NeuroLogic A$^\bigstar$esque Decoding: Constrained Text Generation with Lookahead Heuristics

Ximing Lu[‡†] [♡]Sean Welleck[†‡] [♡]Peter West[†]
Liwei Jiang[‡†] Jungo Kasai[‡†] Daniel Khashabi[‡] Ronan Le Bras[‡]
Lianhui Qin[†] Youngjae Yu[‡] Rowan Zellers[†] Noah A. Smith[†‡] Yejin Choi[†‡]
[‡]Allen Institute for Artificial Intelligence
[†]Paul G. Allen School of Computer Science & Engineering, University of Washington

Abstract

The dominant paradigm for neural text generation is left-to-right decoding from autoregressive language models. Constrained or controllable generation under complex lexical constraints, however, requires foresight to plan ahead for feasible future paths.

Drawing inspiration from the A* search algorithm, we propose NeuroLogic A$^\bigstar$esque,[1] a decoding algorithm that incorporates heuristic estimates of future cost. We develop *lookahead* heuristics that are efficient for large-scale language models, making our method a drop-in replacement for common techniques such as beam search and top-k sampling. To enable constrained generation, we build on NeuroLogic decoding (Lu et al., 2021), combining its flexibility in incorporating logical constraints with A$^\bigstar$esque estimates of future constraint satisfaction.

Our approach outperforms competitive baselines on five generation tasks, and achieves new state-of-the-art performance on table-to-text generation, constrained machine translation, and keyword-constrained generation. The improvements are particularly notable on tasks that require complex constraint satisfaction or in few-shot or zero-shot settings. NeuroLogic A$^\bigstar$esque illustrates the power of decoding for improving and enabling new capabilities of large-scale language models.

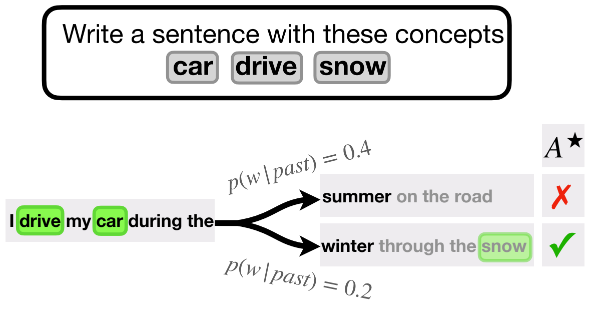


Figure 1: NeuroLogic$^\bigstar$ leverages *lookahead heuristics* to guide generations towards those that satisfy the given task-specific constraints. In this example from the CommonGen task, although **summer** is a *more likely* next word given the already-generated *past*, NeuroLogic$^\bigstar$ looks ahead to see that selecting **winter** results in a generation that incorporates unsatisfied constraint **snow** with a higher probability later on. Thus, **winter** is preferred despite being lower probability than **summer**.

1 Introduction

The dominant paradigm for neural text generation is based on left-to-right decoding from autoregressive language models such as GPT-2/3 (Radford et al., 2019; Brown et al., 2020). Under this paradigm, common decoding techniques such as beam search or top-k/p sampling (Holtzman et al., 2020) determine which token to generate next based on what happened in the past, without explicitly looking ahead into the future. While this lack of foresight often suffices for open-ended text generation – where any coherent text can be acceptable – for constrained text generation, planning ahead is crucial for incorporating all desired content in the generated output (Hu et al., 2017; Dathathri et al., 2019).

Classical search algorithms such as A* search (Hart et al., 1968; Pearl, 1984; Korf, 1985) address the challenge of planning ahead by using *heuristic* estimation of future cost when making decisions. Drawing inspiration from A* search, we develop NeuroLogic A$^\bigstar$esque (shortened to NeuroLogic$^\bigstar$), which combines A*-like heuristic estimates of future cost (e.g., perplexity, constraint satisfaction) with common decoding algorithms for neural text generation (e.g., beam search, top-k sampling), while preserving the efficiency demanded by large-scale neural language models.

As selecting the next token to generate based on the *optimal* future cost is NP-complete (Chen et al., 2018), we develop *lookahead* heuristics, which approximate cost at each decoding step based on con-

[1]Pronounced [ey stɑr ɛsk].

tinuations of the sequence-so-far. Figure 1 shows an example, where NEUROLOGIC A★esque guides generation towards a decision that would have been ignored based on the past alone, but is selected after looking ahead and incorporating the probability that constraints are satisfied in the future.

Our approach builds on NEUROLOGIC Decoding of Lu et al. (2021), a variation of beam-search for controlling generation through rich logic-based lexical constraints expressed in Conjunctive Normal Form (CNF). Our work generalizes Lu et al. (2021) by (1) incorporating novel lookahead heuristics to estimate future contraint satisfaction, and (2) developing additional *unconstrained* variants that can work with an empty set of constraints. These new algorithm variants support broad applications of NEUROLOGIC★, including unconstrained generation, as demonstrated in our experiments.

Our experiments across five generation tasks demonstrate that our approach outperforms competitive baselines. We test NEUROLOGIC★ in conjunction with both supervised and unsupervised models and find that the performance gain is pronounced especially in zero-shot or few-shot settings. On the COMMONGEN benchmark, using NEUROLOGIC★ with an off-the-shelf language model outperforms a host of *supervised* baselines with conventional decoding algorithms, demonstrating that a strong inference-time algorithm such as NEUROLOGIC★ can alleviate the need for costly annotated datasets. Moreover, NEUROLOGIC★ achieves state-of-the-art performance in various settings, including WMT17 English-German machine translation with lexical constraints (Dinu et al., 2019) and few-shot E2ENLG table-to-text generation (Chen et al., 2020b).

In summary, we develop NEUROLOGIC A★esque, a new decoding algorithm for effective and efficient text generation. To our knowledge this is the first A*-like algorithm for guided text generation via lookahead heuristics. Our algorithm is versatile, as it can be applied to a variety of tasks via inference-time constraints, reducing the need for costly labeled data. Extensive experiments show its effectiveness on several important generation benchmarks.

2 NEUROLOGIC A★esque Decoding

We describe NEUROLOGIC A★esque Decoding (shortened as NEUROLOGIC★), our decoding algorithm motivated by A* search (Hart et al., 1968), a best-first search algorithm that finds high-scoring paths using a heuristic estimate of future return. We first introduce the decoding problem, and then describe our heuristics with a novel lookahead procedure for adapting NEUROLOGIC★ search to unconstrained and constrained generation with large-scale autoregressive models.

2.1 Decoding With A★esque Lookahead

Decoding. Sequence-to-sequence generation is the task of generating an output sequence $\mathbf{y}$ given an input sequence $\mathbf{x}$. We consider standard left-to-right, autoregressive models, $p_\theta(\mathbf{y} \mid \mathbf{x}) = \prod_{t=1}^{|\mathbf{y}|} p_\theta(y_t \mid \mathbf{y}_{<t}, \mathbf{x})$, and omit $\mathbf{x}$ to reduce clutter. Decoding consists of solving,

$$\mathbf{y}_* = \arg\max_{\mathbf{y}\in\mathcal{Y}} F(\mathbf{y}), \tag{1}$$

where $\mathcal{Y}$ is the set of all sequences. In our setting, the objective $F(\mathbf{y})$ takes the form $s(\mathbf{y}) + H(\mathbf{y})$, where $s(\mathbf{y})$ is $\log p_\theta(\mathbf{y})$, and $H(\mathbf{y})$ is either zero when no constraints are specified, or is a score for satisfying constraints on $\mathbf{y}$.

Our method takes the perspective of decoding as discrete search, in which *states* are partial prefixes, $\mathbf{y}_{<t}$, *actions* are tokens in vocabulary $\mathcal{V}$ (i.e., $y_t \in \mathcal{V}$), and *transitions* add a token to a prefix, $\mathbf{y}_{<t} \circ y_t$. Each step of decoding consists of (1) expanding a set of candidate next-states, (2) scoring each candidate, and (3) selecting the k best candidates:

$$\begin{aligned} Y_t' &= \{\mathbf{y}_{<t} \circ y_t \mid \mathbf{y}_{<t} \in Y_{t-1}, y_t \in \mathcal{V}\}, \\ Y_t &= \operatorname*{arg\,topk}_{(\mathbf{y}_{<t}, y_t)\in Y_t'} \{f(\mathbf{y}_{<t}, y_t)\}, \end{aligned} \tag{2}$$

where $Y_0 = \{\langle bos \rangle\}$ and $f(\cdot)$ is a scoring function that approximates the objective F. Common decoding algorithms such as beam search score candidates without considering future tokens, e.g., $f(\mathbf{y}_{<t}, y_t) = \log p_\theta(\mathbf{y}_{\leq t})$.

Lookahead heuristics. Our method incorporates an estimate of the future into candidate selection. Ideally, we want to select candidates that are on optimal trajectories, replacing Equation 2 with:

$$Y_t = \operatorname*{arg\,topk}_{(\mathbf{y}_{<t}, y_t)\in Y_t'} \left\{ \max_{\mathbf{y}_{>t}} F(\mathbf{y}_{<t}, y_t, \mathbf{y}_{>t}) \right\}, \tag{3}$$

where $\mathbf{y}_{>t}$ represents future trajectories. However, computing Equation 3 presents two difficulties: 1) the objective $F(\mathbf{y})$ may be unknown or difficult to compute, and 2) the space of $\mathbf{y}_{>t}$ is prohibitively large.

