Mention Flags (MF): Constraining Transformer-based Text Generators

Yufei Wang[1], **Ian D. Wood**[1,2], **Stephen Wan**[2], **Mark Dras**[1] and **Mark Johnson**[3]
Macquarie University, Sydney, Australia[1]
CSIRO Data61, Sydney, Australia[2]
Oracle Digital Assistant, Oracle Corporation[3]
yufei.wang@students.mq.edu.au, ian.wood@mq.edu.au
stephen.wan@data61.csiro.au, mark.dras@mq.edu.au
mark.mj.johnson@oracle.com

Abstract

This paper focuses on *Seq2Seq* (S2S) constrained text generation where the text generator is constrained to mention specific words, which are inputs to the encoder, in the generated outputs. Pre-trained S2S models such as T5 or a Copy Mechanism can be trained to copy the surface tokens from encoders to decoders, but they cannot guarantee constraint satisfaction. Constrained decoding algorithms always produce hypotheses satisfying all constraints. However, they are computationally expensive and can lower the generated text quality. In this paper, we propose Mention Flags (**MF**), which trace whether lexical constraints are satisfied in the generated outputs of an S2S decoder. The **MF** models are trained to generate tokens until all constraints are satisfied, guaranteeing high constraint satisfaction. Our experiments on the Common Sense Generation task (*CommonGen*) (Lin et al., 2020), End2end Data-to-Text task (*E2ENLG*) (Dušek et al., 2020) and Novel Object Captioning task (*nocaps*) (Agrawal et al., 2019) show that the **MF** models maintain higher constraint satisfaction and text quality than the baseline models and other constrained text generation algorithms, achieving state-of-the-art performance on all three tasks. These results are achieved with a much lower run-time than constrained decoding algorithms. We also show that the **MF** models work well in the low-resource setting. [1]

1 Introduction

This paper focuses on *Seq2Seq* (S2S) constrained text generation where a set of encoder input tokens are required to be present in the generated outputs. For example, Keyword-to-Text (Lin et al., 2020), Data-to-Text (Gardent et al., 2017; Dušek et al., 2020) and Image-to-Text (Lin et al., 2014; Agrawal et al., 2019) require the models to mention all or some of the input keywords, key-value pairs and image object labels (respectively), potentially with linguistic variants, in the generated outputs. Large (pre-trained) Transformer-based S2S models such as T5 (Raffel et al., 2019) can be trained (fine-tuned) to perform this task. However, they only learn to copy the surface tokens from encoder inputs to the decoder outputs and there is no underlying mechanism guaranteeing good *constraint satisfaction* (the ratio of satisfied lexical constraints to given lexical constraints). Constrained Beam Search (CBS) (Anderson et al., 2017) and related algorithms can guarantee outputs satisfying all constraints, however they are much slower than the standard beam search algorithm. In addition, as they are all inference-based algorithms, their corresponding models are not aware of the

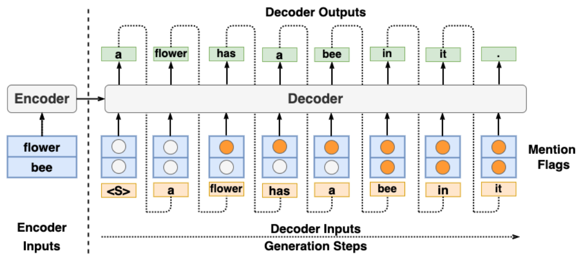


Figure 1: An overview of the Mention Flag mechanism for Transformer-based S2S models. Here, the tokens *flower* and *bee* are required to appear in the generated outputs. Each generated token has a corresponding set of Mention Flags which informs the decoder whether each lexical constraint has been satisfied in the current decoder input sequence. For example, the Mention Flag for *flower* is set (indicated by orange dots) from the third token because it is generated at the second step. Both token and Mention Flag embeddings are the input to the decoder, but Mention Flags are injected into the decoder in a different way to the tokens (see Fig. 3). Note that task specific encoder inputs have been omitted for brevity.

[1]The source code for this paper is released at https://github.com/GaryYufei/ACL2021MF

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 103–113
August 1–6, 2021.

constraint words or phrases, the resulting generation could be poor. Ideally, a method for producing constrained text should: ***a)*** generate high-quality text; ***b)*** achieve high constraint satisfaction; ***c)*** have an efficient inference procedure.

To this end, we propose Mention Flags (**MF**), which trace whether a lexical constraint has been realized in partial decoder outputs. Specifically, each decoder input token is provided with a set of flags indicating which constraints have been satisfied up to that token. As shown in Fig 1, the Mention Flags for *flower* is set from the third step, because *flower* is generated at the second step. We represent the three possible Mention Flags as separate trainable embeddings and inject them into the decoder of the S2S Transformer-based Text generator. The dynamic Mention Flags explicitly inform the model about which constraints have been satisfied, which is helpful for the models to produce high-quality text satisfying the constraints (Goal ***a***). During training, all the mention flags are set when the model is tasked to generate the *End-of-Sequence* (EOS) token, strongly encouraging the model not to stop generation until all constraints are satisfied (Goal ***b***). The **MF** models only require ordinary decoding algorithms. Their inference time and memory requirements are similar to their baseline models (Goal ***c***).

We conduct experiments on three benchmarks: Commonsense Generative Reasoning (*CommonGen*) (Lin et al., 2020), where the only input is a set of words representing concepts, and the output text is constrained to include all of them; End-to-End Data-to-Text (*E2ENLG*) (Dušek et al., 2020), where the constraints are meaning representations with lexicalised attributes and values that the output text should mention; and Novel Object Captioning at scale (*nocaps*) (Agrawal et al., 2019), where constraints are salient image objects that should be mentioned in the generated caption. Compared to the constrained decoding algorithms, the **MF** models can produce higher-quality text with a similar level of constraint satisfaction and much less inference run-time and memory. Mention Flags are a general mechanism that improves constraint satisfaction in the non-pre-trained and pre-trained S2S Transformer-based models. Furthermore, our experiments show that the **MF** models can satisfy *novel constraints* (i.e, involving words or phrases not seen during training) and they work well in low-resource settings. Our **MF** models set a new state-of-the-art in these three tasks.

