InFillmore: Frame-Guided Language Generation with Bidirectional Context

Jiefu Ou[†‡] **Nathaniel Weir**[†] **Anton Belyy**[†] **Felix Yu** **Benjamin Van Durme**
Johns Hopkins University
jouaa@connect.ust.hk
{nweir,abel,fyu17,vandurme}@jhu.edu

Abstract

We propose a structured extension to bidirectional-context conditional language generation, or "infilling," inspired by Frame Semantic theory (Fillmore, 1976). Guidance is provided through two approaches: (1) model fine-tuning, conditioning directly on observed symbolic frames, and (2) a novel extension to disjunctive lexically constrained decoding that leverages frame semantic lexical units. Automatic and human evaluations confirm that frame-guided generation allows for explicit manipulation of intended infill semantics, with minimal loss in distinguishability from human-generated text. Our methods flexibly apply to a variety of use scenarios, and we provide an interactive web demo.[1]

1 Introduction

A popular strategy for automatic story generation is to proceed in a coarse-to-fine manner: first by proposing a *story plan*, and then realizing it into natural language form using large pretrained neural language models (Fan et al., 2018; Goldfarb-Tarrant et al., 2019). In this work, we study the use of FrameNet frames (Baker et al., 1998) as representational units for such plan guidance.

In Frame Semantics (Fillmore, 1976; Fillmore and Baker, 2010), words evoke structural situation types (frames) that describe the common schematic relationships between lexical items. We hypothesize that these structured types can be used to effectively induce the semantic content of text generated by increasingly powerful pretrained language models, yielding a flexible, controllable and domain-general model for surface realization of story plans with a variety of dimensions for user guidance.

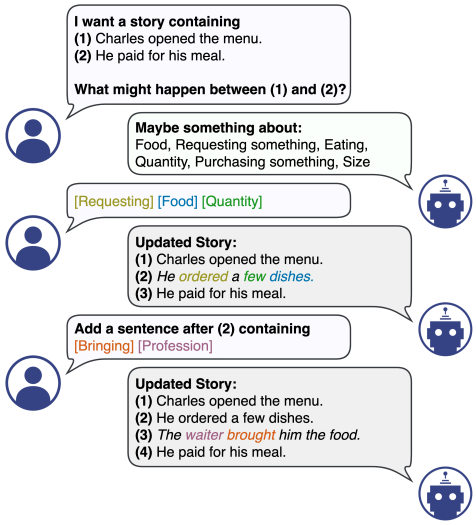

Figure 1: The proposed generation model, applied to the interactive story generation task. Similar to the existing infilling models, a user can insert or rewrite text spans at any position in a story. With the proposed extension, generation can be guided via explicit frame semantic constraints, either provided manually or suggested by the model based on surrounding context.

Based on this supposition, we fine-tune a recent infilling model (Donahue et al., 2020) with a frame-guided denoising objective. We contrast this approach with a novel method for frame-guided generation that modifies only the decoding step of a standard language model through lexical manipulation. The idea originates from the annotation scheme of FrameNet, where each semantic frame is annotated with a set of evocative lexical units (LUs). We posit that it is possible to guide the model's generation with frames *without* modifying its training procedure by instead lexically constraining its generation output to contain frame-associated LUs. Therefore, we develop an extension to lexically-constrained decoding that leverages LUs as ordered disjunctive constraint sets. Given a possibly multi-frame sequence and a generative model, our method en-

[†] Corresponding authors.
[‡] Work done during an internship at the Center for Language and Speech Processing, JHU.
[1] Codebase and demo available from https://nlp.jhu.edu/demos/infillmore.

Proceedings of the 10th Conference on Lexical and Computational Semantics, pages 129–142
August 5–6, 2021, Bangkok, Thailand (online)

forces the generation of one of the associated LUs for each frame in the sequence. This decoding method is implemented as a plug-and-play module that can be imposed on top of any standard generative language model.[2]

We evaluate through a sentence-infilling task based on ROCStories (Mostafazadeh et al., 2016), assessing performance on two dimensions: 1) the *quality* of generation, as measured through perplexity and human evaluation; and 2) the *fidelity*, which scores whether generated text evokes the frames used as guidance. We demonstrate that our methods utilize guidance to generate frame-evoking surface realizations without meaningfully detracting from the contextual narrative coherence. We also demonstrate the practical applicability of frame-guided generation in a variety of example use cases.

2 Related work

Controlled Generation Existing work employs a variety of pretraining strategies to guide and/or diversify text generation. Keskar et al. (2019) train large-scale language models on text prepended with *control codes*, allowing for guided content and style. PPLM (Dathathri et al., 2020) makes use of lightweight *attribute classifiers* that guide generation without requiring language model retraining. For diverse generation of sentences in a more general scenario, Weir et al. (2020) train models to condition on *semantic bit codes* obtained from hashing sentence embeddings.

Constrained Generation Separate lines of work employ *lexical constraints* to achieve the same goal of guided and diverse generation. As such, lexically constrained beam search methods such as Grid Beam Search (Hokamp and Liu, 2017) and Dynamic Beam Allocation (Post and Vilar, 2018; Hu et al., 2019a) were proposed as the decoding methods for causal generation with disjunctive positive constraints (Li et al., 2020b), paraphrasing (Hu et al., 2019b; Culkin et al., 2020), machine translation (Zhang et al., 2021), and abstractive summarization (Mao et al., 2020). Lu et al. (2020) generalize beam-search based methods with an algorithm that supports lexical constraints in the conjunctive normal form.

Parallel are the approaches that handle lexical constraints in an editing manner: starting with a sequence of keyword constraints and fleshing out a sentence via editing operations such as insertion or deletion (Miao et al., 2019; Liu et al., 2019; Sha, 2020; Susanto et al., 2020; **?**; Zhang et al., 2020). Finally, it is possible to satisfy lexical constraints in a soft manner as external memories (Li et al., 2020a, 2019) or constructing constraint-aware training data (Chen et al., 2020).

