STICKING TO THE FACTS: CONFIDENT DECODING FOR FAITHFUL DATA-TO-TEXT GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural conditional text generation systems have achieved significant progress in recent years, showing the ability to produce highly fluent text. However, the inherent lack of controllability in these systems allows them to *hallucinate* factually incorrect phrases that are unfaithful to the source, making them often unsuitable for many real world systems that require high degrees of precision. In this work, we propose a novel confidence oriented decoder that assigns a confidence score to each target position. This score is learned in training using a variational Bayes objective, and can be leveraged at inference time using a calibration technique to promote more faithful generation. Experiments on a structured data-to-text dataset – WikiBio (Lebret et al., 2016) – show that our approach is more faithful to the source than existing state-of-the-art approaches, according to both automatic metrics and human evaluation.

1 INTRODUCTION

Conditional text generation is the task of generating some target text $\boldsymbol{y}$ conditioned on source content $\boldsymbol{x}$. It is the essence of many natural language processing problems, such as text summarization (Mani, 1999), where $\boldsymbol{x}$ is a long document and $\boldsymbol{y}$ is a more concise version, machine translation (Koehn, 2009), where $\boldsymbol{x}$ and $\boldsymbol{y}$ represent equivalent text in different languages, and data-to-text generation (Kukich, 1983; McKeown, 1992), where $\boldsymbol{x}$ is a structured table and $\boldsymbol{y}$ is a textual description.

While traditionally done with template-based approaches (Becker, 2002; Foster & White, 2004; Gatt & Reiter, 2009; Reiter et al., 2005), recently neural encoder-decoder approaches (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014) have become a popular approach. In this formulation, the source content is encoded with a neural architecture, and the decoder autoregressively produces a token at each output position based on its internal state and the source representation. By leveraging continuous representations with rich non-linearities, encoder-decoder approaches can generate highly fluent text (Rush et al., 2015; Radford et al., 2019) without the need for cumbersome handcrafted rules and templates.

However, encoder-decoder architectures are inherently difficult to control, and have been shown to be prone to *hallucination*, i.e., generating text that is fluent but *unfaithful* to the source (Vinyals & Le, 2015; Koehn & Knowles, 2017; Wiseman et al., 2017; Lee et al., 2018). This severe shortcoming can often limit the use of neural approaches in many real world systems, where it is not acceptable to produce output that is even occasionally unfaithful.

In this work, we focus on data-to-text generation since the structured form of source content $\boldsymbol{x}$ makes it relatively easy to evaluate faithfulness using both human evaluation and domain-specific automatic metrics (Dhingra et al., 2019). In particular, we focus on the WikiBio (Lebret et al., 2016) dataset, where the task is to generate a sentence summarizing a tabular biography of a person. Figure 1 shows an example.

First note that the reference contains information such as *bonanno crime family* and *informant* that are true, but cannot be inferred from the source. This source-reference *divergence* exists in many large-scale generation datasets (Wiseman et al., 2017; Dhingra et al., 2019). Secondly, most generation systems are agnostic to this divergence and trained to maximize the log-likelihood of reference. This can often encourage the models to output phrases that are unsupported by the source. For example, Figure 1 shows the output of a state-of-the-art generation baseline, the Pointer-Generator

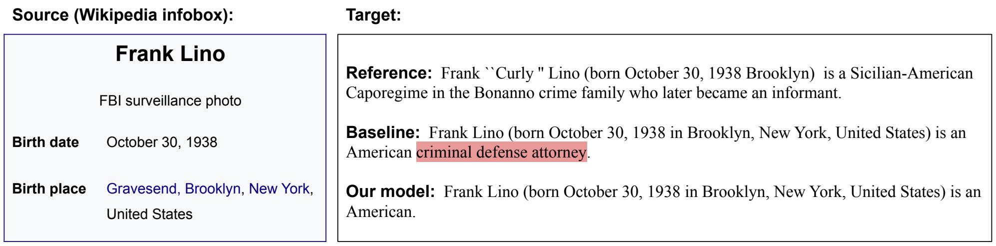

Figure 1: Example in the WikiBio dataset (Lebret et al., 2016) showing the biography of *Frank Lino*. The baseline Pointer-Generator (See et al., 2017) exhibits hallucination.

network (See et al., 2017), which contains the phrase *criminal defense attorney* that is false (but loosely related to *FBI* in the table). Thus, hallucination can often result from the coupling of model shortcomings (e.g. lack of formal reasoning, learning false correlations), and noise/artifacts in the training data.

In this work, we propose a confidence oriented approach which assigns a learned confidence score to each decoder position, and then uses the score in two ways to reduce hallucination: **(1)** In test, it uses confidence to adjust the output probabilities by a calibration technique (Braverman et al., 2019). **(2)** In training, we employ a variational Bayes objective to jointly learn the confidence score while allowing the model to skip tokens with a low confidence score to avoid training on reference phrases that are difficult to infer from the source. In Figure 1, our approach leads to a faithful generation that omits the occupation.

