# Controlling Pretrained Language Generation Models by Learning to Focus

Anonymous ACL submission

## Abstract

Transformer-based language models, which are pretrained on large-scale unsupervised data and then finetuned on task-specific datasets, have become the dominant paradigm for various natural language generation tasks. The finetuning and usages of such models are typically conducted in an end-to-end manner. This work attempts to develop a control mechanism by which a user can select spans of context as "highlights" for the model to focus on, while generating output text. To achieve this goal, we augment a pretrained model with trainable "attention vectors" that are directly applied to the model's embeddings, while the model itself is kept fixed. These vectors, trained on automatic annotations derived from attribution methods, act as indicators for context importance. We test our approach on two core generation tasks: dialogue response generation and abstractive summarization. We also collect evaluation data where the highlight-generation pairs are annotated by humans. Our experiments show that the trained attention vectors are effective in steering the model to generate outputs that are relevant to user-selected highlights.

## 1 Introduction

Transformer-based models pretrained on large-scale text data have become the dominant paradigm for natural language generation (NLG) tasks (Roller et al., 2020; Lewis et al., 2019; Raffel et al., 2020). The attention mechanism (Bahdanau et al., 2016; Vaswani et al., 2017), which aggregates information via a weighted average over word-level embeddings, plays a vital role in these models. The attention mechanism serves two major purposes: (1) It captures linguistic phenomena in the input (Clark et al., 2019; Kovaleva et al., 2019; Kobayashi et al., 2020); (2) It helps the model focus on relevant portions of the input (e.g., alignment in machine translation (Bahdanau et al., 2016) and abstractive summarization (Rush et al., 2015)).

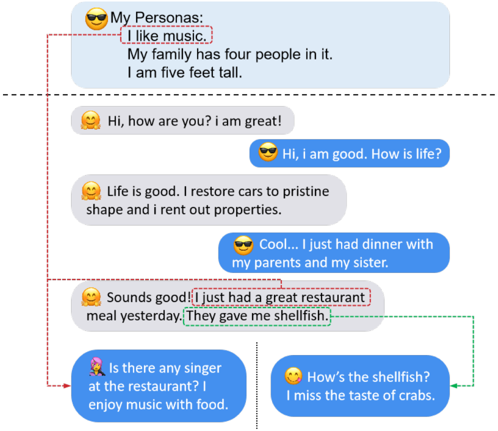

My Personas:
I like music.
My family has four people in it.
I am five feet tall.
Hi, how are you? i am great!
Hi, i am good. How is life?
Life is good. I restore cars to pristine shape and i rent out properties.
Cool... I just had dinner with my parents and my sister.
Sounds good! I just had a great restaurant meal yesterday. They gave me shellfish.
Is there any singer at the restaurant? I enjoy music with food.
How's the shellfish? I miss the taste of crabs.

Figure 1: Illustration of our motivation: different highlights in the input (including persona) lead to different generations. This example is from our collected dialogue data for evaluation (Section 3).

The attention mechanism is particularly useful as it does not require any explicit supervision: the model learns to focus on relevant parts of the input through end-to-end training. However, this property makes it difficult to explicitly control the model's focus across multiple contexts, as the attention distribution is on word-level. Moreover, transformer models have multiple layers of multi-head attention modules (Vaswani et al., 2017), and it is not clear which head's attention distribution should be intervened upon. This is especially sub-optimal in some NLG applications involving a relatively long input such as dialogue or summarization: focusing on different spans of the input could result in completely different generations (illustrated in Figure 1). It would be attractive to give the user an option to steer the model's focus.

This goal brings about significant challenges. For one, many popular NLG datasets are collected in an end-to-end manner, i.e., without annotations of which spans of input are most relevant to the reference target. It would also be ideal for the proposed approach to be compatible with existing pretrained transformer models, as re-training such models is often costly.

In this work, we propose an *attention vector* framework to address the challenges outlined above. Our contributions are as follows:

- To control the model's attention, we augment the pretrained model with trainable attention vectors which are directly applied to the encoder embeddings corresponding to the highlighted input spans. The model itself is kept fixed, and no further changes to the model architecture is needed.
- To train the attention vectors, we utilize attribution methods to derive automatic highlight annotations from existing end-to-end training data, which obviates the need for manual human annotations.
- For principled evaluation and future work in this direction, we collect and release human evaluation data where the highlight-generation pairs are annotated by humans.
- We test our method on two core NLG tasks: dialogue response generation and abstractive summarization. Experiments show that the trained attention vectors are effective in steering the model to generate a relevant output given the selected highlights.

## 2 Model Formulation

We assume the target model is a standard pretrained transformer encoder-decoder model (Vaswani et al., 2017) that has already been finetuned on end-to-end task-specific data (e.g., dialogue or summarization) with the standard negative log-likelihood (NLL) loss. Our goal is to establish a control mechanism whereby the user can highlight several spans of the input, and the model is supposed to generate outputs relevant to the highlighted text. Crucially, this mechanism should not change the base model, in order to allow the user to default back to the original model if desired.

