Model Criticism for Long-Form Text Generation

Yuntian Deng[1], **Volodymyr Kuleshov**[2], **Alexander M. Rush**[2]

[1] Harvard University dengyuntian@seas.harvard.edu

[2] Cornell University {kuleshov,arush}@cornell.edu

Abstract

Language models have demonstrated the ability to generate highly fluent text; however, it remains unclear whether their output retains coherent high-level structure (e.g., story progression). Here, we propose to apply a statistical tool, *model criticism in latent space*, to evaluate the high-level structure of the generated text. Model criticism compares the distributions between real and generated data in a latent space obtained according to an assumptive generative process. Different generative processes identify specific failure modes of the underlying model. We perform experiments on three representative aspects of high-level discourse—coherence, coreference, and topicality—and find that transformer-based language models are able to capture topical structures but have a harder time maintaining structural coherence or modeling coreference.

1 Introduction

It is now broadly accepted that neural language models can consistently generate fluent text (Radford et al., 2019; Shoeybi et al., 2019; Brown et al., 2020; Smith et al., 2022). Yet, while large language models make few local word-level errors, human studies have shown that they still often make "high-level" errors such as incoherence, self-contradictions, and off-topic generations (Dou et al., 2022). We hypothesize that researchers have focused on local fluency partly because it is easy to automatically evaluate through metrics such as perplexity and n-gram matching. Automatic assessment of high-level text generation quality has received less attention, partially because a single general-purpose metric does not exist.

This work takes a step toward the automatic evaluation of the high-level structure of the generated text by applying a tool from statistics, *model criticism in latent space* (Dey et al., 1998; Seth et al., 2019). Under this approach, we first project data to a latent space based on an assumptive generative process, and then compare the implied latent distributions between real data and language model samples. This approach unifies past work for evaluating text generation under a single framework, including existing dimensionality reduction techniques such as probabilistic PCA (Wold et al., 1987), as well as previous applications of model criticism that were restricted to topic models (Mimno and Blei, 2011).

By making different assumptions in the underlying generative process, model criticism in latent space identifies specific failure modes of the generated language. We demonstrate this on three representative high-level properties of the generated discourse—coherence (Barzilay and Lapata, 2005), coreference (Chomsky, 1993), and topicality (Blei and Lafferty, 2006)—as well as on a synthetic dataset for which the true data generating process is known.

Experiments using our proposed framework enable us to make four observations about modern language models. First, we find that it is possible for a model to get strong word-level perplexity, yet fail to capture longer-term dynamics. Second, we find that the transformer language models perform poorly in terms of coherence, in line with previous observations (Dou et al., 2022; Sun et al., 2021; Krishna et al., 2022; Sun et al., 2022), particularly when they do not have access to explicit lexical markers in the context. Third, we show that transformer language models do not model coreference structures well. Last, we show that transformer language models can capture topical correlations (Blei and Lafferty, 2006). All results, data, and code are publicly available at https://github.com/da03/criticize_text_generation.

2 Model Criticism in Latent Space

Model criticism (O Hagan, 2003) quantifies the relationship between a data distribution $P_{\text{data}}(x)$ and a model $P_{\text{model}}(x)$ by comparing statistics over

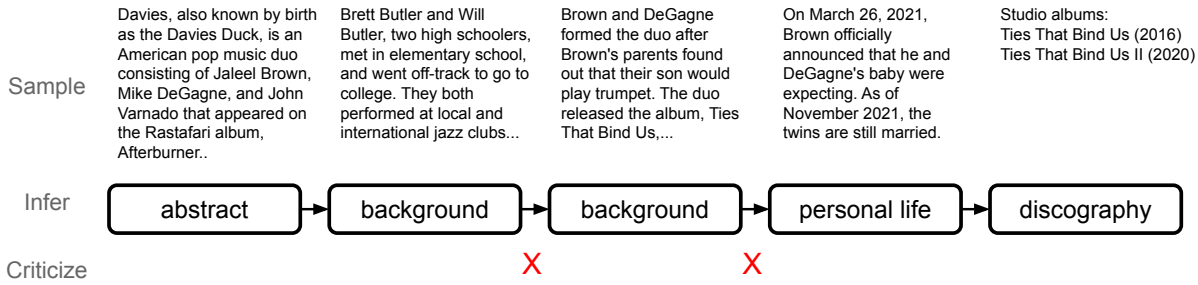

Figure 1: Illustration of applying model criticism in latent space to evaluate discourse coherence. Instead of word-level errors, we identify improper high-level section transitions (those that are rare in real data), as marked by red crosses. The article shown is generated by GPT-2 finetuned on WIKI. See Section 4 for more explanations.

these two distributions. While model criticism can be applied to the observation space, in many applications we are interested in "higher-level" aspects of the data, such as the underlying topics of a document (Mimno and Blei, 2011), or the latent factors of an image (Seth et al., 2019). Model criticism in latent space (Dey et al., 1998; Seth et al., 2019) lifts the criticism approach to a latent space in order to compute higher-level comparative statistics.

How do we critique latent properties of arbitrary, and perhaps unknown, distributions? For example, given a language model, how do we know how well it captures the section transitions at the discourse level (Figure 1)? Lacking access to the generative process, we introduce a critic generative process P_c with latent variables $z \in \mathcal{Z}$ and observations $x \in \mathcal{X}$:

$$z \sim P_c(z) \qquad x \sim P_c(x|z).$$

Based on this generative process, the posterior distribution $P_c(z|x)$ projects x to the latent space. For a single data point x, we can evaluate the negative log-likelihood of the projected latent variables $z \sim P_c(z|x)$ under the prior P_c,

$$\begin{aligned} T_c(x) &\triangleq -\mathbb{E}_{z\sim P_c(z|x)} \log P_c(z) \\ &= H(P_c(z|x), P_c(z)), \end{aligned}$$

where $H(p, q) \triangleq -\mathbb{E}_p \log q$ denotes the cross-entropy between two distributions p and q.[1] This process is illustrated in Figure 2.

Given an arbitrary distribution over x, P_x, we can take an expected negative log-likelihood,

$$T_c(P_x) \triangleq -\mathbb{E}_{x\sim P_x(x)}\mathbb{E}_{z\sim P_c(z|x)} \log P_c(z).$$

We term $T_c(P_x)$ the *Latent NLL*.[2] This value is the cross-entropy between the aggregated posterior distribution and the prior distribution of z:

$$T_c(P_x) = H(\mathbb{E}_{x\sim P_x(x)} P_c(z|x), P_c(z)).$$

In practice, we cannot compute $T(P_x)$ analytically due to the existence of the two expectations $\mathbb{E}_{x\sim P_x(x)}$ and $\mathbb{E}_{z\sim P_c(z|x)}$, but can approximate expectations using Monte-Carlo sampling.

When z is a sequence of M discrete states, we define a metric *Latent PPL* analogous to perplexity:

$$\text{Latent PPL}(P_x) \triangleq \exp[T_c(P_x)/M].$$

With a critic chosen, we can compare $P_{\text{data}}(x)$ and $P_{\text{model}}(x)$ in the latent space by estimating and comparing $T_c(P_{\text{data}})$ and $T_c(P_{\text{model}})$. Similar to a two-sample test (Hotelling, 1951), when P_{data} and P_{model} are the same, the statistics will also stay close. Furthermore, with a powerful critic, $T_c(P_{\text{model}})$ is meaningful by itself: a higher value means that model generations are less likely in the latent space, whereas a lower value implies that samples match the critic along the latent projection.[3] The approach can also be applied to individual points, $T_c(x)$, to identify outliers.

How to select the critic P_c Choosing the critic P_c is obvious only when we know the true latent variables and the generative process of data. In other cases, it depends on the data properties of interest. For example, if we want to criticize the topicality of text, we can use a topic model (Blei et al., 2003) to induce a latent space over topics. Note that the selected critic P_c may underperform P_{model} as a model of x, while still providing useful latent structures. By criticizing strong models, using simpler latent models that are designed to capture a particular aspect of text as P_c, we provide a sanity check for the stronger model along a specific target axis. This property motivates the use of this approach with powerful, yet opaque models.

[1] We discuss the difference between being likely in the latent space versus the observed space in Appendix A.

[2] When z is the same as x ($P_c(z|x) = \mathbb{1}[z = x]$), Latent NLL is the same as the negative log-likelihood of the language model samples under the data distribution (Zhao et al., 2018).

[3] Under a bad critic, the value of $T_c(x)$ might not be meaningful even though we can still use it to compare distributions.

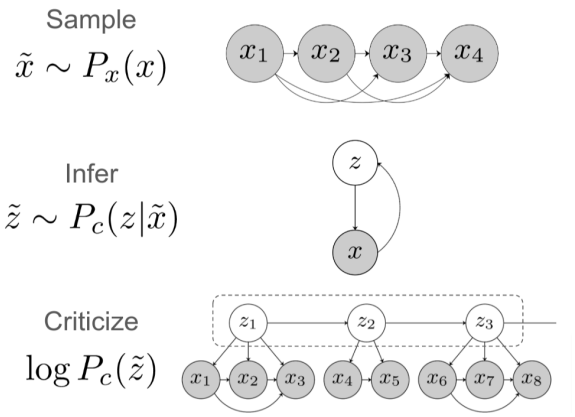

Figure 2: Model criticism in latent space. Given a sample $\tilde{x}$, we first map it to latent states $\tilde{z}$ using $P_c(z|\tilde{x})$. The likelihood of $\tilde{z}$ is evaluated using $P_c(\tilde{z})$ to measure how likely the samples are in the latent space.

Sample x: ... p B W m <s> T c g N f <s>
x i t K a b <s> b A x t N o m U <s> ...
Infer Latent z: ... G B R Y ...
Criticize (Latent NLL): ... 5.1 + 9.9 + 10.3 + 4.2 ...

Figure 3: Applying model criticism to synthetic data.

	Trans-LM	HSMM-LM
Word-level PPL	2.28	2.05
Latent PPL (data)	44.30	
Latent PPL (model)	64.80	47.24

Table 1: Evaluation results of transformer and HSMM on the synthetic dataset. Word-level PPL values are estimated on the test set, and Latent PPL values are estimated using the same number of samples (6.4k).

3 A Surprising Text Generation Failure

As a preliminary experiment, we show a language model with strong word-level perplexity that fails to capture simple long-term dynamics as demonstrated by model criticism. We assume that P_{data} is known and follows a basic pattern. It has a latent high-level sequence of $M = 50$ discrete states $z_1, z_2, \ldots, z_M$ where each state can take one of 256 possible values. These states are generated from a transition distribution

$$P_{\text{data}}(z_1, \ldots, z_M) = \prod_{m=1}^{M} P_{\text{data}}(z_m|z_{m-1}).^{4}$$

At the observation level, each latent state z_m generates a sub-sequence of words $x_1^m, x_2^m, \ldots, x_{N_m}^m$ conditioned on z_m from an emission distribution $P(x_1^m, \ldots, x_{N_m}^m|z_m)$. We also restrict the model so that each sub-sequence can only come from one latent state. The observed sequence is the concatenation of all sub-sequences. The joint distribution of the latent states and the tokens forms:

$$P_{\text{data}}(x, z) = P_{\text{data}}(z)P_{\text{data}}(x|z)$$
$$= \prod_{m=1}^{M} \left[P_{\text{data}}(z_m|z_{m-1})P_{\text{data}}(x_1^m, \ldots, x_{N_m}^m|z_m)\right].$$

With this generative process, we sample a dataset.[5]

We apply a transformer language model as P_{model} and train it on this dataset. Given the simplicity of the generative process and the small vocabulary size, we expect this model to do quite well. And in fact we do see that the model achieves a strong perplexity of 2.28, which nearly matches the true data P_{data} perplexity of 1.99.

Model criticism gives a different method for quantifying model fit. Since the true data generating process is known, we can directly use $P_c = P_{\text{data}}$ as the critic to induce the latent space. To project an observation x to the latent space, we need to perform posterior inference $P_c(z|x)$. By construction, this mapping is deterministic, since each sub-sequence comes from a unique latent state (see Appendix D for details). We then apply model criticism T_c by sampling a sequence of transformer outputs, mapping them to a sequence of latent states, counting to compute the aggregated posterior, and then comparing to the known prior. This process is shown in Figure 3.

Table 1 presents the results. Surprisingly, transformer gets a much worse Latent PPL compared to a hidden semi-Markov model (HSMM, the true model class) fit to data (66.80 v.s. 47.24), which has a near-optimal Latent PPL. This result implies that even though the transformer is nearly as good at predicting the next word in the sequence, it has not learned the higher-level transition structures. Seemingly, it can produce reasonable estimates of the next token which does not reflect the ability to capture longer-range dynamics of this system.

