, there are many cases that this scheme successfully explains; we formalize our assumption as follows:

Assumption 1 *If the model executed on partial input nearly reproduces the next word distribution of the full model, then we view that partial context as a **sufficient** (but perhaps not necessary) input to explain the model's behavior.*

Here we define *partial input* as either just the decoded summary so far or the summary and partial context. In practice, we see two things. First, when considering just the decoder context (i.e., behaving as an LM), the partial model may reproduce the full model's behavior (e.g., *Khan* in Figure 1). We do not focus on explaining these cases in further detail. While conceivably the actual conditional model might internally be doing something different (a risk noted by Rudin (2019)), this proves the existence of a decoder-only proxy model that reproduces the full model's results, which is a criterion used in past work (Li et al., 2020). Second, when considering partial inputs, the model frequently requires one or two specific sentences to reproduce the full model's behavior, suggesting that the given contexts are both necessary *and sufficient*.

Because these analyses involve using the model on data significantly different than that which it is trained on, we want another way to quantify the importance of a word, span, or sentence. This brings us to our second assumption:

Assumption 2 *In order to say that a span of the input or decoder context is important to the model's prediction, it should be the case that this span is demonstrated to be important in counterfactual settings. That is, modified inputs to the model that include this span should yield closer predictions than those that don't.*

This criterion depends on the set of counterfactuals that we use. Rather than just word removal (Ribeiro et al., 2016), we will use a more comprehensive set of counterfactuals (Miller, 2019; Jacovi and Goldberg, 2020) to quantify the importance of input tokens. We describe this more in Section 5.

2.4 Distance Metric

Throughout this work, we rely on measuring the distance between distributions over tokens. Although KL divergence is a popular choice, we found it to be very unstable given the large vocabulary size, and two distributions that are completely different would have very large values of KL. We instead use the L_1 distance between the two distributions: $D(P, Q) = \sum_i |p_i - q_j|$. This is similar to using the Earth Mover's Distance (Rubner et al., 1998) over these two discrete distributions, with an identity transportation flow since the distributions are defined over the same set of tokens.

3 Ablation: Mapping Model Behavior

Based on Assumption 1, we can take a first step towards understanding these models based on the partial models described in Section 2.3. Previous work (See et al., 2017; Song et al., 2020) has studied model behavior based on externally-visible properties of the model's generation, such as identifying novel words, differentiating copy and generation, and prediction confidence, which provides some insight about model's behavior (Xu et al., 2020a). However, these focus more on shallow comparison of the input document, the generated summary, and the reference summary, and do not focus as strongly on the model.

We propose a new way of mapping the prediction space, with maps[3] for XSum and CNN/DM shown in Figure 2. Each point in the map is a single subword token being generated by the decoder on the development set at inference time; that is, each point corresponds to a single invocation of the model. This analysis does not depend on the reference summary at all.

The x-axis of the map shows the distance between $\mathrm{LM}_\emptyset$ and $\mathrm{S}_{\mathrm{full}}$, using the metric defined in Section 2.4 which ranges from 0 to 2. The y-axis shows the distance between $\mathrm{S}_\emptyset$ and $\mathrm{S}_{\mathrm{full}}$. Other choices of partial models for the axes are possible (or more axes), but we believe these show two important factors. The x-axis captures **how much the generic pre-trained language model agrees with the full model's predictions**. The y-axis captures **how much the decoder-only summarization model agrees with the full model's predictions**. The histogram on the sides of the map show counts along with each vertical or horizontal slice.

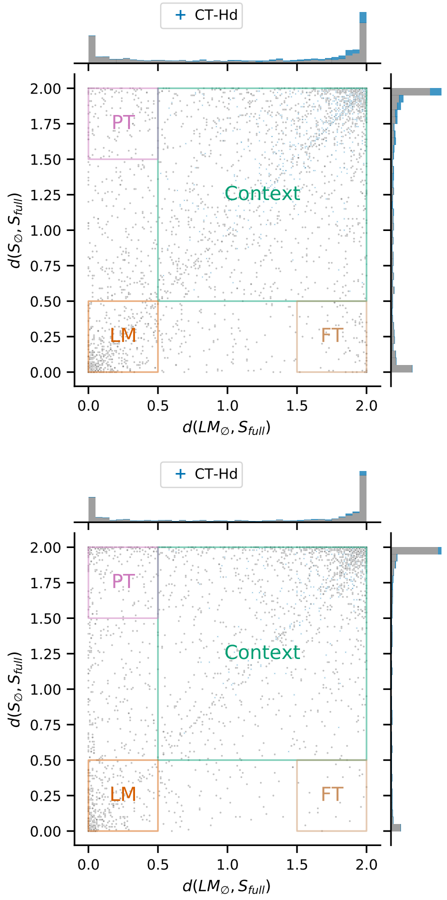

Figure 2: Map of model behavior on XSum (top) and CNN/DM (bottom). The x-axis and y-axis show the distance between $\mathrm{LM}_\emptyset$ and $\mathrm{S}_{\mathrm{full}}$, and distance between $\mathrm{S}_\emptyset$ and $\mathrm{S}_{\mathrm{full}}$. The regions characterize different generation modes, defined in Section 3.

Modes of decisions We break these maps into a few coarse regions based on the axis values. We list the coordinates of the bottom left corner and the upper right corner. These values were chosen by inspection and the precise boundaries have little effect on our analysis, as many of the decisions fall into the corners or along sides.