Empirically, when evaluated on the WikiBio dataset (Lebret et al., 2016), we show that our approach is considerably more faithful to the source than existing state-of-the-art solutions, according to both PARENT precision (Dhingra et al., 2019) and human evaluation.

2 Related Work

Improving the fidelity and accuracy of text generation systems is an important research topic that has spawned a variety of different approaches. Some focus on blending extractive and abstractive approaches, e.g., allowing the model to copy tokens directly from the source (Gu et al., 2016; See et al., 2017), separating content selection from generation (Zhou et al., 2017; Gehrmann et al., 2018) and utilizing topic information from the source to make informed generation (Narayan et al., 2018).

Other approaches have proposed generating more accurate text using semiparametric approaches (Guu et al., 2018; Pandey et al., 2018; Peng et al., 2019), reinforcement learning-based rewards (Paulus et al., 2018; Pasunuru & Bansal, 2018), semi-Markov models to learn neural templates (Wiseman et al., 2018), content planning (Puduppully et al., 2019a), and constrained vocabulary decoding (Wu et al., 2017). While many leverage the structure of the source (Liu et al., 2018) or task-based insights (Puduppully et al., 2019b), our approach is complementary in that it uses general machine learning techniques to build a confidence oriented decoder, that is more faithful to the source and robust to divergence/noise in the training data. Furthermore, many previous works rely on automatic metrics such as BLEU, which can be poorly correlated with human judgment of faithfulness (Wiseman et al., 2017; Dhingra et al., 2019). In contrast, we evaluate on PARENT precision (Dhingra et al., 2019), a metric specifically designed to capture faithfulness in data-to-text generation, and conduct a rigorous human evaluation to assess hallucination in our models.

3 Preliminaries

Before describing our approach, we first review the existing encoder-decoder framework (Sutskever et al., 2014; Bahdanau et al., 2014) with one stop-gradient (SG) tweak. Let $\boldsymbol{x} = x_1 x_2 \dots x_S$, be the source input of length S and $\boldsymbol{y} = y_1 y_2 \dots y_T$ be the target sequence of length T. Each token x_i, y_i takes one value from a vocabulary V. Our goal is to model the conditional distribution $P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{t=1}^{T} P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$, where $\boldsymbol{y}_{<t} = y_1 \dots y_{t-1}$ is the prefix of $\boldsymbol{y}$ up to the $(t-1)^{\text{th}}$ token.

The source can be encoded by any neural network function **enc**, such as a convolutional neural network (CNN, LeCun et al., 1990), long-short-term memory (LSTM, Hochreiter & Schmidhuber, 1997), or Transformer (Vaswani et al., 2017). Let $\boldsymbol{s}_1, ..., \boldsymbol{s}_S = \textbf{enc}(x_1, ..., x_S)$.

Define $\boldsymbol{e}_x \in \mathbb{R}^d$ as the d dimensional embedding of token x. Then, the probability of each target token is computed as:

$$P\left(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}\right) = \frac{\exp\left(\boldsymbol{v}_t^\top \boldsymbol{e}_{y_t}\right)}{\sum_y (\exp \boldsymbol{v}_t^\top \boldsymbol{e}_y)} \tag{1}$$

where

$$\boldsymbol{v}_t = \boldsymbol{a}_t + \boldsymbol{h}_t = \sum_{s=1}^{S} \alpha_{s,t} \boldsymbol{s}_s + \boldsymbol{h}_t \tag{2}$$

The first term on the right hand side in Eq. 2 represents a Luong-style attention (defined in Eq. 3, Luong et al., 2015) while the second term represents the hidden state at position t that is modelled with an RNN[1] (defined in Eq. 4):

$$\alpha_{s,t} = \frac{\exp(\boldsymbol{s}_s^\top \boldsymbol{W} \boldsymbol{h}_t)}{\sum_{s'} \exp(\boldsymbol{s}_{s'}^\top \boldsymbol{W} \boldsymbol{h}_t)} \tag{3}$$

$$\boldsymbol{h}_t = \text{RNN}(\boldsymbol{h}_{t-1}, [\boldsymbol{e}_{y_{t-1}}, \text{SG}(\boldsymbol{a}_{t-1})]) \tag{4}$$

We have made one change to the conventional input-feeding approach as defined by Bahdanau et al. (2014), and apply a stop-gradient (SG) to the attention vector $\boldsymbol{a}_{t-1}$ above. In this work, we use SG as a control on the information flow during training, making our model match the intended design. The above SG prevents information at the current step from being propagated to previous attentions, which is intended by our attention score defined in Section 4.1.