Motivation Given this result, we ask whether similar issues are present in language models applied in more realistic scenarios. We therefore turn to experiments that consider model criticism for long-form generation, and ask whether language models capture properties of discourse coherence (Section 4), coreference (Section 5), and topicality (Section 6).

[4] We assume a special beginning state z_0.

[5] Sub-sequences vary between 4 to 11 words and the vocabulary size is set to 53. There are 51.2k samples for training, 6.4k for validation, and 6.4k for evaluation.

Metric	Model	PUBMED		ARXIV		WIKI	
		W/ Title	W/O Title	W/ Title	W/O Title	W/ Title	W/O Title
PPL	LM_1	11.38	11.50	13.94	14.13	15.38	15.84
	LM_2	10.96	11.09	12.73	12.85	16.35	16.86
Latent PPL	Data	2.58		3.87		4.80	
	LM_1	2.68	3.76	6.72	9.52	4.67	5.47
	LM_2	4.17	7.92	9.01	18.64	6.48	10.22

Table 2: Results of coherence experiments. W/ Title is the setting where section titles are included in the training data for LMs, and W/O Title removes section titles from the training data.

4 Critiquing Discourse Coherence

Text generation from large language models rarely leads to local fluency errors, but there is evidence of failures like those in the previous section (Dou et al., 2022; Sun et al., 2021; Krishna et al., 2022; Sun et al., 2022). In this section, we apply model criticism to assess discourse coherence (Barzilay and Lapata, 2005) of large LMs. We study this through an experiment on generating long-form documents divided into explicit sections. While we do not know the true data generating process, knowing the distribution of section types allows us to assess the latent structure of LM generations.

Figure 1 illustrates the experiment. Here, an LM generates an article. Each word transition is fluent, but the system makes two section transition errors: first, it generates two sections of type "background"; second, it generates a section of type "personal life" following the last "background" section, with both transitions being unlikely in the data.[6] We aim to separate the evaluation of these high-level coherence errors from word-level errors.

To apply model criticism, we posit a simple critic generative process to capture the section changes. We adapt a hidden semi-Markov model (HSMM) which is commonly used to represent segmentations of this form. Specifically, the high-level latent variables $z_1, \ldots, z_M$[7] model transitions among section types and the bottom level generates text conditioned on the current section type:

$$P_c(x, z) = P_c(z)P_c(x|z)$$
$$= \prod_{m=1}^{M} \left[P_c(z_m|z_{m-1})P_c(x_1^m, \ldots, x_{N_m}^m|z_m)\right].$$

We can then evaluate on datasets with known (ground truth) section titles and use these section titles as z. We use three English datasets PUBMED, ARXIV, and WIKI (Cohan et al., 2018).[8] We compare two language modeling settings, one trained with all section titles removed ("W/O Title") and one with section titles before each section ("W/ Title"), since we hypothesize that the existence of explicit section type markers might help the model learn the dynamics, inspired by Nye et al. (2021) and Wei et al. (2022b). Sections are separated by a special marker, and a special end-of-sequence symbol is used to mark the end of the generation. Since all three datasets are relatively small (especially considering that we use them to generate entire articles), we leverage pretrained language models GPT-2 small (LM_1) (Radford et al., 2019), and GPT-Neo small (LM_2) (Black et al., 2021) which is trained on a more diverse dataset (Gao et al., 2020). We finetune these LMs for P_{model}.

To generate, we sample from the language model until we hit the end-of-sequence symbol. No tempering/truncation (Holtzman et al., 2019) is used during sampling, since we are more interested in the learned distribution rather than its mode here. For the "W/ Title" setting, we discard the generated section titles in a postprocessing step.

To infer the section types for a generated article, we need to approximate posterior inference to compute T_c. We make a simplifying assumption that the posterior section title of each section only depends on its corresponding text: $P_c(z|x) \approx \prod_{m=1}^{M} P_c(z_m|x_:^m)$. We then finetune BERT with a classification head to estimate $P_c(z_m|x_:^m)$. At inference time we use the MAP estimate of z instead of maintaining the full distribution $P(z|x)$ (BERT

[6] "background" is usually followed by "reception".

[7] We prepend a special beginning state z_0 and append a special ending state z_{M+1} that do not emit anything.

[8] We adapt PUBMED and ARXIV by filtering out section titles with low frequency. We download and process Wikipedia to get a dataset of the same format as Cohan et al. (2018).

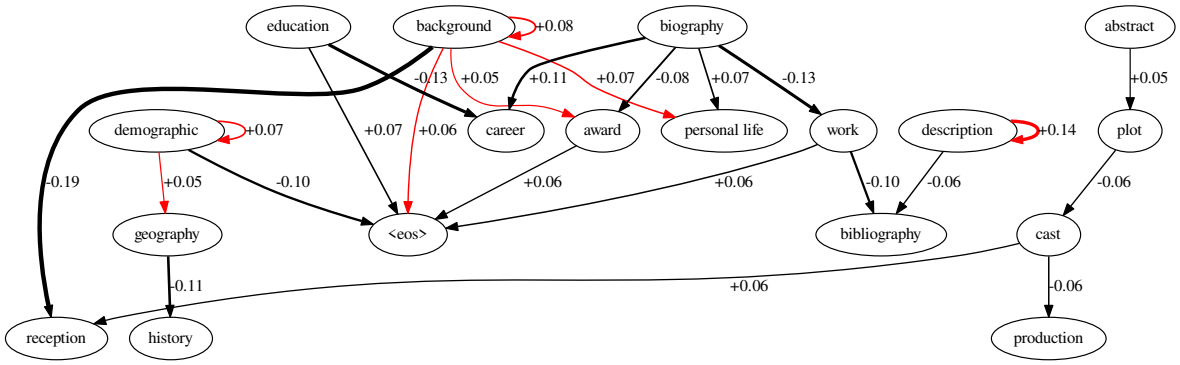

Figure 4: Section transition errors on WIKI, where each edge is labeled with the difference between $P(z_m|z_{m-1})$ of LM_1 (W/O Title) and of data, and its width is proportional to the absolute difference. Red marks unlikely transitions ($P_c(z_m|z_{m-1}) < 0.05$). For clarity, we only show the top 20 section titles and remove singletons.

Metric	Model	W/ Title	W/O Title
Latent PPL	Data	4.78	
	LM_1	5.18	6.24
	LM_2	6.54	10.72

Table 3: Latent PPLs on WIKI-SHORT where all section transitions fit within the context window size.

Metric	LM_2 W/O	LM_2 W/	W/O	W/	Data
PPL↓	11.09	10.96	11.50	11.38	-
MAUVE↑	0.75	0.85	0.91	0.90	0.96
Latent PPL↓	7.92	4.17	3.76	2.68	2.58
Human↑	0.50	0.66	0.71	0.88	0.87

Table 4: Experiments on the correlation of Latent PPL (coherence) with human judgment and automatic metrics (PUBMED). Latent PPLs agree well with human judgments of coherence. W/O: LM_1 W/O Title. W/: LM_1 W/ Title. See Appendix H for evaluation details.

is mostly over 90% certain about its predictions). More details can be found in Appendix E.

Results Table 2 gives results on coherence experiments. We first note that both models have strong word-level perplexity across datasets, with LM_2 doing better on two of the three datasets. We also note that removing titles has a negligible impact on the perplexity of the models. However, Latent PPL tells a different story. We find that LM_1 greatly outperforms LM_2 when criticizing with respect to the latent sections.[9] It is also interesting that transformer LMs are sensitive to title words being explicitly included in the training data (i.e., the W/ Title setting). For example, LM_1 W/ Title gets a Latent PPL of 6.72 on ARXIV, whereas LM_1 W/O Title gets a Latent PPL of 9.52, despite having very close word-level PPLs (13.94 v.s. 14.13). These observations indicate that lacking explicit markers, the tested transformer LMs do not learn the long-term dynamics necessary for discourse coherence. Using explicit section topic markers might serve a similar functionality as using chain-of-thought prompting in language-model-based question answering tasks (Wei et al., 2022b).

[9]For one dataset, LM_1 has a lower Latent PPL than the data distribution. This result is possible as a consequence of our cross-entropy formulation of T_c, under which a mode-seeking distribution can get a lower value than P_{data}, which is the approximate entropy of the latent prior.

One concern is that the difference between W/ Title and W/O Title is a side effect of language models having a limited context window size (1024 for LM_1 and 2048 for LM_2), since two adjacent sections might not fit within the context window size (but one section and the next section title are more likely to fit). To check if this is the case, we filter WIKI to only include articles with maximum section length 500 to form a new dataset WIKI-SHORT. In this dataset, any two adjacent sections can fit within the context window of both LM_1 and LM_2. Table 3 shows that even in this case W/ Title still outperforms W/O Title, indicating that the difference between W/ Title and W/O Title is not due to the limited context window size.

Figure 4 visualizes the section transition errors made by LM_1 (W/O Title) for the most common section types on WIKI. We can find that the language model tends to generate the same section topic repeatedly, although there are other transition errors as well. More detailed error analysis can be found in Appendix E.

Table 4 correlates automatic metrics with human judgments of coherence. Each human annotator first labels the section title of each section (after a training phase where they labeled and received

Metric	Model	W/ Title	W/O Title
PPL	GPT-2 S	15.38	15.84
	GPT-2 M	12.98	13.32
	GPT-2 L	12.19	12.52
	GPT-2 XL	11.60	11.99
Latent PPL	Data	4.80	
	GPT-2 S	4.67	5.47
	GPT-2 M	4.79	5.58
	GPT-2 L	4.90	5.75
	GPT-2 XL	4.75	5.56

Table 5: The results of scaling model size on WIKI. Increasing model size improves PPL but not Latent PPL.

feedback on real data), and then labels whether the organization of the section titles makes sense (Persing et al., 2010). The baseline MAUVE (Pillutla et al., 2021) is a metric that compares the distribution of GPT-3 hidden states between real data and model generations. From this table, we can observe that both MAUVE and Latent PPL align much better with humans than PPLs. Comparing MAUVE and Latent PPL, we can see that Latent PPL aligns better with humans: LM_1 W/O Title is considered to be better than LM_1 W/ Title under MAUVE, but both human evaluation and Latent PPL consider LM_1 W/ Title to be much better.

A natural question is whether increasing model size improves coherence. To this end, in addition to GPT-2 small (GPT-2 S, aka LM_1, 117M parameters), we apply model criticism to GPT-2 medium (GPT-2 M, 345M parameters), GPT-2 large (GPT-2 L, 742M parameters), and full GPT-2 (GPT-2 XL, 1.5B parameters) on WIKI. The results are summarized in Table 5. We can see that increasing model size improves PPL but not Latent PPL.

5 Critiquing Coreference Chains

Coreference tracks how multiple *mention* expressions are used to refer to the same underlying *entity* (Karttunen, 1969; Gordon and Hendrick, 1998). While coreference represents a ubiquitous and important discourse-level phenomenon (Jurafsky and Martin, 1999; Kunz and Hardmeier, 2019), there is evidence that large neural language models make elementary coreference mistakes (Pagnoni et al., 2021), such as referring to non-existent discourse entities (Schuster and Linzen, 2022).

In this experiment, we compare the coreference chain (Jurafsky and Martin, 1999) distributions between real data and LM generations. A coreference chain consists of a sequence of coreferent mentions. To simplify the representation, we use gender features to replace non-pronominal tokens, as illustrated in Figure 5.[10] Presumably these latent chains should be similar in generated text and in real data.

Original Text

... [Lisa]$_0$ runs off to find [him]$_1$ and [they]$_2$ kiss passionately. Afterwards, [Josh]$_1$ tells [her]$_0$ the reason why [he]$_1$'s going to [their]$_2$ first gig, and that [Lisa]$_0$ is going to do it, too...

Coreference Chains z

. [Female]$_0$ [him]$_1$ [they]$_2$. [Male]$_1$ [her]$_0$ [he]$_1$ [their]$_2$ [Female]$_0$

5-gram critic P_c

P_c([Female]$_0$ | previous entity mentions)
$\approx P_c$([Female]$_0$ | [Male]$_1$ [her]$_0$ [he]$_1$ [their]$_2$)

Figure 5: Critiquing coreference chains on a sample from LM_1. We first extract entity mentions from the text and only keep the genders of proper nouns to form **z**, then a 5-gram P_c is used to score **z**. $[]_i$ denotes a mention with entity id i. **.** marks sentence boundaries.