LM ([0, 0], [0.5, 0.5]) contains the cases where $\mathrm{LM}_\emptyset$ and $\mathrm{S}_\emptyset$ both agree with $\mathrm{S}_{\mathrm{full}}$. These decisions are easily made using only decoder information, even without training or knowledge of the input document. These are cases that follow from the constraints of language models, including function words, common entities, or idioms.

[3] While our axes are very different here, our mapping concept loosely follows that of Swayamdipta et al. (2020).

CTX ([0.5, 0.5], [2, 2]) contains the cases where the input is needed to make the prediction: neither decoder-only model can model these decisions.

FT ([1.5, 0], [2, 0.5]) captures cases where the fine-tuned decoder-only model is a close match but the pre-trained model is not. This happens more often on XSum and reflects memorization of training summaries, as we discuss later.

PT ([0, 1.5], [0.5, 2]) is the least intuitive case, where $\text{LM}_\emptyset$ agrees with S_{full} but $\text{S}_\emptyset$ does not; that is, fine-tuning a decoder-only model causes it to work *less well*. This happens more often on CNN/DM and reflects memorization of data in the pre-training corpus.

3.1 Coloring the Map with Context Probing

While the map highlights some useful trends, there are many examples that do rely heavily on the context that we would like to further analyze. Some examples depend on the context in a sophisticated way, but other tokens like parts of named entities or noun phrases are simply copied from the source article in a simple way. Highlighting this contrast, we additionally subdivide the cases by how they depend on the context.

We conduct a sentence-level presence probing experiment to further characterize the generation decisions. For a document with m sentences, we run the S_{part} model conditioned on each of the sentences in isolation. We can obtain a sequence of scalars $P_{\text{sent}} = (P_{\text{part}}(s_i); i \in [1, m])$. We define CTX-Hd ("context-hard") cases as ones where $\max(P_{\text{sent}})$ is low; that is, where no single sentence can yield the token, as in the case of sentence fusion. These also reflect cases of high entropy for S_{full}, where any perturbation to the input may cause a big distribution shift. The first, second and third quartile of $\max(P_{\text{sent}})$ is $[0.69, 0.96, 1.0]$ and $[0.95, 1.0, 1.0]$ on XSum and on CNN/DM.

3.2 Region Count & POS Tags

To roughly characterize the words generated in different regions of the map, in Table 2, we show the percentage of examples falling to each region and the top 3 POS tags for each region on the XSum map. From the frequency of these categories, we can tell more than two-thirds of the decisions belong to the Context category. 17.6% of cases are in LM, the second-largest category. In the LM region, ADP and DET account for nearly half of the data points, confirming that these are largely function words. Nouns are still prevalent, accounting for 13.5% of the category. After observing the data, we found that these points represent commonsense knowledge or common nouns or entities, like "Nations" following "United" or "Obama" following "Barack" where the model generates these without relying on the input. Around 8% of cases fall into gaps between these categories. Only 2.5% and 2.1% of the generations fall into the PT and FT, respectively. These are small but significant cases, as they clearly show the biases from the pre-training corpus and the fine-tuning corpus. We now describe the effects we observe here.

Cat	Freq(%)	Top 3 POS Tags w/ Freq(%)		
LM	17.6%	ADP 28.6%	DET 21.1%	NOUN 13.5%
CTX	69.6%	NOUN 20.3%	VERB 15.9%	PROPN 15.6%
PT	2.5%	PROPN 37.0%	NOUN 13.0%	ADP 13.0%
FT	2.1%	AUX 31.6%	NOUN 23.7%	PROPN 15.8%
ALL	100.0%	NOUN 18.9%	PROPN 14.3%	ADP 13.9%

Table 2: Percentage of examples falling into each region and the top POS tags for each regions in the XSum map.

4 Bias from Training Data

One benefit of mapping the predictions is to detect predictions that are suspiciously likely given one language model but not the other, specifically those in the PT and FT regions. CNN/DM has more cases falling into PT than XSum so we focus on CNN/DN for PT and XSum for FT.

PT: Bias from the Pretraining Corpus The data points falling into the PT area are those where $\text{LM}_\emptyset$ prediction is similar to S_{full} prediction but the $\text{S}_\emptyset$ prediction is very different from S_{full}. We present a set of representative examples from the PT region of the CNN/DM map in Table 3. For the first example, *match* is assigned high probability by $\text{LM}_\emptyset$ and S_{full}, but not by the no-input summarization models. The cases in this table exhibit a suspiciously high probability assigned to the correct answer in the base LM: its confidence about Kylie Jenner vs. Kyle Min(ogue) is uncalibrated with what the "true" probabilities of these seem likely to be to our human eyes.