4 MODEL

Our approach is based on the idea of a learned **confidence score** $C(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ at every decoding position that is a balance of two factors:

- How much the model *should* rely on the source for this position.
- How much the model *actually* relies on the source for this position.

Intuitively, it is reasonable for a system to depend mostly on language modeling to predict function words that make the generation fluent, but it should consult more of the source data to predict content words. For example, given a partial generation "*Christian Campbell is __*", one could predict that the next token is mostly likely "*a*", "*an*" or "*the*", based on language modeling. However, if a model predicts "*American*" as the next token to "*Christian Campbell is an __*", it should be based on a field such as "`Nationality: U.S.`" in the source, rather than the language tendency that "*American*" is likely to appear after the phrase "*is an*". A typical neural network can make predictions based on both reasons with little controllability; this, we contend, is a cause of hallucination.

In order to measure how much the model *should* rely on the source, we compare the encoder-decoder model with an unconditioned language model. The unconditioned language model does not have any access to the source data, so if it can predict a token as precisely as the encoder-decoder, that token is probably an element of a general linguistic construction that does not convey source information.

In order to measure how much the model *actually* relies on the source, we derive an **attention score** A_t of the encoder-decoder from the attention mechanism. If a token is likely a content word (i.e. when its generation probability by the encoder-decoder is much higher than the unconditioned language model), but the attention score is low, then the token might not be predicted based on the source, and could be hallucination. Thus, we design the confidence score to be low in this case.

The confidence score is specified in Section 4.1. There are two ways we leverage the score:

[1] While it is possible our approach could extend to other types of decoders, our current formulation of the confidence score uses this specific form of attention.

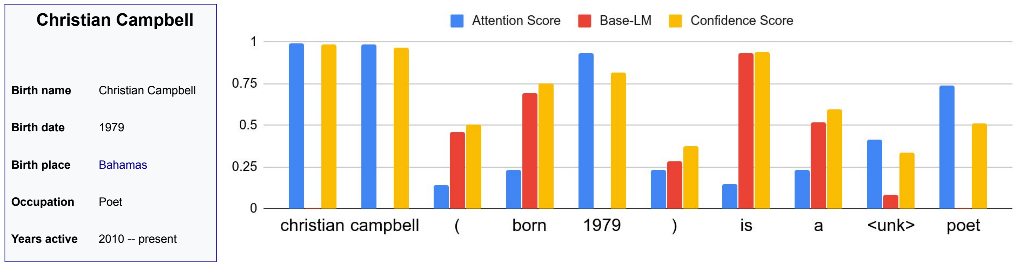

Figure 2: Example of learned attention score, base-LM probability, and confidence score. For content words the base-LM probability is lower, and the confidence score depends more on the attention score.

- At test time, we augment the generation probability with the confidence score, using a calibration technique (Section 4.2). It allows us to weigh more on the confidence of generation, without sacrificing perplexity (Braverman et al., 2019).
- In training, we allow the model to skip some tokens with low confidence score in order to avoid training on noisy references or scarce examples (Section 4.3). However, since the confidence score itself needs to be learned during training, we use a variational Bayes objective to formalize this symbiotic cycle (Kingma & Welling, 2014).

4.1 Confidence Score

We define the **confidence score** as an interpolation between the encoder-decoder model $P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ and the unconditioned language model $P(y_t|\boldsymbol{y}_{<t})$:

$$C\big(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}\big) := A_t P\big(y_t \mid \boldsymbol{y}_{<t}, \boldsymbol{x}\big) + (1 - A_t) P\big(y_t \mid \boldsymbol{y}_{<t}\big), \tag{5}$$

where A_t is the **attention score** measuring how much the encoder-decoder is paying attention to the source:

$$A_t := \frac{\|\boldsymbol{a}_t\|}{\|\boldsymbol{a}_t\| + \|\boldsymbol{h}_t\|}. \tag{6}$$

Here, $\|\cdot\|$ is the Euclidean norm.

For function words, we expect both $P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ and $P(y_t|\boldsymbol{y}_{<t})$ to be high, so the confidence score defined in Eq. 5 will be high no matter what the attention score is. On the other hand, we expect $P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ to be higher than $P(y_t|\boldsymbol{y}_{<t})$ for content words, so the confidence score will largely depend on the attention score in this case.

We refer to the unconditioned language model in the definition of confidence score as the **base-LM**. In this work, we use an RNN for base-LM, but modify the input-feeding of the RNN as following:

$$\boldsymbol{g}_t = \text{RNN}\Big(\boldsymbol{g}_{t-1}, \text{SG}\big((1 - A_{t-1}) P(y_{t-1}|\boldsymbol{y}_{<t-1})\big) \boldsymbol{e}_{y_{t-1}}\Big). \tag{7}$$

Here, $\boldsymbol{g}_t$ is the hidden-state of the base-LM, SG means stop-gradient, and the input embedding is weighted with a component of the previous confidence score. We found this weighting scheme to decrease dependence of the base-LM on content words, seemingly resulting in a model of soft templates (Figure 2).