We begin by establishing notation. We denote the end-to-end training data by $\{\mathbf{x}, \mathbf{y}\}$, where $\mathbf{x} = \{x_1, ..., x_n\}$ refers to the input token sequence, and $\mathbf{y}$ refers to the corresponding reference target token sequence. During evaluation, some spans of the input $\mathbf{x}$ will be highlighted, and we use a binary indicator $c_i$ to indicate whether the $i^{th}$ input token is to be highlighted during generation. This work only considers complete sentences as a valid highlight span. This design choice is mainly for convenience during our human-annotated evaluation data collection. But our framework can readily be generalized to phrase-level or even discontiguous highlights.

Suppose the encoder model is composed of $L$ transformer layers. We denote the $d$-dimensional output embedding of the $i^{th}$ position on the $l^{th}$ encoder layer by $\mathbf{h}_i^l$. We use $\{\mathbf{h}_i^0\}$ to denote the input embeddings. Each decoder layer performs multi-head cross-attention on the outputs of the encoder, where the attention weight computation for the $h^{th}$ head on the $l^{th}$ decoder layer is formulated as below:

$$\alpha_{i,j}^{h,l} = \operatorname*{softmax}_{i\in\{1...n\}} \left( \frac{k(\mathbf{h}_i^L) \cdot \mathbf{q}_j^{h,l}}{\sqrt{d}} \right). \quad (1)$$

Here $k(\cdot)$ is a linear transform, and $\alpha_{i,j}$ is the attention weight assigned to encoder output $\mathbf{h}_i^L$, for the $j^{th}$ position decoder query vector $\mathbf{q}_j$. We use $P_{\text{M}}(\mathbf{y}|\mathbf{x})$ to denote the probability assigned to $\mathbf{y}$ given input $\mathbf{x}$ by the original target model. For more details of the transformer encoder-decoder architecture, we refer readers to Vaswani et al. (2017).

Our proposed framework involves two stages. We first obtain automatic highlight annotations using attribution methods. Then, these annotations are used to train the attention vectors. In the next section, we review the attribution methods.

### 2.1 Attribution Methods

Many popular NLG datasets are collected end-to-end, i.e., without annotations of which spans of input are relevant to the reference target. To obtain these annotations (which are needed to train our attention vectors), we make use of existing attribution methods.

Attribution methods (Baehrens et al., 2010; Simonyan et al., 2014; Shrikumar et al., 2017; Adebayo et al., 2018; Sundararajan et al., 2017), also known as *saliency maps*, attribute the prediction of a (potentially black-box) model to its input features. It thus fits our need to extract relevant spans in the input given the reference target. Most saliency methods are originally designed for image classification, where an importance score is assigned for each dimension of the input feature. Therefore, slight modifications (e.g., dot-product with the word embeddings) are needed to apply them to language data (Ding and Koehn, 2021; Denil et al., 2014).

We implement and compare several popular attribution methods, which compute the attribution score for a given sentence $S$ (denoting the set of token indexes in the sentence) in the input $\mathbf{x}$ for the target $\mathbf{y}$ and model $P_{\mathrm{M}}$.

**Leave-one-out (LOO)** We replace the tokens in $S$ by the `<pad>` token, and compute the difference in NLL:

$$A(S) = \log P_{\mathrm{M}}(\mathbf{y}|\mathbf{x}) - \log P_{\mathrm{M}}(\mathbf{y}|\mathbf{x}_{S\text{-padded}}). \quad (2)$$

LOO is also referred to as an *occlusion-based method* (Zeiler and Fergus, 2014; Li et al., 2016) in the literature.

**Attention-weight** We sum up the attention weights assigned to tokens in $S$ for all attention heads across all decoder layers:

$$A(S) = \sum_{i \in S} \sum_{j,h,l} \alpha_{i,j}^{h,l}. \quad (3)$$

**Grad-norm** We sum the norm of gradient for the input word embeddings in $S$:

$$A(S) = \sum_{i \in S} ||\nabla_{\mathbf{h}_i^0} \log P_{\mathrm{M}}(\mathbf{y}|\mathbf{x})||_2. \quad (4)$$

**Grad-input-product** Instead of taking vector norm, we compute the dot-product between the input embedding and its gradient:

$$A(S) = \sum_{i \in S} \left( \nabla_{\mathbf{h}_i^0} \log P_{\mathrm{M}}(\mathbf{y}|\mathbf{x}) \right) \cdot \mathbf{h}_i^0. \quad (5)$$

While more sophisticated attribution method have been proposed in the literature (Lei et al., 2016; Sundararajan et al., 2017; Bastings et al., 2019), we mainly experiment with the above methods due to their simplicity and popularity. Attribution methods have been used for interpreting black-box models—applying them to derive labels that can further be used to control a model has to our knowledge not been explored before.