One explanation which we investigate is whether the validation and test sets of benchmark datasets

Prefix **Target**	Relevant Context	$\text{LM}_\emptyset$	$\text{S}_\emptyset$	$\text{S}_{\emptyset\text{X}}$	S_{full}
Danny Welbeck was named man of the **match**	[...] , the booming PA system kicked in and proclaimed that Danny Welbeck was England's man of the match.	0.99 match	0.99 year	0.99 year	0.99 match
Gail Scott was desperate to emulate Kylie **Jenner**	Gail Scott was desperate to emulate Kylie Jenner's famous pout but didn't want to spend [...]	0.99 Jenner	0.99 Min	0.99 Min	0.80 Jenner
Some 1,200 of the Reagan's crew will be executing what the **Navy**	Some 1,200 of the Reagan's crew will be executing what the Navy calls a three-hull swap, [...]	0.78 Navy	0.96 president	0.96 president	0.97 Navy
Mason was drafted into the England squad following the withdrawal of Adam **Lallana**	Mason was drafted into the England squad following the withdrawal of Adam Lallana and [...]	0.96 L	0.34 F	0.29 Ant	0.99 L

Table 3: Examples of bias from the pre-trained language model (PT) on CNN/DM. The model's predicted token is in bold following the decoder prefix, then we list relevant context from the corresponding input document and the top-1 predicted token along with probability of $\text{LM}_\emptyset$ (BART language model), $\text{S}_\emptyset$, $\text{S}_{\emptyset\text{X}}$ (the XSum model with no input) and S_{full}. Suspiciously, the LM without fine-tuning is very confident, more so than the no-input summarization model. We show more examples in Table 9.

like CNN/DM are contained in the pre-training corpus, which could teach the base LM these patterns. Several web crawls have been used for different models, including C4 (Raffel et al., 2020), OpenWebText (Radford et al., 2019), CC-News (Liu et al., 2019). Due to the availability of the corpus, we only check OpenWebText, which, as part of C4, is used for models like GPT-2, PEGASUS and T5.

According to Hermann et al. (2015), the validation and test sets of CNN/DM come from March and April 2015, respectively. We extract the March to May 2015 dump of OpenWebText and find that 4.46% (512 out of 11,490) test examples and 3.31% (442 out of 13,368) validation examples are included in OpenWebText.[4] Our matching criteria is more than three 7-gram word overlaps between the pre-training document and reference summaries from the dataset; upon inspection, over 90% of the cases flagged by this criterion contained large chunks of the reference summary.

Our conclusion is that the pre-trained language model has likely memorized certain articles and their summaries. Other factors could be at play: other types of knowledge in the language model (Petroni et al., 2019; Shin et al., 2020; Talmor et al., 2020) such as key entity cooccurrences, could be contributing to these cases as well and simply be "forgotten" during fine-tuning. However, as an analysis tool, ablation suggested a hypothesis about data overlap which we were able to partially confirm, which supports its utility for understanding summarization models.

Group/Bigram	$\frac{\#(w_{t-1},w_t)}{\#w_{t-1}}$	
	XS	CD
of **letters**	0.001	0.000
letters **from**	0.494	0.026
African **journalists**	0.091	0.000
m **(£**	0.420	0.300
(Close	0.058	0.000
Britain**'s**	0.586	0.291
All FT cases	0.162	0.060

Table 4: Example patterns from FT. w_t is in bold. We show the relative frequency counts of each bigram. In aggregate (last row), bigrams in FT cases are much more frequent in the XSum training data than in CNN/DM.

FT: Bias from Fine-tuning Data We now examine the data points falling in the bottom right corner of the map, where the fine-tuned LM matches the full model more closely than the pre-trained LM.

In Table 4, we present some model-generated bigrams found in the FT region of XSum and compare the frequency of these patterns in the XSum and CNN/DM training data. Not every generation instance of these bigrams falls into the FT region, but many do. Table 4 shows the relative probabilities of these counts in XSum and CNN/DM, showing that these cases are all very common in XSum training summaries. The aggregate over all decisions in this region (the last line) shows this pattern as well. These can suggest larger patterns: the first three come from the common phrase *in our series of letters from African journalists* (starts 0.5% of

[4] This is an approximation since we cannot precisely verify the pre-training datasets for each model, but it is more likely to be an underestimate than an overestimate. We only extract pre-training documents from cnn.com and dailymail.co.uk from a limited time range, so we may fail to detect snippets of reference summaries that show up in other time ranges of the scrape or in other news sources, whether through plagiarism or re-publishing.

Target	w_{attr}	DISPTOK $n = 0 \rightarrow 1$	RMTOK $n = 0 \rightarrow 1$
Cameron	minister	**0.01** → **0.90**	0.99 → 0.99
for	Labour	0.96 → 0.94	0.98 → 0.91
mayoral	100	0.01 → 0.01	0.57 → 0.57
S(adiq)	Khan	0.01 → 0.01	**0.97** → **0.38**
Khan	Jeremy	0.99 → 0.99	0.99 → 0.99

Table 5: Examples of DISPTOK and RMTOK. We show the change of the prediction probability of the target token when displaying or masking the w_{attr} token, which is the highest rank token from the occlusion method. Significant change is marked in bold.

summaries in XSum). Other stylistic markers, such as ways of writing currency, are memorized too.

5 Attribution

As shown in Table 2, more than two thirds of generation steps actually do rely heavily on the context. Here, we focus specifically on identifying which aspects of the input are important for cases where the input *does* influence the decision heavily using attribution methods.

Each of the methods we explore scores each word w_i in the input document with a score α_i. The score can be a normalized distribution, or a probability value ranging from 0 to 1. For each method, we rank the tokens in descending order by score. To confirm that the tokens highlighted are meaningfully used by the model when making its predictions, we propose an evaluation protocol based on a range of counterfactual modifications of the input document, taking care to make these compatible with the nature of subword tokenization.