Motivated by A* search (Hart et al., 1968), a best-first search algorithm that finds high-scoring paths by selecting actions that maximize:

$$f(a) = s(a) + h(a),$$

where $s(a)$ is the score-so-far and $h(a)$ is a heuristic estimate of the future score, we approximate the objective using a lightweight *heuristic* $h(\cdot)$:

$$Y_t = \underset{\mathbf{y}_{\leq t} \in Y_t'}{\arg\operatorname{topk}} \left\{ s(\mathbf{y}_{\leq t}) + \max_{\mathbf{y}_{>t}} h(\mathbf{y}_{<t}, y_t, \mathbf{y}_{>t}) \right\}, \tag{4}$$

where $s(\mathbf{y}_{\leq t}) = \log p_\theta(\mathbf{y}_{\leq t})$. To make the search tractable, we search over a set of *lookahead* continuations, approximating Equation 3 as,

$$Y_t = \underset{\mathbf{y}_{\leq t} \in Y_t'}{\arg\operatorname{topk}} \left\{ s(\mathbf{y}_{\leq t}) + \max_{\mathcal{L}_\ell(\mathbf{y}_{\leq t})} h(\mathbf{y}_{\leq t+\ell}) \right\}, \tag{5}$$

where each element $\mathbf{y}_{t+1:t+\ell}$ of $\mathcal{L}_\ell(\mathbf{y}_{\leq t})$ is a length-ℓ continuation of $\mathbf{y}_{\leq t}$. Beam search corresponds to setting ℓ and h to 0.

A★esque decoding. Beam search, A* search, and our method fall under a general class of algorithms that differ based on (1) which candidates are expanded, (2) which candidates are pruned, (3) how candidates are scored (Meister et al., 2020). We inherit the practical advantages of beam search-style expansion and pruning, while drawing on A*-like heuristics to incorporate estimates of the future, and refer to our method as **A★esque decoding**.

Generating lookaheads. We compare several methods for generating the lookaheads $\mathcal{L}_\ell(\mathbf{y}_{\leq t})$.

The *greedy* lookahead produces a single sequence, $\mathcal{L}_\ell = \{\mathbf{y}_{t+1:t+\ell}\}$, starting from $\mathbf{y}_{\leq t}$ and selecting each token according to $y_{t'} = \arg\max_{y \in \mathcal{V}} p_\theta(y \mid \mathbf{y}_{<t'})$.

We also consider a *soft* lookahead which interpolates between providing the greedy token and a uniform mixture of tokens as input at each step. Specifically, we adjust the model's probabilities with a temperature, $\tilde{p}_\theta(y_t \mid \mathbf{y}_{<t}) = \text{softmax}(s_t/\tau)$, where $s_t \in \mathbb{R}^{|\mathcal{V}|}$ is a vector of logits, and feed the expected token embedding as input at step t,

$$e_t = \mathbb{E}_{y_t \sim \tilde{p}(y_t \mid \mathbf{y}_{<t})}[E(y_t)], \tag{6}$$

where $E \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the model's token embedding matrix. The soft lookahead moves from providing the greedy token as input ($\tau \to 0$) to a uniform mixture of tokens ($\tau \to \infty$) based on the value of temperature τ. When using the soft lookahead, we use $\tilde{p}$ in place of p when scoring tokens. The soft (and greedy) lookahead is efficient, but only explores a single trajectory.

The *beam* lookahead trades off efficiency for exploration, returning a set $\mathcal{L}_\ell$ containing the top-k candidates obtained by running beam search for ℓ steps starting from $\mathbf{y}_{<t}$.

Finally, the *sampling* lookahead explores beyond the highly-probable beam search continuations, generating each $\mathbf{y}_{t+1:t+\ell} \in \mathcal{L}_\ell$ using,

$$y_{t'} \sim p_\theta(y \mid \mathbf{y}_{<t'}),$$

for t' from t+1 to t+k.

Next, we move to our proposed lookahead heuristics, starting with the unconstrained setting.

2.2 Unconstrained Generation with NeuroLogic★

First we consider a standard decoding setting,

$$\arg\max_{\mathbf{y} \in \mathcal{Y}} \log p_\theta(\mathbf{y} \mid \mathbf{x}).$$

We score candidates based on a combination of the *history* and *estimated future*, by using the likelihood of the lookahead as a heuristic. That is, at the tth step of decoding, we use Equation 5 with:

$$h(\mathbf{y}_{\leq t+\ell}) = \lambda \log p_\theta(\mathbf{y}_{t+1:t+\ell} \mid \mathbf{y}_{\leq t}, \mathbf{x}), \tag{7}$$

where λ controls how much we rely on the estimated future versus the history, similar to weighted A* (Pohl, 1970).

2.3 NeuroLogic★ for Constrained Generation

Our lookahead heuristics lend themselves to decoding with lexical constraints in a way that standard beam search does not. For constrained generation, we build on and generalize NeuroLogic decoding algorithm of Lu et al. (2021)—a beam-based search algorithm that supports a wide class of logical constraints for lexically constrained generation—with estimates of future constraint satisfaction.

Background of NeuroLogic. NeuroLogic (Lu et al., 2021) accepts lexical constraints in CNF:

$$\underbrace{(D_1 \vee D_2 \cdots \vee D_i)}_{C_1} \wedge \cdots \wedge \underbrace{(D_{i'} \vee \cdots \vee D_N)}_{C_M}$$

where each D_i represents a single positive or negative constraint, $D(\mathbf{a}, \mathbf{y})$ or $\neg D(\mathbf{a}, \mathbf{y})$, enforcing the phrase $\mathbf{a}$ to be included in or omitted from $\mathbf{y}$. Lu et al. (2021) refer to each constraint D_i as a *literal*, and each disjunction C_j of literals as a *clause*.

NeuroLogic is a beam-based approximate search for an objective which seeks fluent sequences in which all clauses are satisfied:

$$\arg\max_{\mathbf{y}\in\mathcal{Y}} p_\theta(\mathbf{y} \mid \mathbf{x}) - \lambda' \sum_{j=1}^{M} (1 - C_j),$$

where $\lambda' \gg 0$ penalizes unsatisfied clauses. At each step of the search, NeuroLogic scores each of the $k \times |\mathcal{V}|$ candidates $(\mathbf{y}_{<t}, y_t)$ based on whether they (partially) satisfy new constraints,

$$f(\mathbf{y}_{\leq t}) = \log p_\theta(\mathbf{y}_{\leq t} \mid \mathbf{x}) + \lambda_1 \max_{D(\mathbf{a},\mathbf{y}_{\leq t})} \frac{|\hat{\mathbf{a}}|}{|\mathbf{a}|}, \quad (8)$$

where the maximization is over a set of unsatisfied multi-token constraints $\mathbf{a}$ tracked by NeuroLogic, and $\hat{\mathbf{a}}$ is the prefix of $\mathbf{a}$ in the ongoing generation. For example, for $\mathbf{y}_{\leq t}$ ="*The boy climbs an apple*" and constraint $\mathbf{a}$="*apple tree*", $\hat{\mathbf{a}}$ is "*apple*". Intuitively, this function rewards candidates that are in the process of satisfying a constraint.

In lieu of taking the top-k scoring candidates (Equation 5), NeuroLogic prunes candidates that contain clauses that violate constraints, groups the candidates to promote diversity, and selects high-scoring candidates from each group. We use the same pruning and grouping approach, and refer the reader to Lu et al. (2021) for further details.

NeuroLogic★ decoding. Our method improves upon the NeuroLogic scoring function with an estimate of future constraint satisfaction. Our key addition is a lookahead heuristic that adjusts a candidate $(\mathbf{y}_{<t}, y_t)$'s score proportional to the probability of satisfying additional *unsatisfied* constraints in the lookahead $\mathbf{y}_{t+1:t+\ell}$:

$$h_{\text{future}}(\mathbf{y}_{\leq t+\ell}) = \lambda_2 \max_{D(\mathbf{a},\mathbf{y}_{\leq t})} \log p_\theta(D(\mathbf{a}, \mathbf{y}_{t+1:t+\ell}) \mid \mathbf{x}, \mathbf{y}_{\leq t}), \quad (9)$$

where we define the probability that constraint $\mathbf{a}$ is satisfied using the most probable subsequence,

$$p_\theta(D(\mathbf{a}, \mathbf{y}_{t+1:t+\ell}) \mid \mathbf{x}, \mathbf{y}_{\leq t}) = \max_{t' \in [t, t+\ell]} p_\theta(\mathbf{y}_{t':t'+|\mathbf{a}|} = \mathbf{a} \mid \mathbf{x}, \mathbf{y}_{<t'}), \quad (10)$$

λ_2 is a scaling hyperparameter for the heuristic.

Intuitively, this lookahead heuristic brings two benefits. When y_t is a token that would satisfy a multi-token constraint, the lookahead incorporates the score of the *full* constraint. When y_t is a token that is not part of a constraint, the lookahead allows for incorporating the score of a future constraint that would be satisfied if y_t was selected.

We add our lookahead heuristic to the NeuroLogic scoring function (Equation 8), and call the resulting decoding procedure NeuroLogic A★esque (or, NeuroLogic★ in short).

3 Experiments

We first consider constrained generation benchmarks: CommonGen (§3.1), constrained machine translation (§3.2), table-to-text generation (§3.3), and constrained question generation (§3.4). NeuroLogic★ consistently outperforms previous approaches, especially in zero-shot and few-shot cases. These low-resource settings are particularly important, as many practical tasks face data scarcity. Finally, we find that A★esque lookahead is useful even without constraints, as shown in unconstrained story generation task (§3.5).

Metrics. As automatic metrics, we use: BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), METEOR (Banerjee and Lavie, 2005), CIDEr (Vedantam et al., 2015), SPICE (Anderson et al., 2016) and NIST (Lin and Hovy, 2003).

3.1 Constrained Commonsense Generation

CommonGen (Lin et al., 2020) is a commonsense generation task with lexical constraints: given a set of concepts (e.g., {throw, run, javelin, track}), models need to generate a coherent sentence describing a plausible scenario using all given concepts (e.g., "a man runs on a track and throws a javelin.").

Approach and Baselines. Following Lu et al. (2021), we enforce that each concept c_i appear in output $\mathbf{y}$ under some morphological inflection. We test in both supervised and zero-shot settings. In the supervised setting, we finetune GPT-2 (Radford et al., 2019) as a sequence-to-sequence model. In the zero-shot setting, we use GPT-2 off-the-shelf (no fine-tuning) and rely on constrained decoding to guide generation. We compare with previous constrained decoding algorithms CBS (Anderson et al., 2017), GBS (Hokamp and Liu, 2017), DBA (Post and Vilar, 2018a), NeuroLogic (Lu et al., 2021) and TSMH (Zhang et al., 2020).