2 Background

In this paper, we focus on constraining transformer-based text generation models due to their popularity and success in various domains, especially in large-scale pre-trained language models (Raffel et al., 2019; Lewis et al., 2020). Previous work can be roughly categorized into two streams: S2S training approaches and Constrained decoding approaches:

Training S2S Models S2S models can implicitly capture the co-occurrence between encoder and decoder sequences, particularly pre-trained ones such as T5 (Raffel et al., 2019) and BART (Lewis et al., 2020). Wen et al. (2015) uses a special gate to control what information will be generated in the following steps. Kale and Rastogi (2020) have shown that the T5 models achieve state-of-the-art results in various Data-to-Text tasks, requiring copying from encoder to decoder, after fine-tuning. As an alternative, the Copy Mechanism (Gu et al., 2016) explicitly learns where to copy the input constraints into the output by adding an extra copy pathway to the models. However, these approaches cannot control or guarantee their constraint satisfaction. Lin et al. (2020) also have observed lower constraint satisfaction in the above methods, compared to the constrained decoding approaches.

Constrained Decoding These algorithms, including Constrained Beam Search (CBS) (Anderson et al., 2017) and Grid Beam Search (GBS) (Hokamp and Liu, 2017), maintain a set of states which have their own size-k beams and only allow hypotheses satisfying specific constraints to be considered during inference. Each CBS state corresponds to the hypotheses satisfying different constraints (exponential in the number of constraints) and the GBS states correspond to the hypotheses satisfying the same number of constraints (linear to constraint number). Balakrishnan et al. (2019); Juraska et al. (2018); Dušek and Jurčíček (2016) also modify their inference algorithm in a similar way to fulfill specific output requirements. However, they significantly increase the inference run-time and memory and can produce sub-optimal outputs.

3 Method

This section first formulates constrained text generation tasks, then introduces Mention Flags and their integration with Transformer-based text generators.

3.1 S2S Constrained Text Generation

In the S2S constrained text generation tasks, we are given encoder inputs $\boldsymbol{x} = [x_1, \ldots, x_{l_x}] \in \mathbb{X}$ that describe the task, where some x_i correspond to lexical constraints that must be satisfied in the generated outputs. At generation step t, the decoder takes as input the tokens generated so far $\boldsymbol{y}_{:t} = [y_1, \cdots, y_t] \in \mathbb{Y}$ and generates the next output token y_{t+1}.

3.2 Mention Flag

At generation step t, a set of Mention Flags indicates whether each lexical constraint has been satisfied up to this step (i.e., in the decoder input sequence $\boldsymbol{y}_{:t}$). Formally, they can be defined as $\mathrm{m} : \mathbb{X} \times \mathbb{Y} \rightarrow \{0, 1, 2\}^{l_x}$ where $|\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})| = |\boldsymbol{x}|$. Specifically, Mention Flag $\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})_i$ is for the input token x_i in $\boldsymbol{x}$:

$$\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})_i = \begin{cases} 0 & x_i \text{ is not a constraint} \\ 1 & x_i \text{ is not mentioned in } \boldsymbol{y}_{:t} \\ 2 & x_i \text{ is mentioned in } \boldsymbol{y}_{:t} \end{cases} \quad (1)$$

The values 1 and 2 represent the status of constraint satisfaction. Once $\boldsymbol{y}_{:t}$ satisfies the constraints, the value of the corresponding Mention Flag(s) are updated from 1 to 2. Value 0 is a static default value for all tokens x_i that do not correspond to any constraints. They are not required to be mentioned in the outputs. These typically act as instructions to the model. At the start, Mention Flags $\mathrm{m}(\boldsymbol{x}, \varepsilon) \in \{0, 1\}^{l_x}$ where ε is the empty string because the empty string does not mention anything. During generation, m is monotonic in $\boldsymbol{y}_*$: given decoder input sequence $\boldsymbol{y}_{:t}$ and $\boldsymbol{y}_{:(t+1)}$, $\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})_i \leq \mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:(t+1)})_i$. The Mention Flags for any token x_i can only remain unchanged or update from value 1 to 2.

Example In Figure 2, given encoder input tokens $\boldsymbol{x}$ = [name, Tetas, area, South, Bank], we start from $\mathrm{m}(\boldsymbol{x}, \varepsilon) = [0, 1, 0, 1, 1]$ because *name* and *area* are not lexical constraints. At step 4, $\mathrm{m}(\boldsymbol{x}$, [Tetas, is, located]$) = [0, 2, 0, 1, 1]$ because *Tetas* has already been mentioned in the current decoder input sequence [Tetas, is, located].

Value Update for Multi-Word Constraints As shown in Figure 2, Mention Flags for the tokens corresponding to the same constraint are updated together. Given encoder input tokens $x_i, \cdots, x_j$, forming a multi-word constraint, we require that $\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_*)_i = \cdots = \mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_*)_j$ for all (partial) outputs $\boldsymbol{y}_*$, and $\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})_i = \cdots = \mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})_j = 2$ *iff* $x_i, \cdots, x_j$ are mentioned in $\boldsymbol{y}_{:t}$. We use conventions from the relevant data set to determine whether a constraint is a multi-word constraint. This avoids false update when the models only generate the prefix of the constraints, rather than the full constraints. For example, given constraint "washing machine", the output could be "I put my washing in the new washing machine." The situation becomes more complicated when both *washing* and *washing machine* are given lexical constraints. When we find this case, we delay the value 2 update for *washing* until the word *in* is generated. Modern tokenization methods, such as BPE (Sennrich et al., 2016), make this situation frequent.