Which attribution method best reflects the model's inner working is still an active research area (Ding and Koehn, 2021; Adebayo et al., 2018). The present work is primarily concerned with how well the attribution scores align with human-annotated highlights. In our experiments, we find that leave-one-out (LOO) has the best correlation on the human-annotated development set (Table 1). We therefore adopt LOO to derive the automatic highlight annotations.

More specifically, for the input-output pairs in the training set, we sort the LOO attribution scores of the sentences in the input from large to small, and mark the tokens in the first few sentences (the exact number varies by task) as highlights. We denote the highlight labels obtained from this automatic procedure by a binary indicator variables $\mathbf{c}^{\text{attr}} = \{c_1^{\text{attr}}, \ldots, c_n^{\text{attr}}\}$, which will be used to train the attention vectors.

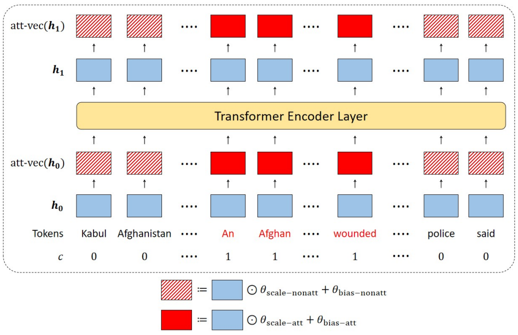

att-vec($\boldsymbol{h}_1$)
$\boldsymbol{h}_1$
Transformer Encoder Layer
att-vec($\boldsymbol{h}_0$)
$\boldsymbol{h}_0$
Tokens
Kabul
Afghanistan
An
Afghan
wounded
police
said
$c$
0
0
1
1
1
0
0
$:= \odot\, \theta_{\text{scale-nonatt}} + \theta_{\text{bias-nonatt}}$
$:= \odot\, \theta_{\text{scale-att}} + \theta_{\text{bias-att}}$

Figure 2: Illustration of our proposed attention vectors applied to a one-layer transformer encoder. The parameters of the transformer model are kept fixed. The highlighted spans are filled by red.

## 2.2 Attention Vectors

To control the model's focus, we introduce a set of $d$-dimensional vectors $\theta$, named *attention vectors (att-vec)*. They are designed to act as *indicators* for the model, designating which parts of the input to focus on. We now assume the training set contains $\{\mathbf{x}, \mathbf{c}^{\text{attr}}, \mathbf{y}\}$ triples, where $\mathbf{c}^{\text{attr}}$ is obtained from the attribution method from the previous section. Attention vectors modify the forward pass of the encoder model by applying a simple transformation on the output embeddings of each layer (including the input layer):

$$\text{att-vec}(\mathbf{h}_i^l) = \begin{cases} \mathbf{h}_i^l \odot \theta_{\text{scale-att}}^l + \theta_{\text{bias-att}}^l, & \text{if } c_i^{\text{attr}} = 1 \\ \mathbf{h}_i^l \odot \theta_{\text{scale-nonatt}}^l + \theta_{\text{bias-nonatt}}^l, & \text{if } c_i^{\text{attr}} = 0 \end{cases}. \quad (6)$$

We provide an illustration in Figure 2. The total number of parameters introduced by the attention vectors is therefore $4 \times (L+1) \times d$, which is negligible in comparison to the large number of parameters of the fixed transformer model. We note that as the attention vectors operate directly on the encoder embeddings, it does not require an explicit attention module to exist in the model and is therefore applicable to non-attentional architectures such as LSTMs (Huang et al., 2015).

We train the attention vectors using the standard NLL loss with stochastic gradient descent (SGD):

$$\mathcal{L}(\mathbf{x}, \mathbf{y}, \mathbf{c}^{\text{attr}}; \theta) = -\log P_{\text{att-vec}}(\mathbf{y}|\mathbf{x}, \mathbf{c}^{\text{attr}}), \quad (7)$$

where $P_{\text{att-vec}}(\cdot|\mathbf{x}, \mathbf{c}^{\text{attr}})$ denotes the distribution over the output after the attention vectors are applied. We re-iterate that during training of the attention vectors, the transformer model is kept fixed. This allows the user to default back to the pretrained model (i.e., without applying the attention vectors), if the user prefers not to specify any highlights.

Readers may wonder what the difference is between our approach and standard end-to-end training, as both cases use the same $\mathbf{x}, \mathbf{y}$ pairs. This is related to our key assumption that *different focus of the input lead to different generations*, and the fact that $\mathbf{c}^{\text{attr}}$ is the relevant span for $\mathbf{y}$ in the ideal case. Therefore, the attention vectors have the opportunity to *guide* the model and give information about which span is more relevant to $\mathbf{y}$ before the model observes $\mathbf{y}$ on the decoder side. To reduce the loss $-\log P_{\text{att-vec(M)}}(\mathbf{y}|\mathbf{x}, \mathbf{c}^{\text{attr}})$, the attention vectors need to steer the model's focus towards the spans marked by $\mathbf{c}^{\text{attr}}$.