Story Generation Inspired by the traditional pipeline of Reiter and Dale (2000), recent work tackles generation of stories in a coarse-to-fine manner (Fan et al., 2018): based on a premise, a structured *outline* is generated first, and then an outline-condition model generates the full story. To represent the story outline, existing approaches typically either model it as a latent variable, or use symbolic representations such as key phrases (Xu et al., 2018; Yao et al., 2019; Goldfarb-Tarrant et al., 2019; Gupta et al., 2019; Rashkin et al., 2020), short summaries (Jain et al., 2017; Chen et al., 2019), verb-argument tuples (Martin et al., 2018), or PropBank predicates and arguments (Fan et al., 2019; Goldfarb-Tarrant et al., 2020). Our work can be viewed as an extension of this direction, where a *Content Planner* model generates an outline as a sequence of FrameNet frames, and our methods generate a surface form story.

3 Data

FrameNet FrameNet is a lexical database of English based on Fillmore's theory of Frame Semantics. It defines more than 1200 frames spanning various semantic domains, where each frame schematically describes a type of event, relation, or entity. A frame is defined with a set of corresponding *Frame Elements* (FEs): the participants in the frame with relational roles, and a set of *Lexical Units* (LUs): words that evoke the frame in text.

For example, the **Apply_heat** frame that describes the concept of cooking consists of core FEs *Food*, *Cook*, *Container*, *Heating_instrument*, and *Temperature_setting*, and has evocative LUs that include *fry, bake, boil*, and *broil*. Frame annotations provide a partial (albeit rich) picture of sentence meaning, i.e. information not governed by the syntax/semantics interface. We find that they serve as an effective, theory-grounded formalism for discrete semantic guidance of generation.

Conceptually, our choice to use FrameNet as guiding semantics builds upon trends in generative modeling of discourse (Ferraro and Van Durme, 2016) that treat text documents as mixtures of hi-

[2]Fairseq-based implementation and data to be released.

Story	Charles went shopping. He bought fruit. Then he left.
ILM	Charles went shopping. [blank] Then he left. *[sep] He bought fruit.*
S-FFL	[sep] [Food] He bought fruit.
A-FFL	[sep] [Commerce_buy] [Food] He bought fruit.

Figure 2: Training examples for frame-guided ILM models. Examples are fed from left to right, with the *italicized* portion of the ILM example replaced by the frame-injected sequences for FFL examples.

erarchical latent variables in accordance with classical theories of frame semantics (e.g. Minsky (1974); Fillmore (1976)). As described by Ferraro and Van Durme (2016), FrameNet frame information can be used to learn a hierarchical latent representation of sentence-level semantics that produces discourse models that better fit to natural text data. Our work then asks whether this information can be used to harness the increasingly powerful ability of recent neural language models for the purposes of controlled story generation.

ROCStories Mostafazadeh et al. (2016) introduce the ROCStories corpus, which comprises over 98K 5-sentence simple stories that can serve as a resource for commonsense narrative schema learning and story generation (Ippolito et al., 2020). We use this dataset to evaluate the performance of our methods (described in section 5).

4 Approach

4.1 Model Architecture

The Infilling by Language Modelling (ILM, Donahue et al., 2020) framework fine-tunes pretrained unidirectional language model such as GPT-2 (Radford et al., 2019) to generate target infill spans with bidirectional contexts. This allows the ILM model to flexibly generate text at any position in a document, as shown in Figure 1. In this work, we introduce FrameNet frame guidance into the ILM pipeline. We propose and compare methods based on 1) fine-tuning on frame-annotated data (4.2), and 2) imposing lexically-constrained beam search during decoding (4.3) with the original ILM.

4.2 Fine-Tuned "Framefilling" (FFL)

The ILM task definition comprises a context passage x containing **[blank]** tokens at points where the new spans must be generated.[3] The passage x is concatenated with a **[sep]** token and golden span infills (each separated by another **[sep]**) to form a fine-tuning instance for an off-the-shelf unidirectional language model such as GPT-2. We build on this setup by adding one or more frame ID tokens $F_1, F_2, \ldots$ (e.g. **[Food]**) as prefixes of each golden infill span, as shown in Figure 2. A model fine-tuned on this modified formulation, which we call a "framefilling" model (FFL for short), therefore conditions each infill on the bidirectional context as well as one or more control codes that guide the infill's semantic content. If an example contains multiple infills, subsequent infills are conditioned on the frames and text of previous infills.

We experiment with multiple variants of the FFL model, varying primarily in the level of frame guidance. We train a variant on infilling examples that contain a single frame ID (S-FFL), another on examples with a set of one or multiple frames (M-FFL; number of frames sampled from a geometric distribution with $p = .4$), and a final variant conditioned on all frames (covered by FrameNet v1.7) triggered by the infill (A-FFL). In all cases, the frame ID tokens are predicted by a state-of-the-art neural FrameNet parser (Xia et al., 2021).[4]

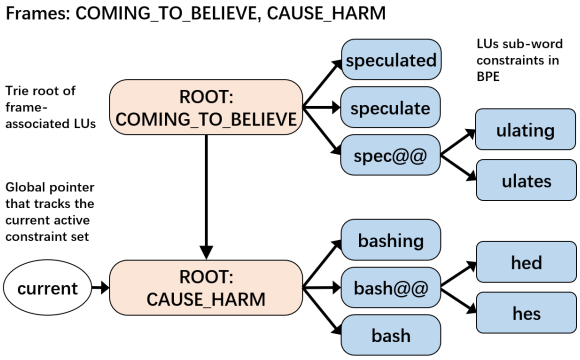

Figure 3: Example of LCD constraint representation in a list of 2 tries, corresponding to the given frames **Coming_to_believe** and **Cause_harm** (red). Other LUs are omitted for simplicity.

4.3 Lexically Constrained Decoding (LCD)

Given a sequence of frame ID tokens $F_1, F_2, ..., F_n$, we build a corresponding sequence of disjunctive lexical constraint sets $C_1, C_2, ..., C_n$, where C_i consists of all LUs of F_i with their morphologi-

[3] Our work focuses primarily on infilling single sentence spans, leaving arbitrary length spans, e.g. n-grams or full paragraphs, to the future work.

[4] We choose to evaluate these three variants in order to compare the coherence of a model trained with only low frame guidance (S-FFL), a model trained with only high (A-FFL), and a model (M-FFL) trained on a distribution of examples that comprises a superset of the first two.

cal variants. During decoding, our method forces the output to contain $c_1, c_2, ...c_n$, where $c_i \in C_i$.