5.1 Evaluation by Adding and Removing

Our evaluation focuses on the following question: given a budget of tokens or sentences, how well does the model reconstruct the target token y_t when shown the important content selected by the attribution method? Our metric is the cross entropy loss of predicting the model-generated next token given different subsets of the input.[5]

Methods based on adding or removing single tokens have been used to evaluate before (Nguyen, 2018). However, for summarization, showing the model partial or ungrammatical inputs in the source may significantly alter the model's behavior. To address this, we use four methods to evaluate under a range of conditions, where in each case the model has a specific budget. Our conditions are: 1. DISPTOK selects n tokens as the input. 2. RMTOK shows the document with n tokens *masked* instead of deleted.[6] 3. DISPSENT selects n sentences as the input, based on cumulative attribution over the sentence. 4. RMSENT removes n sentences from the document as the input.

Table 5 shows examples of these methods applied to the examples from Figure 1. These highlight the impact of key tokens in certain generation cases, but not all.

We describe the details of how we feed or mask the tokens in TOK in Appendix. C. The sentence-level methods are guaranteed to return grammatical input. Token-based evaluation is more precise which helps locating the exact feature token, but the trade-off is that the input is not fully natural.

5.2 Methods

We use two baseline methods: **Random**, which randomly selects tokens or sentences to display or remove, and **Lead**, which selects tokens or sentences according to document position, along with several attribution methods from prior work. **Occlusion** (Zeiler and Fergus, 2014) involves iteratively masking every single token or remove each sentence in the document and measuring how the prediction probability of the target token changes. Although **attention** has been questioned (Jain and Wallace, 2019), it still has some value as an explanation technique (Wiegreffe and Pinter, 2019; Serrano and Smith, 2019). We pool the attention heads from the last layer of the Transformer inside our models, ignoring special tokens like SOS.

Finally, we use two gradient-based techniques (Bastings and Filippova, 2020). **Input Gradient** is a saliency based approach taking the gradient of the target token with respect to the input and multiplying by the input feature values. **Integrated Gradients** Sundararajan et al. (2017) computes gradients of the model input at a number of points interpolated between a reference "baseline" (typically an all-MASK input) and the actual input. This computes a path integral of the gradient.

[5] The full model is not a strict bound on this; restricting the model to only see salient content could actually increase the probability of what was generated. However, because we have limited ourselves to CTX examples and are aggregating across a large corpus, we do not observe this in our metrics.

[6] Note that we do not directly remove the tokens because this approach typically makes the sentence ungrammatical. Token masks are a more natural type of input to models that are pre-trained with these sorts of masks anyway.

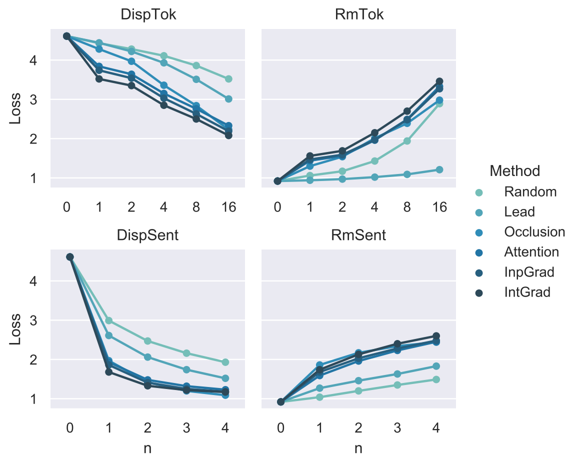

Figure 3: Four-way evaluation for our content attribution methods. The reported value is the NLL loss with respect to the predicted token. Lower is better for display methods and higher is better for removal methods (we "break" the model more quickly). $n = 0$ means the baseline when there is no token or sentence displayed in DISP or removed or masked in RM.

Attribution Aggregation for Sentence-level Evaluation We have described the six methods we use for token-level evaluation. To evaluate these methods on the sentence level benchmark, we aggreagate the attributions in each sentence $attr(s_i) = \sum_{j=0}^{d} attr(w_j)/d$. Hence we can obtain a ranking of sentences by their aggregated attribution score.

5.3 Results

In Figure 3, we show the token-level and sentence-level comparison of the attribution methods on the CTX examples in XSum. IntGrad is the best technique overall, with InpGrad achieving similar performance. Interestingly, occlusion underperforms other techniques when more tokens are removed, despite our evaluation being based on occlusion; this indicates that single-token occlusion is not necessarily the strongest attribution method. We also found that all of these give similar results, regardless of whether they present the model with a realistic input (sentence removal) or potentially ungrammatical or unrealistic input (isolated tokens added/removed).

Our evaluation protocol shows better performance from gradient-based techniques. The combination of four settings tests a range of counterfactual inputs to the model and increases our confidence in these conclusions.

Prob	Content
0.00	1. Atherton, 28, has won all seven races this season and 13 in a row, a run stretching back to 2015.
0.09	2. The world champion had already sealed the 2016 World Cup crown in Canada last month but won in Andorra on Saturday to end the World Cup season unbeaten.
0.01	3. She has now won five overall World Cup titles in downhill. [...]
0.16	5. Trek Factory Racing's Atherton won the final race by 6.5 seconds ahead of Australian Tracey Hannah and Myriam Nicole of France.
Prob	Comb. & Predict Summary
0.71	(2, 5) Britain's Laura Atherton has won the **UCI** Mountain Bike World Cup [...]
Prob	Content
0.01	1. Dujardin, 30, and Valegro won individual and team dressage gold for Britain at London 2012 and have since won World and European titles.
0.00	2. But, she says, the Olympics in Brazil next summer will be the horse's last.
0.01	3. "This will be Valegro's retirement after Rio so I want to go out there and want to enjoy every last minute," Dujardin told BBC Points West.
Prob	Comb. & Predict Summary
0.63	(1, 2) Olympic dressage champion Charlotte Dujardin says she will **retire** from the sport after Rio Olympics.