At test time, the user will highlight several sentences in the input which we denote by $\mathbf{c}^{\text{user}}$. We apply the trained att-vec according to Equation 6, and decode the output from $P_{\text{att-vec}}(\cdot|\mathbf{x}, \mathbf{c}^{\text{user}})$.

## 3 Datasets

We test our method on two NLG tasks: dialogue response generation with the PersonaChat dataset (Zhang et al., 2018), and abstractive summarization with the CNN/Dailymail dataset (Hermann et al., 2015; Nallapati et al., 2016) .

**PersonaChat** *PersonaChat* is an open domain multi-turn chit-chat dataset, where two participants are required to get to know each other by chatting naturally. Each of them is given a *persona*: several pieces of personal information such as *"I major in Computer Science"*, serving as background information. The participants are required to reflect their assigned persona in the conversation. The dataset contains 8,939 dialogues for training, 1,000 for validation, and 968 for test. For each turn in the dialogue, we concatenate the persona of the speaker and the dialogue history as input, and train the base model to generate the current utterance. In some cases, the dialogue history is long and exceeds the input limit of the model, in which case we truncate the dialogue at the sentence level. The average number of sentences is around 11 after truncation.

**CNN/Dailymail** *CNN/Dailymail* is a standard dataset for end-to-end abstractive summarization. It contains 287,113 training examples, and 13,368 / 11,490 examples for validation / test. We apply the same truncation strategy as *PersonaChat* during preprocessing. The processed articles have an average length of 748 tokens, and the reference summaries have an average length of 67 tokens.

| Attribution Method | PersonaChat<br>P@1(%) | CNN/Dailymail<br>P@1(%) |
|---|---|---|
| attention-weight | 29.18 | 40.31 |
| grad-norm | 54.00 | 43.87 |
| grad-input-product | 44.05 | 32.60 |
| leave-one-out | **62.31** | **64.43** |

Table 1: Top-1 precision (%) of different attribution methods on the human-labeled development set.

**Human-annotated Evaluation Data** Both PersonaChat and CNN/Dailymail are created end-to-end and do not contain annotated highlight spans. For principled evaluation, we utilize the Amazon Mechanical Turk (AMT) platform to collect evaluation sets where the highlight-generation pairs are annotated by humans.

For PersonaChat, each turker is shown a dialogue history and the corresponding persona of the speaker. The dialogue history is randomly selected from the original test set of PersonaChat. Then the turker is required to choose 1-3 sentences as highlights (for example, one sentence in persona, and one sentence in dialogue history), and write down a dialogue response that not only continues the current dialogue, but also is relevant to the chosen highlights. Finally, we ask the turker to repeat the above process, but select a different set of highlights and provide another response. After a few preliminary trials and modifications to our instructions / rewards, we find that turkers comply nicely with our instructions and provide high-quality highlight-response pairs.

For CNN/Dailymail however, we first found that turkers had difficulty writing a high-quality summary for a given news article, with many turkers giving random responses even after we increased the reward. This is perhaps unsurprising given that writing a good summary is challenging and the reference summaries are written by experts. After a few disappointing iterations, we turn to a compromise: we directly provide the turkers with the reference summary, and only ask them to select 2-5 relevant sentences in the article. This greatly simplifies the task, and we are able to collect high-quality labels. This compromise is not ideal, as it reverses the order of highlighting and generation. However, we find that in most cases, the reference summaries in CNN/Dailymail were well covered by several "key" sentences in the article, which are highlighted by the turkers. Therefore, we believe this compromise does not hurt the soundness of our evaluation.

In order to ensure high data quality for both dialogue and summarization, we design a qualification test that turkers need to pass before conducting the actual tasks. Several automatic checks and a minimal time limit are added in the scripts to prevent trivial answers. We also manually monitor the incoming submissions, and ban misbehaving turkers and discard their submissions. More details about our AMT setup are provided in Appendix B.

Our final collected datasets include 3,902 highlight-generation pairs for PersonaChat, and 4,159 pairs for CNN/Dailymail. They are randomly splitted 50/50 into dev/test sets. We include a number of samples of our collected data in the supplementary materials. Our code and the collected dataset will be released in the public version of this manuscript. We hope that this evaluation data could facilitate future research in this direction.

**Comparison of Attribution Methods** We use the collected highlight-generation pairs in the dev set to compare which attribution method aligns best with human-annotated highlights. In particular, we compute the top-one precision of the sentence ranked highest by the attribution method. The result is shown in Table 1. We find that for both PersonaChat and CNN/Dailymail, LOO has the best alignment. We therefore use LOO to obtain automatic annotations for attention vector training. Interestingly, we observe low alignment between attention weight-derived attribution scores and human judgment, which potentially indicates that controlling model generatons via intervening on the attention distributions may not optimal. Finally, we note that this result does not mean LOO is the "best" attribution method, as attribution method is supposed to reflect the model's inner working, instead of a human's.