For the critic P_c, we use a 5-gram language model with Kneser–Ney smoothing (Ney et al., 1994) over chains. To infer **z**, we use an off-the-shelf coreference resolution tool.[11] To avoid data sparsity issues, we relabel entity clusters within each n-gram. We apply model criticism to compare real data and LMs trained on WIKI (W/ Title), after filtering data to only consider articles about films since they contain richer reference structures.

Results Table 7 shows the Latent PPLs on real data and LM generations. We can see that in general there is a mismatch of coreference distributions. Interestingly, while LM_1 models outperformed LM_2 models on discourse coherence, for this task LM_2 models are better.

Table 6 shows the 10 coreference chain n-grams that most contributed to this difference. Some are intuitively implausible: in the fourth row, [His]$_1$ does not have a local antecedent; in the second to the last row, [himself]$_2$ also does not have a local

[10] We discuss the ethical considerations of using gender as features in Section 10.

[11] https://github.com/huggingface/neuralcoref

z_1	z_2	z_3	z_4	z_5	$\hat{P}_{\text{data}}$	$\hat{P}_{\text{LM}}$
			.	M_0	0.10	0.12
.	M_0	.	M_0	M_0	0.01	0.03
.	.	M_0	his_0	he_0	0.04	0.05
.	.	.	N_0	His_1	0.00	$0.00^\dagger$
M_0	M_1	.	M_0	M_0	0.00	0.01
N_0	N_1	M_2	M_3	We_4	0.00	$0.00^\dagger$
.	F_0	her_0	.	F_0	0.03	0.04
M_0	his_0	.	M_1	M_1	$0.00^\dagger$	0.01
.	.	N_0	N_1	h.self_2	0.00	$0.00^\dagger$
.	.	M_0	his_0	M_0	0.01	0.01

Table 6: Coreference chain n-grams ranked by contribution to the difference in Latent NLL (LM_1). $\hat{P}$ denotes the empirical frequency of an n-gram in percentages. M (Male), F (Female), and N (None of the above). Blank: padding. †: small positive numbers truncated to 0.

Data	LM_1	LM_2
6.26	7.22	6.93

Table 7: Latent PPLs of coreference chains.

Model	Latent PPL
Data	6.26
GPT-2 S	7.22
GPT-2 M	7.64
GPT-2 L	7.27
GPT-2 XL	7.62

Table 8: Critiquing coreference chains on larger models. Increasing model size does not improve Latent PPL.

antecedent. Others are rare but possible: in the last row, a proper noun $[\text{Male}]_0$ is used after a pronoun $[\text{his}]_0$ is used in the same sentence to refer to the same entity.[12]

The learned critic P_c can also be used to identify unlikely coreference chains, as shown in Table 15 in Appendix G. Appendix G also has more qualitative examples and analyses.

Lastly, we evaluate whether scaling model size improves coreference modeling. The results are summarized in Table 8. We can see that increasing model size does not improve Latent PPL, similar to our observations on critiquing discourse coherence.

[12]Different from prescriptive linguistic theories on coreference (Chomsky, 1993; Büring, 2005), the differences identified by model criticism only reflect differences in empirical distributions and do not necessarily mean coreference errors.

6 Critiquing Topic Correlations

Topical structure is another important aspect of long-form document generation (Serrano et al., 2009). Certain topics are more likely to appear together, for example, a document containing a topic related to "poets" is more likely to also contain one related to "publisher" relative to one related to "football". A text generation model should capture these topical relations. For this experiment, we again sample documents from the trained language model P_{model}. Specifically, we utilize the transformer-based LMs trained on the datasets in Section 4 (W/O Title).

To explore the topical structure in the generated documents, we need a critic P_c. While LDA (Blei et al., 2003) is the most commonly used generative process for topic modeling, the Dirichlet prior does not explicitly model topic correlations in documents. We therefore use the correlated topic model (CTM) specifically designed to model topical structures (Blei and Lafferty, 2006). Model criticism will then compare the latent space of the real data with the generated texts.

For each document, a CTM with M topics first generates a topic coefficient latent variable $z \in \mathbb{R}^M$ from a multivariate Gaussian distribution $P_c(z) \triangleq \mathcal{N}(z; \mu, \Sigma)$.

Each coefficient of z can be interpreted as the "strength" of a topic in a document, so the covariance matrix Σ captures the correlations among different topics. These weights z are then normalized using a softmax function, the result of which is used to parameterize the distribution over topic t_n for the n-th word. Each topic t_n induces a categorical distribution over word types $P(x_n|t_n) = \phi_{t_n,x_n}$, where ϕ_{ij} parameterizes the probability of emitting word type j conditioned on topic i. The joint probability of a document with N words is:

$$P_c(x, t|z; \phi) = \prod_{n=1}^{N} [\text{softmax}(z)]_{t_n} \phi_{t_n,x_n}.$$

Since we are only interested in criticizing the document-level z, we marginalize out the topic assignments of individual words:

$$P_c(x|z) = \prod_{n=1}^{N} \sum_{i=1}^{M} [\text{softmax}(z)]_i \phi_{i,x_n}.$$

To fit this generative process on data, we use variational inference and maximize the ELBO following Blei and Lafferty (2006). We set M to 100.

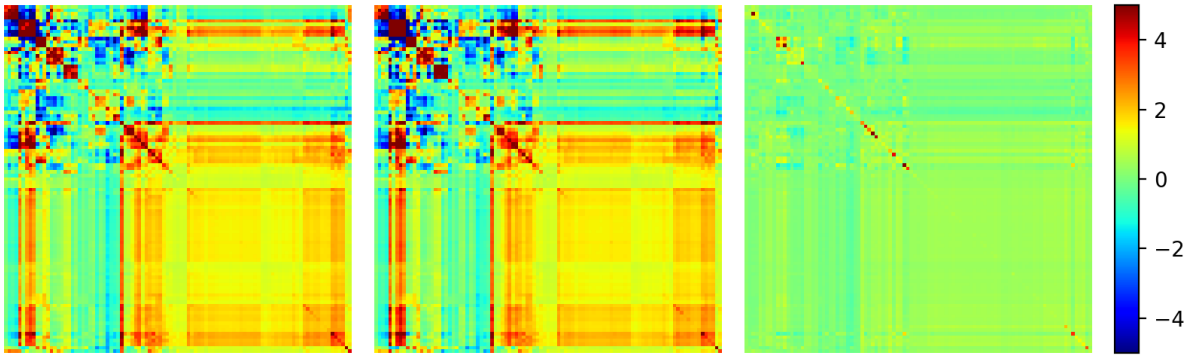

Figure 6: Topic covariance matrix for the induced z (on WIKI). Left: Test set (P_{data}). Middle: LM_1 generations (P_{model}). Right: generations of LM_1 trained on PUBMED as a visual baseline. The Latent NLLs are: 124.70, 123.30, and 140.10. Topic ids are rearranged using hierarchical clustering to facilitate visual comparison.

Metric	Model	PUBMED	ARXIV	WIKI
Latent NLL	Data	174.43	163.95	124.70
	LM_1	172.70	161.40	123.30
	LM_2	172.81	163.17	124.35

Table 9: Latent NLL of topic correlation modeling. Transformer LMs perform similarly to the real data.

Since analytical posterior inference is intractable, we use variational inference to estimate $P_c(z|x)$.

Results Table 9 shows the main results. The Latent NLLs of LM generations and real data are close on all three datasets (there are outlier pathological generations that we can identify using $T(x)$, as shown in Appendix F). In Figure 6, we visualize and compare the covariance matrices of the aggregated posterior distributions of LM generations and real data, and find that transformers are able to model the correlations among topics well. These results indicate that topic correlation is well represented in text generation systems, and is likely an easier task to model than ordered coherence.

7 Related Work

Text Generation Evaluation Traditional evaluation metrics include perplexity, and n-gram overlap metrics for translation-type problems such as BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Lavie and Agarwal, 2007), and NIST (Martin and Przybocki, 2000). In recent years, with the emergence of neural models that learn contextual representations (Devlin et al., 2019; Liu et al., 2019), researchers propose to project text to contextual representations and compute distance in this space (Zhang et al., 2019; Zhao et al., 2019; Pillutla et al., 2021). The closest work to ours is Eikema and Aziz (2020), which evaluates different decoding strategies in machine translation by comparing the statistics of the produced text. While these past works mainly concern word-level string/meaning representation matching, the goal of our work is to check the high-level aspects of the generated text such as coherence. Besides, word-level matching is not suitable for evaluating open-ended generation tasks due to the existence of too many plausible references (Celikyilmaz et al., 2020), while our work projects text to a more manageable lower-dimensional latent space to make the evaluation of open-ended generation feasible.

Evaluation of Long-Form Text There is a long line of research evaluating the discourse coherence of text (Grosz et al., 1995; Poesio et al., 2004; Barzilay and Lapata, 2005; Lai and Tetreault, 2018; Logeswaran et al., 2018; Persing et al., 2010). Most learn a predictor that maps features such as the distribution of entities (Barzilay and Lapata, 2005) or the transitions of topics (Persing et al., 2010) to manually-labeled coherence scores. Our work differs in two important ways: first, we unify the evaluation of different high-level aspects of text using the formalism of model criticism; second, we do not assume any annotated coherence scores—we only specify a generative process in order to project text to a latent space for the comparison between machine-generated text and real text. Recently, there have been works targeting the evaluation of discourse-level coherence, such as BARTScore (Yuan et al., 2021) and DiscoScore (Zhao et al., 2022). These methods presume either a conditional generation setting or require textual references. We also note that model criticism does not use a generic neural representation, but focuses

on specific user-specified high-level aspects of text. In this respect, our work is similar in spirit to some recently proposed suite-based metrics, such as Language Model Evaluation Harness (Gao et al., 2021) and BIG-bench (Srivastava et al., 2022) that utilize many different skill-based metrics.

High-Level Issues of Text Generation Models Concurrent with our work, several other groups also notice that existing LMs fail to capture some high-level aspects of text. For example, similar to our findings of LMs being not strong at discourse coherence, Sun et al. (2022) observe that large LMs including GPT-3 fail to assign a higher probability to ground truth chapter continuations compared to distractor chapters given a prefix, and yet a simple classifier trained on this identification objective can achieve a much higher accuracy. Similar to our findings of LMs being not strong at modeling coreference, Papalampidi et al. (2022) find that LMs fail to maintain long-range entity consistency and coherency in the generated narrative stories.

Model Criticism Model criticism, also known as model checking, is a general framework for checking if a generative model fits the data well (Box, 1980; Gelman et al., 1995; Stern and Sinharay, 2005; O Hagan, 2003). Model criticism is different from aforementioned metrics such as PPL and is similar to two-sample tests (Hotelling, 1951) in that it computes and compares some statistics on the real data and on the samples to determine if they are close enough. While the statistics may be directly computed in the observation space, in many applications we are interested in criticizing some latent aspects of data such as topics (Mimno and Blei, 2011) or latent factors (Seth et al., 2019). To this end, Dey et al. (1998) introduce model criticism in latent space, which measures the discrepancy between real data and model generations in the latent space induced by a generative model (Chaloner and Brant, 1988; O Hagan, 2003; Seth et al., 2019; Dey et al., 1995; Weiss, 1995; Dey et al., 1998). Recently, Barkhof and Aziz (2022) propose to use model criticism to evaluate VAEs. Model criticism in latent space forms the basis of our work, with two major differences: first, we apply model criticism to models with a point estimate of parameters such as commonly-used neural language models instead of models with uncertainties in their parameters. Second, we allow for using a different generative model to induce the latent space from the model that we criticize. By separating out the model to be criticized and the generative process used for projecting data to the latent space, our approach allows for criticizing different views of the data depending on user needs and for criticizing generative models without any latent variables such as neural language models. For qualitative analysis and outlier identification, our work applies visual posterior predictive checks (Gabry et al., 2019; Gelman, 1997), a graphical version of model criticism.

8 Limitations

One limitation of the proposed approach is its reliance on choosing a critic generative process P_c, which presumes some knowledge of a true data generating process. For an improperly specified critic, it does not expose the latent space that we intend to criticize. However, since we compare statistics between real data and model generations (similar to two-sample tests), for a good model the statistics should be close even with improper critics.

Another limitation is that not observing any differences does not imply that the model generations conform to the unknown data distribution—it simply means that they are close with regard to the latent aspects that we criticize (O Hagan, 2003).

Recently, researchers found that certain capabilities such as reasoning under augmented prompts only emerge in large LMs beyond tens of billions of parameters (Wei et al., 2022a). Since the largest LM tested in this paper only has 1.5 billion parameters, future work is required to investigate whether the high-level issues observed in this paper can be solved by further scaling model size.