Table 6: Examples of sentence fusion in the DISPSENT setting. We list the single sentence probability on the left side with the document, and the best combination with its probability at the bottom. We underline the tokens according to the top attributions of occlusion. Articles are truncated.

6 Case Study: Sentence Fusion

We now present a case study of the sort of analysis that can be undertaken using our two-stage interpretation method. We conduct an analysis driven by sentence fusion, a particular class of `CTX-Hd` cases. Sentence fusion is an exciting capability of abstractive models that has been studied previously (Barzilay and McKeown, 2005; Thadani and McKeown, 2013; Lebanoff et al., 2019, 2020).

We broadly identify cases of cross-sentence information fusion by first finding cases in `CTX-Hd` where the $\max(P_{sent}) < 0.5$, but two sentences combined enable the model to predict the word. We search over all $\binom{m}{2}$ combinations of sentences (m is the total number of sentences) and run the S_{part} model on each pair of sentences. We identify 16.7% and 6.0% of cases in CNN/DM and XSum, respectively, where conditioning on a pair of sentences increases the probability of the model's generation by at least 0.5 over any sentence in isolation.

In Table 6, we show two examples of sentence fusion on XSum in this category, additionally analyzed using the DISPSENT attribution method. In the first example, typical in XSum, the model has to predict the event name *UCI* without actually seeing it. The model's reasoning appears distributed over the document: it consults entity and event descriptions like *world champion* and *France*, perhaps to determine this is an international event. In the second example, we see the model again connects several pieces of information. The generated text is factually incorrect: the horse is retiring, and not Dujardin. Nevertheless, this process tells us some things that are going wrong (the model disregards the horse in the generation process), and could potentially be useful for fine-grained factuality evaluation using recent techniques (Tian et al., 2019; Kryscinski et al., 2020; Goyal and Durrett, 2020; Maynez et al., 2020).

The majority of the "fusion" cases we investigated actually reflect content selection at the beginning of the generation. Other cases we observe fall more cleanly into classic sentence fusion or draw on coreference resolution.

7 Related Work

Model interpretability for NLP has been intensively studied in the past few years (Ribeiro et al., 2016; Alvarez-Melis and Jaakkola, 2018; Jacovi et al., 2018; Chen et al., 2020a; Jacovi and Goldberg, 2020; DeYoung et al., 2020; Pruthi et al., 2020; Ye et al., 2021). However, many of these techniques are tailored to classification tasks like sentiment. For post-hoc interpretation of generation, most work has studied machine translation (Ma et al.; Li et al., 2020; Voita et al., 2020). Li et al. (2020) focus on evaluating explanations by finding surrogate models that are similar to the base MT model; this is similar to our evaluation approach in Section 5, but involves an extra distillation step. Compared to Voita et al. (2020), we are more interested in highlighting how and why changes in the source article will change the summary (*counterfactual explanations*).

To analyze summarization more broadly, Xu et al. (2020a) provides a descriptive analysis about models via uncertainty. Previous work (Kedzie et al., 2018; Zhong et al., 2019; Kryscinski et al., 2019; Zhong et al., 2019) has conducted comprehensive examination of the limitations of summarization models. Filippova (2020) ablates model input to control the degree of hallucination. Miao et al. (2021) improves the training of MT by comparing the prediction of LM and MT model.

Finally, this work has focused chiefly on abstractive summarization models. We believe interpreting extractive (Liu and Lapata, 2019) or compressive (Xu and Durrett, 2019; Xu et al., 2020b; Desai et al., 2020) models would be worthwhile to explore and could leverage similar attribution techniques, although ablation does not apply as discussed here.

8 Recommendations & Conclusion

We recommend a few methodological takeaways that can generalize to other conditional generation problems as well.

First, **use ablation to analyze generation models.** While removing the source forms inputs not strictly on the data manifold, ablation was remarkably easy, robust, and informative in our analysis. Constructing our maps only requires querying three models with no retraining required.

Second, to understand an individual decision, **use feature attribution methods on the source *only*.** Including the target context often muddies the interpretation since recent words are always relevant, but looking at attributions over the source and target together doesn't accurately convey the model's decision-making process.

Finally, to probe attributions more deeply, **consider adding or removing various sets of tokens.** The choice of counterfactuals to explain is an ill-posed problem, but we view the set used here as realistic for this setting (Ye et al., 2021).

Taken together, our two-step framework allows us to identify generation modes and attribute generation decisions to the input document. Our techniques shed light on possible sources of bias and can be used to explore phenomena such as sentence fusion. We believe these pave the way for future studies of targeted phenomena, including fusion, robustness, and bias in text generation, through the lens of these interpretation techniques.

Acknowledgments

Thanks to the members of the UT TAUR lab for helpful discussion, especially Tanya Goyal, Yasumasa Onoe, and Xi Ye for constructive suggestions. This work was partially supported by a gift from Salesforce Research and a gift from Amazon. Thanks as well to the anonymous reviewers for their helpful comments.