Decoding with Ordered/Unordered Disjunctive Constraint Sets We develop a disjunctive lexically constrained decoding method (**LCD**) that extends implementations in Post and Vilar (2018); Hu et al. (2019a) and Li et al. (2020b). We also use Dynamic Beam Allocation (DBA) (Post and Vilar, 2018; Hu et al., 2019a) for beam assignment and next token selection, but we track our constraints differently. As shown in Figure 3, LCD represents a sequence of disjunctive constraint sets as a list of tries, one per frame, each covering a set of disjunctive lexical units (with morphological variants) based on the Byte Pair Encoding (BPE, Sennrich et al., 2016) adopted by GPT-2.

Based on this representation, we develop two versions of LCD: LCD-ordered and -unordered, the former of which requires that the constraint sets be completed in the order that the corresponding frame ID tokens are specified. By providing these two versions, we offer the user the flexibility to either enforce the frame-evoking narration being triggered in their desire order, or leave it to be determined by the generative model and decoder.

To track the generation progress through constraint sets, we use a global pointer to the currently active disjunctive set. Whenever the active set C_i is completed, the pointer is set to null. If unsatisfied sets remain, the next possible set(s) to be completed is C_{i+1} for LCD-ordered and $\{C_j : j \neq i \in \{1, 2, ..., n\} \mid C_j \text{ is not completed}\}$ for LCD-unordered. At the beginning of generation when no set is active, the next possible set(s) is C_0 for LCD-ordered and all sets for LCD-unordered. During the generation, when the pointer is null and a constraint token that starts any of the next possible set(s) is picked by DBA, the global pointer is set to the corresponding disjunctive set. Apart from the global pointer, the bookkeeping and unwinding mechanism within each trie is similar to the implementations in (Hu et al., 2019a) and (Li et al., 2020b), except that a trie is marked as finished and the global pointer is updated once any path in the trie is completed.

We implement LCD as an extension of the token generation constraint implementation in the `fairseq` library. Our LCD works very similarly to the disjunctive positive constraints decoding in (Li et al., 2020b), where the disjunctive sets are maintained in a single trie rather than our "list of tries" approach. However, we support explicit ordering of constraint sets, and we don't prune a sub-trie when the corresponding constraint set is finished.

5 Experiments

We test the effectiveness of our models on a frame-guided sentence infilling task derived from ROCStories. We use a state-of-the-art neural FrameNet parser (Xia et al., 2021) to obtain the set of frames evoked by each sentence in the dataset. We then present models with a five-sentence ROC story with one masked out. The model must infill the missing sentence given one or many frame ID tokens parsed from the masked-out sentence. For evaluations requiring generated outputs (all but perplexity), we use beam search with beam size 20. We find that beam search achieves higher frame fidelity and coherence than the random sampling approach used by Donahue et al. (2020).

We train our models (all GPT-2 'base') using the provided train split of ROCStories. For S/M/A-FFL, each example contains one/multiple/all frame ID tokens sampled randomly from the parser output. To test LCD, we re-train the original ILM using the identical ROCStories training data to our FFL models but without frame tokens (training details described in A.2). Unlike Donahue et al. (2020), we do not include story titles. We also use this ILM as a baseline with no guidance.

To investigate whether enforcing generated frame order impacts model performance, we evaluate both LCD-ordered and -unordered; we also evaluate FFL-ordered models fine-tuned to generate frames in the order in which they are provided.

5.1 Automatic Evaluation

We evaluate our frame-guided generation methods by measuring the rate at which they produce sentences that trigger the desired frame(s) and by measuring the perplexity score of the framefilling-trained language model on test examples.

Frame Fidelity We automatically evaluate whether a produced sequence triggers a given set of frames by running it through the same neural frame parser used to determine the desired frame from a gold human-generated sentence. Table 1 shows the rates at which methods correctly produce sentences that contain every specified frame.[5] For each

[5]Methods that condition on fewer frame IDs are evaluated using subsets of those for multi-frame models; e.g. if the

Fidelity ↑	Recall			Perfect Recall	
# Frames	Single	Multi	All	Multi	All
ILM (no guidance)	.169	.166	.165	.091	.026
ILM + LCD	**.584**	.595	.610	.418	.232
ILM + LCD-ord	–	**.598**	.626	**.427**	.255
FFL	.518	.559	.640	.381	.259
FFL (rand sample)	.461	.511	.601	.338	.224
FFL-ord	–	.585	**.669**	.415	**.298**

Table 1: Frame fidelity, computed as frame recall according to the neural frame parser (left). The per-example rate at which models perfectly predict frame sets is also given (right). Higher is better.

model, we evaluate the top-1 decoded sequence.

Perplexity The typical automatic evaluation metric for a language model is test data perplexity (PPL). Since LCD requires no training modification to the ILM model, we only compute the PPL for S/M/A-FFL and ILM on a test set of stories in which one of five sentences has been masked out.[6] Following Donahue et al. (2020), we evaluate models' PPL specifically on infill tokens and also compute PPL including the surrounding special tokens (separators and frame IDs). Because sequences for FFL models include one or more frame ID tokens, the token length for a given story example is different for ILM and each FFL variant; PPL therefore cannot be directly compared. To construct a scenario in which the ILM and FFL model perplexities *are* directly comparable, we train variants of both models for which every infill sequence is prepended with 5 special tokens, thus regularizing token length for every evaluated model.

5.2 Human Evaluation

In addition to automatic evaluation, we collect human judgements to assess models' ability to maintain coherent and plausible generation. We conduct two human evaluations that ask annotators to tell apart model- and human-generated sentences (Indistinguishability task) and rank model-generated sentences relative to one another (Relative Plausibility task). Details of our collection protocols and example interfaces are provided in Appendix D.

Indistinguishability Following Donahue et al., we present annotators with 5-sentence stories in which one sentence has been replaced by the output of an infilling model. Annotators must identify which sentence is model-generated.

For each model, we calculate the confusion rate $r = \frac{N_{confused}}{N_{all}}$, where $N_{confused}$ is the number of stories for which a human annotator fails to identify the machine-generated content, and N_{all} is the total number of stories. Results are shown in Table 3. Higher confusion rate is posited to mean more natural text infilling. Optimal performance is 80%, meaning the annotator is performing at chance.

Perplexity ↓	ILM	S-FFL	M-FFL	A-FFL	A-FFL (ord)
Infill Text	12.85	11.7	9.84	6.19	**5.05**
+ Sp Toks	7.24	8.32	9.5	8.95	**7.04**
w/ 5 Fr Slots	**4.06**	5.12	6.34	7.32	6.03

Table 2: Model perplexity over infill text tokens and infill text tokens + special tokens (<start to infill>, <end of infill>, <infill mask>). Lower is better.