	$\boldsymbol{x}$ \ $\boldsymbol{y}_{:t}$	<S>	*Tetas*	*is*	*located*	*in*	*the*	*South*	*Bank*	*.*
✗	*name*	0	0	0	0	0	0	0	0	0
✓	*Tetas*	1	2	2	2	2	2	2	2	2
✗	*area*	0	0	0	0	0	0	0	0	0
✓	*South*	1	1	1	1	1	1	1	2	2
✓	*Bank*	1	1	1	1	1	1	1	2	2

Figure 2: An example of Mention Flag Matrix. ✓ for constrained encoder input tokens and ✗ for non-constrained ones. Both *name* and *area* start with value 0 because they are not parts of lexical constraints. The lexical constraints *Tetas* and *South Bank* start from Value 1. The Mention Flags are updated to value 2 when $\boldsymbol{y}_{:t}$ satisfies the constraints. The Mention Flags for multi-word constraints are updated simultaneously.

Definition of Mentions We deliberately allow a flexible notion of *mentions* in the Function $\mathrm{m}()$. We can define various types of *mentions* to fulfill the requirements of different applications and tasks. With this flexibility, the end-users can use Mention Flags in many constraint scenarios. For tasks with strict constraints, we define mentions to be the exact string match in $\boldsymbol{y}_{:t}$. Otherwise, inflectional variants or synonyms of words in the lexical constraints are allowed when checking for *mentions*. Our Mention Flag mechanism thus supports lexical constraints with multiple verbalizations. We leave more sophisticated constraints (e.g., using NLP parsers) to future work.

Mention Flag Matrix Given $\boldsymbol{x}$, $\boldsymbol{y}_{:t}$, We define the two-dimensional *Mention Flag Matrix* $F \in$

$\{0,1,2\}^{l_x \times t}$ as follows:

$$F = [\mathrm{m}(\boldsymbol{x}, \varepsilon); \mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:1}); \cdots; \mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})] \quad (2)$$

During training, given $\boldsymbol{x}$ and ground-truth output Y^{gt} (with l_{gt} tokens), we can construct the ground-truth Mention Flag Matrix $F^{gt} \in \{0,1,2\}^{l_x \times l_{gt}}$ by finding the mentioning position of tokens in the lexical constraints in Y^{gt}. F^{gt} follows the same masking strategy as the decoder input tokens $\boldsymbol{y}_{:t}$. For the tokens whose corresponding lexical constraints having no alignment with Y^{gt}, their Mention Flags are also assigned value 0. During inference, we build the Mention Flag matrix incrementally, starting from $F^{inf,0} = [\mathrm{m}(\boldsymbol{x}, \varepsilon)] \in \{0,1\}^{l_x \times 1}$. In step t, we add a new column $\mathrm{m}(\boldsymbol{x}, \boldsymbol{y}_{:t})$ to $F^{inf,t-1} \in \{0,1,2\}^{l_x \times (t-1)}$ and obtain the new Mention Flag matrix $F^{inf,t} \in \{0,1,2\}^{l_x \times t}$.

Why Mention Flags work During the training of MF models, the ground-truth always has all MFs set to "completed" before stopping the generation (i.e., before generating EOS Token). This provides a strong signal to satisfy all constraints before completing generation. The value update from 1 to 2 in MF provides implicit signals about where the constraints are satisfied during training. Otherwise, the model has to learn this information via the co-occurring sub-sequences between input sequence and output sequence. These two signals allow the model to achieve high constraint satisfaction and help to maintain high text quality (Sec. 4.5). Since there are only 3 added embeddings, learning does not require a substantial amount of training data (Sec. 4.7). Since these embeddings are independent of particular lexical constraints, we expect that performance on novel constraints, not seen during training, is improved (Sec. 4.5).

3.3 Integration with S2S Transformer

As shown in Figure 3, Mention Flags are injected into the Transformer decoder. We first review the standard S2S Transformer proposed in Vaswani et al. (2017), then discuss how to inject Mention Flags information into the S2S Transformer model.

Standard S2S Transformer Model The encoder input tokens $\boldsymbol{x}$ is fed into the Transformer Encoder $\boldsymbol{h}^e = Enc(\boldsymbol{x})$ where $\boldsymbol{h}^e \in \mathbb{R}^{l_x \times d}$ and d is the model hidden size. In the Transformer decoder, there are two self-attention modules, Self Multi-Head Attention (SA) which handles the current decoder input sequence $\boldsymbol{y}_{:t}$, and Cross Multi-Head

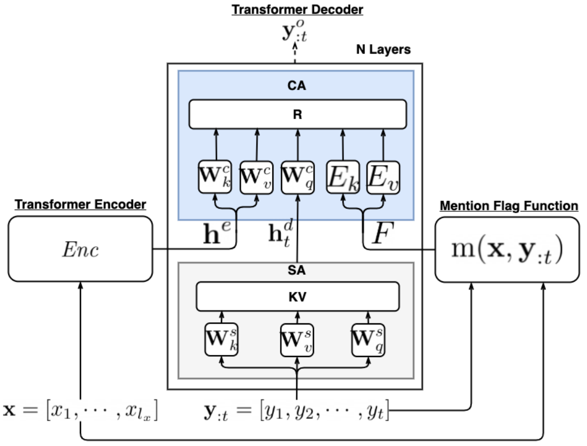


Figure 3: In each decoder layer, the Cross-Attention (CA) module (light blue) integrates Mention Flags as additional inputs describing relationship between encoder contents and decoder input tokens. There are separated representations for Mention Flags in different decoder layers.