References

David Alvarez-Melis and Tommi S. Jaakkola. 2018. Towards robust interpretability with self-explaining neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 7786–7795, Red Hook, NY, USA. Curran Associates Inc.

Regina Barzilay and Kathleen R. McKeown. 2005. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155, Online. Association for Computational Linguistics.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2020. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*.

Hanjie Chen, Guangtao Zheng, and Yangfeng Ji. 2020a. Generating hierarchical explanations on text classification via feature interaction detection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5578–5593, Online. Association for Computational Linguistics.

Yiran Chen, Pengfei Liu, Ming Zhong, Zi-Yi Dou, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2020b. CDEvalSumm: An empirical study of cross-dataset evaluation for neural summarization systems. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3679–3691, Online. Association for Computational Linguistics.

Shrey Desai, Jiacheng Xu, and Greg Durrett. 2020. Compressive summarization with plausibility and salience modeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6259–6274, Online. Association for Computational Linguistics.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458, Online. Association for Computational Linguistics.

Katja Filippova. 2020. Controlled hallucinations: Learning to generate faithfully from noisy data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 864–870, Online. Association for Computational Linguistics.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Evaluating factuality in generation with dependency-level entailment. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3592–3603, Online. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2021. Annotating and modeling fine-grained factuality in summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563, Online. Association for Computational Linguistics.

Yotam Hechtlinger. 2016. Interpretation of prediction models using the input gradient. *arXiv preprint arXiv:1611.07634*.

Karl Moritz Hermann, Tomás Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. 2018. Understanding convolutional neural networks for text classification. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–65, Brussels, Belgium. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Chris Kedzie, Kathleen McKeown, and Hal Daumé III. 2018. Content selection in deep learning models of summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1818–1828, Brussels, Belgium. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, pages 1885–1894. PMLR.

Wojciech Kryscinski, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 540–551, Hong Kong, China. Association for Computational Linguistics.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346, Online. Association for Computational Linguistics.

Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020. Learning to fuse sentences with transformers for summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4136–4142, Online. Association for Computational Linguistics.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019. Analyzing sentence fusion in abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 104–110, Hong Kong, China. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Jierui Li, Lemao Liu, Huayang Li, Guanlin Li, Guoping Huang, and Shuming Shi. 2020. Evaluating explanation methods for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 365–375, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

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

Xutai Ma, Ke Li, and Philipp Koehn. An analysis of source context dependency in neural machine translation. In *21st Annual Conference of the European Association for Machine Translation*, page 189.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Mengqi Miao, Fandong Meng, Yijin Liu, Xiao-Hua Zhou, and Jie Zhou. 2021. Prevent the language model from being overconfident in neural machine translation.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Dong Nguyen. 2018. Comparing automatic and human evaluation of local explanations for text classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1069–1078, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4782–4793, Online. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. *OpenAI Blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SIGKDD)*.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. 1998. A metric for distributions with applications to image databases. In *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, pages 59–66. IEEE.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Abigail See, Peter J. Liiu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Kaiqiang Song, Bingqing Wang, Zhe Feng, Ren Liu, and Fei Liu. 2020. Controlling the amount of verbatim copying in abstractive summarization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8902–8909.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics-On What Language Model Pre-training Captures. *Transactions of the Association for Computational Linguistics*, 8:743–758.

Kapil Thadani and Kathleen McKeown. 2013. Supervised sentence fusion with single-stage inference. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1410–1418, Nagoya, Japan. Asian Federation of Natural Language Processing.

Ran Tian, Shashi Narayan, Thibault Sellam, and Ankur P Parikh. 2019. Sticking to the facts: Confident decoding for faithful data-to-text generation.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Elena Voita, Rico Sennrich, and Ivan Titov. 2020. Analyzing the source and target contributions to predictions in neural machine translation. *arXiv preprint arXiv:2010.10907*.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *arXiv preprint arXiv:1910.03771*.

Jiacheng Xu, Shrey Desai, and Greg Durrett. 2020a. Understanding neural abstractive summarization models via uncertainty. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6275–6281, Online. Association for Computational Linguistics.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3292–3303, Hong Kong, China. Association for Computational Linguistics.

Jiacheng Xu, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020b. Discourse-aware neural extractive text summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5021–5031, Online. Association for Computational Linguistics.

Xi Ye, Rohan Nair, and Greg Durrett. 2021. Evaluating explanations for reading comprehension with realistic counterfactuals. *arXiv preprint arXiv:2104.04515*.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. Proceedings of Machine Learning Research. PMLR.

Ming Zhong, Pengfei Liu, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. 2019. Searching for effective neural extractive summarization: What works and what's next. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1049–1058, Florence, Italy. Association for Computational Linguistics.

	Complexity	Time	Memory
Occlusion	$\mathcal{O}(n^3)$	$\sim$ 33x	$\sim$ 2.1x
S+Occlusion	$\mathcal{O}(s \times d^2 + d^3)$	1x	1x

DISPTOK	0	1	2	4	8
Occlusion	4.61	4.28	3.97	3.36	2.84
S+Occlusion		4.27	3.93	3.31	**2.71**

Table 7: (Upper) The complexity, actual time and GPU memory comparison of Occlusion and S+Occlusion. We set the same environment for both experiments. (Bottom) Token-level selection evaluation on Occlusion and S+Occlusion. The reported number is the NLL loss of w.r.t. the token predicted by S_{full}.