Relative Plausibility We present human annotators with a 5-sentence story where one sentence is missing, and 10 candidate replacement sentences (the gold plus the infills of 9 different models). Annotators are tasked with ranking the candidate sentences (via drag-and-drop) based on how plausible they are relative to each other. Upon aggregating judgements, each model's score is calculated as the average relative rank of its output sentences that are assigned by annotators, as shown in Table 4.

6 Analysis

Fidelity From the results in Table 1, we find that ILM+LCD, FFL and FFL-ordered all perform similarly while substantially outperforming the baseline unguided ILM. This shows that our methods effectively produce text evoking the desired frame semantic content. Both methods benefit from the inclusion of gold frame order, more so for FFL.

There is a considerable gap between the performance of our models and perfect performance (1.0). This is because FFL operates only with soft "control code" constraints, and although LCD is strictly required to generate trigger LUs for every frame, it does not produce sentences that always successfully evoke the frame. While some of this gap might be the result of imperfections of the parser, we find word sense ambiguity to be a contributing problem. Many LUs, such as *work.v*, *see.v*, or *call.v* have multiple senses each associated with a different frame. Since neither LCD nor FFL imposes hard constraints on word sense, it is entirely

M-FFL model must generate a set **{[Food] [Size]}**, the S-FFL must predict one of **[Food]** or **[Size]**.

[6] See Appendix C for model perplexity trained and evaluated with all five sentences having been masked.

Confusion rate (%) ↑	# Frames		
	Single	Multi	All
ILM (no guidance)	**41**	**41**	**41**
ILM+LCD	35	31*	20*
FFL	33*	39	38
FFL-ordered	33*	38	37

Table 3: Confusion rate computed as the percentage of stories for which the annotator picks a wrong sentence as machine-generated. Higher is better. * denotes significant difference from the baseline (ILM) result, according to the two-sided McNemar test with $p < 0.05$.

Average rank (1..10) ↓	# Frames		
	Single	Multi	All
ILM (no guidance)	**5.48**	**5.48**	5.48
ILM+LCD	5.85*	6.38*	7.50*
FFL	5.88*	5.57	5.11
FFL-ordered	5.88*	5.53	**5.02***

Table 4: Average relative plausibility rank by human annotators. Lower is better. * denotes significant difference from the baseline (ILM) result, according to the two-sided Wilcoxon signed-rank test with $p < 0.05$.

Story
· · · I danced terribly and broke a friend's toe. [blank]
Frame: [Request]

Gold [Request]
The next weekend, I was **asked** to please stay home.

ILM+LCD [Contacting]
I went home to **call** my friend and tell her I broke her toe.

Figure 4: From the lexical constraints on the LUs of the frame **Request** (as *asked.v* in the gold infill), the decoder selects *call.v*, but in the generated context *call* becomes a surfaces realization of frame **Contacting**.

possible for an unintended sense to be generated.

As illustrated in Figure 4, LCD forces picking the LU *call.v* for the target frame **Request**, but given the subsequent output *call my friend to tell her I was hurt*, the *call.v* unit takes on a sense that triggers the incorrect frame **Contacting**.

Perplexity Table 2 shows that the perplexity over purely the infill tokens is inversely proportional to the amount of frame guidance provided to the language model. However, we find that under the directly comparable 5 slot scenario, PPL computed over the infill tokens plus all surrounding special/frame tokens is *worse* for models with more frame tokens. As this work is predominantly concerned with the quality of generation given gold frame IDs, this is less of a concern; that the perplexity of infill tokens decreases considerably with the introduction of frame guidance shows that neural language models can be explicitly guided towards specific semantic spaces in accordance with the conceptual semantic structures underpinning human understanding of language.

Generation Quality Table 4 shows that in terms of human-judged relative plausibility, FFL outperforms all other models (including the unconstrained ILM) when conditioning on all frames, and underperforms ILM with only a small margin with multi-frame guidance. Table 3 shows that ILM outperforms FFL models and LCD on the Indistinguishability task in all cases, but with only a small margin in multi/all-frame cases comparing with FFLs. This is unsurprising, as ILM is optimized to replicate human-produced text under no constraints via semantic guidance. We observe as in the fidelity evaluation that LCD slightly outperforms FFL under single frame constraints in both human evaluations. From these results we can conclude that in the process of achieving effective controlled frame-guided language generation, the fine-tuned FFL model achieves competitive performance to its unconstrained ILM counterpart, especially in the presence of increased guiding information. Moreover, the compromise in quality for the LCD method is minimal particularly for single frame guidance.

Effect of Different Levels of Guidance Table 3 and Table 4 show that as the level of guidance (number of frames provided) increases, FFL and LCD models show opposite trends in quality: the former improves whereas the latter gets worse. We illustrate this effect in Figure 5.

For FFL, this indicates that generative capabilities would improve if the model were trained with more information about semantic content. This is a somewhat counterintuitive finding, given the effectiveness of the ILM model trained with no semantic information whatsoever beyond surface-level lexical information (words in the context).

For LCD, we posit that the increase in the size of lexical unit constraint sets amplifies the negative effects of the lexical units' word sense ambiguity, resulting in the downward trend. With more guiding frames, LCD has to search through a larger space of possible LU combinations and is therefore more prone to the misuse of LU (sense). More-

Story	**FFL**	**ILM+LCD**
Ari spends $20 a day on pickles. He decides to make his own to save money. He puts the pickles in brine. [blank] Ari opens the jar to find perfect pickles.	**Single Frame:** [Transition_to_State]	
Gold Ari waits 2 weeks for his pickles to get sour.	He ends up with a jar full of pickles.	He gets the pickles and puts them in jars.
ILM Baseline He puts the pickles in a jar.	**Multiple Frames:** [Cardinal_Numbers] [Transition_to_State]	
	He ends up with 5 jars of pickles.	He puts one in the jar and opens it to get a drink.
	All Frames: [Cardinal_Numbers] [Measure_duration] [Transition_to_State][Chemical-sense_description]	
	He waits for a week for the pickles to get sour.	He waits for the pickles to thaw out of the jar to thaw one day he gets the pickles and eats them delicious.