9 Conclusions

We consider the problem of evaluating long-form text generation for specific discourse properties. We propose a statistical tool, model criticism in latent space, which projects text to a latent space based on an assumptive generative process, and compares the implied latent distribution. Different critic generative processes focus on different properties of data. We apply this tool to analyze three representative document properties: coherence, coreference, and topicality, using transformer-based language models. Experiments find that while transformer LMs can capture topical structures well, they are not currently strong at modeling discourse coherence without explicit markers or at modeling coreference.

10 Ethical Considerations

In our experiment of critiquing coreference chains, we used a gender binary (Hyde et al., 2019) to categorize proper nouns, but there are many individuals who do not adhere to the gender binary that this simple categorization fails to consider (Bamman et al., 2014). The reason for the gender binary is primarily because personal pronouns are typically gendered in English which makes the qualitative and statistical analysis more clear. For example, one coreference error detected by the approach is to use pronouns of different genders to refer to the same person, as shown in Appendix G. In Appendix G, we describe the exact procedure through which the genders of proper nouns are determined to make explicit what our "gender" definition is (Larson, 2017). Going forward, exploring other features of proper nouns such as their syntactic features (Shieber and Tao, 2003) to replace gender assignments here might further mitigate this concern.

Acknowledgements

YD is supported by an Nvidia Fellowship. AR is supported by NSF CAREER 2037519, NSF 1704834, and a Sloan Fellowship. We would also like to thank Harvard University FAS Research Computing for providing computational resources.

References

David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. 2014. Gender identity and lexical variation in social media. *Journal of Sociolinguistics*, 18(2):135–160.

Claartje Barkhof and Wilker Aziz. 2022. Statistical model criticism of variational auto-encoders. *arXiv preprint arXiv:2204.03030*.

Regina Barzilay and Mirella Lapata. 2005. Modeling local coherence: An entity-based approach. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 141–148, Ann Arbor, Michigan. Association for Computational Linguistics.

Steven Bird and Edward Loper. 2004. NLTK: The natural language toolkit. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

David Blei and John Lafferty. 2006. Correlated topic models. *Advances in neural information processing systems*, 18:147.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

George EP Box. 1980. Sampling and bayes' inference in scientific modelling and robustness. *Journal of the Royal Statistical Society: Series A (General)*, 143(4):383–404.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Daniel Büring. 2005. *Binding theory*. Cambridge University Press.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*.

Kathryn Chaloner and Rollin Brant. 1988. A bayesian approach to outlier detection and residual analysis. *Biometrika*, 75(4):651–659.

Noam Chomsky. 1993. *Lectures on government and binding: The Pisa lectures*. 9. Walter de Gruyter.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Kevin Crowston. 2012. Amazon mechanical turk: A research tool for organizations and information systems scholars. In *Shaping the future of ict research. methods and approaches*, pages 210–221. Springer.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dipak K Dey, Alan E Gelfand, Tim B Swartz, and Pantelis K Vlachos. 1995. Simulation based model checking for hierarchical models. *Technical Report*, pages 95–29.

Dipak K Dey, Alan E Gelfand, Tim B Swartz, and Pantelis K Vlachos. 1998. A simulation-intensive approach for checking hierarchical models. *Test*, 7(2):325–346.

Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A. Smith, and Yejin Choi. 2022. Is GPT-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7250–7274, Dublin, Ireland. Association for Computational Linguistics.

Bryan Eikema and Wilker Aziz. 2020. Is MAP decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. 2019. Visualization in bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 182(2):389–402.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Leo Gao, Jonathan Tow, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Kyle McDonell, Niklas Muennighoff, Jason Phang, Laria Reynolds, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2021. A framework for few-shot language model evaluation.

Andrew Gelman. 1997. Bayesian computation.

Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. 1995. *Bayesian data analysis*. Chapman and Hall/CRC.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Peter C Gordon and Randall Hendrick. 1998. The representation and processing of coreference in discourse. *Cognitive science*, 22(4):389–424.

Barbara J. Grosz, Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–225.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Harold Hotelling. 1951. A generalized t test and measure of multivariate dispersion. In *Proceedings of the second Berkeley symposium on mathematical statistics and probability*, pages 23–41. University of California Press.

Janet Shibley Hyde, Rebecca S Bigler, Daphna Joel, Charlotte Chucky Tate, and Sari M van Anders. 2019. The future of sex and gender in psychology: Five challenges to the gender binary. *American Psychologist*, 74(2):171.

Daniel Jurafsky and James H Martin. 1999. Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition.

Lauri Karttunen. 1969. Discourse referents. In *International Conference on Computational Linguistics COLING 1969: Preprint No. 70*, Sånga Säby, Sweden.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kalpesh Krishna, Yapei Chang, John Wieting, and Mohit Iyyer. 2022. Rankgen: Improving text generation with large ranking models. *arXiv preprint arXiv:2205.09726*.

Jenny Kunz and Christian Hardmeier. 2019. Entity decisions in neural language modelling: Approaches and problems. In *Proceedings of the Second Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 15–19, Minneapolis, USA. Association for Computational Linguistics.

Alice Lai and Joel Tetreault. 2018. Discourse coherence in the wild: A dataset, evaluation and methods. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 214–223, Melbourne, Australia. Association for Computational Linguistics.

Brian N Larson. 2017. Gender as a variable in natural-language processing: Ethical considerations. *EACL 2017*, page 1.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. 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. *arXiv preprint arXiv:1907.11692*.

Lajanugen Logeswaran, Honglak Lee, and Dragomir Radev. 2018. Sentence ordering and coherence modeling using recurrent neural networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Alvin Martin and Mark Przybocki. 2000. The nist 1999 speaker recognition evaluation—an overview. *Digital signal processing*, 10(1-3):1–18.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. *http://mallet. cs. umass. edu*.

David Mimno and David Blei. 2011. Bayesian checking for topic models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 227–237, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Hermann Ney, Ute Essen, and Reinhard Kneser. 1994. On structuring probabilistic dependences in stochastic language modelling. *Computer Speech & Language*, 8(1):1–38.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

Anthony O Hagan. 2003. Hsss model criticism. *Oxford Statistical Science Series*, pages 423–444.

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. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Artidoro Pagnoni, Vidhisha Balachandran, and Yulia Tsvetkov. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4812–4829, Online. Association for Computational Linguistics.

Pinelopi Papalampidi, Kris Cao, and Tomas Kocisky. 2022. Towards coherent and consistent use of entities in narrative generation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 17278–17294. PMLR.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Isaac Persing, Alan Davis, and Vincent Ng. 2010. Modeling organization in student essays. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 229–239, Cambridge, MA. Association for Computational Linguistics.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34.

Massimo Poesio, Rosemary Stevenson, Barbara Di Eugenio, and Janet Hitzeman. 2004. Centering: A parametric theory and its instantiations. *Computational Linguistics*, 30(3):309–363.

GV Puskorius and LA Feldkamp. 1994. Truncated backpropagation through time and kalman filter training for neurocontrol. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 4, pages 2488–2493. IEEE.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Alexander Rush. 2020. Torch-struct: Deep structured prediction library. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 335–342, Online. Association for Computational Linguistics.

Sebastian Schuster and Tal Linzen. 2022. When a sentence does not introduce a discourse entity, transformer-based models still sometimes refer to it. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 969–982, Seattle, United States. Association for Computational Linguistics.

M Ángeles Serrano, Alessandro Flammini, and Filippo Menczer. 2009. Modeling statistical properties of written text. *PloS one*, 4(4):e5372.

Sohan Seth, Iain Murray, and Christopher KI Williams. 2019. Model criticism in latent space. *Bayesian Analysis*, 14(3):703–725.

Stuart M. Shieber and Xiaopeng Tao. 2003. Comma restoration using constituency information. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 221–227.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.

Hal S Stern and Sandip Sinharay. 2005. Bayesian model checking and model diagnostics. *Handbook of Statistics*, 25:171–192.

Andreas Stolcke. 2002. Srilm-an extensible language modeling toolkit. In *Seventh international conference on spoken language processing*.

Simeng Sun, Kalpesh Krishna, Andrew Mattarella-Micke, and Mohit Iyyer. 2021. Do long-range language models actually use long-range context? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 807–822, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Simeng Sun, Katherine Thai, and Mohit Iyyer. 2022. ChapterBreak: A challenge dataset for long-range language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3704–3714, Seattle, United States. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022b. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*.

Robert E Weiss. 1995. Residuals and outliers in repeated measures random effects models. In *Expected Total*. Citeseer.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*.

Svante Wold, Kim Esbensen, and Paul Geladi. 1987. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Junbo Zhao, Yoon Kim, Kelly Zhang, Alexander Rush, and Yann LeCun. 2018. Adversarially regularized autoencoders. In *International conference on machine learning*, pages 5902–5911. PMLR.

Wei Zhao, Maxime Peyrard, Fei Liu, Yang Gao, Christian M. Meyer, and Steffen Eger. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. 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 563–578, Hong Kong, China. Association for Computational Linguistics.

Wei Zhao, Michael Strube, and Steffen Eger. 2022. Discoscore: Evaluating text generation with bert and discourse coherence. *arXiv preprint arXiv:2201.11176*.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2019. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.

Appendix

A Interpretation of Latent NLL

In Section 3 we termed $T(x)$ the Latent NLL, and a lower $T(x)$ indicates being more likely in the latent space. What does it mean to be "more likely in the latent space"? How is it reflected in the marginal likelihood $P(x)$? In this section we answer this question by decomposing the log marginal likelihood $\log P(x)$ into three components (for brevity we use P instead of P_c in this section):[13]

$$
\begin{aligned}
&\log P(x) \\
&= \frac{\log P(x,z)}{\log P(z|x)} \ (\forall z) \text{ (Bayes' theorem)} \\
&= \mathop{\mathbb{E}}_{z\sim P(z|x)} \frac{\log P(x,z)}{\log P(z|x)} \\
&= \mathop{\mathbb{E}}_{z\sim P(z|x)} \frac{\log P(x|z) + \log P(z)}{\log P(z|x)} \\
&= \underbrace{\mathop{\mathbb{E}}_{P(z|x)} \log P(z)}_{(1)} + \underbrace{\mathop{\mathbb{E}}_{P(z|x)} \log P(x|z)}_{(2)} + \underbrace{H(P(z|x))}_{(3)}
\end{aligned}
$$

The first term (1) can be interpreted as how likely the posterior latent variable of x is under the prior distribution $P(z)$ in expectation, and it is the negative Latent NLL ($-T(x)$) according to the definition of T. The second term (2) can be understood as how likely it is to realize the observation x given the posterior latent codes z. The third term measures the diversity of the posterior distribution, which is not reflected in our evaluation metric. In fact, if we combine (1) and (3) we would get $-\mathrm{KL}(P(z|x)||P(z))$:

$$
\log P(x) = \mathop{\mathbb{E}}_{P(z|x)} \log P(x|z) - \mathrm{KL}(P(z|x)||P(z))
$$

Therefore, the proposed evaluation metric $T(x)$ (hence $T(P_x)$) can be complemented using a diversity measure for completeness, which we leave for future work.

B The Optimal Critic Prior $P_c(z)$

For the quantity $T(x)$ to be meaningful, $P_c(z)$ should not be an uninformative prior. For example, if $P_c(z)$ is uniform, then $T(x)$ hence $T(P_x)$ would be a constant. We will show below that the optimal $P_c(z)$ that maximizes the data likelihood is exactly the aggregated posterior distribution under the data distribution ($P_{\text{agg}}(z) \triangleq \mathbb{E}_{x\sim P_{\text{data}}(x)} P_c(z|x)$).

To find the optimal $P_c(z)$ that maximizes the data likelihood, we use the equation from Appendix A that $\log P_c(x) = \mathbb{E}_{P_c(z|x)} \log P_c(z) + \mathbb{E}_{P_c(z|x)} \log P_c(x|z) + H(P_c(z|x))$, and take the expectation on both sides w.r.t. $P_{\text{data}}(x)$:

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{x\sim P_{\text{data}}} \log P_c(x) \\
&= \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} \mathop{\mathbb{E}}_{P_c(z|x)} \log P_c(z) \\
&\quad + \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} \mathop{\mathbb{E}}_{P_c(z|x)} \log P_c(x|z) \\
&\quad + \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} H(P_c(z|x)) \\
&= \mathop{\mathbb{E}}_{z\sim P_{\text{agg}}} \log P_c(z) + \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} \mathop{\mathbb{E}}_{P_c(z|x)} \log P_c(x|z) \\
&\quad + \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} H(P_c(z|x)) \\
&= -\mathrm{KL}(P_{\text{agg}}(z)||P_c(z)) + H(P_{\text{agg}}(z)) \\
&\quad + \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} \mathop{\mathbb{E}}_{P_c(z|x)} \log P_c(x|z) \\
&\quad + \mathop{\mathbb{E}}_{x\sim P_{\text{data}}} H(P_c(z|x)).
\end{aligned}
$$

In the right-hand side of the above equation, the only term containing $P_c(z)$ is the first term $-\mathrm{KL}(P_{\text{agg}}(z)||P_c(z))$. Therefore, the optimal $P_c(z)$ that maximizes the likelihood of data is $P_{\text{agg}}(z)$, although the optimization algorithm is not guaranteed to find this optimum.