Metrics. We report standard automatic metrics as well as *coverage*, the average percentage of concepts present in generations. Additionally, we conduct human evaluation on 100 test examples using Amazon Mechanical Turk (AMT), with 3 annotators per example (template in Appendix D). Workers rate each generation on *language quality*, *sce-*

Decode Method	Automatic Evaluation						Human Evaluation			
	ROUGE-L	BLEU-4	METEOR	CIDEr	SPICE	Coverage	Quality	Plausibility	Concepts	Overall
Supervised										
CBS (Anderson et al., 2017)	38.8	20.6	28.5	12.9	27.1	97.6	2.27	2.35	2.51	2.23
GBS (Hokamp and Liu, 2017)	38.2	18.4	26.7	11.7	26.1	97.4	2.06	2.17	2.29	2.01
DBA (Post and Vilar, 2018a)	38.3	18.7	27.7	12.4	26.3	97.5	2.23	2.30	2.43	2.15
NeuroLogic (Lu et al., 2021)	42.8	26.7	30.2	14.7	30.3	<u>97.7</u>	2.54	2.56	2.67	2.50
NeuroLogic* (greedy)	<u>43.6</u>	<u>28.2</u>	**30.8**	15.2	<u>30.8</u>	**97.8**	2.66	<u>2.67</u>	2.73	2.59
NeuroLogic* (sample)	43.4	27.9	**30.8**	<u>15.3</u>	**31.0**	<u>97.7</u>	2.64	2.64	2.74	2.58
NeuroLogic* (beam)	43.2	<u>28.2</u>	<u>30.7</u>	15.2	**31.0**	97.6	<u>2.68</u>	<u>2.67</u>	<u>2.76</u>	<u>2.60</u>
Unsupervised										
TSMH (Zhang et al., 2020)	24.7	2.2	14.5	3.6	15.4	71.5	1.85	1.92	1.95	1.63
NeuroLogic (Lu et al., 2021)	41.9	24.7	29.5	14.4	27.5	96.7	2.64	2.52	2.68	2.50
NeuroLogic* (greedy)	**44.3**	**28.6**	<u>30.7</u>	**15.6**	29.6	97.1	**2.78**	**2.70**	**2.77**	**2.70**

Table 1: Performance of various decoding methods with *supervised* or *off-the-shelf* GPT-2 on the CommonGen test set, measured with automatic and human evaluations. We only tried NeuroLogic* (greedy) in the unsupervised setting because of the computational cost. The best numbers are **bolded** and the second best ones are <u>underlined</u>.

Words	Method	Generation
cut	GBS	Cut a piece of wood to use as a fence.
piece	DBA	Cut a piece of wood to use as a fence.
use	NeuroLogic	Piece of wood used for cutting.
wood	NeuroLogic*	**A man cuts a piece of wood using a circular saw.**
ball	GBS	A dog is run over by a ball and mouth agape.
dog	DBA	A dog is run over by a ball and bites his mouth.
mouth	NeuroLogic	A dog is running and chewing on a ball in its mouth.
run	NeuroLogic*	**A dog running with a ball in its mouth.**
dog	GBS	Soap and water scrubbed dog with a towel.
scrub	DBA	Soap and water on a dog and scrubbed skin.
soap	NeuroLogic	A dog is scrubbing his paws with soap and water.
water	NeuroLogic*	**A man is scrubbing a dog with soap and water.**

Table 2: Example generations for the CommonGen task across supervised NeuroLogic* and baselines, including GBS (Hokamp and Liu, 2017), DBA (Post and Vilar, 2018a), and NeuroLogic (Lu et al., 2021).

Method	Dinu et al.		Marian MT	
	BLEU	Term%	BLEU	Term%
Unconstrained	25.8	76.3	32.9	85.0
train-by-app.	26.0	92.9	–	–
train-by-rep.	26.0	94.5	–	–
Post and Vilar (2018a)	25.3	82.0	33.0	94.3
NeuroLogic	26.5	95.1	33.4	<u>97.1</u>
NeuroLogic* (greedy)	**26.7**	**95.8**	**33.7**	**97.2**
NeuroLogic* (sample)	<u>26.6</u>	<u>95.4</u>	**33.7**	**97.2**
NeuroLogic* (beam)	<u>26.6</u>	**95.8**	<u>33.6</u>	**97.2**

Table 3: Results on constrained MT. The left section uses the same two-layer transformer as Dinu et al. (2019), while the right one uses a stronger Marian MT EN-DE model. The highlighted methods modify training data specifically for constrained generation, and thus cannot be applied to off-the-shelf models. The best numbers are **bold**, second best are <u>underlined</u>.

nario plausibility, *coverage of given concepts*, and an *overall score* on a 3-point Likert scale.[2]

Results. Table 1 compares different constrained decoding methods on top of the finetuned and off-the-shelf GPT-2, in supervised and zero-shot settings respectively. The key observations are:

1. NeuroLogic* outperforms all previous constrained-decoding methods in both supervised and zero-shot settings. Surprisingly, *unsupervised* NeuroLogic* outperforms all supervised methods based on human evaluation.
2. Compared to vanilla NeuroLogic, NeuroLogic* improves generation quality while maintaining high constraint satisfaction. The difference is especially substantial in the zero-shot setting. Intuitively, this setting leaves more room for incorporating constraint-driven signals due to the lack of supervision.
3. NeuroLogic* reaches similar performance using different lookahead strategies, among which beam lookahead slightly outperforms the others based on human evaluation, and greedy lookahead has the lowest runtime. We analyze lookahead strategies further in Appendix A.

3.2 Constrained Machine Translation

Next, we test on constrained machine translation (MT). It is often critical to have control over MT systems, such as to incorporate domain-specific terminology (Post and Vilar, 2018a; Dinu et al., 2019). To achieve this goal, recent work proposed constrained decoding algorithms (Chatterjee et al., 2017; Hokamp and Liu, 2017; Hasler et al., 2018; Hu et al., 2019, *inter alia*) or specialized training (Dinu et al., 2019). We demonstrate that

[2]Agreement by ordinal Krippendorff alpha ($0 \leq \alpha \leq 1$) (Krippendorff, 2007) is 0.40, 0.46, 0.36, and 0.44 (respectively) indicating fair to moderate agreement.

# T	# Sents.	Decode Method	BLEU	Term%
1	378	Beam search	25.4	79.6
		NeuroLogic	26.2	95.2
		NeuroLogic★	**26.3**	**95.8**
2+	36	Beam search	28.1	85.0
		NeuroLogic	28.9	93.7
		NeuroLogic★	**29.3**	**96.5**

Table 4: Constrained MT performance broken down by the number of constraint terms (# T). All configurations use the two-layer tranformer from Dinu et al. (2019). The best numbers are **bolded** and the second best ones are underlined.

NeuroLogic★ can be readily applied to off-the-shelf MT systems for constrained machine translation. We follow Dinu et al. (2019) and evaluate on the WMT17 EN-DE test set (Bojar et al., 2017). The constraint here is to integrate given custom terminologies into the translation output; constraint terms are automatically created from the IATE EU terminology database for 414 test sentences.

Approach, Baselines, and Metrics. We experiment with two MT systems: Dinu et al. (two-layer transformer) and the off-the-shelf Marian MT (Junczys-Dowmunt et al., 2018). We compare with previous constrained decoding algorithms, including DBA (Post and Vilar, 2018a), NeuroLogic (Lu et al., 2021), and also specialized training proposed by Dinu et al. (2019). Following Dinu et al. (2019), we report BLEU and term use rates, i.e., percentage of times given constraint terms were generated out of total number of constraint terms.

Results. Table 3 presents experimental results with Dinu et al.'s model and Marian MT. In both cases, NeuroLogic★ outperforms prior methods in BLEU and term coverage. Besides higher quality and coverage, NeuroLogic★ is plug-and-play, working with any off-the-shelf MT system, unlike previous training-based methods. Table 4 breaks down the performance by the number of constraint terms. We see that the improvement brought by NeuroLogic★ is especially large when given complex constraints with multiple terms. (e.g., 96.5 vs. 93.7 from NeuroLogic in term of coverage).

3.3 Table-to-text Generation

Next we test on the table-to-text task, where models need to generate natural language for structured table data. Constrained generation ensures that the output text is factual and consistent with the input data. We follow the few-shot setup of Chen et al. (2020b) on the E2ENLG (Dušek et al., 2018) dataset, where randomly-sampled 0.1%, 0.5%, 1%, or 5% of training instances are used for finetuning.

Decode Method	NIST	BLEU	METEOR	CIDEr	ROUGE	Coverage
Beam Search	3.82	42.8	32.6	10.8	57.8	73.6
CBS	6.50	42.3	36.4	13.0	54.3	91.6
GBS	6.26	40.7	36.7	12.9	54.2	94.1
NeuroLogic	6.95	47.6	38.9	16.3	58.7	97.6
NeuroLogic★ (greedy)	**7.11**	49.2	40.0	**17.5**	**60.0**	**100.0**
NeuroLogic★ (beam)	7.01	48.9	40.0	17.2	59.8	99.9
NeuroLogic★ (sample)	**7.11**	**49.3**	**40.1**	**17.5**	**60.0**	**100.0**

Table 5: Performance of different decoding methods with few-shot GPT-2 finetuned on 0.1% E2ENLG data. The best numbers are **bold**, second best are underlined.

Method	0.1%	0.5%	1%	5%
TGen (Dušek and Jurčíček, 2016)	3.6	27.9	35.2	57.3
Template-GPT-2 (Chen et al., 2020a)	22.5	47.8	53.3	59.9
KGPT-Graph (Chen et al., 2020b)	39.8	53.3	55.1	61.5
KGPT-Seq (Chen et al., 2020b)	40.2	53.0	54.1	61.1
GPT-2	42.8	57.1	56.8	61.1
GPT-2 + NeuroLogic	47.6	56.9	58.0	62.9
GPT-2 + NeuroLogic★ (greedy)	**49.2**	**58.0**	**58.4**	**63.4**

Table 6: Few-shot results (BLEU-4) on E2ENLG test set with 0.1%, 0.5%, 1%, 5% of training instances. The best numbers are **bold**, second best are underlined.