Figure 5: Example infills by FFL, LCD and ILM baseline under single, multiple, and all frame guidance. Under single frame guidance, all decoding methods perform interchangeably. As the number of frames increases, FFL approaches a surface realization of frame-specified semantic content that resembles that of the gold infill. The unguided baseline ILM generates something relatively incoherent. Under "all frame" guidance, LCD fails to satisfy all constraints in one sentence and generates an additional sentence that corrupts quality.

over, we observe that in some cases with many (e.g. $\geq$ 5) frames, LCD cannot satisfy all constraints within one sentence and will start new sentences to complete unmet constraints. This is likely a contributing factor to LCD's lower scores under human evaluations.

7 Case Study: Interactive Generation

To demonstrate the practical applicability of our frame-guided infilling methods, we qualitatively explore them in a variety of human-in-the-loop use cases based on recent work in text generation. In the following cases, we use models for both frame ID inference and text infilling conditioned on surrounding context. For frame inference, we use the forward frame token probability of an unordered-frame M-FFL model trained as in Section 3, with the modification that training examples have between 0 and 4 surrounding sentences as context. This allows for more flexibilty than a model trained only on complete 5-sentence stories. We modify the training data by taking a random contiguous slice of each 5-sentence example. Figure 6 shows examples of each scenario. For infilling, we use FFL for **A**, **B** and **D** and LCD for **C**.

A. Iterative Story Refinement For a maximally free-form and extensible use case, we devise a scenario in the spirit of Goldfarb-Tarrant et al. (2019) in which a user interfaces with a model to collaboratively construct an open-domain story given any combination of text and/or frames. Over the course of a human-system dialog, the user can iteratively either choose for the model to predict new frames at specified locations in the context or select from candidate infills conditioned on selected frames. As discussed in Goldfarb-Tarrant et al. (2019), this type of process allows for a symbiotic relationship in which the user can correct, suggest or revise content generated by the machine and vice versa. Injecting frame guidance into this scenario enables for an extra degree of interactive flexibility in both suggestion and specification.

B. Generation from Story Skeleton Recent work (Fan et al., 2018; Goldfarb-Tarrant et al., 2019) has used pretrained neural language models for surface realization of structured story content. We approximate this task by having a model accept a seed sentence (i.e. a prompt) plus an ordered sequence of sets of frames specifying the content to appear in a story. We then use the frame-guided conditional generation to complete the text. Without the ability to handle explicit frame semantic guidance, this task would be incredibly difficult for a neural generation model.

C. Diverse Candidate Generation Weir et al. (2020) explore the task of diverse causal generation, in which a model must propose a set of semantically distinct causes or effects of an input sentence. Following their two-step approach, we devise a frame semantic model that 1) predicts the distinct frames that are likely to appear at a specified index before (for causes) or after (effects) the input sentence, then 2) run a separate beam search conditioned on each top-k predicted

A. Iterative Refinement

User: (I) Alice went to the grocery store. (II) [Commerce_buy]
System: (IIA) She bought all the ingredients for a cake. (IIB) She bought a new pair of shoes. (IIC) She bought a lot of fruits and veggies.
U: Choose (IIA) as (II) and infer content after (II)
S: [Food], [Cooking], [Ingredients], [Desirability], [Time_Collocation]
U: Choose [Desirability] [Cooking]
S: (I) Alice went to the grocery store. (II) She bought all the ingredients for a cake. *(III) She made the best cake she ever had.*
U: Infer content to replace (I)
S: [Food], [Deciding], [Social_Event], [Building], [Quantity]
U: choose [Social_Event]
S: *(I) Mary wanted to make a cake for her birthday.* (II) She bought all the ingredients for a cake. (III) She made the best cake she ever had.
U: Insert sentence about [Motion] at (II) and sentence about [Temporal_Collocation] at (IV)
S: (I) Mary wanted to make a cake for her birthday. *(II) She went to the store.* (III) She bought all the ingredients for a cake. *(IV) That afternoon, she baked the cake in the oven.* (V) She made the best cake she ever had.

B. Generation from Skeleton

User: (I) Bob went to Paris. (II) [Buildings] (III) [Experience] (IV) [Motion] 5 [Calendric_Unit]
System: (I) Bob went to Paris. *(II) He went to see the Louvre museum. (III) He saw lots of paintings and sculptures. (IV) He went to see the Eiffel Tower. (V) It was the best day of Bob's life.*

U: (I) Emma loved writing and wanted to be a writer. (II) [Deciding] (III) [Practice] (IV) [Publishing][Text] (V) [Fame]
S: (I) Emma loved writing and wanted to be a writer. *(II) She decided to enter a writing contest. (III) Emma's practice paid off and she won first prize. (IV) Emma was so proud of herself that she published a book. (V) Now Emma is a famous author.*

C. Diverse Candidate Generation

User: Infer candidates after (I) Charles slipped on a banana peel.
System: (IIA [Body_Parts]) He skinned his knee badly. (IIB [Motion_Directional]) He fell down on the ground. (IIC [Spatial_Contact]) He landed on his arm and broke it. (IID [Kinship]) His mother brought him to the emergency room. (IIE [Causation]) It caused him to fall down the stairs.

U: Infer candidates before (II) She hired a lawyer.
S: (IA [Submitting_Documents]) She filed a civil lawsuit. (IB [Trial]) She sued for breach of contract. (IC [Personal_Relationship]) She filed for divorce. (ID [Awareness]) She didn't know how to defend herself. (IE [Desiring]) she did not want to go to jail.