At its optimality, $P_c(z)$ is the same as the aggregated posterior distribution $P_{\text{agg}}(z)$, in which case $T(P_x)$ can also be interpreted as the cross-entropy between the aggregated posterior under model generations ($\mathbb{E}_{x\sim P_x(x)} P_c(z|x)$) and the aggregated posterior under the real data distribution ($\mathbb{E}_{x\sim P_{\text{data}}(x)} P_c(z|x)$):

$$
T(P_x) = H(\mathbb{E}_{x\sim P_x(x)} P_c(z|x), P_{\text{agg}})
$$

C Detection of Code-Mixing

In this section, we show that model criticism can generalize some previous high-level evaluation metrics. In particular, we replicate the machine translation experiment in Zhou et al. (2019) under our framework. In this experiment, an English sentence might be translated into Spanish, German, or French, but never a mix of different languages. Therefore, one failure mode of a model is to generate text that contains code-mixing.

To criticize the existence of code-mixing, we need a model that can model the mixing of languages of a document. LDA (Blei et al., 2003)

[13] The decomposition is in fact the evidence lower bound (ELBO) where the expectation is taken w.r.t. the true posterior distribution, so the inequality becomes tight.

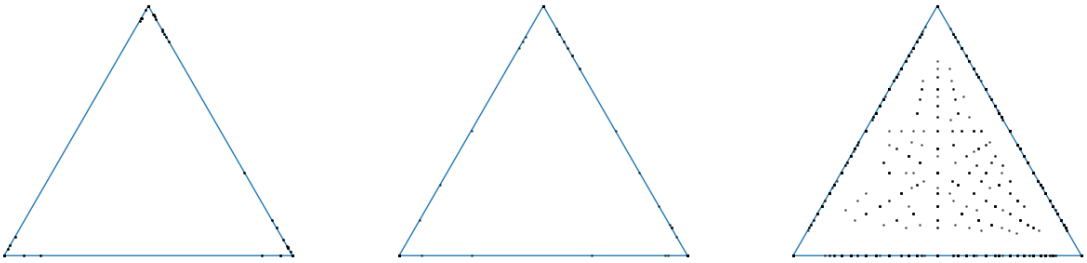

Figure 7: Scatterplot of samples from the topic posterior $\mathbb{E}_{P_x} P_c(z|x)$. Left: real data. Middle: autoregressive LM generations. Right: non-autoregressive LM generations. Real data and autoregressive LM generations contain much less code-mixing compared to non-autoregressive LM generations.

Topic 1 (German)	die der und in zu den von für dass ist wir des nicht auf das eine werden es im auch
Topic 2 (Spanish)	de la que en y el a los las del se una para un por no con es al sobre
Topic 3 (French)	de la et des à les le que en ' du dans nous pour qui une un est au pas

Table 10: Learned topics largely correspond to languages. The top 20 words per topic are shown.

is suitable for this purpose, as each language is analogous to a topic, and the document-topic latent variable parameterizes how topics (languages) are mixed in a document.

In LDA, each document is associated with a topic coefficient latent variable $z \in \Delta(N)$, where $N = \{1, 2, \ldots, M\}$ is a set of M topics and $\Delta(N)$ is a probability simplex over these topics such that z can be used to parameterize a categorical distribution. The prior over z is modeled using a Dirichlet distribution with parameters α:

$$P(z) \triangleq \text{Dirichlet}(z; \alpha).$$

The document-topic coefficient latent variable z defines a categorical distribution over the topics t_n for each word x_n in the document, and each topic in turn induces a categorical distribution over word types $P(x_n|t_n) = \phi_{t_n,x_n}$, where ϕ_{ij} parameterizes the probability of observing a word type j conditioned on a topic i, so the joint probability of topics and words for a document with N words is:

$$P(x, t|z; \phi) = \prod_{n=1}^{N} z_{t_n} \phi_{t_n,x_n}.$$

Since we are only interested in criticizing the document-topic coefficient latent variable z, we marginalize out the topic assignments of each word. Assuming there are M topics, the marginal distribution is:

$$P(x|z; \phi) = \prod_{n=1}^{N} \sum_{i=1}^{M} z_i \phi_{i,x_n}.$$

To fit this generative process on data, we set $M = 3$ (since there are three target languages). We treat ϕ as a latent variable with prior Dirichlet$(z; \alpha)$, and then use collapsed Gibbs sampling to sample topic assignments t from $P(t|x)$ (both z and ϕ are collapsed). β is fixed at 0.01, and α is optimized every 100 iterations with initial value $\frac{1}{M}$, and we use the MAP of $P(\phi|t, x)$ as a point estimate of ϕ^*.[14] For posterior inference, we use a two-stage sampling approach: Since $P(z, t|x; \phi^*) = P(t|x; \phi^*)P(z|t, x; \phi^*)$, we again apply collapsed Gibbs sampling to sample from $P(t|x; \phi^*)$ first, and then sampling from $P(z|t, x; \phi^*)$ is trivial since $P(z|t, x; \phi^*) = P(z|t; \phi^*) = \text{Dirichlet}(z; \alpha')$ where $\alpha'_i = \alpha_i + \sum_{n=1}^{N} \mathbb{1}[t_n = i]$.

We evaluate two probabilistic formulations of transformer LMs in terms of code-mixing. The first model is an autoregressive LM which assumes that each word depends on all previous words, and the second model is a non-autoregressive LM which assumes that different words are generated independent of each other (Gu et al., 2018). We train both LMs on the same English-German/Spanish/French dataset as in Zhou et al. (2019).[15]

Model Settings For both autoregressive and non-autoregressive LMs we use a transformer with 6 layers, 8 attention heads, model dimension 512,

[14] We use MALLET 2.0.8 (McCallum, 2002) to process data and learn the topic model.

[15] We randomly split 80% for training, 10% for validation, and 10% for testing, and the split might be different from Zhou et al. (2019).

hidden dimension 2048.[16] The autoregressive LM has 64.80M parameters, and the non-autoregressive LM has 65.98M parameters. Training takes about 16 hours on a single Nvidia A100 GPU.

Results Table 10 shows the learned topics, which largely correspond to the three underlying languages. Figure 7 visualizes samples from the posterior $P(z|x)$. We can see that for both the ground truth data and the autoregressive LM generations, the posterior is concentrated at the corners (hence it appears that there are fewer points), indicating that each translation contains mostly the same topic (underlying language). On the other hand, the posterior for non-autoregressive LM generations is dispersed, indicating that it's unable to fully commit to a single topic during generation due to the strong independence assumption. This result is the same as Zhou et al. (2019) without relying on external lexicons.

D Details of "A Surprising Text Generation Failure"

Data We set M to 50, $|\mathcal{Z}|$ to 256, $\mathcal{V}$ to the set of upper- and lower- case letters plus a special end-of-sequence symbol (so $|\mathcal{V}| = 53$. We uniformly sample 10k distinct subsequences of tokens $x_1^m, \ldots, x_{N_m}^m$ by first sampling uniformly a length between 4 and 11, and then each token is drawn uniformly from the set of letters (except for the last token $x_{N_m}^m$, which is always end-of-sequence). For each subsequence of tokens, we sample uniformly from $\mathcal{Z}$ and only allow emissions from the sampled state to this subsequence (such that the posterior $P(z_m|x_1^m, \ldots, x_{N_m}^m)$ is a delta distribution). The entries in the transition matrix $P_c(z_m|z_{m-1})$ are initialized with a normal distribution, divided by temperature 0.5, and then normalized using softmax. The entries in the emission matrix $P_c(x_1^m, \ldots, x_{N_m}^m|z_m)$ are initialized with a normal distribution and divided by temperature 0.3. Then we mask out emissions not allowed and normalize the matrix using softmax.

Posterior Inference Given a sequence x, the goal of posterior inference is to infer $P(z|x)$. This can be done in two steps: first, we segment x into subsequences (each subsequence corresponds to one hidden state). This segmentation is deterministic due to the end-of-sequence tokens. Next, we map each subsequence to its hidden state z_m by a simple lookup operation because we masked the emission matric to only allow one hidden state per subsequence. Therefore, $P(z|x)$ is a delta distribution.

Model The HSMM LM has 800 states. It is parameterized with the logits of its transition matrix $P(z_m|z_{m-1})$, its length emission matrix $P(N_m|z_m)$, and its emission matrix $P(x_1^m, \ldots, x_{N_m}^m|z_m, N_m)$. N_m ranges from 1 to 11. To parameterize the emission matrix, we take the 250k most common n-grams in the training dataset for n from 1 to 11.[17] It has 1.61B parameters due to the large number of possible emissions (the true data distribution P_{data} only has 2.63M parameters). The transformer LM has 6 layers, 4 attention heads, model dimension 512, and hidden dimension 1024.[18] It has 18.94M parameters.

Optimization We optimize HSMM using stochastic gradient descent (SGD) on the log marginal likelihood $\log P(x)$.[19] To marginalize out z_m and N_m, we use PyTorch-Struct (Rush, 2020). The model parameters are initialized with Xavier (Glorot and Bengio, 2010). We use a batch size of 8 and train the model for 10 epochs with the Adam optimizer (Kingma and Ba, 2014) on an Nvidia A100 GPU. The learning rate is initialized to 3e-1 and halved when the validation log marginal likelihood does not improve for 240 steps, with a minimal learning rate 3e-4. We found it necessary to pretrain the emission matrix $P(x_1^m, \ldots, x_{N_m}^m|z_m, N_m)$ for one epoch using a learning rate of 1e-1 while fixing other parameters to avoid the under-utilization of states. Pretraining takes about a day and training takes about a week, due to the large number of parameters and the small batch size that we can afford. The transformer LM is optimized with Adam as well, but with a batch size of 4096 tokens, 4k learning rate warmup steps to maximum learning rate 5e-4. It is optimized to 120k steps in total (about 19 epochs), following fairseq's default setting for conditional language modeling on IWSLT14

[16] We use the transformer_wmt_en_de implementation in fairseq.

[17] We cannot use the ground truth 10k valid subsequences of tokens since that would give HSMM LM an unfair advantage over the transformer LM.

[18] We use the transformer_iwslt_de_en implementation in fairseq (Ott et al., 2019).

[19] While HSMMs are usually optimized using the EM algorithm, we used SGD to be more comparable to transformers.

De-En.[20] Training the transformer LM takes about 4 hours on an Nvidia A100 GPU.

E Details of "Critiquing Discourse Coherence"

Data - PUBMED and ARXIV The PUBMED and ARXIV datasets in Section 4 are based on the datasets of Cohan et al. (2018), where each article consists of a list of sections with section titles.[21] We process the dataset in a few steps: First, we standardize the section titles by lemmatizing each word in the section title,[22] removing any numbers, and mapping each word to a standard spelling (e.g., "acknowledgement" is mapped to "acknowledgment"). Next, we remove from each article "see also", "external link", "reference", "further reading", "note", and "source" sections. Then we filter articles with fewer than 3 remaining sections, or with sections of more than 2k tokens or fewer than 30 tokens (the number of tokens is counted according to the GPT-2 tokenizer). Finally, we remove articles containing infrequent section titles, where the threshold is 500 for PUBMED and 200 for ARXIV (all counted on the training dataset).

Data - WIKI We download the English Wikipedia dumped on Dec 1, 2021.[23] We then use a Python package mwparserfromhell[24] to extract top-level sections from each article. We ignore redirect pages, disambiguation pages, links, files, and images, and strip away code. We also ignore articles about years. Then we process the dataset in the same way as how we processed PUBMED and ARXIV, except that we remove articles with fewer than 4 sections (since we always count the introductory paragraph of each article in Wikipedia as an "abstract" section), and that we removed infrequent section titles that appear fewer than 4k times in the training data.

Dataset Statistics The statistics of all three datasets can be found at Table 11.