Copy mechanism In case the encoder-decoder is equipped with a copy mechanism, the generation probability is mixed with a probability of copying from the source (Gu et al., 2016; See et al., 2017):

$$\tilde{P}(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}) = p_t^{\text{gen}} P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}) + (1 - p_t^{\text{gen}}) \sum_{s: x_s = y_t} \beta_{s,t}, \tag{8}$$

where p_t^{gen} is the probability of doing generation instead of copying at step t, and $\beta_{s,t}$ is an attention weight that the copy mechanism is paying to position s in the source. The sum is taken over all positions s where the word x_s is the same as y_t. When the copy mechanism is incorporated, we re-define the attention score as

$$\tilde{A}_t := p_t^{\text{gen}} A_t + (1 - p_t^{\text{gen}}), \tag{9}$$

and the confidence score is re-defined accordingly.

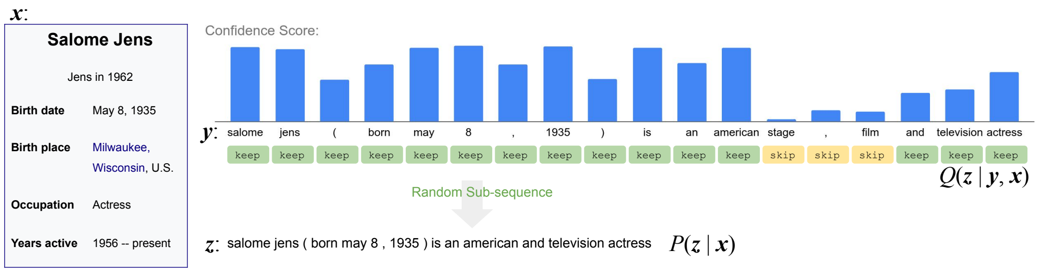

Figure 3: Example of sampling a sub-sequence according to the confidence score. Our variational Bayes objective combines the sampling probability $Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})$ and the generation probability $P(\boldsymbol{z}|\boldsymbol{x})$.

4.2 Calibration

In order to use the confidence score to promote faithful generation, we apply a calibration technique (Braverman et al., 2019) which augments the generation probability as follows:

$$\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t},\boldsymbol{x}) := \frac{\mathrm{SG}(C(y_t|\boldsymbol{y}_{<t},\boldsymbol{x}))^{\kappa}\mathrm{SG}(P(y_t|\boldsymbol{y}_{<t},\boldsymbol{x}))}{\sum_{w\in V}\mathrm{SG}(C(w|\boldsymbol{y}_{<t},\boldsymbol{x}))^{\kappa}\mathrm{SG}(P(w|\boldsymbol{y}_{<t},\boldsymbol{x}))}. \tag{10}$$

Here, $\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ is a one-parameter family of probability distributions called the calibration of $P(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ to $C(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$. Note that $\mathrm{SG}(\cdot)$ stops gradients so that only the parameter κ is updated during training. In order to learn κ, we minimize the negative log-likelihood of $\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ jointly with $P(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ and $P(y_t|\boldsymbol{y}_{<t})$:

$$\mathcal{L}_{\text{joint}}(\boldsymbol{y},\boldsymbol{x}) := \sum_{t=1}^{T} -\log\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t},\boldsymbol{x}) - \log P(y_t|\boldsymbol{y}_{<t},\boldsymbol{x}) - \log P(y_t|\boldsymbol{y}_{<t}). \tag{11}$$

Since $P(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ is a special case of $\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ (namely at $\kappa=0$), the training perplexity of $\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$ is at most $P(y_t|\boldsymbol{y}_{<t},\boldsymbol{x})$. In practice, κ is initialized as 0 and found converging to a positive value (Section. 5.4). Therefore, the calibration trick can improve confidence of generation without sacrificing perplexity.

4.3 Training with a Variational Bayes Objective

In practice, the training data of a conditional text generation task almost always contain noises and/or scarce examples. In order to reduce the impact of such outliers and train a confident generation model, we allow the model to skip some tokens in the training data when it feels unconfident. For this purpose, we use the confidence score to sample a sub-sequence of each training target, and minimize the negative log-likelihood on that confident sub-sequence. However, since the confidence score itself needs to be trained, we use a variational Bayes objective to formalize the problem.