D. Counterfactual Story Rewriting

User: (I) Alec's daughter wanted more blocks to play with. (II) Alec figured that blocks would develop her scientific mind. (III) Alec bought blocks with letters on them. (IV) Alec's daughter made words with them rather than structures. (V) Alec was happy to see her developing her verbal ability.
Replace (II) with "Alec could not afford to buy new blocks for his daughter" and rewrite the last three sentences.
Parser: (III) [Containers] (IV) [Text_Creation] (V) [Emotion_directed]
System: (I) Alec's daughter wanted more blocks to play with. (II) Alec could not afford to buy new blocks for his daughter. *(III) Alec's daughter begged him to buy her blocks. (IV) Alec wrote a letter to Santa Claus himself. (V) She was very happy when he wrote back.*

Figure 6: Example use cases of frame-guided infilling. **A.** depicts *human-in-the-loop iterative story refinement*, in which a user provides an initial context and/or intended frame semantic content and interacts with the model to predict and user-select new frame content and surface-realized context. **B.** depicts surface realization from a *frame semantic story skeleton*, i.e. a seed sentence and a sequence of frame sets to appear in the specified order. **C.** depicts *semantically diverse candidate generation* using model frame inference to identify distinct semantic content then using conditional generation to realize each candidate. **D.** depicts *counterfactual story revision*, in which one sentence (II) is replaced and subsequent sentences are rewritten using frames parsed from the originals.

frame. Using a frame-infused generation model for this purpose leverages the hierarchical semantic delineations contained within FrameNet, selecting human-interpretable semantic spaces from which to generate content. This is compared to other methods for diverse sampling, such as random and nucleus sampling (Holtzman et al., 2020), in which there is no notion of higher level semantic reasoning and a tendency to hallucinate content, or COD3S (Weir et al., 2020), which enables only moderate interpretability not based–as FrameNet is–in cognitive theories of semantic organization.

D. Counterfactual Story Revision Qin et al. (2019) introduce the task of generative counterfactual reasoning in narratives. Given an original story and a counterfactual event (i.e. the replacement of one original sentence), the task is to minimally revise the rest of the story according to the counterfactual replacement. We devise a frame semantic model for this task that 1) parses the frames of sentences following the replacement and 2) conditions the generation model on the replacement text and a sampled sequence of the parsed frames so as to produce a revised story whose frame semantics are similar to the original's. While previous approaches to this generation task condition only on surrounding context, our frame-injected model allows for explicit retention of semantic spaces.

8 Conclusion

We propose the application of frame semantics in the context of controlled text generation. We in-

troduce two extensions of neural text generation that leverage FrameNet frames as guiding signals: 1) model fine-tuning with a frame-guided infilling objective; and 2) disjunctive lexically constrained decoding with frame-associated lexical units. Experimental results on a sentence infilling task and the case study involving an interactive story generation setup show that both of our methods can properly leverage the frame information to trigger surface realization of frame semantic content. Our results show that our methods enable explicit manipulation of semantics at the frame level with competitive generation quality, and we exhibit a variety of use cases that enable new dimensions of user guidance on generation.

Acknowledgments

This work was supported by DARPA KAIROS and NSF grant no. BCS-2020969. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes. The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsement.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Gang Chen, Yang Liu, Huanbo Luan, Meng Zhang, Qun Liu, and Maosong Sun. 2019. Learning to predict explainable plots for neural story generation. *arXiv preprint arXiv:1912.02395*.

Guanhua Chen, Yun Chen, Yong Wang, and Victor O. K. Li. 2020. Lexical-constraint-aware neural machine translation via data augmentation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3587–3593. ijcai.org.

Ryan Culkin, J. Hu, Elias Stengel-Eskin, Guanghui Qin, and Benjamin Van Durme. 2020. Iterative paraphrastic augmentation with discriminative span alignment. *arXiv preprint arXiv:2007.00320*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2650–2660, Florence, Italy. Association for Computational Linguistics.

Francis Ferraro and Benjamin Van Durme. 2016. A unified bayesian model of scripts, frames and language. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2601–2607. AAAI Press.

Charles J Fillmore. 1976. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech*, volume 280, pages 20–32. New York.

Charles J Fillmore and Collin Baker. 2010. A frames approach to semantic analysis. In *The Oxford handbook of linguistic analysis*.

Seraphina Goldfarb-Tarrant, Tuhin Chakrabarty, Ralph Weischedel, and Nanyun Peng. 2020. Content planning for neural story generation with aristotelian rescoring. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4319–4338, Online. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Haining Feng, and Nanyun Peng. 2019. Plan, write, and revise: an interactive system for open-domain story generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 89–97, Minneapolis, Minnesota. Association for Computational Linguistics.

Prakhar Gupta, Vinayshekhar Bannihatti Kumar, Mukul Bhutani, and Alan W Black. 2019. WriterForcing: Generating more interesting story endings. In *Proceedings of the Second Workshop on Storytelling*, pages 117–126, Florence, Italy. Association for Computational Linguistics.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1535–1546, Vancouver, Canada. Association for Computational Linguistics.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. *International Conference on Learning Representations*.

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019a. Improved lexically constrained decoding for translation and monolingual rewriting. 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 839–850, Minneapolis, Minnesota. Association for Computational Linguistics.

J. Edward Hu, Rachel Rudinger, Matt Post, and Benjamin Van Durme. 2019b. PARABANK: monolingual bitext generation and sentential paraphrasing via lexically-constrained neural machine translation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6521–6528. AAAI Press.

Daphne Ippolito, David Grangier, Douglas Eck, and Chris Callison-Burch. 2020. Toward better storylines with sentence-level language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7472–7478, Online. Association for Computational Linguistics.

Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. Story generation from sequence of independent short descriptions. *arXiv preprint arXiv:1707.05501*.

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

Huayang Li, Guoping Huang, and L. Liu. 2020a. Neural machine translation with noisy lexical constraints. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1864–1874.

Ya Li, Xinyu Liu, Dan Liu, Xueqiang Zhang, and J. Liu. 2019. Learning efficient lexically-constrained neural machine translation with external memory. *arXiv preprint arXiv:1901.11344*.

Zhongyang Li, Xiao Ding, Ting Liu, J. Edward Hu, and Benjamin Van Durme. 2020b. Guided generation of cause and effect. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3629–3636. ijcai.org.

Dayiheng Liu, Jie Fu, Q. Qu, and J. Lv. 2019. Bfgan: Backward and forward generative adversarial networks for lexically constrained sentence generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Neurologic decoding: (un)supervised neural text generation with predicate logic constraints. *arXiv preprint arXiv:2010.12884*.

Yuning Mao, X. Ren, Huai zhong Ji, and Jiawei Han. 2020. Constrained abstractive summarization: Preserving factual consistency with constrained generation. *arXiv preprint arXiv:2010.12723*.

Lara J. Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O. Riedl. 2018. Event representations for automated story generation with deep neural nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 868–875. AAAI Press.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: constrained sentence generation by metropolis-hastings sampling. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6834–6842. AAAI Press.

Marvin Minsky. 1974. A framework for representing knowledge.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of*

the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1314–1324, New Orleans, Louisiana. Association for Computational Linguistics.