# 4 Experiments

## 4.1 Experiment Setting and Baselines

We use Blenderbot (Roller et al., 2020) as the base model for PersonaChat and BART (Lewis et al., 2019) for CNN/Dailymail, both of which are standard encoder-decoder transformer models. Our code is based on the *transformers* library (Wolf et al., 2020). We load the pretrained weights from `facebook/blenderbot-400M-distill` and `facebook/bart-base`. Blenderbot has 2 encoder layers and 12 decoder layers, while Bart has 6 encoder layers and 6 decoder layers. To help Blenderbot cope with long dialogue context in PersonaChat, we extend its maximum position embedding index from 128 to 256. We use beam-search for decoding, where we follow the recommended configuration (Roller et al., 2020; Lewis et al., 2019) and use a beam size of 10 for Blenderbot and a beam size of 4 for Bart.

For both tasks, we first finetune the base model on the original training set in the standard end-to-end manner. The model is then fixed and used to obtain automatic labels $\mathbf{c}^{\text{attr}}$ with the LOO attribution method on the same training set. For each training sample, we select the top-$k$ sentences in the input ranked by LOO. Since we do not know the best value for $k$, we set it to be a random number from 1 to 3 for PersonaChat, and from 2 to 5 for CNN/Dailymail.

While the highlight labels in the training set used to train attention vectors are derived automatically, we use the human-labeled dev set for hyper-parameter tuning. This is to facilitate fair comparison with other baseline approaches which also utilize the human-labeled dev set. In our ablation study, we will show that this dependence on human-labeled dev set is not crucial for our approach to achieve strong performance. We perform a grid search over learning rate with $\{1, 3, 5\} \times \{1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}\}$. The Adam optimizer (Kingma and Ba, 2014) is used with $\beta_1 = 0.9, \beta_2 = 0.999$, and a L2 decay weight of $0.01$. For both tasks, we set the mini-batch size to be 16.

We compare the proposed att-vec approach with several baselines:

**Vanilla:** The vanilla model, without any modification in both the model and the input.

**Padding:** One trivial way to control the model's attention is to replace all input by the `<pad>` token, except the spans highlighted by the user. However, we find that this direct padding during evaluation results in drastically worse perplexity. To alleviate this problem, we randomly pad a portion of sentences in the input during the standard end-to-end finetuning, to make the model aware that only partial input would be provided.

| Model | PersonaChat |  |  | CNN/Dailymail |  |  |
|---|---|---|---|---|---|---|
|  | PPL | ROUGE-1/2/L | BERTScore | PPL | ROUGE-1/2/L | BERTScore |
| padding | 38.93 | 16.69/2.80/13.72 | 84.42 | 19.62 | 39.31/18.44/28.67 | 88.34 |
| vanilla | 28.73 | 17.02/2.73/14.52 | 85.41 | 4.51 | 43.48/21.01/30.98 | 89.23 |
| att-offset | 23.79 | **21.10**/3.77/17.54 | 86.04 | 4.49 | 43.96/20.64/31.26 | 89.28 |
| att-vec | **22.51** | 20.81/**3.98/17.58** | **86.13** | **4.48** | **45.92/23.03/32.98** | **89.78** |

Table 2: Main results on the PersonaChat and CNN/Dailymail datasets. The proposed attention vector approach shows strong performance across different metrics.

**Att-offset:** As a direct way to control the model's attention, we add a positive scalar offset $s^{\text{offset}}$ to the cross-attention heads before the softmax operation (Equation 1), for the highlighted spans. A similar technique has been used in Dong et al. (2021) to *modulate* the attention distribution to tackle neural text degeneration problems (Holtzman et al., 2019). This approach modifies the attention weights via:

$$\alpha'_{i,j} = \underset{i\in\{1\ldots n\}}{\text{softmax}} \left( \frac{k(\mathbf{h}_i^L)\cdot \mathbf{q}_j}{\sqrt{d}} + s^{\text{offset}} \cdot \mathbb{1}_{[c_i=1]} \right), \quad (8)$$

where $s^{\text{offset}}$ is a hyper-parameter, and is applied to all cross-attention heads in the decoder. We tune $s^{\text{offset}}$ on the human-annotated development set in a fine-grained manner. More details are given in Appendix A.

Whether the attention distribution faithfully explains a model's predictions is the subject of much current debate (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Bastings and Filippova, 2020). Therefore this direct modification of the attention head may not be the optimal solution for achieving attention control. Our proposed att-vec framework, on the other hand, utilizes attribution methods, and directly operates on the encoder embeddings.

### 4.2 Results and Analysis

During evaluation, human-annotated highlights are fed to the model. In addition to perplexity, we evaluate the generations from different approaches using two popular NLG metrics: ROUGE (Lin, 2004), and BERTScore (Zhang et al., 2019).

We show the main results in Table 2. As expected, the padding baseline has poor performance, as a large portion of input is masked out. Compared to the vanilla baseline, att-vec obtains significantly improved ROUGE and BERTScore on both tasks. This validates the motivation of this work: att-vec is effective in steering the model's focus, which leads towards the desired generation. For CNN/Dailymail, the perplexity of att-vec is close to the vanilla model even though there is a large difference in ROUGE. We believe this is due to the constrained nature of the summarization task and how perplexity is computed: once the model observes the first few tokens, it is easy to figure out what the current highlight is. The other two metrics, on the other hand, are based on the actual generation, and therefore does not have this issue.