Specifically, for each target $\boldsymbol{y}=y_1y_2\ldots y_T$, we define $\boldsymbol{z}=z_1z_2\ldots z_R=y_{\iota(1)}y_{\iota(2)}\cdots y_{\iota(R)}$ as a latent sub-sequence of $\boldsymbol{y}$, which consists of confident tokens of length R. Here, $\iota:|R|\to|T|$ is an inclusion of indices. We assume that $\boldsymbol{z}$ is generated by the calibrated probability $\hat{P}^{\kappa}\big(z_r \mid \boldsymbol{z}_{<r},\boldsymbol{x}\big)$:

$$P\big(\boldsymbol{z}\mid\boldsymbol{x}\big) = \prod_{r=1}^{R}\hat{P}^{\kappa}\big(z_r\mid\boldsymbol{z}_{<r},\boldsymbol{x}\big). \tag{12}$$

Then, we connect $P(\boldsymbol{z}|\boldsymbol{x})$ to the probability of training target $P(\boldsymbol{y}|\boldsymbol{x})$. From Bayes rule:

$$P\big(\boldsymbol{y}\mid\boldsymbol{x}\big) = \frac{P(\boldsymbol{y}|\boldsymbol{z},\boldsymbol{x})\,P(\boldsymbol{z}|\boldsymbol{x})}{P(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}. \tag{13}$$

We assume that $P(\boldsymbol{y}|\boldsymbol{z},\boldsymbol{x})=1$ for every training example because the training data is uniquely given. Then, we regard $\boldsymbol{z}$ as a sequential "`keep`/`skip`" labeling over $\boldsymbol{y}$, and define a probability distribution

$$Q_t := \begin{cases} Q_t(\texttt{keep}) \propto C(y_t|\boldsymbol{z}_{\iota(s)<t},\boldsymbol{x})^{\rho} & \text{Where } \texttt{keep} \text{ means } y_t \text{ is in } \boldsymbol{z} \\ Q_t(\texttt{skip}) \propto \gamma^{\rho} & \text{Where } \texttt{skip} \text{ means } y_t \text{ is not in } \boldsymbol{z} \end{cases} \tag{14}$$

to sample a sub-sequence according to the confidence score (Figure 3). Here, ρ and γ are hyper-parameters. Now let

$$Q(\boldsymbol{z} \mid \boldsymbol{y}, \boldsymbol{x}) = \prod_{t=1}^{T} Q_t, \tag{15}$$

and the idea is to use $Q(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})$ as an approximation to the unknown posterior $P(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})$ in Eq. 13. By taking $-\log(\cdot)$ of both sides in Eq. 13 and trivially introducing $\log Q(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})$, we get

$$-\log P(\boldsymbol{y} \mid \boldsymbol{x}) = -\log \frac{Q(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})}{P(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})} + \log Q(\boldsymbol{z} \mid \boldsymbol{y}, \boldsymbol{x}) - \log P(\boldsymbol{z} \mid \boldsymbol{x}). \tag{16}$$

Then, we take the expectation $\mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}[\cdot]$ of both sides and note that $\mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}[\log \frac{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}{P(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}] = KL[Q(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})||P(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})] \geq 0$, so

$$-\log P(\boldsymbol{y} \mid \boldsymbol{x}) \leq \mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}\Big[\log Q(\boldsymbol{z} \mid \boldsymbol{y}, \boldsymbol{x}) - \log P(\boldsymbol{z} \mid \boldsymbol{x})\Big]. \tag{17}$$

The variational Bayes objective is to minimize the upper bound on the right hand side of Eq. 17. In practice, it is computationally expensive to explicitly calculate $\mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}[\cdot]$ by enumerating all sub-sequences of $\boldsymbol{y}$, because the number of sub-sequences is exponential to the length T. Thus, we apply a Monte Carlo method which calculates $\mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}[\cdot]$ by sampling from $Q(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})$. In order to back-propagate gradients through the expectation $\mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}[\cdot]$ as well, the loss function is given as follows (Paisley et al., 2012):

$$\begin{aligned}\mathcal{L}_{\text{var}}(\boldsymbol{y}, \boldsymbol{x}) := \frac{1}{K} \sum_{\substack{k=1 \\ \boldsymbol{z}_k \sim Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}}^{K} & \log Q(\boldsymbol{z}_k \mid \boldsymbol{y}, \boldsymbol{x}) - \log P(\boldsymbol{z}_k \mid \boldsymbol{x}) \\ & + \lambda\, \mathrm{SG}\Big(\log Q(\boldsymbol{z}_k \mid \boldsymbol{y}, \boldsymbol{x}) - \log P(\boldsymbol{z}_k \mid \boldsymbol{x})\Big) \log Q(\boldsymbol{z}_k \mid \boldsymbol{y}, \boldsymbol{x}).\end{aligned} \tag{18}$$