Lianhui Qin, Antoine Bosselut, Ari Holtzman, Chandra Bhagavatula, Elizabeth Clark, and Yejin Choi. 2019. Counterfactual story reasoning and generation. 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 5043–5053, Hong Kong, China. Association for Computational Linguistics.

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

Hannah Rashkin, Asli Celikyilmaz, Yejin Choi, and Jianfeng Gao. 2020. PlotMachines: Outline-conditioned generation with dynamic plot state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4274–4295, Online. Association for Computational Linguistics.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703, Online. Association for Computational Linguistics.

Raymond Hendy Susanto, Shamil Chollampatt, and Liling Tan. 2020. Lexically constrained neural machine translation with Levenshtein transformer. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3536–3543, Online. Association for Computational Linguistics.

Nathaniel Weir, João Sedoc, and Benjamin Van Durme. 2020. COD3S: Diverse generation with discrete semantic signatures. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5199–5211, Online. Association for Computational Linguistics.

Patrick Xia, Guanghui Qin, Siddharth Vashishtha, Yunmo Chen, Tongfei Chen, Chandler May, Craig Harman, Kyle Rawlins, Aaron Steven White, and Benjamin Van Durme. 2021. LOME: Large ontology multilingual extraction. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pages 149–159, Online. Association for Computational Linguistics.

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4306–4315, Brussels, Belgium. Association for Computational Linguistics.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *AAAI*.

Jiacheng Zhang, Huanbo Luan, Maosong Sun, Feifei Zhai, Jingfang Xu, and Yang Liu. 2021. Neural machine translation with explicit phrase alignment. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1001–1010.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020. POINTER: Constrained progressive text generation via insertion-based generative pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8649–8670, Online. Association for Computational Linguistics.

A Training Details

A.1 FFL

We finetune GPT-2 on examples of frame-guided infilling using the same training parameters (to the extend possible) as Donahue et al. (2020). We use the `fairseq` library to perform training and inference using the pretrained GPT-2 parameters provided by HuggingFace[7]. Training takes 1.5 hours using 8 Quadro RTX 6000 GPUs.

```
fairseq-train
--task framefilling
--sample-break-mode eos
--arch ilm_gpt2
--dropout 0.1
--attention-dropout 0.1
--clip-norm 1
-optimizer adam --adam-eps 1e-08
--lr 5e-5
--weight-decay 0.0
--max-epoch 100
--patience 3
```

A.2 ILM

To compare ILM with FFL on a uniform basis, we retrain ILM on sentence level infilling using the code provided by Donahue et al. (2020),[8] with same parameters and stopping criterion.

It is worth noticing that the original ILM is trained on stories from the ROCStories dataset with titles provided. However, the test set portion of ROCStories on which we formulate the frame-guided sentence infilling task are provided without title. We observe that the original ILM trained with title is problematic in infilling the first sentence of a story without title (Sometimes it outputs full stop only, or generate a new title in addition to the sentence). Therefore, we delete all titles in the training data when retraining ILM.

B LCD Diversification

Although the LCD algorithm will explore the prefix of each of the dozens of constraints typically associated with a frame, a few LUs will tend to dominate the final candidates throughout beam search — this is also observed in Li et al. (2020b). This problem is exacerbated by the rather broad definitions of some frames that cover both general, common LUs, and more specific LUs, whose likelihood will be dwarfed during decoding by the former. For example, the **Collaboration** frame contains LUs that depict the concept of collaboration from various perspectives: the act of collaborating (e.g. *collaborate.v*, *team up.v*), the participants in the collaboration (e.g. *collaborator.n*, *partner.n*), and the state of being in collaboration (e.g. *in cahoots.a*, *together.adv*), etc. However, in practice the general unit *together.adv* is more often selected by beam search to satisfy the constraint because of its generally higher likelihood. This dominant LU prevents other potentially diverse surface realizations of the frame triggered by other LUs.

To improve the lexical and semantic diversity in triggering frames, we construct disjunctive sets on a more fine-grained semantic level. We divide each set of LUs into k subsets using hierarchical clustering over the GloVe embeddings of LUs (Pennington et al., 2014). In particular, we use the `AgglomerativeClustering` class of **scikit-learn** [9] to perform hierarchical clustering over the GloVe embedding of LUs to divide each set of frame-associate LUs into subsets. In the experiments, we set number of clusters to 8. For multi-frame constraints, we set number of clusters to 4 for the frame with the most number of LUs and 2 for the frame with the second most of, we do not divide any LU sets for remaining frames (if any), this could ensure the total combination of multi-frame LU subsets equals 8. Figure 7 shows the clustering results of three frames: **Collaboration**, **Ingestion** and **Departing**, with number of clusters set to 4.

To ensure that the decoder will be able to explore all possible combinations of LUs, we build lists of tries for every combination of LU subsets. The constrained beam search is then run separately on each of them. To ensure that candidates from each LU subset are considered, final candidates are selected in a round-robin manner: the top-1 scored hypothesis is picked for each subset, followed by the top-2, and so on.

C Perplexity

We repeat the perplexity experiment from subsection 5.1, but instead of masking one out of five of a story's sentences at a time, we mask all five. This

[7] https://github.com/pytorch/fairseq/blob/master/fairseq/models/huggingface/hf_gpt2.py

[8] https://github.com/chrisdonahue/ilm

[9] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

Frames	LUs clusters
Collaboration	**cluster 1**: conspire, conspiracy, collusion, collude **cluster 2**: together, in league, in cahoots, work together, team up **cluster 3**: confederate **cluster 4**: partner, jointly, cooperation, associate, affiliated, collaboration, collaborator, cooperate, collaborate
Ingestion	**cluster 1**: have, put away, lap, put back, down **cluster 2**: feed, lunch, breakfast, snack, eat, drink **cluster 3**: swig, ingestion, quaff, swill, guzzle, sup, nosh, gulp, devour, gobble, ingest, consume, dine, nibble, imbibe, slurp, sip **cluster 4**: tuck, munch, feast, nurse
Departing	**cluster 1**: departure, depart, exit, leave **cluster 2**: vamoose, decamp, skedaddle **cluster 3**: exodus, disappearance, escape **cluster 4**: disappear, vanish, emerge

Figure 7: clustering examples of frame **Collaboration**, **Ingestion**, and **Departing**, morphological variants are excluded for demonstration purpose.

scenario can be considered a fully generative model of text in which no context is provided except for frame IDs specifying general semantic content for each sentence. Table 5 shows the resulting model perplexities.