The performance gap (in ROUGE/BERTScore) between att-vec and att-offset is larger on the CNN/Dailymail dataset. We believe this is because the BART model has a deeper encoder than the Blenderbot model. As the encoder grows deeper, the embeddings become more "contextualized" and its *identifiability* (Brunner et al., 2020) degrades. And since the decoder attends to the last layer of the encoder, this direct manipulation of attention weights could be ineffective with deep encoders.

Table 3 shows generation samples from different attention control approaches for PersonaChat. Spans of the generation that are relevant to the highlighted persona are marked in red. Comparing to the generation from the vanilla model, the generations from both att-offset and att-vec are highly relevant to the respective highlighted persona. One generation from att-offset is a little erratic (*"I am petro, my dog"*), which may be due to the inflexibility of att-offset. We defer the generation examples for CNN/Dailymail to Table 6 and Table 7 (Appendix C) due to space constraints. We observe that the generation from att-vec is quite different from the vanilla model, and is more focused on the highlighted inputs. On the other hand, att-offset's generation still remains similar to the vanilla model.

*Does att-vec really change the model's "attention"*? In Figure 3, we study how the outputs of attribution methods (attention-weight and grad-norm) change with different baselines (vanilla, att-vec and att-offset) for the CNN/Dailymail example (Table 6). Note that in this analysis, for the attribution methods we set the target $\mathbf{y}$ to be the decoded output from the respective modeling, instead of the reference summary. The highlighted sentences are marked by the red rectangles.

| | | |
|---|---|---|
| Persona: | **I have a dog named pedro.** I like to eat muffins. I am five feet tall. I work in the healthcare industry. | I have a dog named pedro. **I like to eat muffins.** I am five feet tall. I work in the healthcare industry. |
| Context: | [P-1]: Hello, how are you today?<br>[P-2]: [TO ANSWER] | [P-1]: Hello, how are you today?<br>[P-2]: [TO ANSWER] |
| reference: | Scared, i'm looking for my dog, his name is pedro. have you seen him? | Just baking some muffins, if you got good vibes come on over and share a muffin with me! |
| vanilla: | Hi! I am doing well, how about yourself? What do you do for a living? | Hi! I am doing well, how about yourself? What do you do for a living? |
| att-offset: | I am doing well, and you? I am petro, my dog, and you? | I am doing well. I just finished eating a delicious muffin. How about you? |
| att-vec: | I am good. Just got back from walking my dog. Do you have any pets? | I am doing well. What do you do for a living? I am a muffin maker. |

Table 3: Generation samples from different attention control approaches for PersonaChat. The inputs on both sides are identical, except a different piece of persona is highlighted (marked in bold), which leads to different generations from att-offset and att-vec. Another example is provided in Table 5 (Appendix C).

We observe that for both attention-weight and grad-norm, the application of att-vec make the highlighted sentences obtain the highest attribution scores, and the scores differ significantly from the vanilla model. In some of the non-highlighted sentences (marked by the blue rectangles), att-offset is not strong enough to significantly reduce its attribution. We also tried larger values of $s^{\text{offset}}$ for att-offset but found it lead to performance degradation. This analysis shows that despite the small number of parameters associated with the attention vectors, they are able to effectively steer the model's focus. We provide a simple visualization of the trained att-vec parameters in Figure 3 (Appendix C).

**Ablation Studies** Table 4 shows several variants of att-vec on CNN/Dailymail. We first tune the hyper-parameters of att-vec only with the original dev set with $\mathbf{c}^{\text{attr}}$, instead of human-annotated highlights. Despite this discrepancy, att-vec still achieves strong performance on the test set. This result shows that the use of human-annotated dev set is not crucial for our framework. We then conduct an ablation study where we only apply att-vec on the first or last layer of the encoder, which reduces the number of parameters. We find that this results in marginal performance degradation. Finally, we jointly finetune attention vectors and the whole model with the same loss function (Equation 7), where a separate and smaller learning rate is used for the model. Interestingly, the gain from model finetuning is very limited, which demonstrates the effectiveness of att-vec.

## 5 Related Work

Our proposed attention vector framework is closely related to the research topics of controllable text generation, LM adaptation, and attention/attribution analysis, which we review below.

| Model | CNN/Dailymail | | |
|---|---|---|---|
| | PPL | ROUGE-1/2/L | BERTScore |
| all-layer* | 4.48 | 45.92/23.03/32.98 | 89.78 |
| ori-dev with $\mathbf{c}^{\text{attr}}$ | 4.50 | 46.41/22.69/32.48 | 89.62 |
| only first layer | 4.48 | 45.67/22.63/32.45 | 89.59 |
| only last layer | 4.48 | 46.06/22.84/32.69 | 89.69 |
| plus model finetune | 4.49 | 46.65/23.54/33.30 | 89.82 |

Table 4: Performance of different variants of att-vec trained on CNN/Dailymail. all-layer* refers to our proposed modelling (also reported in Table 2).