Approach, Baselines, and Metrics. Following Shen et al. (2019), we linearize data tables into strings and finetune GPT-2 with few-shot examples. We compare NeuroLogic★ with three previous constrained decoding algorithms: CBS (Anderson et al., 2017), GBS (Hokamp and Liu, 2017), and NeuroLogic (Lu et al., 2021), based on few-shot GPT-2 finetuned with 0.1% data. Then we compare NeuroLogic★ on top of GPT-2, with previous table-to-text methods, including TGen (Dušek and Jurčíček, 2016), Template-GPT-2 (Chen et al., 2020a), KGPT (Chen et al., 2020b), in multiple few-shot settings with various numbers of training instances. We report standard automatic metrics, as well as information coverage, i.e., percentage of information present in the generation.

Results. Table 5 compares various decoding methods with few-shot GPT-2 finetuned on 0.1% of the data. NeuroLogic★ substantially outperforms previous methods on all metrics, consistently improving quality while achieving near-perfect constraint satisfaction. Previous work (CBS and GBS) improves constraint satisfaction, but negatively affects quality, indicated by drops in BLEU and ROUGE. Table 6 compares NeuroLogic★ on top of GPT-2 with previous table-to-text approaches. As before, NeuroLogic★ outperforms past approaches by a large margin, even if the latter ones leverage specialized model architectures or addi-

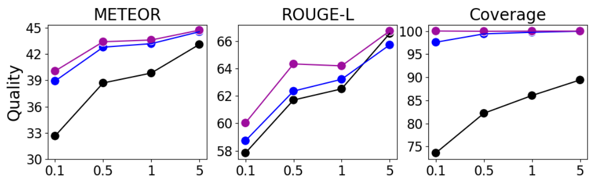


Figure 2: Performance (y-axis) of supervised GPT-2 on E2ENLG, with a varying percentage of training data for supervision (x-axis). The **purple**, **blue**, and **black** lines denote decoding with NEUROLOGIC★, NEUROLOGIC and conventional beam search, respectively.

tional pretraining on massive table-to-text corpora. Additionally, Figure 2 compares the performance (y-axis) of few-shot GPT-2 with NEUROLOGIC★ (**purple line**), NEUROLOGIC (**blue line**), and conventional beam search (**black line**) as a function of the varying percentage of training instances (x-axis). The benefit of NEUROLOGIC★ grows as data size is reduced. Indeed, constrained decoding enables impressive low-resource performance.

3.4 Constrained Question Generation

Next, we consider constrained question generation (Zhang et al., 2020), where models need to generate interrogative questions using given keywords. This task is zero-shot without any training data, further testing the capacity of NEUROLOGIC★ to guide off-the-shelf models without finetuning.

Approach, Baselines, and Metrics. We use GPT-2 off-the-shelf and compare NEUROLOGIC★ with previous constrained decoding methods, including CGMH (Miao et al., 2019), TSMH (Zhang et al., 2020) and NEUROLOGIC (Lu et al., 2021). We report standard generation metrics and keyword coverage as in §3.1. We conduct human evaluation following subsection 3.1, to measure grammar, fluency, meaningfulness, and overall quality of the generated questions, using a 3-point Likert scale[3] (template in Appendix D).

Results. Table 7 presents comparisons across different decoding methods based on off-the-shelf language models. NEUROLOGIC★ outperforms all previous methods with respect to both automatic and manual metrics; it enhances the generation quality while achieving perfect constraint satisfaction. The difference between NEUROLOGIC and NEUROLOGIC★ is particularly large compared to other tasks. We suspect that the search problem is much harder here, due to the lack of supervision and complex logical constraints involving both keywords and syntax. As a whole, the results demonstrate the effectiveness of NEUROLOGIC★ in tackling challenging constrained generation problems.

3.5 *Unconstrained* Story Generation

Finally, we demonstrate NEUROLOGIC★ can also improve *unconstrained* generation. We investigate whether A★esque decoding with our unconstrained lookahead heuristic (Equation 7) can (1) improve beam search, which typically struggles in open-ended settings (Holtzman et al., 2020; Welleck et al., 2019b), and (2) improve *sampling* algorithms that are commonly used in open-ended generation. We consider conditional story generation on the RocStories dataset (Mostafazadeh et al., 2016): given a first sentence $\mathbf{x}$, generate the full story $\mathbf{y}$.

Approach, Baselines and Metrics. We use GPT-2, fine-tuned on the RocStories training set. We apply A★esque decoding to (1) beam search, the setting used so far in the experiments, and (2) top-k sampling (Fan et al., 2018), a commonly used sampling algorithm in open-ended generation. For top-k sampling, we use the heuristic to adjust the probability scores, then renormalize. We use standard automatic metrics: perplexity and BLEU for fluency, and unique n-grams as a measure of diversity. We conduct human evaluation following subsection 3.1, for story flow and overall quality on a 3-point Likert scale[4] (template in Appendix D).

Results. Table 8 presents the results of beam search and top-k sampling with and without A★esque heuristics. A★esque heuristics result in more fluent, coherent and interesting stories for both beam search and top-k sampling. For beam search, A★esque not only enhances generation quality– e.g. improving human evaluation scores from 2.32 to 2.63–but also boosts generation diversity, reflected by number of unique n-grams. For top-k sampling, A★esque heuristics improve quality, while maintaining comparable diversity. We further analyze quality and diversity tradeoff in Appendix A. Moreover, we notice that beam lookahead works the best for beam search, and greedy lookahead works the best for top-k sampling. We suspect that beam lookahead gives the most accurate estimate of future beam path, while greedy

[3]Agreement by ordinal Krippendorff alpha ($0 \leq \alpha \leq 1$) (Krippendorff, 2007) is 0.27, 0.28, 0.25 and 0.30, indicating fair agreement.

[4]Agreement by ordinal Krippendorff alpha ($0 \leq \alpha \leq 1$) (Krippendorff, 2007) of 0.24 and 0.22 (respectively), indicating fair agreement.

Decode Method	Automatic Evaluation						Human Evaluation			
	ROUGE	BLEU	METEOR	CIDEr	SPICE	Coverage	Grammar	Fluency	Meaningfulness	Overall
CGMH (Miao et al., 2019)	28.8	2.0	18.0	5.5	21.5	18.3	2.28	2.34	2.11	2.02
TSMH (Zhang et al., 2020)	42.0	4.3	25.9	10.4	37.7	92.7	2.35	2.28	2.37	2.22
NeuroLogic (Lu et al., 2021)	38.8	11.2	24.5	18.0	41.7	90.6	2.78	2.71	2.49	2.51
NeuroLogic★ (greedy)	**43.7**	**14.7**	28.0	**20.9**	47.7	**100.0**	**2.83**	2.77	2.74	**2.76**
NeuroLogic★ (beam)	42.9	14.4	27.8	20.3	46.9	**100.0**	2.81	**2.86**	**2.76**	2.75
NeuroLogic★ (sample)	43.5	14.6	**28.2**	20.8	**47.8**	**100.0**	**2.83**	2.75	**2.76**	2.73

Table 7: Performance of different unsupervised decoding algorithms on constrained question generation.

Decode Method	Fluency			Diversity			Human Eval	
	PPL	BLEU-1	BLEU-2	Uniq. 2-gram	Uniq. 3-gram	Uniq. 4-gram	Coherence	Overall
beam search	2.24	33.7	16.5	20.13k	34.09k	41.91k	2.46	2.32
beam search + A★esque (greedy)	**2.11**	34.3	16.7	20.63k	34.94k	43.02k	2.56	2.57
beam search + A★esque (beam)	2.14	**34.4**	**16.8**	20.68k	35.03k	43.12k	**2.62**	**2.63**
beam search + A★esque (sample)	2.16	**34.4**	16.7	**20.78k**	**35.41k**	**43.64k**	2.59	2.57
top-k sample	4.01	31.4	13.9	28.54k	48.36k	56.62k	2.23	2.15
top-k sample + A★esque (greedy)	**3.68**	32.1	14.3	28.47k	**48.44k**	**56.63k**	**2.48**	**2.47**
top-k sample + A★esque (beam)	3.75	**32.2**	**14.4**	28.53k	48.27k	56.36k	2.39	2.34
top-k sample + A★esque (sample)	3.70	32.0	14.2	**28.57k**	48.04k	56.15k	2.47	2.44

Table 8: Performance of different decoding algorithms on RocStories test set.

lookahead provides an estimate which better resembles a continuation from top-k sampling.

4 Related Work

A* search in NLP. Many classical NLP problems (e.g., parsing, text alignment) can be seen as structured prediction subject to a set of task-specific constraints. For many such problems, A* search has been used effectively (Och et al., 2001; Haghighi et al., 2007; Hopkins and Langmead, 2009; Meister et al., 2020). For example, Klein and Manning (2003); Zhang and Gildea (2006); Auli and Lopez (2011); Lee et al. (2016) have used it in the context of parsing. Similar approaches are used for finding high-probability alignments (Naim et al., 2013). Despite these applications, applying informed heuristic search to text generation with autoregressive language models (this work's focus) has been underexplored.

Decoding strategies for text generation. The rise of autoregressive language models like GPT (Radford et al., 2018) has inspired work on decoding strategies (Post and Vilar, 2018a; Ippolito et al., 2019; Zheng et al., 2020; Leblond et al., 2021; West et al., 2021). These works often focus on incorporating factors like diversity (Ippolito et al., 2019), fluency (Holtzman et al., 2020), or constraints (Anderson et al., 2017; Hokamp and Liu, 2017; Post and Vilar, 2018b; Miao et al., 2019; Welleck et al., 2019a; Zhang et al., 2020; Qin et al., 2020; Lu et al., 2021). Constrained beam search (Anderson et al., 2017) and grid beam search (Hokamp and Liu, 2017) extend beam search to satisfy lexical constraints during generation. Lu et al. (2021) incorporate logic-based constraints into beam search, which we extend with lookahead heuristics.