Attention (CA) which handles the interaction between encoder output $\boldsymbol{h}^e$ and $\boldsymbol{y}_{:t}$:

$$SA(\boldsymbol{y}_{:t}) = KV(W_q^s \boldsymbol{y}_{:t}, W_k^s \boldsymbol{y}_{:t}, W_v^s \boldsymbol{y}_{:t}) \quad (3)$$

$$CA(\boldsymbol{h}_t^d, \boldsymbol{h}^e) = KV(W_q^c \boldsymbol{h}_t^d, W_k^c \boldsymbol{h}^e, W_v^c \boldsymbol{h}^e) \quad (4)$$

where $\boldsymbol{h}_t^d = SA(\boldsymbol{y}_{:t})$. KV is the standard key-value self-attention proposed in Vaswani et al. (2017). The outputs of $CA(\boldsymbol{h}_t^d, \boldsymbol{h}^e)$ further determine the model output y_{t+1} via a Feed Forward layer, a Residual Connection and a $\mathrm{softmax}$ layer.

Incorporating Mention Flag Matrix Our two-dimensional Mention Flag matrix $F \in \{0,1,2\}^{l_x \times t}$ is associated with the elements from encoder output $\boldsymbol{h}^e$ and current decoder input $\boldsymbol{y}_{:t}$. The optimal way is to incorporate the full F matrix into a component in the Transformer decoder. We note that the CA module in the Transformer decoder already uses $\boldsymbol{y}_{:t}$ as query and $\boldsymbol{h}^e$ as key. The resulting query-key similarity matrix has the same size of our Mention Flag matrix, making it suitable to incorporate F.

Mention Flag Matrix as Relative Position Inspired by Shaw et al. (2018) which incorporates token relative positions into the SA module, we propose to inject Mention Flags as the "relative positions" between encoder output $\boldsymbol{h}^e$ and current decoder input $\boldsymbol{y}_{:t}$ in the CA module. In each decoder layer, we represent F as two sets of trainable embeddings *Mention Flag key* $\boldsymbol{m}^k = E_k(F)$ and *Mention Flag Value* $\boldsymbol{m}^v = E_v(F)$ where

$E_k, E_v \in \mathbb{R}^{3\times d}$ are the Mention Flag embedding tables. $\boldsymbol{m}^k$ and $\boldsymbol{m}^v \in \mathbb{R}^{l_x \times t \times d}$. We have separated Mention Flags representations for each decoder layer. Eq. 4 is changed to:

$$CA(\boldsymbol{h}^d_t, \boldsymbol{h}^e, \boldsymbol{m}^k, \boldsymbol{m}^v) = R(W^c_q \boldsymbol{h}^d_t, W^c_k \boldsymbol{h}^e, W^c_v \boldsymbol{h}^e, \boldsymbol{m}^k, \boldsymbol{m}^v) \quad (5)$$

where R is the Self-Attention function with relative position, defined as follows:

$$R(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}, \boldsymbol{m}^k, \boldsymbol{m}^v)_j = \sum_{i=1}^{l_x} \boldsymbol{a}_{i,j}(\boldsymbol{v}_i + \boldsymbol{m}^v_{i,j}) \quad (6)$$

$$\boldsymbol{a}_{*,j} = Softmax(\boldsymbol{e}_{*,j}) \quad (7)$$

$$\boldsymbol{e}_{i,j} = \frac{\boldsymbol{q}_j(\boldsymbol{k}_i + \boldsymbol{m}^k_{i,j})^T}{\sqrt{d}} \quad (8)$$

As an alternative to representing F as $\boldsymbol{m}^k$ and $\boldsymbol{m}^v$, we could follow the approach to relative position in the T5 model (Raffel et al., 2019) and represent F as scalars that are added to the corresponding logits $\boldsymbol{e}_{i,j}$ in Eq. 7 used for computing the attention weights. However, we find this scalar approach less effective than our proposed one in Sec. 4.6.

4 Experiments

We conduct experiments on three benchmarks with different forms of constraints including Commonsense Generative Reasoning (*CommonGen*) (Lin et al., 2020) with keyword constraints, End-to-End restaurants dialog (*E2ENLG*) (Dušek et al., 2020) with key-value constraints, and Novel Object Captioning at scale (*nocaps*) (Agrawal et al., 2019) with visual object word constraints. We integrate Mention Flags with a three-layer standard S2S Transformer models (*Trans, L3*) (Vaswani et al., 2017) and pre-trained T5 models (Raffel et al., 2019) for each task. The T5 models achieve state-of-the-art results in various Data-to-Text tasks (Kale and Rastogi, 2020). For the *T5-Base* and *T5-Large* models, we use the implementation of T5 models in the *huggingface transformers* [2]. The Trans, L3 models share the same implementation of the *T5-Base* models, except that it is not initialized with the pre-trained parameters and it only uses 3 layers, rather than 12 layers, for both encoder and decoder. In addition, to improve the generalization of our pre-trained model, we freeze the parameters in the Self-Attention module and Feed-Forward Layers in each layer of the T5 decoder. This parameters freezing technology is applied to both T5 baseline models and the **MF** models in all of our experiments. We report *constraint satisfaction* for all tasks. We use GBS in the *CommonGen* task (max 5 constraints) and CBS in the *E2ENLG* (max 1 constraint) and *nocaps* (max 2 constraints) task.

4.1 *CommonGen*

In this task, the encoder input is a sequence of concepts $C = [c_1, \cdots, c_k], k \leq 5$. The models should generate a coherent sentence describing all concepts in C. $\mathrm{m}(C, \varepsilon) = [1, 1, \cdots, 1]$ and m allows inflectional variants to satisfy lexical constraints. We train (fine-tune) *Trans, L3*, *T5-Base* and *T5-Large* model as our baselines. We apply Mention Flags to the T5-Base and T5-Large model (+ **MF**). Following the suggestions in Lin et al. (2020), we report CIDEr (Vedantam et al., 2015) and SPICE (Anderson et al., 2016) as generated text quality metrics. We calculate constraint satisfaction for all constraints (ALL), novel constraints (Novel) and seen constraints (Seen).