**Controllable Text Generation** Prior work on controllable summarization introduced various types of control mechanisms. Fan et al. (2017); Saito et al. (2020) extract entity, keyword or length, as additional supervision during training. Gehrmann et al. (2018) trains a token-level content selection module, where the supervision is by aligning the summaries to the documents. (Song et al., 2021) proposes a two-staged generation strategy and Goyal et al. (2021) incorporates multiple decoders into a transformer framework. Some recent work (He et al., 2020; Dou et al., 2020) uses prompts to control the generation.

Existing work on controllable dialogue response generation include using conditional variational autoencoders (Zhao et al., 2017; Li et al., 2020), and incorporating external knowledge into the conversational agent using knowledge graphs (Cui et al., 2021; Moon et al., 2019), unstructured documents (Kim et al., 2020), or dialogue context (Zhao et al., 2020).

In open-ended language generation, a series of approaches have been proposed to control for some attribute (e.g., topic) of the generation (Keskar

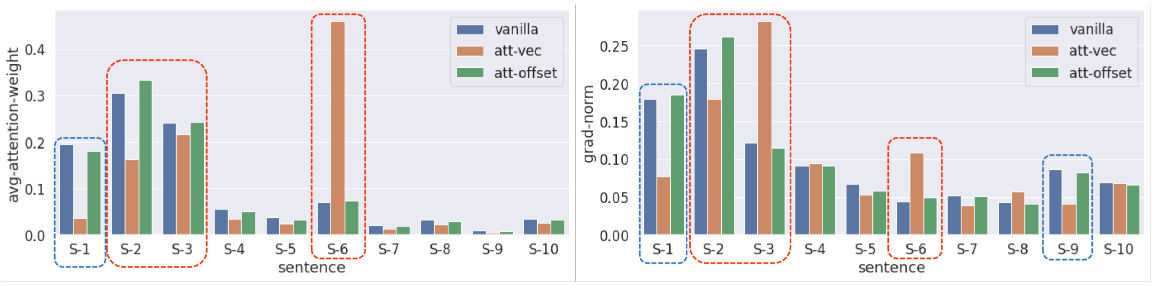


Figure 3: Attribution scores for each sentence in the input, with different attention control approach applied to BART. The highlighted sentences are marked by the red rectangles. The corresponding example is given in Table 6 (Appendix C).

et al., 2019; Dathathri et al., 2020; Krause et al., 2020; Yang and Klein, 2021). Some of these studies utilize a trained classifier to guide the generative model towards the desired attribute.

**LM Adaptation** Our proposed attention vector framework is also inspired by a series of recent works on prompting or light-weight LM adaptation. Li and Liang (2021), followed by Lester et al. (2021) and Zhong et al. (2021), propose *prefix tuning*, where continuous task-specific input vectors are tuned to adapt the pretrained LM to a downstream task with supervised data, and the model is kept fixed.

There is also a line of works on *adapter-tuning*, which insert and finetune task-specific layers (adapters) between each layer of the pretrained LM (Houlsby et al., 2019; Lin et al., 2020; Pfeiffer et al., 2021). More recently, Guo et al. (2021) and Ben-Zaken et al. (2020) propose to finetune only a small subset of a pretrained model's parameters, and achieves strong performance on the GLUE benchmark (Wang et al., 2018).

**Attention Analysis and Attribution Methods** Due to the ubiquity of the attention module in current NLP models, various work has studied how the module captures linguistic phenomena in the input (Clark et al., 2019; Kovaleva et al., 2019; Kobayashi et al., 2020). It has also been used as a tool to interpret the model's predictions (Wang et al., 2016; Lee et al., 2017; Ghaeini et al., 2018).

Recently, there have been a series of studies discussing the use of attention weights for interpretability (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Bastings and Filippova, 2020; Serrano and Smith, 2019), and it has been argued that attribution methods are a better choice to explain the model's predictions. The poor alignment performance of attention weights that we get in Table 1, on some level, are in agreement with that argument. Our work is also related to the line of work on interpreting black box models through *rationales* (Lei et al., 2016; Bastings et al., 2019), which are typically (discrete) subsets of the input that are used to predict the output. Finally, several recent works (Xu and Durrett, 2021; Ding and Koehn, 2021) have compared different attribution methods for interpreting NLP models.

In comparison to the aforementioned works, our major innovations are two fold: (1) Our goal is to control the *focus* of pretrained models, and thereby steer the model's generation, and our proposed attention vectors are compatible with the standard transformer architecture; (2) We utilize attribution methods to obtain automatic annotations for att-vec training. Therefore, our framework can be applied to a wide range of NLG applications.

## 6 Conclusion

In this work we propose the attention vector framework as a light-weight solution to control the focus of pretrained transformer models. It has two major advantages: (1) Attention vectors act as simple transformations to the embeddings in the encoder, and the transformer model is kept fixed; (2) Attribution methods are utilized to get automatic highlight labels for training attention vectors.