Here, K is the number of samples taken, and λ is a hyper-parameter controlling how fast the gradients go through $\mathbb{E}_{Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}[\cdot]$. However, since we define $P(\boldsymbol{z}|\boldsymbol{x})$ in Eq. 12 by the calibrated probability, which only learns one parameter κ, we add joint learning terms into Eq. 18 to make the final objective:

$$\begin{aligned}\mathcal{L}(\boldsymbol{y}, \boldsymbol{x}) := \frac{1}{K} \sum_{\substack{k=1 \\ \boldsymbol{z}_k \sim Q(\boldsymbol{z}|\boldsymbol{y},\boldsymbol{x})}}^{K} & \log Q(\boldsymbol{z}_k \mid \boldsymbol{y}, \boldsymbol{x}) + \mathcal{L}_{\text{joint}}(\boldsymbol{z}_k, \boldsymbol{x}) \\ & + \lambda\, \mathrm{SG}\Big(\log Q(\boldsymbol{z}_k \mid \boldsymbol{y}, \boldsymbol{x}) - \log P(\boldsymbol{z}_k \mid \boldsymbol{x})\Big) \log Q(\boldsymbol{z}_k \mid \boldsymbol{y}, \boldsymbol{x}).\end{aligned} \tag{19}$$

In order to sample sub-sequences from $Q(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{x})$, we apply the same Gumbel-max trick as in Kool et al. (2019). Although the random sampling is purely based on a learned probability distribution, without any constraints to make it fluent, surprisingly the model still learns to generate fluent text.

5 Experiments

Although our approach could apply to many conditional text generation tasks, in this work we consider data-to-text generation, in which the source is some structured data and the target is natural language text describing the data. Usually, the data have concise semantics and simple structure, which makes it easy to check the facticity of the generation.

5.1 Dataset and Evaluation Metrics

The WikiBio dataset (Lebret et al., 2016) contains 728,321 biographies paired with infoboxes, taken from the Sep-2015 dump of English Wikipedia, and splitted into train/valid/test sets in a $8:1:1$ ratio. The biography text is the first sentence of the Wikipedia page (26.1 words on average). Infoboxes have 12.1 non-empty fields on average.

For automatic metrics, we report BLEU (Papineni et al., 2002), as well as PARENT (Dhingra et al., 2019), a metric designed to mitigate the shortcomings of BLEU on structured data-to-text generation.

Model	Warmup Steps	Learning Rate	RNN Dropout	RNN Dim.	Beam Size	ρ	γ	K	λ
Confident BERT-to-RNN	40000	0.05	0.1	768	8	0.75	1/8	4	1/64
Confident Pointer-Generator	None	0.0005	0.2	200	8	0.5	1/16	4	1/4

Table 1: Hyper-parameters. We did a hyper-parameter search for ρ, γ and λ, within the range $[0.25, 0.5, 0.75, 1]$, $[1/16, 1/8, 1/4]$ and $[1/64, 1/16, 1/4, 1]$, respectively. Model selection is based on PARENT F1.

For human evaluation, we obtain crowd-source annotations on examples randomly chosen from predictions on the WikiBio test set, the same 1000 for each model. We instruct raters to grade on each of 3 criteria: faithfulness, coverage, and fluency, as below:

- Faithfulness (precision) - We define a sentence to be faithful if all the information in the proposed sentence is supported by the table or the reference. A single hallucinated piece of information makes the sentence non-faithful.
- Coverage (recall) - The number of table cells that contain information present in the sentence.
- Fluency - A sentence is defined to be fluent if it is clear, natural, and grammatically correct. Raters choose among three options: *Fluent*, *Mostly Fluent*, *Not Fluent*.

An ideal system would always produce fluent and faithful text with high coverage.

5.2 Experiment Setting

We compare the following systems:

- BERT-to-BERT (Rothe et al., 2019): A Transformer encoder-decoder model (Vaswani et al., 2017) where the encoder and decoder are both initialized with BERT (Devlin et al., 2018).
- Structure Aware Seq2Seq (Liu et al., 2018): A state-of-the-art method on the WikiBio dataset in terms of BLEU.
- Pointer-Generator (See et al., 2017): A Seq2Seq with attention and copy mechanism (our implementation).
- Confident BERT-to-RNN (This Work): A Transformer encoder initialized with BERT checkpoint, and a GRU (Cho et al., 2014) decoder with our confident decoding method.
- Confident Pointer-Generator (This Work): Pointer-Generator model with confident decoding.

We built our systems using Tensorflow (Abadi et al., 2016). Infoboxes are linearized into sequences, with field names and values separated by special tokens. For BERT-to-BERT and Confident BERT-to-RNN, we pre-trained a BERT checkpoint on the Books corpus (Zhu et al., 2015) only, since the original BERT was trained on Wikipedia that overlaps with the test targets in WikiBio. For Pointer-Generator and Confident Pointer-Generator, we use GloVe (Pennington et al., 2014) as the input word embedding, and the two models share the same hyper-parameter settings. For our confident decoding models, there are additional hyper-parameters ρ, γ as defined in Eq. 14, and K, λ as defined in Eq. 18. The hyper-parameters are given in Table 1. The optimizer is Adam (Kingma & Ba, 2015).