Method	CIDEr	SPICE	Constraint Seen	Constraint Novel	Constraint ALL
w/o Pre-training					
Trans, L3	79.5	20.1	62.6	2.3	58.0
Trans, L3 + **MF**	113.9	24.6	93.8	49.2	90.4
LevenTrans.♣	74.5	16.8	-	-	63.8
ConstLeven.♣	108.0	20.1	-	-	94.5
w/ Pre-training					
T5-Base	164.4	32.1	95.7	94.6	95.6
T5-Base + G	110.7	27.8	**100**	**100**	**100**
T5-Base + **MF**	<u>170.1</u>	<u>32.7</u>	<u>99.6</u>	<u>99.2</u>	<u>99.6</u>
T5-Base + **MF** + G	115.0	27.6	**100**	**100**	**100**
T5-Large	167.3	33.0	93.9	93.8	93.9
T5-Large + **MF**	**174.8**	**33.4**	99.2	99.0	99.1
Liu et al. (2021)	168.3	32.7	-	-	98.6

Table 1: Experiment Results on *CommonGen* Test Split. The T5-Base + **MF** model achieves high text quality with high constraint satisfaction. G for GBS. ♣ results taken from Lin et al. (2020). **Bold** is the highest score and <u>underline</u> is the second highest score.

Results Table 1 shows that the **MF** model improves the constraint satisfaction over the baselines for all cases, achieving close to 100% (i.e., 99.6% and 99.1%). Notably, Mention Flags improve novel constraint satisfaction from 2.3% to 49.2% in the randomly initialized Transformer models. Compared to the LevenTrans (Gu et al., 2019) and Con-

[2] https://github.com/huggingface/transformers

stLeven (Susanto et al., 2020) models, our *Trans, L3 +* ***MF*** model achieves higher CIDEr and SPICE scores with constraint satisfaction 4.1% lower than the non-autoregressive ConstLeven model. While GBS provides a way to maximise constraint satisfaction (i.e., 100%), doing so significantly degrades the output text quality (more than 50 CIDEr). Our **MF** model achieves near optimum constraint satisfaction while improving text quality (5.7 CIDEr score improvement in *T5-Base* and 6.5 CIDEr score improvement in *T5-Large*). Finally, our *T5-Large +* ***MF*** model outperforms the previous state-of-the-art result (Liu et al., 2021), which integrates the *ConceptNet* (Speer et al., 2017) into the BART model, by 6.5 CIDEr and 0.7 SPICE, suggesting that pre-trained language models with textual concepts may provide sufficient information for this task.

4.2 *E2ENLG*

In this task, the encoder input is a sequence of key-value meaning representations $C = [k_1, v_1, \cdots, k_n, v_n], n \leq 8$. We lists all given key-value information as a space-separated string. $\mathrm{m}(C, \varepsilon) = [0, 1, 0, 1, \cdots, 0, 1]$ and m allows synonyms to satisfy lexical constraints. For example, *welcome children* and *is family friendly* are both mentions of *familyFriendly[yes]*. The models must generate a fluent and coherent dialog response using all key-value pairs in the encoder. *E2ENLG* includes 79 different in-domain key-value constraints. We use the scripts from Dušek et al. (2019) [3] to construct the synonyms set for these inputs. We use Trans, L3 and T5-Base model as our baselines. We use CBS to constrain the T5 model to satisfy all missing constraints (T5-Base + C). We report NIST (Lin and Hovy, 2003), BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005) as they are common metrics for evaluating the quality of long text in the *E2ENLG* outputs (more than 20 tokens).

Results Table 2 shows that the **MF** models consistently achieve higher output text quality and constraint satisfaction than the baseline models (99.9% vs. 95.1% and 100% vs. 96.6%). CBS improves the *T5* model's constraint satisfaction, but negatively affects the text quality (0.3 BLUE points lower). Shen et al. (2019), the previous state-of-the-art, trained the model via a complex *speaker-listener* approach inspired by cognitive science. With a much simpler model architecture (S2S), our *T5 +* ***MF*** model achieves full constraint satisfaction and outperforms Shen et al. (2019) by 0.2 NIST and 0.3 METEOR.

Method	BLEU	NIST	METEOR	Constraint
w/o Pre-training				
Trans, L3	64.7	8.5	43.8	95.1
Trans, L3 + **MF**	65.4	8.6	44.9	99.9
w/ Pre-training				
T5	67.4	8.7	45.5	96.6
T5 + CBS	67.1	8.7	45.6	**100.0**
T5 + **MF**	<u>68.3</u>	**8.9**	**45.6**	**100.0**
Shen et al. (2019)	**68.6**	8.7	45.3	-

Table 2: Experiment Results in the *E2ENLG* Test Split. The *T5 +* ***MF*** model achieves high text quality with high constraint satisfaction.

4.3 *nocaps*

Using T5 for Image Captioning In Image Captioning, each input image is represented by a sequence of visual objects. Each of these objects is assigned (by the object detector) with a textual label. The encoder input is a sequence of objects followed by the same textual labels $C = [\boldsymbol{v}_1^1, \cdots, \boldsymbol{v}_1^{s_1}, l_1, \cdots, \boldsymbol{v}_k^1, \cdots, \boldsymbol{v}_k^{s_k}, l_k]$ where $\boldsymbol{v}_i^*$ is the visual feature vector (similar to the one in Li et al. (2020)) and l_i is the corresponding textual label. The visual features are used in the same way of normal textual tokens in the T5 models. We find this approach works well for both *nocaps* and standard COCO image captioning task.