Other work addresses the mismatch between monotonic decoding and satisfying constraints that can depend on a full generation, through MCMC sampling (Miao et al., 2019; Zhang et al., 2020), recursive non-monotonic generation (Welleck et al., 2019a), continuous optimization (Qin et al., 2020), or generated contexts (West et al., 2021). Unlike these past works, NeuroLogic A★esque explicitly decodes future text to estimate the viability of different paths for satisfying constraints.

5 Conclusion

Inspired by the A* search algorithm, we introduce NeuroLogic A★esque decoding, which brings A*-like heuristic estimates of the *future* to common *left-to-right* decoding algorithms for neural text generation. A★esque lookahead heuristics improve over existing decoding methods (*e.g.*, NeuroLogic, beam, greedy, sample decoding methods) in both *constrained* and *unconstrained* settings across a wide spectrum of tasks. Our work demonstrates the promise of moving beyond the current paradigm of unidirectional decoding for text generation, by taking bidirectional information from both the *past* and *future* into account to generate more globally coherent text.

Acknowledgment

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) (funding reference number 401233309), DARPA MCS program through NIWC Pacific (N66001-19-2-4031), Google Cloud Compute, and Allen Institute for AI, Microsoft PhD Fellowship.

Broader Impact and Ethical Implications

Our method deals with improving neural text generation, thus inheriting the potential impact and risks brought by text generation applications (e.g. dual use, see Pandya (2019); Brown et al. (2020)). Constraining generation through logical constraints offers the promise of improved control, consistency, and human-machine collaboration in high-impact applications such as translation, machine-aided writing, and education. On the other hand, constrained generation methods could foreseeably be used to generate text that contains biased, offensive, and/or hateful keywords (e.g., extremist texts; McGuffie and Newhouse, 2020). For a broader discussion of these risks, and of the risks of large pretrained language models in general, refer to discussions in Brown et al. (2020); Bender et al. (2021).

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. Spice: Semantic propositional image caption evaluation. In *European conference on computer vision*, pages 382–398. Springer.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2017. Guided open vocabulary image captioning with constrained beam search. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 936–945, Copenhagen, Denmark. Association for Computational Linguistics.

Michael Auli and Adam Lopez. 2011. Efficient CCG parsing: A* versus adaptive supertagging. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1577–1585, Portland, Oregon, USA. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Emily Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*, pages 169–214, Copenhagen, Denmark. Association for Computational Linguistics.

T. Brown, B. Mann, Nick Ryder, Melanie Subbiah, J. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, G. Krüger, T. Henighan, R. Child, Aditya Ramesh, D. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, E. Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, J. Clark, Christopher Berner, Sam McCandlish, A. Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Rajen Chatterjee, Matteo Negri, Marco Turchi, Marcello Federico, Lucia Specia, and Frédéric Blain. 2017. Guiding neural machine translation decoding with external knowledge. In *Proceedings of the Second Conference on Machine Translation*, pages 157–168, Copenhagen, Denmark. Association for Computational Linguistics.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020a. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, Online. Association for Computational Linguistics.

Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020b. KGPT: Knowledge-grounded pretraining for data-to-text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8635–8648, Online. Association for Computational Linguistics.

Yining Chen, Sorcha Gilroy, Andreas Maletti, Jonathan May, and Kevin Knight. 2018. Recurrent neural networks as weighted language recognizers. 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 2261–2271, New Orleans, Louisiana. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Georgiana Dinu, Prashant Mathur, Marcello Federico, and Yaser Al-Onaizan. 2019. Training neural machine translation to apply terminology constraints. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3063–3068, Florence, Italy. Association for Computational Linguistics.

Ondřej Dušek and Filip Jurčíček. 2016. Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 45–51, Berlin, Germany. Association for Computational Linguistics.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG Challenge. In *Proc. of the 11th International Conference on Natural Language Generation*, pages 322–328, Tilburg, The Netherlands. 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.

Aria Haghighi, John DeNero, and Dan Klein. 2007. Approximate factoring for A* search. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 412–419, Rochester, New York. Association for Computational Linguistics.

Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Eva Hasler, Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2018. Neural machine translation decoding with terminology constraints. 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 506–512, New Orleans, Louisiana. 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, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Mark Hopkins and Greg Langmead. 2009. Cube pruning as heuristic search. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 62–71.

J. Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. 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.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning*, pages 1587–1596. PMLR.

Daphne Ippolito, Reno Kriz, João Sedoc, Maria Kustikova, and Chris Callison-Burch. 2019. Comparison of diverse decoding methods from conditional language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3752–3762.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2003. A* parsing: Fast exact Viterbi parse selection. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 119–126.

Richard E Korf. 1985. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial intelligence*, 27(1):97–109.

Klaus Krippendorff. 2007. Computing krippendorff's alpha reliability. *Departmental papers (ASC)*, page 43.

Rémi Leblond, Jean-Baptiste Alayrac, Laurent Sifre, Miruna Pislar, Jean-Baptiste Lespiau, Ioannis Antonoglou, Karen Simonyan, and Oriol Vinyals. 2021. Machine translation decoding beyond beam search. *arXiv preprint arXiv:2104.05336*.

Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2016. Global neural CCG parsing with optimality guarantees. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2366–2376, Austin, Texas. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Bill Yuchen Lin, Ming Shen, Wangchunshu Zhou, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of EMNLP*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, Online. Association for Computational Linguistics.

Kris McGuffie and Alex Newhouse. 2020. The radicalization risks of gpt-3 and advanced neural language models. arXiv.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020. Best-first beam search. *Transactions of the Association for Computational Linguistics*, 8:795–809.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, San Diego, California. Association for Computational Linguistics.

Iftekhar Naim, Daniel Gildea, Walter Lasecki, and Jeffrey P Bigham. 2013. Text alignment for real-time crowd captioning. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 201–210.

Franz Josef Och, Nicola Ueffing, and Hermann Ney. 2001. An efficient a* search algorithm for statistical machine translation. In *Proceedings of the ACL 2001 Workshop on Data-Driven Methods in Machine Translation*.

Jayshree Pandya. 2019. The dual-use dilemma of artificial intelligence. *Forbes Magazine*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Judea Pearl. 1984. Heuristics - intelligent search strategies for computer problem solving. In *Addison-Wesley series in artificial intelligence*.

Ira Pohl. 1970. *First Results on the Effect of Error in Heuristic Search*.

Matt Post and David Vilar. 2018a. 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.

Matt Post and David Vilar. 2018b. 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.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Backpropagation-based decoding for unsupervised counterfactual and abductive reasoning. In *EMNLP*.

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

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

Sheng Shen, Daniel Fried, Jacob Andreas, and Dan Klein. 2019. Pragmatically informative text generation. 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 4060–4067, Minneapolis, Minnesota. Association for Computational Linguistics.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Sean Welleck, Kianté Brantley, Hal Daumé Iii, and Kyunghyun Cho. 2019a. Non-monotonic sequential text generation. In *International Conference on Machine Learning*, pages 6716–6726. PMLR.

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

Peter West, Ximing Lu, Ari Holtzman, Chandra Bhagavatula, Jena D. Hwang, and Yejin Choi. 2021. Reflective decoding: Beyond unidirectional generation with off-the-shelf language models. In *ACL/IJCNLP*.

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 (EMNLP): System Demonstrations*.

Hao Zhang and Daniel Gildea. 2006. Efficient search for inversion transduction grammar. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 224–231.

Maosen Zhang, Nan Jiang, Lei Li, and Yexiang Xue. 2020. Language generation via combinatorial constraint satisfaction: A tree search enhanced Monte-Carlo approach. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1286–1298, Online. Association for Computational Linguistics.

Renjie Zheng, Mingbo Ma, Baigong Zheng, Kaibo Liu, and Liang Huang. 2020. Opportunistic decoding with timely correction for simultaneous translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 437–442.

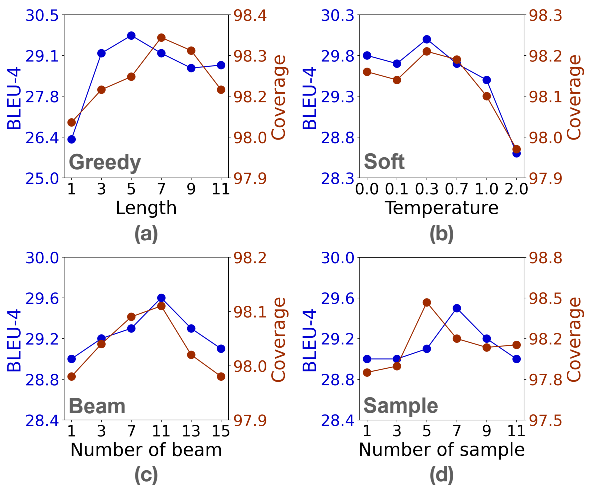


Figure 3: Effect of varying the primary hyperparameter for each lookahead strategy (§2.1) – (a) greedy (lookahead length), (b) soft (temperature), (c) beam (number of beams), and (d) sample (number of samples). Performance is measured on the CommonGen validation set, using **BLEU-4** and **Coverage**.

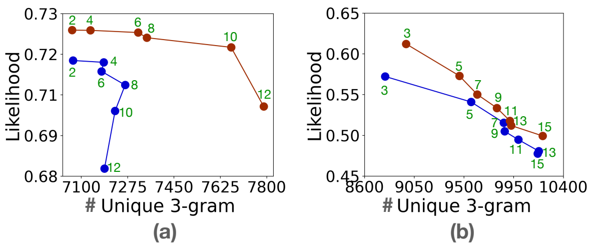


Figure 4: Likelihood (y-axis) vs. number of unique 3-grams (x-axis) using supervised GPT-2 on RocStories. Figure **(a)** denotes decoding with beam search, with a varying amount of beam size. Figure **(b)** denotes decoding with top-k sampling, with a varying amount of k value. The **brown** and **blue** lines denote with and without A★esque heuristics separately.