5.3 Results

Table 2 shows the results. According to human evaluation, our approach gives a clear improvement in faithfulness over the baselines, with some drop in coverage. To further measure the validity of our confidence score, we postprocessed the output to remove words with lower confidence than 0.125. This thresholding technique gives further gains to faithfulness, while sacrificing some fluency.

Among the automatic metrics, PARENT precision and recall seem correlated to faithfulness and coverage respectively, and our approach achieves the highest precision score. BLEU, perhaps because of its length penalty that rewards longer generations, seems more correlated to coverage rather than faithfulness. Regarding the baselines, we see that BERT-to-BERT is the most fluent while Pointer Generator is the most faithful, suggesting that pretraining might help fluency while the copy mechanism can be valuable for faithfulness.

	Automatic Metrics			Human evaluation		
Model	**BLEU**	**PARENT (Prec. / Rec. / F1)**	**Avg Len.**	**Faithful %**	**Avg Cov.**	**Fluency %**
BERT-to-BERT (Rothe et al., 2019)	44.83	77.62 / 43.00 / 53.13	20.9	77.6	4.33	**98.5** / 99.4
Structure-Aware Seq2Seq (Liu et al., 2018)	**45.36**	73.98 / **44.02** / 52.81	23.1	66.1	**4.47**	88.6 / 99.7
Pointer-Generator (See et al., 2017)	41.07	77.59 / 42.12 / 52.10	19.1	80.3	4.24	93.1 / 96.0
Confident BERT-to-RNN (This Work)	33.30	77.98 / 37.21 / 47.90	16.6	**85.2***	3.90	92.3 / 94.1
Confident Pointer-Generator (This Work)	38.10	**79.52** / 40.60 / 51.38	17.0	**86.8***	4.05	95.4 / 96.3
+threshold=0.125	36.62	**80.15** / 39.59 / 50.50	16.4	**90.7***	4.01	91.6 / 92.2

Table 2: Performance on WikiBio test set. Two Fluency measures differ in whether to include sentences graded as *Mostly Fluent*. Starred numbers are statistically significant against baselines ($p < .001$), by bootstrap test.

	BLEU	**PARENT (Prec. / Rec. / F1)**	**Avg Len.**
Confident Pointer-Generator (This Work)	38.10	**79.52** / 40.60 / 51.38	17.0
No base-LM	39.39	78.77 / 41.55 / 52.08	17.9
No calibration	37.89	79.47 / 40.47 / 51.26	16.9
No variational	**41.29**	78.25 / **42.40** / **52.52**	18.9
Pointer-Generator	41.07	77.59 / 42.12 / 52.10	19.1
Truncated	35.50	77.68 / 38.16 / 48.66	17.1

Table 3: Ablative tests on three components of our confident decoding method, and a truncation test.

5.4 Ablations

Our confident decoding method has three novel components: **(1)** The use of a base-LM to define confidence score; **(2)** The calibration technique to adjust output probability; and **(3)** The variational Bayes objective to train a confident model. In this section, we assess the effects of each component by an ablative study. We start from the Confident Pointer-Generator, and in each test replace one component by a trivial alternative: **(1)** In order to assess the effects of using the base-LM in the confidence score, we instead use $P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ directly as confidence, and train models with the same hyper-parameter search. The results on WikiBio are shown in Table 3 as "No base-LM". **(2)** We use $P(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ instead of $\hat{P}^{\kappa}(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x})$ at test time ('No calibration"), to assess the effects of calibration. The model is the same as Confident Pointer-Generator. The learned κ was 0.035. **(3)** Instead of the variational Bayes objective, we use the joint training loss $\mathcal{L}_{\text{joint}}$ in Eq. 11 without sampling sub-sequences from training targets (No variational).

As we can see from Table 3, all three components improve PARENT precision. While the improvement by calibration is the smallest, the technique also improves PARENT recall and BLEU score at the same time, making it an easy choice. The other techniques trade recall for precision, making them useful for tasks that require a high degree of faithfulness. When all three components are disabled, the model is exactly the same as our implementation of the Pointer-Generator. Every component improves PARENT precision upon it as well. Especially, comparing Pointer-Generator with "No variational" shows again that joint training with calibration improves all metrics.

We also note that the average lengths of generations by our confident decoding models are shorter. While there exists heuristics such as length penalty (Wu et al., 2016) to encourage longer generation at inference time, shorter generation is not trivial. In the "Truncated" setting, we truncate predictions by the Pointer-Generator two words each to match the average length of Confident Pointer-Generator. The PARENT precision by our confident decoding method is not trivially achieved by truncation.