Experiment Setup Traditional image captioning models select and describe a subset of input objects jointly (Anderson et al., 2018). However, Puduppully et al. (2019) shows the benefits of separating content selection and text planning steps for general data-to-text tasks. Following this, we propose to first select salient objects and incorporate the selected objects into the description using Mention Flags. $\mathrm{m}(C, \varepsilon) = [0, 0, \cdots, 1, \cdots, 0, 0, \cdots, 1]$ where only salient object labels receive value 1. $\mathrm{m}()$ allows inflectional variants to satisfy lexical constraints. We use T5-base model in this experiment. The *T5 + C* and *T5 +* ***MF*** *+ C* models are constrained with CBS. Following Wang et al. (2021), we report CIDEr and SPICE as output text quality metrics and constraint satisfaction for novel constraints (Novel) and all constraints (ALL). We present the performance for all evaluation images

[3] https://github.com/tuetschek/e2e-cleaning/blob/master/slot_error.py

(**Overall**) and for the challenging images with only novel objects (*out-of-domain* split).

Salient Object Selector We use a transformer-based salient object detector to select a subset of object labels as lexical constraints. The visual representations of detected image objects are first fed into the 3-layer standard Transformer model without any positional embedding. We train this detector using binary Cross-Entropy loss averaged over all detected input objects. The training data for salient object detection is the training data in *nocaps*. We use COCO 2017 Dev set as the evaluation dataset to select the best checkpoint.

Method	*out-of-dom.*		**Overall**		Constraint	
	CIDEr	S	CIDEr	S	Novel	ALL
nocaps *Val. (w/o Pre-training)*						
Trans, L3	34.2	8.6	58.7	10.6	16.3	35.8
Trans, L3 + **MF**	39.8	9.1	60.4	11.2	49.3	71.5
ECOL w/o LM◇	34.8	9.2	58.0	11.2	-	-
nocaps *Val. (w/ Pre-training)*						
T5	63.4	9.9	72.7	11.3	35.8	47.5
T5 + C	**80.2**	10.5	79.2	11.6	**100**	**100**
T5 + **MF**	<u>79.9</u>	**10.8**	**79.9**	**11.9**	<u>96.9</u>	<u>98.3</u>
T5 + **MF** + G	79.6	10.6	79.2	11.8	**100**	**100**
T5 + **MF** + C	79.7	10.7	79.5	11.8	**100**	**100**
OSCAR_L + C♡	77.4	10.5	78.6	11.8	-	-
VIVO + C§	83.0	10.7	85.3	12.2	-	-
nocaps *Test*						
T5 + **MF**	71.5	10.4	77.7	12.1	96.3	97.8
UpDown (E&C)♠	66.7	9.7	73.1	11.2	-	-
ECOL + IB◇	67.0	10.3	76.0	11.9	-	-

Table 3: Evaluation Results for *nocaps*. The *T5 + **MF*** model produces high-quality text with high constraint satisfaction, setting a new state-of-the-art among the comparable previous works. C: CBS. G: GBS. S: SPICE. Con.: Constraint Satisfaction. § Hu et al. (2020), a non-comparable model that uses additional visual-text aligned training data. ♠ Agrawal et al. (2019). ♡ Li et al. (2020). ◇ Wang et al. (2021).

Results Mention Flags achieve optimal constraint satisfaction in almost all cases. In particular the *Trans, L3 + **MF*** model shows marked improvement (i.e., from 16.3% to 49.3%) on novel constraints, despite the fact that the corresponding token embeddings are not changed from their random initialisation. The generated text quality is also improved, particularly in the *out-of-domain* split. The *T5 + C* model is 0.3 SPICE lower in both overall and the *out-of-domain* split than the T5 + **MF** model, indicating that the **MF** model correctly captures more long-range relationships (calculated by the parsing trees used in SPICE) among the (novel) objects than CBS. Our *T5 + **MF*** model outperforms the existing state-of-the-art end-to-end single-stage image captioning systems (Agrawal et al., 2019; Li et al., 2020; Wang et al., 2021) by 1.3 CIDEr and 0.1 SPICE on the validation set and 1.7 CIDEr and 0.2 SPICE on the test set, showing the advantage of our two-stage captioning model empowered by Mention Flags. *VIVO + C* (Hu et al., 2020) is not comparable as it uses additional visual-text aligned training data. Finally, we investigate the relatively lower constraint satisfaction in *nocaps* (98.3% vs. 99.5+%) compared to the **MF** models in the other two tasks and find that missing cases frequently happen in the instances with two constraints involving *a)* (near-) synonymy (e.g., mule and horse) and *b)* hyponymy (e.g., hot dog and fast food). A more advanced salient object detector would solve this issue.

4.4 Model Efficiency

The **MF** models use standard beam search and run much faster with less memory than the constrained beam search algorithms. For comparison, we select the GBS algorithm because its resource use is linear in the number of constraints and uses less run time and memory than CBS. We run the **MF** models and the models with GBS using beam size 5 and compare their run time (RT) and memory requirement (#M) in Table 4. Compared to the **MF** models, GBS runs one to two orders of magnitude slower, and uses 4.4 to 23.4 times more memory. Compared to the *T5-Base* model, the **MF** models only increases the inference time slightly.

Task	*E2ENLG*		*CommonGen*		*nocaps*	
	RT	#M	RT	#M	RT	#M
T5-Base + G	438 m	16.9	645 m	23.4	93 m	4.4
T5-Base + **MF**	19 m	1	10 m	1	18 m	1
T5-Base	17 m	1	8 m	1	16 m	1

Table 4: Efficiency of the **MF** and GBS model. RT: inference Run Time (in minutes). #M: the number of GBS states (indicating the memory required).