A Further Experiments

A.1 Constrained Commonsense Generation

Studying Lookahead Strategies. We further use CommonGen to study the lookahead strategies for NeuroLogic★ proposed in §2.1 (Figure 3). With infinite lookahead length ℓ and number of lookaheads $|\mathcal{L}_\ell|$, lookahead decoding exactly solves Equation 3, finding an optimal trajectory. In practice these are finite, meaning that the quality of the lookahead approximation can depend on the lookahead strategy and its hyperparameters. For practical choices of ℓ and $|\mathcal{L}_\ell|$, we empirically study how varying the lookahead strategy and hyperparameters affects performance. In Figure 3, we study the greedy, soft, beam, and sampling lookahead strategies.

Figure 3(a) shows the effect of increasing the lookahead length ℓ for the greedy lookahead strategy. Increasing the length improves up to one point – e.g., 5-7 steps – then decreases thereafter, likely due to the difficulty of long-horizon approximation.

Figure 3(b) studies the temperature in the soft lookahead, showing that greedy ($\tau = 0.0$) performs well, with slight gains if τ is carefully selected. The results suggest that one can safely bypass tuning τ using fast, greedy lookahead.

Next, Figure 3(c) shows that with beam lookahead, increasing the beam width improves performance up to a certain point (here, 11). Similarly, increasing the number of samples with sampling lookahead improves over a single sample, and then reaches an inflection point (Figure 3(d)).

A.2 *Unconstrained* Story Generation

Fluency and Diversity Tradeoff We study the effect of A★esque decoding in unconstrained generation with different decoding hyperparameters: beam size in beam search and k value in top-k sampling. Figure 4 plots the fluency (measured by likelihood) versus diversity (measured by unique 3-grams) for generations with various beam sizes or top-k values. Ideally, we want generations to be both fluent and diverse (top right). However, we observe a fluency and diversity tradeoff in practice. A★esque decoding flattens this trend and results in larger area under the curve. The effect is especially strong with beam search. In summary, A★esque decoding yields a more favorable balance of fluency and diversity compared to conventional decoding methods, regardless of hyperparameters.

B Runtime

Decoding Method	Runtime
Beam Search	0.20
NeuroLogic	2.04
NeuroLogic A★esque	19.24

Table 9: Runtime (seconds per sentence) of different decoding algorithms with finetuned GPT2-L on the CommonGen dataset

C Experimental Details

C.1 Off-the-Shelf Models

We download off-the-shelf models, including pre-trained GPT-2 and Marian MT, from HuggingFace

Transformers (Wolf et al., 2020), which are implemented in the PyTorch deep learning framework.

C.2 Model Training Details

All training is performed on a single NVIDIA Quadro RTX 8000 GPU and costs about 100 GPU hours in total. Our method is implemented with PyTorch an the Huggingface Transformers library.

C.2.1 CommonGen

For supervised setting, we finetune GPT-2 for conditional generation. We follow Lu et al. (2021)'s setup and use their hyperparameters for finetuning, as shown in Table 10.

Hyperparameter	Assignment
model	GPT2-Large
number of parameters	774M
number of steps	15 epochs
batch size	64
learning rate optimizer	Adam
Adam epsilon	1e-8
Adam initial learning rate	1e-5
learning rate scheduler	linear with warmup
warmup steps	1.5 epoch
weight decay	0

Table 10: Hyperparameters for finetuning GPT-2 on CommonGen dataset.

C.2.2 Constrained Machine Translation

For fair comparison, we reproduced MT model (two-layer transformer) used by Dinu et al. (2019), using the same setup and hyperparameters reported in their original paper.

C.2.3 Table-to-text Generation

We finetune GPT-2 with random sampled few-shot training instances from E2ENLG dataset. We used the same hyperparameters for finetuning with Li and Liang (2021), as shown in Table 11.

Hyperparameter	Assignment
model	GPT2-Large
number of parameters	774M
number of steps	5 epochs
batch size	5
learning rate optimizer	Adam
Adam epsilon	1e-8
Adam initial learning rate	5e-5
learning rate scheduler	linear with warmup
warmup steps	100
weight decay	0

Table 11: Hyperparameters for finetuning GPT-2 on E2ENLG dataset.

C.2.4 Unconstrained Story Generation

We finetune GPT-2 for conditional story generation on the RocStories dataset: given a first sentence $\mathbf{x}$, generate the full story $\mathbf{y}$. Hyperparameters for finetuning are given in Table 12.

Hyperparameter	Assignment
model	GPT2-Large
number of parameters	774M
number of steps	10 epochs
batch size	64
learning rate optimizer	Adam
Adam epsilon	1e-8
Adam initial learning rate	1e-5
learning rate scheduler	linear with warmup
warmup steps	1 epoch
weight decay	0

Table 12: Hyperparameters for finetuning GPT-2 on the RocStories dataset.

C.3 Generation Details

All generation is performed on a single NVIDIA Quadro RTX 8000 GPU and costs about 100 GPU hours in total.

C.3.1 CommonGen

NeuroLogic★ hyperparameters for CommonGen in supervised and zero-shot setting are shown in Table 13 and Table 14 separately. We use the same NeuroLogic hyperparameters with Lu et al. (2021), including beam size, α, β and λ_1. We performed a hyperparameter grid search for the scaling factor λ_2 over the range $[0, 0.3]$, for the look ahead step over the the range $[1, 15]$, for the look ahead temperature over the the range $[0, 1.0]$, for the look ahead beam width over the the range $[1, 10]$, and for the look ahead number of sample over the the range $[1, 10]$, using a small subset of CommonGen development set.

Hyperparameter	Assignment
beam size	20
pruning threshold α	50
pruning threshold β	2
scaling factor λ_1	0
scaling factor λ_2	0.25
look ahead step	5
look ahead (greedy) temperature	0
look ahead (beam) beam width	5
look ahead (sample) number of sample	4

Table 13: NeuroLogic★ hyperparameters for CommonGen in supervised setting.

Hyperparameter	Assignment
beam size	20
pruning threshold α	500000
pruning threshold β	2
scaling factor λ_1	0
scaling factor λ_2	0.175
look ahead step	5
look ahead (greedy) temperature	0

Table 14: NEUROLOGIC★ hyperparameters for COMMONGEN in zero-shot setting.

C.3.2 Constrained Machine Translation

NEUROLOGIC★ hyperparameters for constrained machine translation are shown in Table 15. We use the same beam size with Dinu et al. (2019) for fair comparison. We performed a hyperparameter grid search for the pruning threshold α over the range $[50, 300]$, for the pruning threshold β over the range $[1, 3]$, for the scaling factor λ_1 over the range $[0, 1.0]$, for the scaling factor λ_2 over the range $[0, 0.3]$, for the look ahead step over the the range $[5, 40]$, using a subset of WMT2013 IATE development set. We use the same hyperparameters for look ahead temperature, look ahead beam width, and look ahead number of sample with supervised COMMONGEN and omit the hyperparameter search due to the computational cost.

Hyperparameter	Assignment
beam size	5
pruning threshold α	200
pruning threshold β	2
scaling factor λ_1	0.25
scaling factor λ_2	0.05
look ahead step	35
look ahead (greedy) temperature	0
look ahead (beam) beam width	5
look ahead (sample) number of sample	4

Table 15: NEUROLOGIC★ hyperparameters for constrained machine translation.

C.3.3 Table-to-text Generation

NEUROLOGIC★ hyperparameters for table-to-text generation are shown in Table 16. We performed a hyperparameter grid search for the scaling factor λ_2 over the range $[0, 0.3]$, for the look ahead step over the the range $[1, 15]$, using E2ENLG development set. For other hyperparameters, we use the same value with supervised COMMONGEN and omit the hyperparameter search due to the computational cost.

Hyperparameter	Assignment
beam size	20
pruning threshold α	50
pruning threshold β	2
scaling factor λ_1	0
scaling factor λ_2	0.05
look ahead step	7
look ahead (greedy) temperature	0
look ahead (beam) beam width	5
look ahead (sample) number of sample	4

Table 16: NEUROLOGIC★ hyperparameters for table-to-text generation.

C.3.4 Constrained Question Generation

NEUROLOGIC★ hyperparameters for constrained question generation are shown in Table 17. The task is zero-shot and doesn't provide train or development set, so we use the same decoding hyperparameters with zero-shot COMMONGEN.

Hyperparameter	Assignment
beam size	20
pruning threshold α	500000
pruning threshold β	2
scaling factor λ_1	0
scaling factor λ_2	0.175
look ahead step	5
look ahead (greedy) temperature	0
look ahead (beam) beam width	5
look ahead (sample) number of sample	4

Table 17: NEUROLOGIC★ hyperparameters for constrained question generation.

C.3.5 Unconstrained Story Generation

A★esque hyperparameters with beam search and top-k sampling for unconstrained story generation are shown in Table 18 and Table 19 separately. We performed a hyperparameter grid search for the scaling factor λ_2 over the range $[0, 1.0]$, for the look ahead step over the the range $[1, 15]$, for the look ahead temperature over the the range $[0, 1.0]$, for the look ahead beam width over the the range $[1, 15]$, and for the look ahead number of sample over the the range $[1, 15]$, using a small subset of RocStories development set.

C.4 Dataset Details

Details of datasets used for downstream tasks are provided in Table 22.

D Human Evaluation

We include screenshots of the human evaluation templates for CommonGen (Figure 5), Constrained

Hyperparameter	Assignment
beam size	4
scaling factor λ_2	0.6
look ahead step	4
look ahead (greedy) temperature	0
look ahead (beam) beam width	4
look ahead (sample) number of sample	15

Table 18: A$^\bigstar$esque hyperparameters with beam search for unconstrained story generation.

Hyperparameter	Assignment
k value	5
scaling factor λ_2	0.5
look ahead step	3
look ahead (greedy) temperature	0
look ahead (beam) beam width	4
look ahead (sample) number of sample	15

Table 19: A$^\bigstar$esque hyperparameters with top-k sampling for unconstrained story generation.

Question Generation (Figure 6), and RocStories (Figure 7) tasks. We ensure the annotators are paid adequately for at least $15 per hour and we inform annotators that their annotations are used for model evaluation purpose.