Posterior Inference We finetune a BERT classifier (Devlin et al., 2019) to estimate $P(z_m|x_m)$ using the Adam optimizer. We use a batch size of 32, learning rate of 2e-5, and finetune for 3 epochs. The validation accuracies are 89.48%, 72.52%, and 88.15% on PUBMED, ARXIV, and WIKI respectively. Finetuning takes up to a few hours on a single Nvidia A100 GPU.

Language Models We use the base version of GPT-2[25] (LM_1, which has about 117M parameters) and the 125M version of GPT-Neo (LM_2)[26]. We use Adam to finetune all LMs.[27] Since both LM_1 and LM_2 have a limited context window size, we use truncated BPTT (Puskorius and Feldkamp, 1994) with context window size set to the maximum value possible (1024 for LM_1 and 2048 for LM_2). At generation time, we generate one token at a time and truncate the context to fit within the context window. We use a special symbol <endoftext> to mark article boundaries. For optimization we use a batch size of 8 (for GPT-Neo-based LMs we use a batch size of 4 but update parameters every two steps), a learning rate of 5e-5 (we did an initial learning rate search from {5e-6, 5e-5, 5e-4, 5e-3} on PUBMED and found 5e-5 to perform the best), and train the model for 20 epochs. Model checkpoints with the best validation loss (the lowest validation PPL) are used for the final evaluation. Training takes up to 24 hours on a single Nvidia A100 GPU.

Error Analysis Table 12 presents the most common section transition errors across all section types for different settings (W/ Title and W/O Title). We notice again that a very common transition error is to generate a section repeatedly. However, even in the ground truth data, there are repeated sections, such as "career → career" (appearing 0.08%), which is due to the misclassification by the BERT classifier used for the inference network.[28]

Repetition Errors Repetition is a common type of error found by model criticism (see Table 12). For example, on the WIKI dataset, repetition errors account for 25.93% of all errors (using the

[20] https://github.com/pytorch/fairseq/blob/5e343f5f23b4a90cca2beec416b87d4dd7a4264f/examples/translation/README.md#iwslt14-german-to-english-transformer

[21] They use an Apache-2.0 license.

[22] We use the lemmatizer of NLTK 3.6.7 (Bird and Loper, 2004).

[23] https://dumps.wikimedia.org/enwiki/20211201/

[24] https://github.com/earwig/mwparserfromhell (version 0.7.dev0)

[25] https://huggingface.co/gpt2

[26] https://huggingface.co/EleutherAI/gpt-neo-125M

[27] We use the training script from https://github.com/huggingface/transformers/blob/master/examples/legacy/run_language_modeling.py.

[28] In the training data there also exist some rare repetitions, such as https://web.archive.org/web/20220307192058/https://en.wikipedia.org/wiki/Leanna_Brown.

Dataset	#Train	#Val	#Test	#Sect Types	Med #Sect	Med Sect Len	Max Sect Len
PUBMED	32.35k	1.80k	1.84k	27	4	518	1986
ARXIV	4.91k	0.17k	0.15k	50	4	787.5	1965
WIKI	111.40k	13.97k	13.98k	96	4	122	1999
WIKI-SHORT	69.30k	8.56k	8.68k	96	4	101	500

Table 11: Data Statistics. Section length is measured using the GPT-2 tokenizer (Radford et al., 2019; Wolf et al., 2020). Section statistics are based on the validation set. More details on data processing can be found in Appendix E.

z_{m-1}	z_m	Frequency (%)
W/O Title		
career	career	0.26
abstract	life and career	0.24
reception	reception	0.14
abstract	playing career	0.14
plot	plot	0.14
total failures	-	7.54
W/ Title		
abstract	playing career	0.16
abstract	life and career	0.14
career	career	0.13
abstract	production	0.11
total failures	-	5.46

Table 12: The top 5 section transition errors on WIKI. Frequency is the frequency of observing the specific transition error across all transitions in the (generated) dataset. A transition is deemed an error if $P_c(z_m|z_{m-1}) < 0.01$. Here we use the better-performing GPT-2-based LMs (LM_1).

same criterion as in Table 12) on LM_1 W/O Title and 17.89% on LM_1 W/ Title. While previous works have shown that neural language models tend to repeat at the level of phrases (Holtzman et al., 2019) and sentences (Welleck et al., 2019), our work found that the repetition might even happen at a higher level, as shown in the qualitative example in Table 13.

F Details of "Critiquing Topic Correlations"

Data We use the same datasets as in Section 4. For topic modeling, we remove word types that appear in more than 50% of training documents, and we also remove LaTeXcommands such as `\xmath`.

Topic Model Training We use $M = 100$ topics. To learn the topic model, we use variational EM to optimize the ELBO with the default training settings in David Blei's CTM implementation.[29] At inference, we also use variational inference (without the M step), also with the default inference settings in David Blei's CTM implementation. Training takes up to a few days using a single Intel Xeon Platinum 8358 CPU.

Outlier Detection While Section 6 has shown that in aggregate the Latent NLL of the LM_1 generations is close to that of real data, we can identify outliers by finding x for which $T(x) = -\mathbb{E}_{z\sim P_c(z|x)} \log P_c(z)$ is high. We find that those outliers are usually pathological cases that result in a very different distribution of topics, as shown in Table 14.

More Visualizations Section 6 visualized the covariance matrices on WIKI. We also plot the covariance matrices on PUBMED and ARXIV in Figure 8 and Figure 9. Note that we use hierarchical clustering of the covariance matrix on the test set to reorder topics, and we clamp the values in the covariance matrix to be in the range of [-5, 5] for plotting.

G Details of "Critiquing Coreference Chains"

Data All experiments in this section use a subset of the WIKI dataset: we apply a simple filter to only consider articles about films, by matching the first section of the article with the regular expression `.*is a.*film.*`.

Coreference Resolution We use an off-the-shelf neural coreference resolution system neuralcoref[30] to infer $\mathbf{z}$ given an article. We limit our studies to only consider person entities.

[29] http://www.cs.columbia.edu/~blei/ctm-c/

[30] https://github.com/huggingface/neuralcoref/tree/60338df6f9b0a44a6728b442193b7c66653b0731

Inferred Section Title	Section Text
abstract	Toledo Township is a township in Dauphin County, Pennsylvania, United States. As of the 2010 census, the township population was 1,655. It is part of the Dutchess/Berwick Micropolitan Statistical Area. Toledo Township was organized in 1867, and named for the United States senator, Judge John Toledo.
geography	Toledo Township is in northern Dutchess County, bordered by the city of Altoona to the north, the borough of Ritzel to the southeast, and Tuskegee Township and the unincorporated hamlet of Westmoreland Township to the south. According to the United States Census Bureau, the township has a total area of, of which is land and, or 0.62%, is water. It is bordered on the south by the Tullahonas River, on the west by the Delaware Channel, on the south by the Mohawk River and on the west by Tullahonas Creek, whose tributaries are the Westmoreland and Trenton rivers. Pennsylvania Route 11, which runs between Routes 11 and N, crosses the township via the Tuskegee River ...
demographic	As of the census of 2000, there were 1,638 people, 809 households, and 595 families residing in the township. The population density was 787.1 people per square mile (285.2/km2). There were 944 housing units at an average density of 331.2 per square mile (126.5/km2). The racial makeup of the township was 95.07% White, 1.81% African American, 0.46% Native American, 0.36% Asian, 0.06% Pacific Islander, 0.42% from other races, and 1.06% from two or more races. Hispanic or Latino of any race were 1.13% of the population. There were 809 households, out of which 32.4% had children under the age of 18 living with them, 49.0% were married couples living together, 11.1% had a female householder with no husband present, and 30.0% were non-families. 26.5% of all households were made up of individuals, and 12.9% had someone living alone who was 65 years of age or older ...
demographic	Census 2010 As of the 2010 United States Census, there were 1,655 people, 613 households, and 585 families residing in the township. The population density was 847.8 people per square mile (287.1/km2). There were 640 housing units at an average density of 296.1 per square mile (110.2/km2). The racial makeup of the township was 95.17% White, 1.81% African American, 0.41% Native American, 0.12% Asian, 1.00% from other races, and 0.49% from two or more races. Hispanic or Latino of any race were 2.67% of the population. There were 613 households, out of which 33.8% had children under the age of 18 living with them, 56.9% were married couples living together, 12.7% had a female householder with no husband present, and 29.7% were non-families. 24.6% of all households were made up of individuals, and 13.1% had someone living alone who was 65 years of age or older.
notable people	Joseph R. Clements (May 17, 1911 – June 23, 1998) Mayor of Mount Pleasant, South Carolina Jefferson Daugherty (born 1935 in Chatham) U.S. Senator, United States House of Representatives ...

Table 13: An example section-level repetition error (LM_1 W/O Title on WIKI). The most common section type after "demographic" is "education". The structures of the repeated sections are similar yet the facts are different.

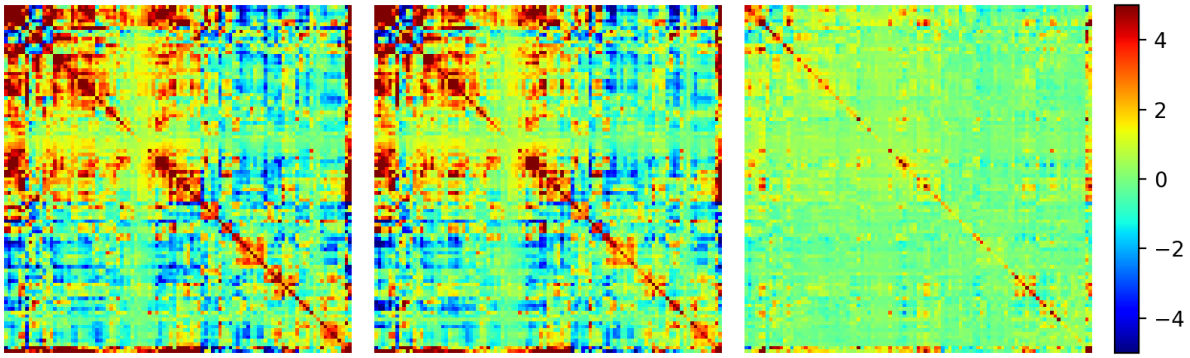

Figure 8: Topic covariance matrix for the induced z (on PUBMED). Left: Test set (P_{data}). Middle: LM_1 generations (P_{model}). Right: generations of LM_1 trained on ARXIV as a visual baseline.

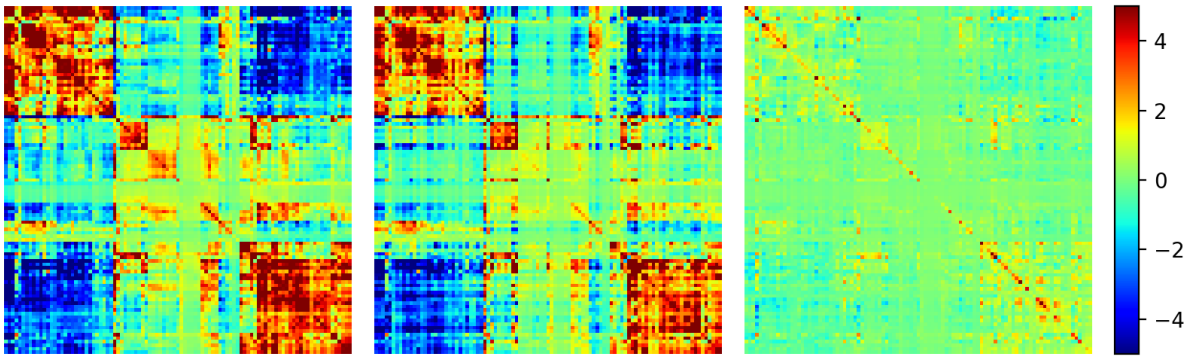

Figure 9: Topic covariance matrix for the induced z (on ARXIV). Left: Test set (P_{data}). Middle: LM_1 generations (P_{model}). Right: generations of LM_1 trained on PUBMED as a visual baseline.

$T(x)$	Text
280.88	... to the best of our knowledge, this result, together with previous results, supports the conclusion that there is no difference in the spin concentration between bp and qp versions of the hamiltonian in the quenched version of the hamiltonian in the quenched version of the hamiltonian in the qp version of the hamiltonian in the quenched version of the hamiltonian in the quenched version of the hamiltonian in the quenched version of the hamiltonian in the quenched ver sion of the hamiltonian in the quenched version of the hamiltonian in the quenched version of the hamiltonian in the quenched version of the hamiltonian in the quenched version ...
224.74	... if we now use the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential of the electrostatic potential ...

Table 14: Top 2 outliers identified by $T(x)$ on the generations from LM_1 finetuned on the ARXIV dataset (W/O Title). The average $T(x)$ (Latent NLL) is 161.40.