4.5 Main Result Discussion

Constraint Satisfaction & Text Quality In all tasks, **MF** models improve the text quality over their baselines (including CBS and GBS) while achieving constraint satisfaction that is close to 100%.

This supports the claim in Sec 3.2 that training signals from Mention Flags can help to improve constraint satisfaction and text quality.

Non-Pre-trained vs. Pre-trained Models In all tasks, Mention Flags have a similar effect (higher text quality and constraint satisfaction) on both non-pre-trained and pre-trained models. This indicates that Mention Flags do not rely on information from pre-trained models to be effective.

Novel Constraints In the *CommonGen* and *nocaps* tasks, the *Trans, L3* + ***MF*** model achieve much higher coverage (i.e., 2.3% to 49.2% in *CommonGen*; 16.3% to 49.3% in *nocaps*) for constraints with novel lexical items than the baseline models. Here, the **MF** models can satisfy novel constraints, even where the corresponding token representations did not receive any training signals. As Mention Flags decouples with model representations, the **MF** models learn lexicon-independent indicators to mention the novel words.

E2ENLG	BLEU	NIST	METEOR	Con.
Scalar **MF**	67.1	8.8	45.3	100
Static **MF**	67.7	8.8	**45.8**	100
Merged **MF**	67.7	8.8	45.3	98.9
Shared **MF**	67.2	8.8	45.5	99.9
MF	**68.3**	**8.9**	<u>45.6</u>	**100.0**
CommonGen	CIDEr	SPICE	C-Novel	C-ALL
Scalar **MF**	166.9	32.7	97.5	97.1
Static **MF**	160.5	32.0	93.5	94.5
Shared **MF**	168.1	32.8	99.0	99.4
MF	**170.1**	**32.7**	**99.4**	**99.6**
nocaps	METEOR	CIDEr	SPICE	Con.
Scalar **MF**	25.3	79.3	11.8	91.5
Static **MF**	25.3	**80.4**	11.7	87.2
Shared **MF**	25.4	78.7	11.8	95.8
MF	**25.6**	79.9	**11.9**	**98.3**

Table 5: Ablation Study For **MF** Status. Static **MF** removes value 2 and Merged **MF** merges value 0 and 1. Full **MF** achieves the highest constraint satisfaction and output text quality among all other variants. Con., C-Novel, C-ALL: constraint satisfaction (resp. for novel/all constraints).

4.6 Design Choices for Mention Flags

We conduct experiments for following choices of Mention Flag: *Static **MF*** where value 2 (*is mentioned*) and 1 (*not mentioned*) are merged; *Merged **MF*** where value 0 (*not a constraint*) is merged with value 1; *Scalar **MF*** where Mention Flags are represented as scalars added to the attention logits in the *CA* module; and *Shared **MF*** where all decoder layers use the same Mention Flag embeddings. We apply *Static **MF***, *Scalar **MF*** and *Shared **MF*** to all three tasks. We only use Merged **MF** in *E2ENLG* because a *CommonGen* model does not include value 0 and a *nocaps* model without value 0 cannot distinguish between constrained and non-constrained objects. As shown in Table 5, in the *CommonGen* and *nocaps* tasks, the Static **MF** models achieve much lower constraint satisfaction, 99.6% vs. 94.5% and 98.3% vs. 87.2% respectively. The explicit update from value 1 to 2 is important for high constraint satisfaction. The merged **MF** model produces lower constraint satisfaction (100% to 98.9%) and generated text quality (68.3 BLEU to 67.7 BLEU) in *E2ENLG*, indicating the utility of value 0 in this task. Compared to the **MF** models, Scalar **MF** models produce lower constraint satisfaction in the *CommonGen* and *nocaps* task (99.6% to 97.1%, 98.3% to 91.5%, respectively) and lower-quality generated text in all three tasks (1.2 BLEU, 3.2 CIDEr and 0.6 CIDEr lower). Representing Mention Flags as Key and Value dense vectors works better than scalars. Finally, using shared **MF** across all decoder layers has negative impact (e.g., all constraint satisfaction ratio drop) in all three tasks.

4.7 Low-Resource Learning

This section shows that Mention Flags are still useful for improving the constraint satisfaction and generated text quality when trained with many fewer instances. We use 0.1%, 1% and 10% of the original training instances to train the models. In the first two tasks (*E2ENLG* and *CommonGen*), we compare the **MF** models with T5-Base models. In the *nocaps* task, we additionally compare the T5-Base + **MF** model with the T5-Base + C model. We report BLEU in *E2ENLG* CIDEr in *CommonGen* and *nocaps*. As shown in Table 6, the **MF** models consistently generate higher-quality text (higher METEOR or CIDEr Score) and achieve higher constraint satisfaction than the baseline models. The **MF** models reach 97+% when only training with 10% of the *E2ENLG* and *CommonGen* training data. This confirms our claim in Sec. 3.2 that the three added Mention Flag embeddings can be learned with relatively little training data.

4.8 Qualitative Analysis

We chose three representative examples that illustrate successful use of Mention Flags (Table 7).

Training Sample	0.1 %		1 %		10 %	
E2ENLG	BLEU	Con.	BLEU	Con.	BLEU	Con.
T5-Base	51.3	83.5	60.5	94.7	67.1	95.9
T5-Base + **MF**	**52.4**	**87.4**	**61.1**	**99.8**	**67.3**	**99.9**
CommonGen	CIDEr	Con.	CIDEr	Con.	CIDEr	Con.
T5-Base	77.9	87.2	95.4	81.5	140.6	91.1
T5-Base + **MF**	**78.5**	**89.5**	**98.7**	**85.4**	**149.4**	**97.6**
nocaps	CIDEr	Con.	CIDEr	Con.	CIDEr	Con.
T5-Base	43.5	46.2	49.4	44.0	60.8	48.2
T5-Base + C	50.7	72.4	58.7	82.8	69.3	92.7
T5-Base + **MF**	**51.7**	**72.4**	**60.2**	**82.8**	**71.9**	**92.7**

Table 6: Low-resource Learning. We use 0.1%, 1% and 10% of the training instances to train the models. Con.: constraint satisfaction.