A Validity of Decoder-Only Model in $S_\emptyset$ Setting

We use an off-the-shelf BART summarization model as the decoder-only model for the ablation study. To guarantee the validity of the usage of the off-the-shelf model for ablation study, we also *fine-tuned* a BART language model where encoding input is empty and the decoding target is the reference summary. We compare the model output with the $S_\emptyset$ output in the paper. For 55% of cases the top-1 predictions of these two models agree with each other. This is pretty high, and suggests that the $S_\emptyset$ is at least doing reasonably. Note that fine-tuning will probably give rise to different behavior on the 70% of CTX cases, since the $S_\emptyset$ will hallucinate differently than the newly fine-tuned model (which further suggests why our analysis should focus on $S_\emptyset$).

B Examples of PT

We present more examples of bias from the pre-trained language model on CNN/DM in Table 9. In Table 3 we have shown the cases where the memorized phrases are proper nouns or nouns. Here we provide examples of other types like function words. The memorization of function words like *with* or *and* can be challenging to spot using other means due to their ubiquity.

C Implementation Detail for TOK

We rank the attribution score of all subword tokens rather than words. However, to provide necessary context for DISPTOK and to avoid information leakage in RMTOK, we extend the selection by a context window to collect neighboring word pieces. We illustrate the way of fulfilling budget with an example.

Labels: LM CTX CTX-Hd PT FT

Examples from XSum
Hundreds of people have attended a memorial service in Liverpool.
Two code violations for Nicolas Almagro and Pablo Cuevas at the Australian Open were described as disgraceful.
In our series of letters from African journalists film maker and columnist Farai Sevenzo looks at the challenges facing Nigeria's President Muhammadu Buhari.
Four people have been arrested after a BBC Panorama investigation uncovered shocking abuse at a private hospital.
West Indies Shabnim Ishaq has been ruled out of the rest of the Women's World Cup.

Examples from CNN/DM
In the worst cases, doctors have reported patients showing up because they were hungover, their false nails were hurting or they had paint in their hair. More than four million visits a year are unnecessary and cost the NHS £290million annually.
Elski Felson of Los Angeles, California, decided to apply for a Community Support Specialist role at Snapchat via the social media app. In just over three minutes, the tech enthusiast created a video resume.
Chelsea supporters have been involved in the highest number of reported racist incidents as they travelled to and from matches on trains. The information, gathered from 24 police forces across the country, shows there have been over 350 incidents since 2012.
Kris-Deann Sharpley was on maternity leave and had just given birth to her first child. Her body was found in the bathroom of her father's home.

Table 8: More examples of predicted summaries with the colors following the map. For LM and punctuation we use the default color. The majority of CNN/DM predictions are continuous spans of CTX excluding CTX-Hd, meaning the model is frequently copying.

Bur	#berry	bets	on	**new**	branding
①	②		④	③	⑤

In this example "Bur" receives the highest score and "new" the second. We use a context windows of size 1 and a budget of $n = 4$ tokens. In DISPTOK, the input will be "⟨sos⟩Burberry, on new⟨eos⟩"; In RMTOK, the input will be ⟨sos⟩## bets## branding⟨eos⟩ where # stands for the MASK token. If $n = 5$, *branding* will be added or masked.

D Efficient Two-Stage Selection Model

For long documents in summarization, attribution methods can be computationally expensive. Occlusion requires running inference once for each token in the input document. Gradient-based methods store the gradients and so require a lot of GPU

Prefix **Target**	Relevant Context	$\text{LM}_{\emptyset}$	$\text{S}_{\emptyset}$	$\text{S}_{\emptyset\text{X}}$	S_{full}
Labour released five mugs to coincide with the Launch of Ed **Miliband**	Labour released five mugs to coincide with the Launch of Ed Miliband's five election pledges.	0.95 Miliband	0.94 ible	0.68 ible	0.99 Miliband
British supermodel, Georgia **May**	British supermodel, Georgia May Jagger, 23, poses next to a floral plane designed by Masha Ma.	0.93 May	0.34 -	0.25 [SPACE]	0.99 May
Peter Schmeichel has urged Manchester United to sign Zlatan Ibrahimovic. Ibrahimovic has been linked **with**	The well travelled Sweden international has been linked with a move to Old Trafford in the past and, ...	0.99 with	0.98 to	0.99 to	0.99 with
Tunisian security forces kill two attackers as they end the siege at the Bardo Museum. The death toll, which included 17 tourists **and**	But the death toll, which included 17 tourists and at least one Tunisian security officer, could climb.	0.99 and	0.94 ,	0.99 ,	0.99 and
The costume was designed by three-**time**	The costume was designed by three-time Oscar-winner Colleen Atwood, ...	0.98 time	0.97 year	0.73 and	0.99 time
U.S. State **Department**	What has U.S. State Department subcontractor Alan Gross been up to since ...	0.99 Department	0.96 of	0.34 University	0.99 Department

Table 9: More examples of PT cases from the pre-trained language model.