Gender Assignment To avoid the open vocabulary problem of proper nouns and also due to the fact that personal pronouns are usually gendered in English, we replace proper nouns with their genders (Male/Female/Plural/None of the above). In order to identify genders of proper nouns, we use the majority voting of the genders of the pronouns that corefer with them (for example, "she" corresponds to female, "he" corresponds to male, and "they" corresponds to plural). If there are no gen-

dered pronouns that corefer with the given proper noun, we assign "None of the above" as the gender. The caption of Figure 13 presents an example of the gender assignment procedure.

Language Models We use LM_1 (W/O Title) trained on WIKI. We apply the same filtering process as applied to real data to only consider generations about films.

z_1	z_2	z_3	z_4	z_5	$\log P_c$	z_5^*	$\log P_c(z_5^*\|z_{<5})$
$[M]_0$	$[he]_0$	$[M]_0$	.	$[They]_1$	-13.37	$[M]_0$	-1.22
.	$[M]_0$	$[M]_1$	$[M]_2$	$[she]_2$	-14.72	$[his]_2$	-2.72
	.	$[M]_0$	.	$[her]_1$	-11.29	$[He]_0$	-1.67
$[N]_0$	.	$[M]_1$	$[N]_0$	$[she]_2$	-8.05	$[his]_1$	-2.15
$[F]_0$	$[her]_0$	$[he]_1$	$[M]_2$	$[he]_2$	-7.03	$[his]_1$	-2.22
$[he]_0$	$[her]_1$	$[she]_1$	.	$[He]_2$	-8.14	$[M]_0$	-1.87
.	.	$[M]_0$	$[he]_1$	$[she]_0$	-14.18	$[his]_1$	-1.48
.	$[N]_0$	.	$[he]_1$	$[her]_2$	-10.82	$[his]_1$	-1.48
$[M]_0$	.	$[her]_1$	.	$[He]_2$	-8.74	$[M]_0$	-2.20
$[F]_0$	$[she]_0$	.	$[F]_0$	$[him]_1$	-8.18	$[her]_0$	-1.28
$[F]_0$	$[M]_1$	.	$[F]_2$	$[him]_3$	-8.56	$[her]_2$	-1.41
$[F]_0$	$[F]_1$	$[her]_1$	$[her]_1$	$[he]_1$	-10.25	$[her]_1$	-1.72
$[F]_0$	$[her]_0$	$[her]_0$	$[he]_0$	$[P]_1$	-7.28	$[her]_0$	-1.23

Table 15: Unlikely $z_5|z_{<5}$ and the corresponding $\log P_c(z_5|z_{<5})$ in LM_1 generations according to the learned critic ($\log P_c(z_5|z_{<5}) < -7$). To get a better sense of what is considered likely by the critic, we also showed $z_5^* = \arg\max_{z_5} P_c(z_5|z_{<5})$ as well as $\log P_c(z_5^*|z_{<5})$. M (Male), F (Female), P (Plural), and N (None of the above). Blank: padding.

Critic We use a 5-gram language model with Kneser-Ney smoothing (Ney et al., 1994) to fit the critic distribution, where we used $\frac{\text{\#unique (n-1)-grams}}{\text{\#unique (n-1)-grams}+2\text{\#unique n-grams}}$ as the discount factor (Stolcke, 2002).

What does this critic learn? Table 15 shows a random subset of unlikely coreference chain n-grams generated by LM_1 according to the critic. We can see that the learned critic makes sense intuitively. For example, in the first row, $[They]_1$ is created even though the previous context only contains a single entity;[31] in the second row, "she" is used to refer to a male; in the third row, "her" doesn't have any antecedent.[32]

More Results Table 16 shows the coreference chains that occur more frequently in LM generations than in real data (we again used 5-gram LM with Kneser-Ney smoothing to estimate the probabilities). We can see that some of these are implausible similar to the observation in the main paper: for example, in the second to last row a proper noun $[Male]_0$ is used after a pronoun $[he]_0$ is used to refer to the same entity in the sentence. In Table 17 we show the other direction: the coreference chains that occur more frequently in real data than LM generations. We can see that while this also shows the places where the coreference distributions do not match, the coreference structures here are not unlikely, since they appear frequently in real data.

z_1	z_2	z_3	z_4	z_5	P_{data}	P_{LM}
.	.	$[M]_0$	$[F]_1$	$[M]_0$	0.01	0.06
$[M]_0$	.	$[N]_1$	$[M]_0$	$[M]_2$	0.04	0.17
$[M]_0$	.	$[N]_1$	$[M]_0$	$[M]_0$	0.04	0.17
.	.	$[M]_0$	$[he]_0$	$[M]_0$	0.01	0.04
.	$[He]_0$	$[his]_0$	.	$[His]_0$	0.02	0.06

Table 16: The top 5 coreference chain n-grams with the largest log probability differences between LM generations and real data ($\log P_{\text{LM}}(z_5|z_{<5}) - \log P_{\text{data}}(z_5|z_{<5})$). We only consider n-grams that appear more than (including) 5 times in both test set and LM generations. M (Male), F (Female), and N (None of the above).

z_1	z_2	z_3	z_4	z_5	P_{data}	P_{LM}
.	$[N]_0$	$[F]_1$	.	$[M]_2$	0.17	0.05
$[him]_0$	$[she]_1$	$[him]_0$	.	.	0.48	0.14
$[M]_0$	$[him]_0$	$[he]_0$	.	.	0.36	0.11
$[N]_0$	$[N]_0$	$[N]_0$	.	$[N]_0$	0.18	1.16
			.	$[P]_0$	0.01	0.00

Table 17: The top 5 coreference chain n-grams with the *lowest* log probability differences between LM generations and real data ($\log P_{\text{LM}}(z_5|z_{<5}) - \log P_{\text{data}}(z_5|z_{<5})$). We only consider n-grams that appear more than (including) 5 times in both test set and LM generations. M (Male), F (Female), P (Plural), and N (None of the above). Blank: padding.

Qualitative Examples Figure 10 and Figure 11 show two examples where coreference abnormalities are successfully detected by the model. Figure 12 shows an example where due to the limited context window size of the 5-gram critic, a pronoun is identified as unlikely due to its antecedent falling outside the context window even though it is appropriate. Figure 13 shows an example where due to coreference resolution errors are intertwined with coreference errors. This type of errors would likely go away as more powerful coreference resolution systems are developed.

[31] That being said, it is possible that outside this context window there are other entities that makes using "They" possible.

[32] Since this 5-gram starts with padding, there is nothing to the left of the context window.

Tokenized Text

... . After the marriage , [Jyothish]$_0$ finds out that Raja [Rao]$_1$ had raped [Sridevi]$_2$. and [he]$_1$ also tells [him]$_1$ that Raja [Rao]$_1$ is [his]$_1$ father , so [he]$_1$ tries to kill them both ...

Coreference Chains z

. [Female]$_0$ [Male]$_1$ [None]$_2$. [he]$_1$ [him]$_1$ [Male]$_1$ [his]$_1$ [he]$_1$

5-gram critic $\mathbf{P_c}$

$$P_c(\text{[Male]}_1 \mid \text{previous entity mentions}) \approx P_c(\text{[Male]}_1 \mid \text{[None]}_2\ \text{[he]}_1\ \text{[him]}_1) = \exp(-7.29)$$

Figure 10: A qualitative example where the critic correctly identifies an implausible coreference n-gram. The argmax at the circled position is [he]$_1$ with probability $\exp(-2.32)$. We only highlighted the root of each entity mention to avoid clutter.

Tokenized Text

... . [He]$_0$ realizes that [he]$_0$ must put [his]$_0$ life so that the [girl]$_1$ will know anything about it , only that [she]$_1$ wo n't because of [himself]$_0$ ...

Coreference Chains z

. [He]$_0$ [he]$_0$ [his]$_0$ [girl]$_1$ [she]$_1$ [himself]$_0$

5-gram critic $\mathbf{P_c}$

$$P_c(\text{[himself]}_0 \mid \text{previous entity mentions}) \approx P_c(\text{[himself]}_0 \mid \text{[he]}_0\ \text{[his]}_0\ \text{[girl]}_1\ \text{[she]}_1) = \exp(-9.00)$$

Figure 11: A qualitative example where the critic correctly identifies an implausible coreference n-gram. The argmax at the circled position is [him]$_0$ with probability $\exp(-1.43)$. We only highlighted the root of each entity mention to avoid clutter.

Potential Improvements By throwing away all the other words but the entity mentions, we lose much information about the sentence, even syntactic information such as the c-command structures (Chomsky, 1993). By augmenting the entity mentions with syntactic features, the critic is likely to be even more powerful at identifying more nuanced abnormalities of language model generations.

Tokenized Text

... . The next day , [Wayne]$_0$ tells [Jennifer]$_1$ [he]$_0$ wants to tell [her]$_1$ the story and that [she]$_1$ should take care of [herself]$_1$. [Jennifer]$_1$ tells [him]$_0$ that [she]$_1$ feels that because of it , [her]$_1$ life is in danger , so [she]$_1$ asks [him]$_0$ about [him]$_0$ before taking out [her]$_1$ phone , which is later found on [her]$_1$ in the morning ...

Coreference Chains z

. [Male]$_0$ [Female]$_1$ [he]$_0$ [her]$_1$ [she]$_1$ [herself]$_1$. [Female]$_1$ [him]$_0$ [she]$_1$ [her]$_1$ [she]$_1$ [him]$_0$ [him]$_0$ [her]$_1$ [her]$_1$

5-gram critic $\mathbf{P_c}$

$$P_c(\text{[him]}_0 \mid \text{previous entity mentions}) \approx P_c(\text{[him]}_0 \mid \text{[she]}_1\ \text{[herself]}_1\ \text{[Female]}_1) = \exp(-8.76)$$

Figure 12: A qualitative example where the critic incorrectly identifies an implausible coreference n-gram, due to the limited context window not containing the antecedent of the pronoun. The argmax at the circled position is [her]$_1$ with probability $\exp(-1.49)$. We only highlighted the root of each entity mention to avoid clutter.

H Human Evaluation

Inspired by Persing et al. (2010), we evaluate the coherence of an article by asking human annotators to first label the type of each section, and then label whether an article is coherent based on the organization of section types.

Our human evaluation system is based on Amazon Mechanical Turk (Crowston, 2012). Each human annotator needs to first go through a training phase to learn the typical organization of articles in the training dataset, as shown in Figure 14. After this training phase, a human annotator will use the interface shown in Figure 15 to annotate whether an article is coherent or not, where the annotator needs to first label the section types of each sec-

Tokenized Text

... . [Jack]$_0$'s girlfriend , the wealthy Baron von Brühl (Peter Lorre) , also steals the [girl]$_1$. After an unpleasant and embarrassing incident in which [she]$_1$ is forced to drink a pager before going home , the [Baron]$_2$'s [henchwoman]$_3$ is caught and thrown on the balcony of [his]$_3$ inn , where [she]$_1$ is set upon . [Jack]$_0$ rescues [her]$_1$ and takes [her]$_1$ to Austria to live with von [Brühl]$_4$. Von [Brühl]$_4$ is now worried that Jenny (Ann Sheridan) is in love with [Jack]$_0$. After realizing that [she]$_4$ is already engaged to [Jack]$_0$, [he]$_4$ persuades [her]$_4$ to go with [him]$_4$ to Austria as soon as [they]$_5$ can ...

Coreference Chains z

. [None]$_0$ [Female]$_1$. [she]$_1$ [Male]$_2$ [Male]$_3$ [his]$_3$ [she]$_1$. [None]$_0$ [her]$_1$ [her]$_1$ [Female]$_4$. [Female]$_4$ [None]$_0$. [she]$_4$ [None]$_0$ [he]$_4$ [her]$_4$ [him]$_4$ [they]$_5$

5-gram critic $\mathbf{P_c}$

$P_c($[he]$_4$ | previous entity mentions$)$
$\approx P_c($[he]$_4$ | [Female]$_4$ [None]$_6$ [she]$_4$ [None]$_0)$
$= \exp(-11.68)$

Figure 13: A qualitative example where coreference errors are intertwined with coreference resolution errors: the circled position is deemed implausible because it's using a male pronoun [he]$_4$ to refer to a female [Brühl]$_4$. The argmax at the circled position is [she]$_4$ with probability $\exp(-1.80)$. This example also shows how gender assignments are made: since [henchwoman]$_3$ corefers with a male pronoun [his]$_3$, it is labeled as a male. We only highlighted the root of each entity mention to avoid clutter.

tion,[33] and then label if the article is coherent or not based on the labeled section types.