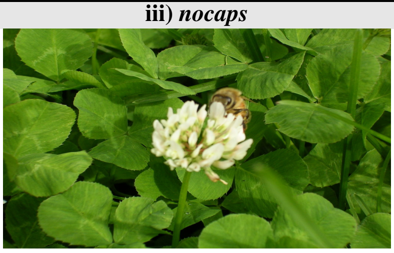
i) *E2ENLG*	
name[Punter], eatType[restaurant], area[riverside], priceRange[£20-25], familyFriendly[yes]	
T5-B	Punter is a restaurant in the £20-25 price range. It is in the riverside area
+ C	Punter is a kid friendly restaurant in the riverside area. It has a price range of £20-25.
+ **MF**	Punter is a kid friendly restaurant in riverside with a price range of £20-25
ii) *CommonGen*	
mother, washer, clothes, toddler, help	
T5-B	a mother helps a toddler to wash his clothes
+ G	mother helping her toddler clothe in washer
+ **MF**	a mother helps a toddler to wash clothes in the washer
GT	the mother helps her toddler put the clothes in the washer
iii) *nocaps*	
	
Salient Obj: *bee*, *flower*; non-Salient Obj: *plant*, *leaf*	
T5-B	a close up of a flower on a tree
+ C	a close up of a bee flower on a tree
+ **MF**	a small white flower with a bee in it
GT	a white flower has a bee on it with green around.

Table 7: Representative examples illustrate successful use of the **MF** models. GT: ground truth text. +C/+G: with constrained/grid beam search. T5-B: T5 base.

i) The **MF** model generates the most concise dialogue response, compared to the baseline and constrained decoding model; **ii)** The **MF** model is the only model that generates a fluent and coherent sentence satisfying all input constraints; **iii)** The **MF** model is the only model that accurately describes the relationship between *bee* and *flower*, grounding to the input images and constraints.

Human Evaluation We have shown that our proposed **MF** model can achieve higher constraint satisfaction ratio and automatic metrics. However, the automatic metrics do not necessarily reflect human preference of the generated text. We therefore select 100 output samples from the T5 baseline and our **MF** model in all three tasks (300 in total). For each sample pair, we ask three annotators to judge which sample is "more human-like". Table 8 shows that more than 70% of output of our **MF** model is generally better or similar than the output of the baseline model, verifying the output quality of our **MF** model.

Task	Baseline	Equal	**MF**
CommonGen	27.3%	22.0%	50.7 %
E2ENLG	30%	25%	45%
nocaps	28%	26.7%	45.3%

Table 8: Human Evaluation over output samples in the *CommonGen*, *E2ENLG* and *nocaps* task.

5 Conclusion and Future Work

In this paper, we propose Mention Flags to constrain Transformer-based text generators via injecting mention status embeddings into text decoders. Our extensive experiments on three different tasks have shown the effectiveness of Mention Flags in maintaining high generated text quality and excellent constraint satisfaction, comparing favourably to competitive constrained decoding algorithms. We plan to expand Mention Flags **i)** to control larger input source text such as constrained text summarization and machine translation; **ii)** to handle larger granularity such as sentence-level.

Acknowledgments

We thank anonymous reviewers for their insightful suggestions to improve this paper. This research was supported by a Google award through the Natural Language Understanding Focused Program, by a MQ Research Excellence Scholarship and a CSIRO's DATA61 Top-up Scholarship, and under the Australian Research Councils Discovery Projects funding scheme (project number DP160102156).

References

Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. 2019. nocaps: novel object captioning at scale. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8948–8957.

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.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Anusha Balakrishnan, Jinfeng Rao, Kartikeya Upasani, Michael White, and Rajen Subba. 2019. Constrained decoding for neural NLG from compositional representations in task-oriented dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 831–844, Florence, Italy. 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, Ann Arbor, Michigan. Association for Computational Linguistics.

Ondřej Dušek, David M. Howcroft, and Verena Rieser. 2019. Semantic noise matters for neural natural language generation. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 421–426, Tokyo, Japan. 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. 2020. Evaluating the State-of-the-Art of End-to-End Natural Language Generation: The E2E NLG Challenge. *Computer Speech & Language*, 59:123–156.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, Santiago de Compostela, Spain. Association for Computational Linguistics.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany. Association for Computational Linguistics.

Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

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.

Xiaowei Hu, Xi Yin, Kevin Lin, Lijuan Wang, Lei Zhang, Jianfeng Gao, and Zicheng Liu. 2020. Vivo: Surpassing human performance in novel object captioning with visual vocabulary pre-training. *arXiv preprint arXiv:2009.13682*.

Juraj Juraska, Panagiotis Karagiannis, Kevin Bowden, and Marilyn Walker. 2018. A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. 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 152–162, New Orleans, Louisiana. Association for Computational Linguistics.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 97–102, Dublin, Ireland. Association for Computational Linguistics.

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

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Computer Vision – ECCV 2020*, pages 121–137, Cham. Springer International Publishing.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

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

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S. Yu. 2021. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

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

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6908–6915.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

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.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. 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 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

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.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4444–4451. AAAI Press.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

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.

Yufei Wang, Ian D. Wood, Stephen Wan, and Mark Johnson. 2021. ECOL-R: Encouraging Copying in Novel Object Captioning with Reinforcement Learning. *arXiv preprint arXiv:2101.09865*.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721, Lisbon, Portugal. Association for Computational Linguistics.