Tok	Disp ↓							Rm ↑						
	0	1	2	4	8	16	−Δ	0	1	2	4	8	16	Δ
Random	4.61	4.43	4.28	4.11	3.86	3.52	0.57	0.92	1.06	1.17	1.43	1.94	2.89	0.78
Lead		4.44	4.22	3.93	3.51	3.01	0.79		0.94	0.97	1.02	1.09	1.21	0.13
Occlusion		4.28	3.97	3.36	2.84	2.23	1.27		1.30	1.54	2.01	2.39	2.98	1.12
Attention		3.84	3.64	3.15	2.76	2.33	1.47		1.44	1.56	1.96	2.49	3.33	1.24
InpGrad		3.74	3.54	3.03	2.63	2.19	1.58		1.47	1.59	1.97	2.48	3.27	1.24
IntGrad		3.52	3.35	2.85	2.50	2.08	**1.75**		1.56	1.69	2.15	2.70	3.46	**1.39**

Table 10: Token-level evaluation for content attribution methods. The reported value is the NLL loss w.r.t. the predicted token. $n = 0$ means the baseline when there is no token displayed in Disp or masked in Rm.

memory when the document is long. These techniques spend time and memory checking words that have little impact on the generation.

In order to improve the efficiency of these methods, we propose an efficient alternative where we first run sentence level presence probing on the full document, and then run attribution methods locally on the top-k sentences. We call the proposed model *S+[method]* where *method* can be arbitrary attribution methods including occlusion, attention, InpGrad and IntGrad.

We define our notation as follows: s, n and d are the number of sentences, the number of tokens in the document, and the number of tokens in each sentence, respectively. For the occlusion method, we can run inference s times to pre-select important sentences, each of which costs $\mathcal{O}(d^2)$ times due to self-attention. The attribution is then applied only to only one or few sentences so the complexity is now $\mathcal{O}(k \times d^2 \times d)$ where k is the number of top sentences used for attribution. In our experiments, we set $k = 2$ and $n \leq 500$. Compared to the complexity of the regular model $\mathcal{O}(n^3)$, the complexity of the two-stage model is only $\mathcal{O}(s \times d^2 + k \times d^2 \times d)$.

In Table 7 we compare the complexity and actual run time and memory usage. We batch the occlusion operation and the batch size is set to 100. We can see a huge reduction in running time and a significant drop in memory usage.

Takeaway A two-stage selection model is much more efficient, yielding a 97% running time reduction on the occlusion method. The downside of this method is that it only produces single-sentence attributions, and so isn't appropriate in cases involving sentence fusion.

Following (Vaswani et al., 2017), we compare the complexity for all methods in Table 12. n is the number of tokens in the document. d is the number of tokens in each sentence. s is the number of sentences in the document. r is the number of steps in the integral approximation of Integrated Gradient. bp indicates the time consumption of one back-

SENT	DISP ↓						RM ↑					
	0	1	2	3	4	$-\Delta$	0	1	2	3	4	Δ
Random	4.61	2.99	2.47	2.16	1.93	2.22	0.92	1.04	1.20	1.35	1.49	0.35
Lead		2.61	2.06	1.74	1.52	2.63		1.27	1.46	1.63	1.83	0.63
Occlusion		1.97	1.42	1.20	1.09	3.19		1.86	2.17	2.34	2.44	1.28
Attention		1.93	1.48	1.32	1.23	3.12		1.59	1.96	2.23	2.45	1.14
InpGrad		1.86	1.41	1.25	1.18	3.19		1.68	2.03	2.28	2.48	1.20
IntGrad		1.68	1.33	1.22	1.17	**3.26**		1.74	2.13	2.40	2.60	**1.30**

Table 11: Sentence-level evaluation for content attribution methods. The reported value is the NLL loss w.r.t. the predicted token. $n = 0$ means the baseline when there is no sentence displayed in DISP or removed in RM.

Method	Regular	Two Stage S+ Base: $\mathcal{O}(s \times d^2)$
Occlusion	$\mathcal{O}(n^2 \times n)$	$+\mathcal{O}(d^2 \times d)$
Attention	$\mathcal{O}(n^2)$	$+\mathcal{O}(d^2 \times d)$
IntGrad	$\mathcal{O}(n^2 \times r + r \times bp)$	$+\mathcal{O}(d^2 \times r + r \times bp)$
InpGrad	$\mathcal{O}(n^2 + bp)$	$+\mathcal{O}(d^2 + bp)$

Table 12: Comparison of complexity of regular methods and their two-stage variants. The time complexity of back propagation bp is hard to define so we just leave it for simplicity.

propagation for gradient based methods. We list the complexity of the original methods in the middle column and the sentence based pre-selection variant in the right column. The base cost for sentence pre-selection model is to run the sentence selection model s times, so it's $\mathcal{O}(s \times d^2)$. The n^2 and d^2 originate from the quadratic operation of self-attentions in Transformer models. We ignore the number of layers in the neural network or other model related hyper-parameters since all of the methods here share the same model.

E Four Way Evaluation

Due to the space limit, we only show the plot of the four way evaluation in Figure 3. To enable future comparisons on the proposed evaluation protocol, we also include the detailed results in Table 10 and Table 11 for TOK and SENT evaluation. The Δ measures how the average performance increase or drop deviates from the original baseline. We abstract the evaluation methods as a function $eval$. The input is the text and the budget n and output is the predicted loss.

$$\Delta = \text{Avg}(eval(i)) - eval(0)$$

For TOK series evaluation, $i \in \{1, 2, 4, 8, 16\}$. For SENT series evaluation, $i \in \{1, 2, 3, 4\}$ because a sentence carries much more information than a token. IntGrad performs the best across all of the evaluation methods.