The instructions for the training phase is shown in Figure 16, and the instructions for the testing phase is shown in Figure 17. These instructions are shown upon clicking the button "Instructions" in the labeling webpage. In the instructions we disclose to the annotators that the data will be used for reasearch and will be made public after anonymizing.

We collected 71-128 annotations per system from five volunteer annotators (all annotators are US-based graduate student volunteers), and we compute the score of each system by computing the percentage of articles labeled as coherent.

The main human evaluation results have been presented in Table 4 in the main paper. In Table 18 we take a deeper look at what type of section organizations are considered incoherent by humans. We can see that while many errors are repetition errors, there are many other types of errors as well. For example, for the most common mistake (the first row of Table 18), a case report is introduced without an introduction section; for the second most common mistake, "material and method" is directly followed by a "discussion" section, skipping results.

[33] All possible section types are provided in the dropdown menu. We used PUBMED for this experiment mainly because it has the fewest number of possible section types.

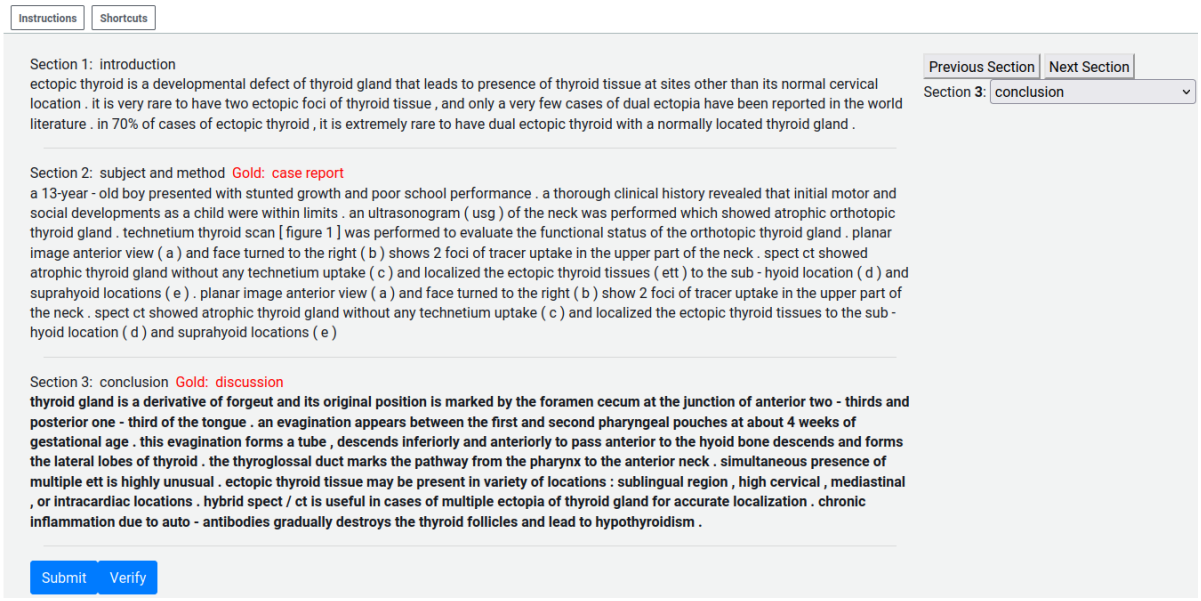

Figure 14: The training interface of human evaluation. Upon clicking "Verify", the selected section types will be compared against gold section types.

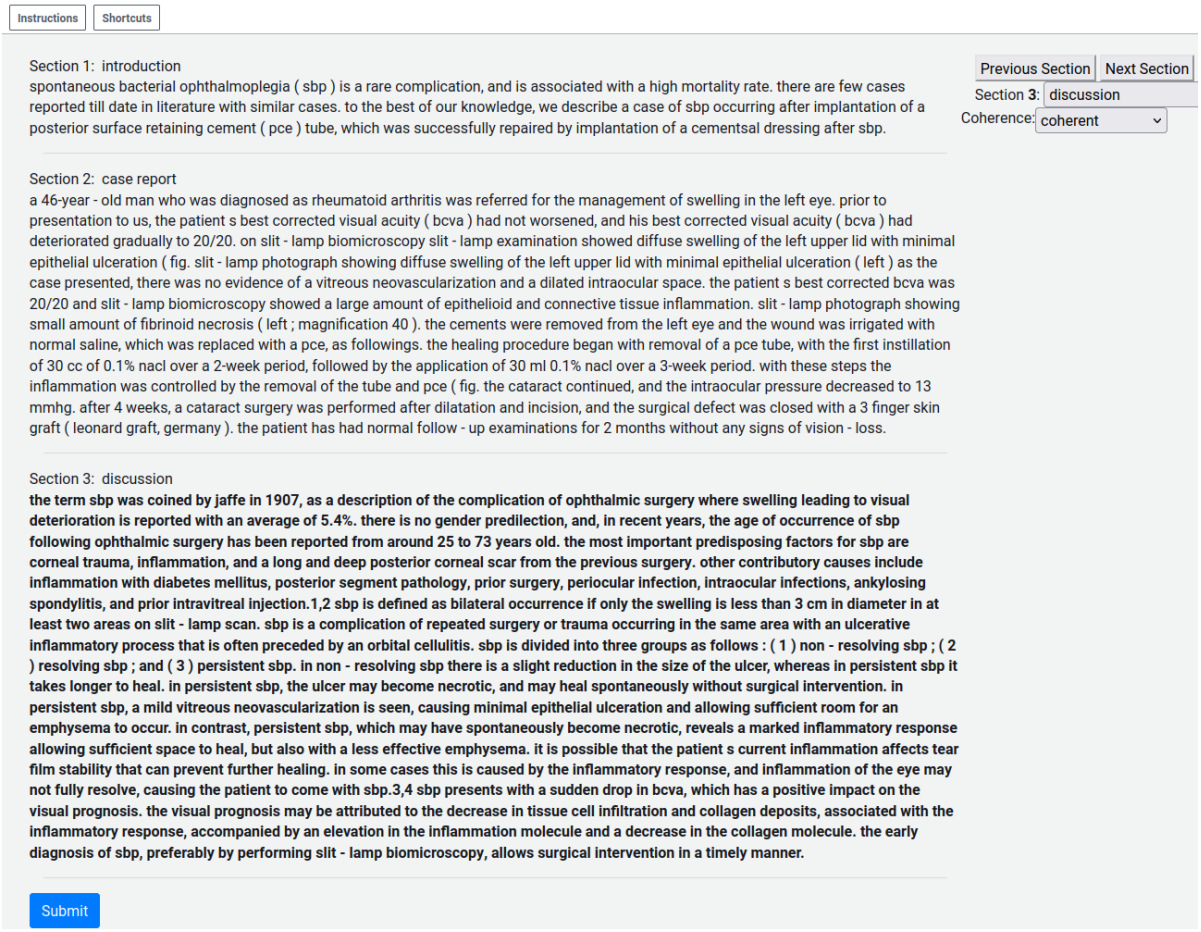

Figure 15: The testing interface of human evaluation. The human annotator needs to first label all section types and then label whether the article is coherent or not based on the labeled section types.

Section 1	Section 2	Section 3	Section 4	Section 5	Section 6	Section 7
case report	discussion					
introduction	material and method	discussion				
introduction						
material and method	result	discussion				
introduction	material and method	result	discussion			
introduction	case report	case report	discussion			
introduction	material and method	material and method	result	discussion	conclusion	
case report	discussion	conclusion				
introduction	conclusion					
introduction	case report					
introduction	result	discussion				
introduction	material and method	result	result	discussion		
introduction	case report	discussion	conclusion			
introduction	material and method	discussion	conclusion			
introduction	case report	case report	case report	discussion		
introduction	material and method	result	discussion	conclusion		
introduction	case report	result	discussion			
introduction	material and method	result	result			
introduction	material and method	result	result	discussion	conclusion	
introduction	result	result and discussion				
discussion						
material and method	discussion					
case report	discussion	case report	conclusion			
introduction	case report	introduction	discussion	discussion		
introduction	case report	result	result	discussion		
introduction	material and method	result	discussion	discussion	conclusion	
introduction	material and method	conclusion				
introduction	introduction					
introduction	material and method	material and method	discussion	conclusion		
introduction	material and method	statistical analysis	result	result	discussion	conclusion

Table 18: The most common incoherent section type organizations according to human evaluation. Note that the fifth row does not seem to have any coherence issues, which is due to section texts that are too bad to support any section type. Note that we instructed annotators that “if the text of a section is too bad to tell which section type it is, you should label the article as "incoherent" (for labeling the section type, just select a random one in this case).”

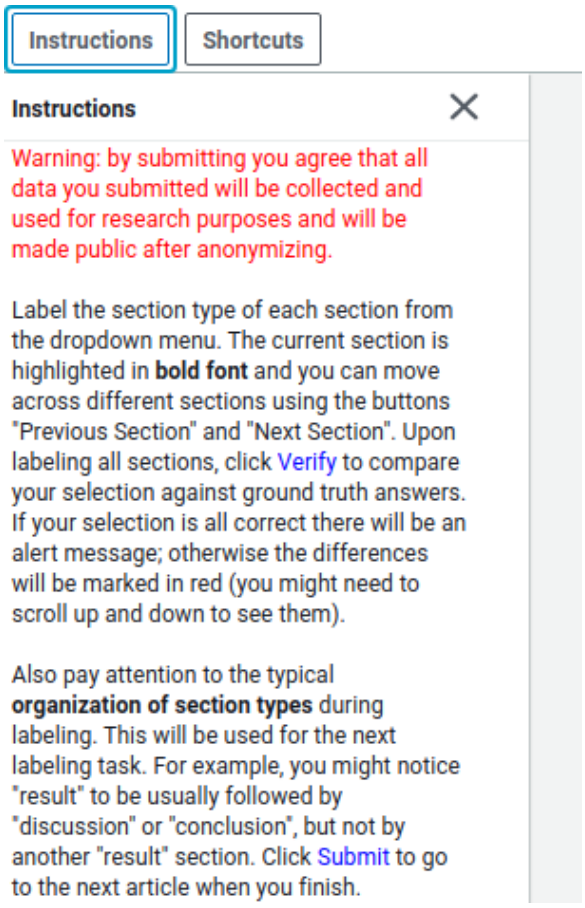

Figure 16: Instructions for the training interface of human evaluation. These instructions are shown upon clicking "Instructions" in the training interface (Figure 14).

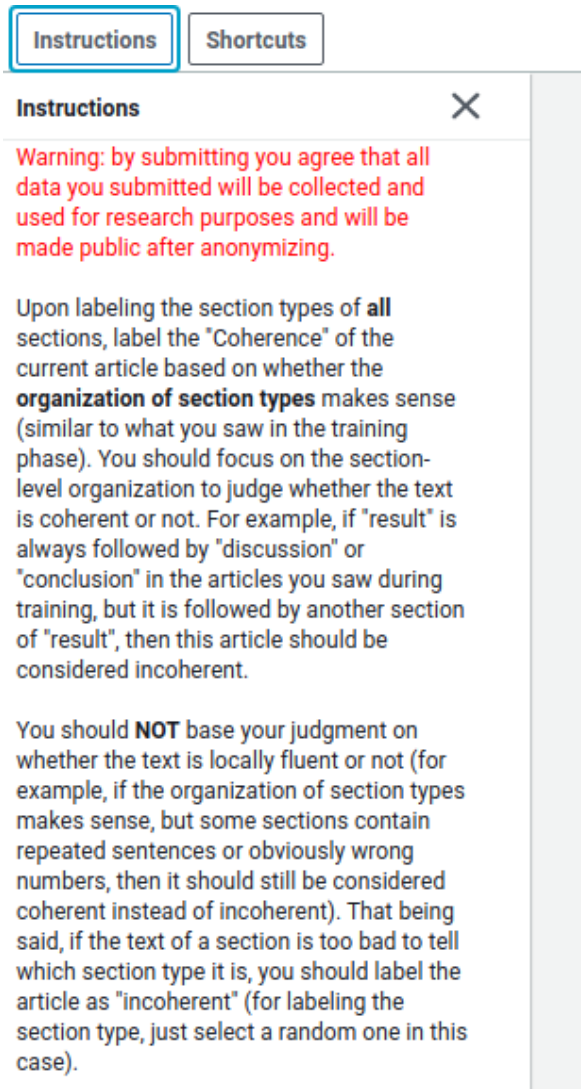

Figure 17: Instructions for the testing interface of human evaluation. These instructions are shown upon clicking "Instructions" in the testing interface (Figure 15).