6 Conclusion

In this work, we proposed a confidence oriented decoder that achieved more faithful generation on the WikiBio dataset than existing state-of-the-art approaches. Our method is general in principle, so it could potentially be adapted to other forms of conditional text generation such as document summarization and machine translation. Another source of future work could be enhancing the confidence score to move beyond shallow alignment and do more complex logical inference (Steedman & Baldridge, 2011; Kamath & Das, 2018).

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Tilman Becker. Practical, template–based natural language generation with tag. In *Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+ 6)*, pp. 80–83, 2002.

Mark Braverman, Xinyi Chen, Sham M Kakade, Karthik Narasimhan, Cyril Zhang, and Yi Zhang. Calibration, entropy rates, and memory in language models. *arXiv preprint arXiv:1906.05664*, 2019.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William W Cohen. Handling divergent reference texts when evaluating table-to-text generation. *arXiv preprint arXiv:1906.01081*, 2019.

Mary Ellen Foster and Michael White. Techniques for text planning with xslt. In *Proceeedings of the Workshop on NLP and XML (NLPXML-2004): RDF/RDFS and OWL in Language Technology*, pp. 1–8, 2004.

Albert Gatt and Ehud Reiter. Simplenlg: A realisation engine for practical applications. In *Proceedings of the 12th European Workshop on Natural Language Generation (ENLG 2009)*, pp. 90–93, 2009.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. Bottom-up abstractive summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 4098–4109. Association for Computational Linguistics, 2018. URL `https://www.aclweb.org/anthology/D18-1443/`.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016.

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. Generating sentences by editing prototypes. *TACL*, 6:437–450, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Aishwarya Kamath and Rajarshi Das. A survey on semantic parsing. *arXiv preprint arXiv:1812.00978*, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL `http://arxiv.org/abs/1312.6114`.

Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*, 2017.

Wouter Kool, Herke van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3499–3508. PMLR, 2019. URL `http://proceedings.mlr.press/v97/kool19a.html`.

Karen Kukich. Design of a knowledge-based report generator. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pp. 145–150. Association for Computational Linguistics, 1983.

Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1203–1213, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1128. URL `https://www.aclweb.org/anthology/D16-1128`.

Yann LeCun, Bernhard E Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne E Hubbard, and Lawrence D Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, pp. 396–404, 1990.

Katherine Lee, Orhan Firat, Ashish Agarwal, Clara Fannjiang, and David Sussillo. Hallucinations in neural machine translation. In *Open Review*, 2018.

Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. Table-to-text generation by structure-aware seq2seq learning. In *Proc. of AAAI*, 2018.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pp. 1412–1421. The Association for Computational Linguistics, 2015. URL `https://www.aclweb.org/anthology/D15-1166/`.

Inderjeet Mani. *Advances in automatic text summarization*. MIT press, 1999.

Kathleen McKeown. *Text generation*. Cambridge University Press, 1992.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL `https://www.aclweb.org/anthology/D18-1206`.

John W. Paisley, David M. Blei, and Michael I. Jordan. Variational bayesian inference with stochastic search. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012. URL `http://icml.cc/2012/papers/687.pdf`.

Gaurav Pandey, Danish Contractor, Vineet Kumar, and Sachindra Joshi. Exemplar encoder-decoder for neural conversation generation. In *Proc. of ACL*, 2018.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Ramakanth Pasunuru and Mohit Bansal. Multi-reward reinforced summarization with saliency and entailment. *arXiv preprint arXiv:1804.06451*, 2018.

Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *Proc. of ICLR*, 2018.

Hao Peng, Ankur P Parikh, Manaal Faruqui, Bhuwan Dhingra, and Dipanjan Das. Text generation with exemplar-based adaptive decoding. *arXiv preprint arXiv:1904.04428*, 2019.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

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

Ratish Puduppully, Li Dong, and Mirella Lapata. Data-to-text generation with entity modeling. *arXiv preprint arXiv:1906.03221*, 2019b.

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

Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1-2):137–169, 2005.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *arXiv preprint arXiv:1907.12461*, 2019.

Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015.

Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In Regina Barzilay and Min-Yen Kan (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pp. 1073–1083. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1099. URL `https://doi.org/10.18653/v1/P17-1099`.

Mark Steedman and Jason Baldridge. Combinatory categorial grammar. *Non-Transformational Syntax: Formal and explicit models of grammar*, pp. 181–224, 2011.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pp. 3104–3112, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. Learning neural templates for text generation. In *Proc. of EMNLP*, 2018.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Yu Wu, Wei Wu, Dejian Yang, Can Xu, Zhoujun Li, and Ming Zhou. Neural response generation with dynamic vocabularies. *CoRR*, abs/1711.11191, 2017. URL `http://arxiv.org/abs/1711.11191`.

Qingyu Zhou, Nan Yang, Furu Wei, and Ming Zhou. Selective encoding for abstractive sentence summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1095–1104, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1101. URL `https://www.aclweb.org/anthology/P17-1101`.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pp. 19–27, 2015.