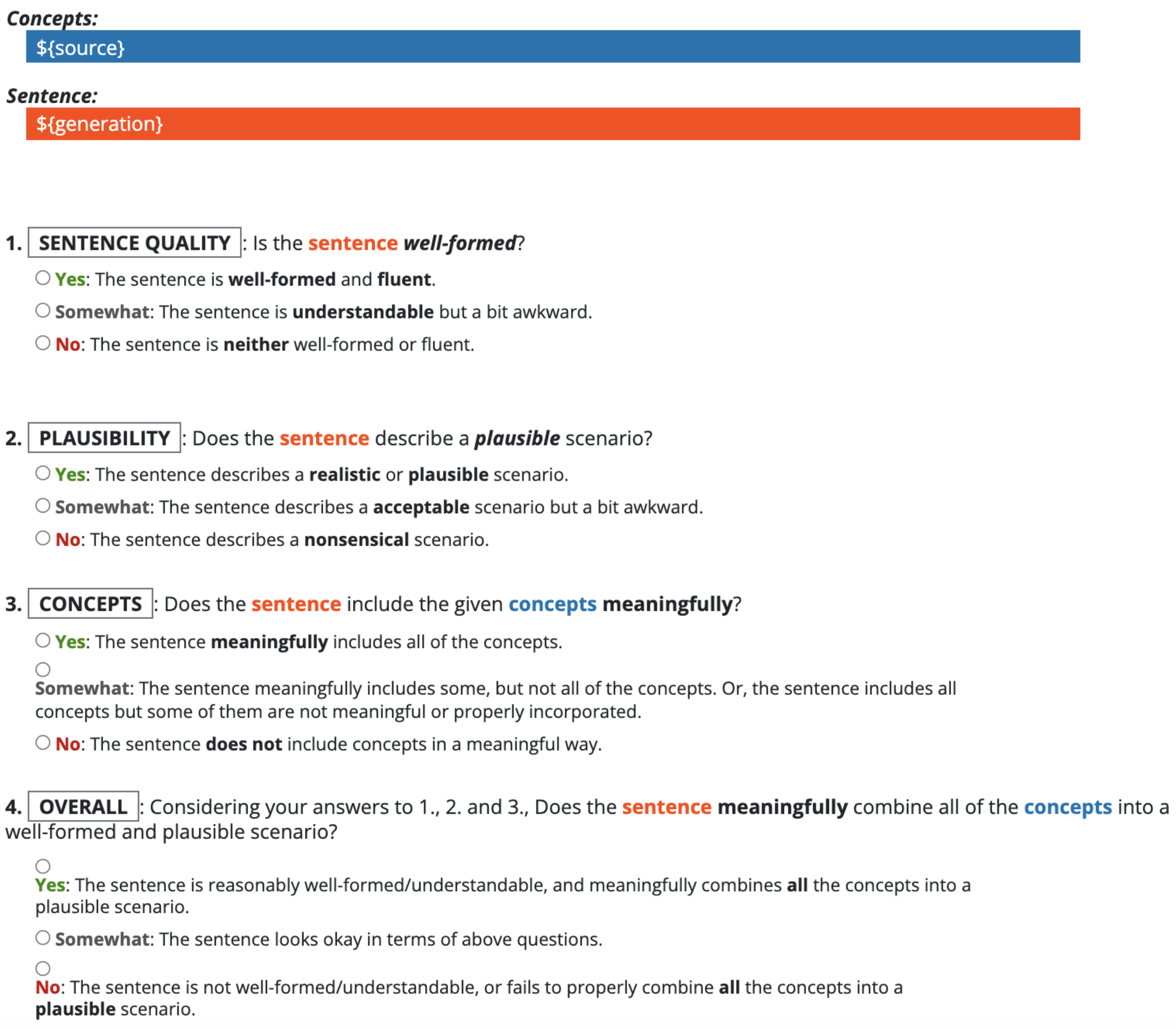
E Qualitative Generation Examples

Qualitative examples of the constrained question generation and unconstrained story generation are shown in Table 21 and 20.

F Limitations and Risks.

Limitations. For constrained generation, NEUROLOGIC A$^\bigstar$esque decoding can only take the constraints that can be formulated as logical expressions as described in the paper; we leave it to future work to expand the scope of such logical constraints.

Risks. Constrained generation methods could foreseeably be used to generate text that contains biased, offensive, and/or hateful keywords. (e.g., extremist texts; McGuffie and Newhouse, 2020). For a broader discussion of these risks, and of the risks of large pretrained language models in general, refer to discussions in Brown et al. (2020); Bender et al. (2021).



Concepts:

${source}

Sentence:

${generation}

1. **SENTENCE QUALITY** : Is the **sentence** ***well-formed***?

○ **Yes**: The sentence is **well-formed** and **fluent**.

○ **Somewhat**: The sentence is **understandable** but a bit awkward.

○ **No**: The sentence is **neither** well-formed or fluent.

2. **PLAUSIBILITY** : Does the **sentence** describe a ***plausible*** scenario?

○ **Yes**: The sentence describes a **realistic** or **plausible** scenario.

○ **Somewhat**: The sentence describes a **acceptable** scenario but a bit awkward.

○ **No**: The sentence describes a **nonsensical** scenario.

3. **CONCEPTS** : Does the **sentence** include the given **concepts** **meaningfully**?

○ **Yes**: The sentence **meaningfully** includes all of the concepts.

○
Somewhat: The sentence meaningfully includes some, but not all of the concepts. Or, the sentence includes all concepts but some of them are not meaningful or properly incorporated.

○ **No**: The sentence **does not** include concepts in a meaningful way.

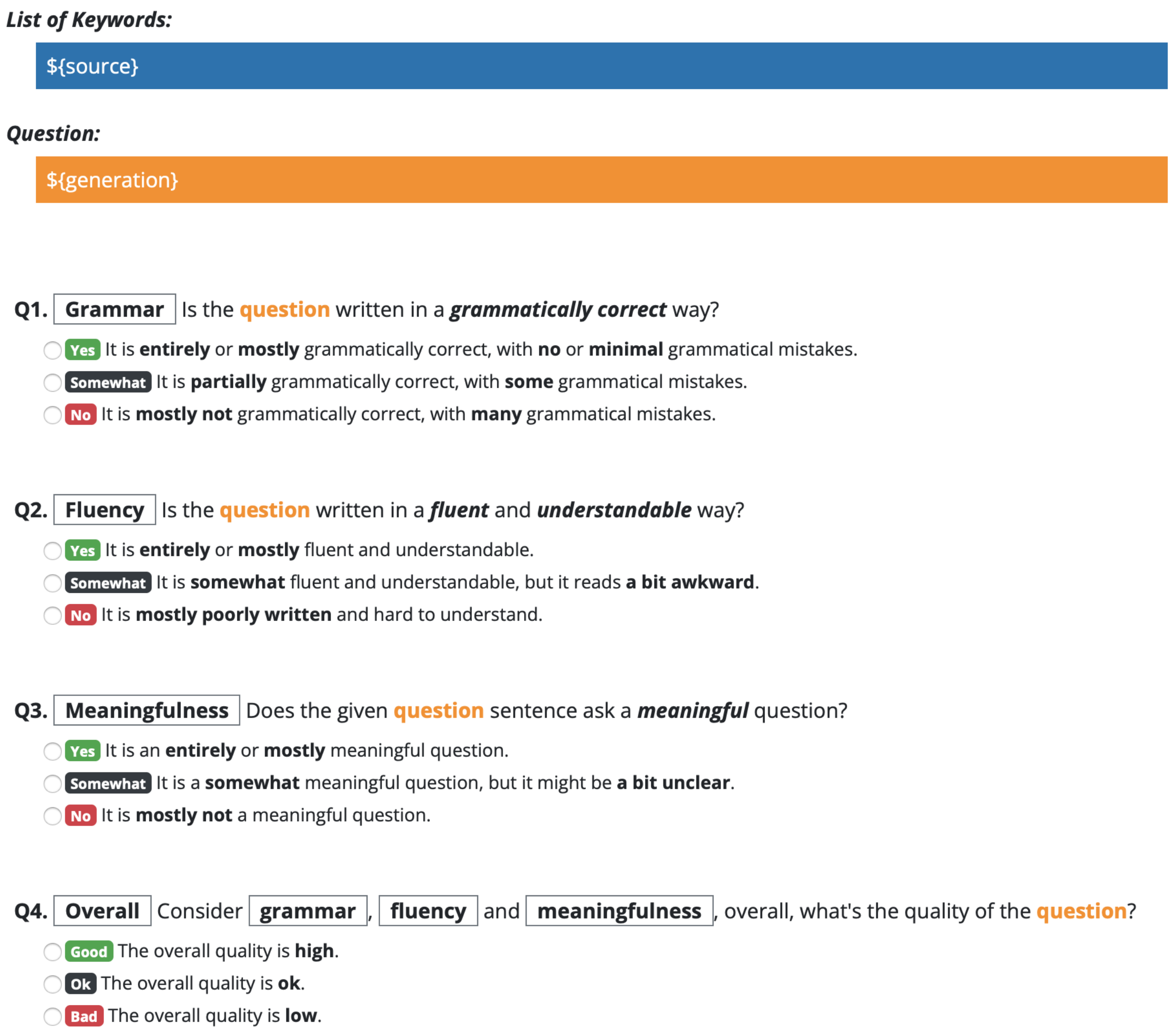
4. **OVERALL** : Considering your answers to 1., 2. and 3., Does the **sentence** **meaningfully** combine all of the **concepts** into a well-formed and plausible scenario?

○
Yes: The sentence is reasonably well-formed/understandable, and meaningfully combines **all** the concepts into a plausible scenario.

○ **Somewhat**: The sentence looks okay in terms of above questions.

○
No: The sentence is not well-formed/understandable, or fails to properly combine **all** the concepts into a **plausible** scenario.

Figure 5: Human evaluation template for the Constrained Commonsense Generation task.



List of Keywords:

${source}

Question:

${generation}

Q1. **Grammar** Is the **question** written in a ***grammatically correct*** way?