D Human Evaluation Details

Akin to Donahue et al. (2020), we sampled 100 stories from the test set of the ROCStories dataset. Masking one sentence at a time in each 5-sentence story, we obtained 500 masked stories. Each model was then tasked to infill a missing sentence in a masked story. We compared 10 models in total: 8 proposed in this paper (S/M/A-FFL, M/A-FFL-ordered, and the ordered variant[10] of S/M/A ILM+LCD), as well as the gold human infill and the ILM model. Below we further specify the details of each of the human evaluation tasks.

D.1 Indistinguishability

To achieve high comparability with Donahue et al. (2020), we conducted this evaluation as a Human Intelligence Task (HIT) on Amazon Mechanical Turk. To filter out malicious workers, we used a control model which always generates "This sentence was generated by a machine." or a synonymous sentence. We also validated that the gold human infill achieves 80% confusion rate (which was attained precisely in our run), which corresponds to picking 1 sentence out of 5 at random. Overall, 12 workers participated in the HIT, of which one was filtered by the control model. The annotator's interface can be seen on Figure 8.

[10]Based on the Frame Fidelity and the pilot HIT results, we chose to only evaluate the ordered variant, as the unordered LCD performed very similarly in terms of those metrics.

Perplexity	ILM	S-FFL	M-FFL	A-FFL	A-FFL (ord)
Infill Text	13.88	11.07	8.76	5.45	**4.69**
+ Sp Toks	8.87	9.16	10.05	9.3	**7.43**
+ 5 Fr Slots	**4.66**	5.51	6.71	7.64	6.23

Table 5: Model perplexity over infill text tokens and infill text tokens + special tokens with all 5 ROCStory sentences masked out.

D.2 Relative Plausbility

Due to a relatively high complexity of this task, compared to the Indistinguishability task, the evaluation was conducted with a team of skilled annotators, comprised of four undergraduate students who have previously participated in NLP/AI annotation projects. On average, ranking 10 models' outputs for one story took 3 minutes 19 seconds for each worker. The annotator's interface can be seen on Figure 9.

You will be presented with **six** short stories, each comprised of **five** sentences.

For each story, please select **one** sentence out of **five** which you think was **the most likely** generated by a machine.

Note that some stories will appear multiple times. When you see a story re-appear, please answer **the same way** as in the previous response.

I. Identify one of the five sentences generated by a machine:

1. ○ Jill had been having a problem with mice in her home.
2. ○ She went to the animal shelter and adopted a cat.
3. ○ The cat turned out to be timid.
4. ○ Jill no longer had a mouse problem.
5. ○ She loved having her cat around.

II. Identify one of the five sentences generated by a machine:

1. ○ Jill had been having a problem with mice in her home.
2. ○ She went to the animal shelter and adopted a cat.
3. ○ Jill's cat took care of the mice.
4. ○ Jill no longer had a mouse problem.
5. ○ She loved having her cat around.

III. Identify one of the five sentences generated by a machine:

1. ○ Jill had been having a problem with mice in her home.
2. ○ She went to the animal shelter and adopted a cat.
3. ○ The cat turned out to be a dog.
4. ○ Jill no longer had a mouse problem.
5. ○ She loved having her cat around.

Figure 8: Interface shown to the workers during collection of indistinguishability judgments.

Given the story, please estimate how plausible the candidate sentences are relative to one another.

- Sort the candidates by relative plausibilities, from **most plausible** to **least plausible**, by drag-and-dropping the candidates into their respective order.
- After sorting the candidates, indicate the **relative plausibilities** using the vertical bar.
- Set the bar pin's value relative to the fixed candidates at the top and bottom of the bar (i.e. the most and least plausible candidates, respectively).
  - To set the plausibility for a particular candidate, select that candidate before clicking the bar.
  - Note that the bar pin does not appear until you have selected the value for the candidate for the first time.
  - Note that the bar pin for a given candidate cannot be moved past the two adjacent candidates.
- To switch the ranking of the candidates after the vertical bar appears, continue using the drag-and-drop feature.

Please calibrate your score for a specific candidate with respect to the scores you've given for other candidates in the same HIT (i.e. not relative to 0% and 100%).

Story: Jill had been having a problem with mice in her home. She went to the animal shelter and adopted a cat. **<X>**. Jill no longer had a mouse problem. She loved having her cat around.

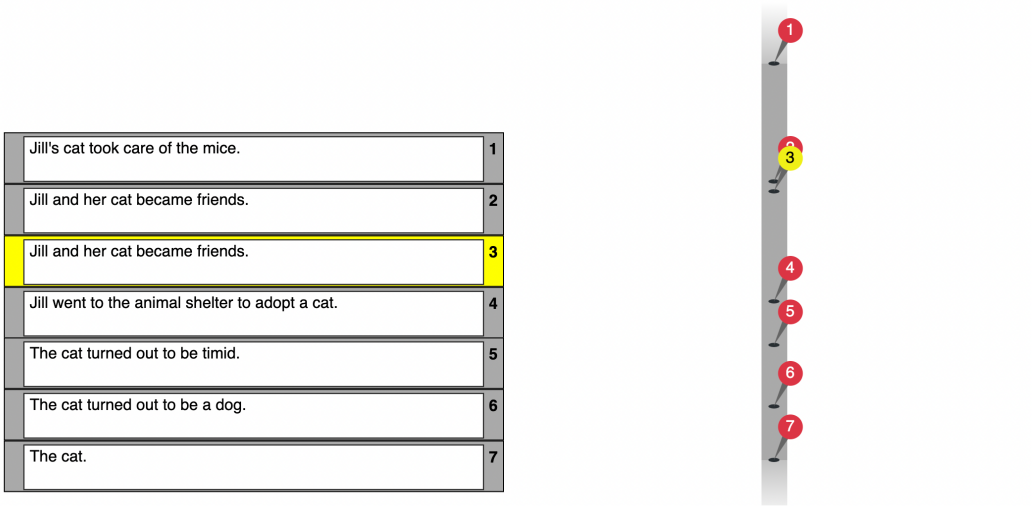

Figure 9: Interface shown to the workers during collection of relative plausibility judgments.