- ○ **Yes** It is **entirely** or **mostly** grammatically correct, with **no** or **minimal** grammatical mistakes.
- ○ **Somewhat** It is **partially** grammatically correct, with **some** grammatical mistakes.
- ○ **No** It is **mostly not** grammatically correct, with **many** grammatical mistakes.

Q2. **Fluency** Is the **question** written in a ***fluent*** and ***understandable*** way?

- ○ **Yes** It is **entirely** or **mostly** fluent and understandable.
- ○ **Somewhat** It is **somewhat** fluent and understandable, but it reads **a bit awkward**.
- ○ **No** It is **mostly poorly written** and hard to understand.

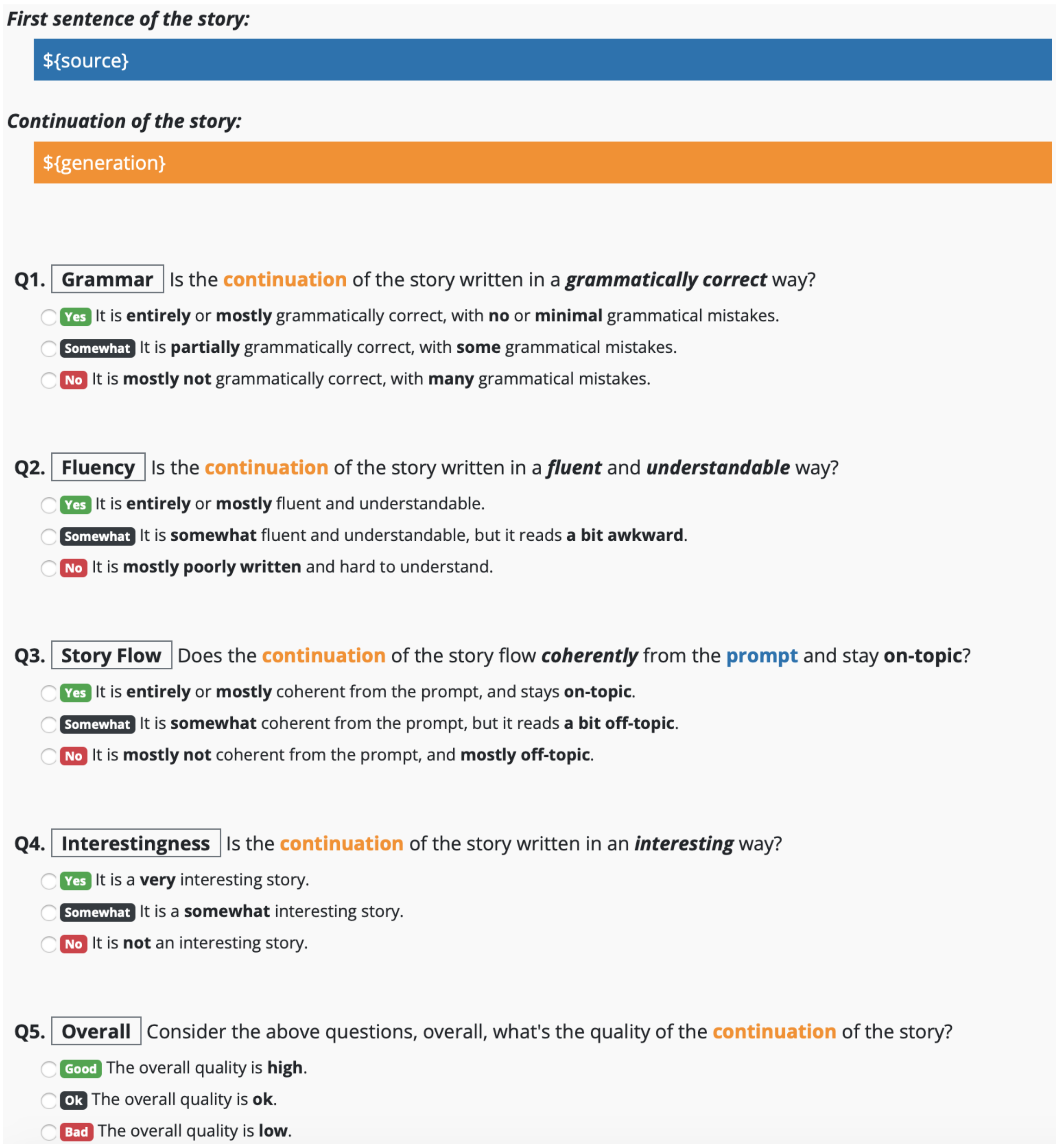
Q3. **Meaningfulness** Does the given **question** sentence ask a ***meaningful*** question?

- ○ **Yes** It is an **entirely** or **mostly** meaningful question.
- ○ **Somewhat** It is a **somewhat** meaningful question, but it might be **a bit unclear**.
- ○ **No** It is **mostly not** a meaningful question.

Q4. **Overall** Consider **grammar**, **fluency** and **meaningfulness**, overall, what's the quality of the **question**?

- ○ **Good** The overall quality is **high**.
- ○ **Ok** The overall quality is **ok**.
- ○ **Bad** The overall quality is **low**.

Figure 6: Human evaluation template for the Interrogative Sentence Generation task.



First sentence of the story:

${source}

Continuation of the story:

${generation}

Q1. **Grammar** Is the **continuation** of the story written in a ***grammatically correct*** way?

- ○ **Yes** It is **entirely** or **mostly** grammatically correct, with **no** or **minimal** grammatical mistakes.
- ○ **Somewhat** It is **partially** grammatically correct, with **some** grammatical mistakes.
- ○ **No** It is **mostly not** grammatically correct, with **many** grammatical mistakes.

Q2. **Fluency** Is the **continuation** of the story written in a ***fluent*** and ***understandable*** way?

- ○ **Yes** It is **entirely** or **mostly** fluent and understandable.
- ○ **Somewhat** It is **somewhat** fluent and understandable, but it reads **a bit awkward**.
- ○ **No** It is **mostly poorly written** and hard to understand.

Q3. **Story Flow** Does the **continuation** of the story flow ***coherently*** from the **prompt** and stay **on-topic**?

- ○ **Yes** It is **entirely** or **mostly** coherent from the prompt, and stays **on-topic**.
- ○ **Somewhat** It is **somewhat** coherent from the prompt, but it reads **a bit off-topic**.
- ○ **No** It is **mostly not** coherent from the prompt, and **mostly off-topic**.

Q4. **Interestingness** Is the **continuation** of the story written in an ***interesting*** way?

- ○ **Yes** It is a **very** interesting story.
- ○ **Somewhat** It is a **somewhat** interesting story.
- ○ **No** It is **not** an interesting story.

Q5. **Overall** Consider the above questions, overall, what's the quality of the **continuation** of the story?

- ○ **Good** The overall quality is **high**.
- ○ **Ok** The overall quality is **ok**.
- ○ **Bad** The overall quality is **low**.

Figure 7: Human evaluation template for the RocStories task.

Prompt	Method	Generation
I wanted to buy a video game console.	beam search	I went to the store and looked at all the options. I decided to go with the PS4. I went home and waited for the PS4 to arrive. When the PS4 arrived, I was very excited.
	beam search + A★esque	I went to the store and looked at all the options. I decided to go with the PS4 because it was the most expensive. When I got home, I plugged the PS4 into my TV. The PS4 turned on and I was happy.
My roommate is from Germany.	beam search	She moved to the United States when she was 16. She has been in the United States for 10 years now. She is a great roommate.
	beam search + A★esque	He came to the United States when he was 10 years old. He had a hard time adjusting to American culture. One day, he decided to move back to Germany. He was very happy to be back in his homeland.
Oliver was nervous about his wedding.	beam search	He had never been to a wedding before. He decided to go to the rehearsal dinner. The rehearsal dinner went very well. Oliver had a great time at his wedding.
	beam search + A★esque	He had never been married before and didn't know what to expect. When the day of the wedding came, he was very nervous. When he walked down the aisle, he knew it was going to be great. Oliver and his wife had a wonderful wedding.
My friends all love to go to the club to dance.	beam search	One day I decided to go with them. We went to the club and had a great time. We danced all night and had a great time. We decided to go back next week.
	beam search + A★esque	One day, I decided to go with them. We went to the club and danced all night. When we got home, I told my friends about the fun we had. They all agreed that it was the best night of their lives.

Table 20: Example generations for unconstrained story generation using beam search with and without A★esque.

Words	Method	Generation
waste water heat	CGMH	what waste is there, it seems now?
	TSMH	where was the waste - water heater?
	NEUROLOGIC	How much water is waste heat?
	NEUROLOGIC★	Why do we waste so much water to heat our homes?
Naples plague killed	CGMH	when would she finally turn twenty - one?
	TSMH	why was the plague epidemic in naples not in fact killed?
	NEUROLOGIC	Who was killed in the plague in Naples?
	NEUROLOGIC★	How many people are killed by the plague in Naples?
controversial aspect imperialism	CGMH	what war was ever fought after american imperialism collapsed?
	TSMH	what are some controversial aspects of present - day american imperialism?
	NEUROLOGIC	Whose imperialism is it, anyway?
	NEUROLOGIC★	What is the most controversial aspect of imperialism?
engines efficient steam	CGMH	or were they the very first steam engines efficient enough for mass - production?
	TSMH	why are steam engines so energy-efficient, just like fossil fuels?
	NEUROLOGIC	Why do you think steam engines are so efficient?
	NEUROLOGIC★	Why are steam engines so efficient?

Table 21: Example generations for constrained question generation with NEUROLOGIC★and baselines, including CGMH (Miao et al., 2019), TSMH (Zhang et al., 2020) and NEUROLOGIC (Lu et al., 2021).

Dataset	train	dev.	test
COMMONGEN (Lin et al., 2020)	32,651	993	1,497
WMT2013/2017 IATE (Dinu et al., 2019)	-	581	414
E2ENLG (Dušek et al., 2018)	4,862	547	630
Interrogative question (Zhang et al., 2020)	-	-	300
RocStories (Mostafazadeh et al., 2016)	45,496	1,871	1,871

Table 22: Details of datasets in downstream tasks.