We test our approach on two tasks: dialogue response generation, and abstractive summarization. For evaluation, we collect data where the highlight-generation pairs are annotated by humans. Experiments show that the trained attention vectors are effective in steering the model to generate output text that is relevant to the specified highlights.

## References

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. 2010. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(61):1803–1831.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Jasmijn Bastings and Katja Filippova. 2020. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 149–155, Online. Association for Computational Linguistics.

Elad Ben-Zaken, Shauli Ravfogel, and Yoav Goldberg. 2020. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2020. On identifiability in transformers. In *International Conference on Learning Representations*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Leyang Cui, Yu Wu, Shujie Liu, and Yue Zhang. 2021. Knowledge enhanced fine-tuning for better handling unseen entities in dialogue generation. *arXiv preprint arXiv:2109.05487*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Misha Denil, Alban Demiraj, and Nando de Freitas. 2014. Extraction of salient sentences from labelled documents. *CoRR*, abs/1412.6815.

Shuoyang Ding and Philipp Koehn. 2021. Evaluating saliency methods for neural language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5034–5052, Online. Association for Computational Linguistics.

Yue Dong, Chandra Bhagavatula, Ximing Lu, Jena D. Hwang, Antoine Bosselut, Jackie Chi Kit Cheung, and Yejin Choi. 2021. On-the-fly attention modulation for neural generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1261–1274, Online. Association for Computational Linguistics.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2020. Gsum: A general framework for guided neural abstractive summarization. *arXiv preprint arXiv:2010.08014*.

Angela Fan, David Grangier, and Michael Auli. 2017. Controllable abstractive summarization. *arXiv preprint arXiv:1711.05217*.

Sebastian Gehrmann, Yuntian Deng, and Alexander M. Rush. 2018. Bottom-up abstractive summarization. *CoRR*, abs/1808.10792.

Reza Ghaeini, Xiaoli Fern, and Prasad Tadepalli. 2018. Interpreting recurrent and attention-based neural models: a case study on natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4952–4957, Brussels, Belgium. Association for Computational Linguistics.

Tanya Goyal, Nazneen Fatema Rajani, Wenhao Liu, and Wojciech Kryściński. 2021. Hydrasum–disentangling stylistic features in text summarization using multi-decoder models. *arXiv preprint arXiv:2110.04400*.

Demi Guo, Alexander Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4884–4896, Online. Association for Computational Linguistics.

Junxian He, Wojciech Kryściński, Bryan McCann, Nazneen Rajani, and Caiming Xiong. 2020. Ctrlsum: Towards generic controllable text summarization. *arXiv preprint arXiv:2012.04281*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28:1693–1701.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. *CoRR*, abs/1902.10186.

Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *CoRR*, abs/1909.05858.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential latent knowledge selection for knowledge-grounded dialogue. *arXiv preprint arXiv:2002.07510*.

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

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4365–4374, Hong Kong, China. Association for Computational Linguistics.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq R. Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. Gedi: Generative discriminator guided sequence generation. *CoRR*, abs/2009.06367.

Jaesong Lee, Joong-Hwi Shin, and Jun-Seok Kim. 2017. Interactive visualization and manipulation of attention-based neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 121–126, Copenhagen, Denmark. Association for Computational Linguistics.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chunyuan Li, Xiang Gao, Yuan Li, Baolin Peng, Xiujun Li, Yizhe Zhang, and Jianfeng Gao. 2020. Optimus: Organizing sentences via pre-trained modeling of a latent space. *arXiv preprint arXiv:2004.04092*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *CoRR*, abs/2101.00190.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 441–459, Online. Association for Computational Linguistics.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Ça glar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *CoNLL 2016*, page 280.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. 2020. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, and Junji Tomita. 2020. Abstractive summarization with combination of pre-trained sequence-to-sequence and saliency models. *arXiv preprint arXiv:2003.13028*.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? *CoRR*, abs/1906.03731.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153. PMLR.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *In Workshop at International Conference on Learning Representations*.

Kaiqiang Song, Bingqing Wang, Zhe Feng, and Fei Liu. 2021. A new approach to overgenerating and scoring abstractive summaries. *arXiv preprint arXiv:2104.01726*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

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 I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615, Austin, Texas. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

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

Jiacheng Xu and Greg Durrett. 2021. Dissecting generation modes for abstractive summarization models via ablation and attribution. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6925–6940, Online. Association for Computational Linguistics.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online. Association for Computational Linguistics.

Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *Computer Vision – ECCV 2014*, pages 818–833, Cham. Springer International Publishing.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. *arXiv preprint arXiv:1703.10960*.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. *arXiv preprint arXiv:2010.08824*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [MASK]: learning vs. learning to recall. *CoRR*, abs/2104.05240.

# Appendices

## A Implementation Details

For the att-vec baseline, we tune the offset $s^{\text{offset}}$ in a fine-grained manner, on the human-annotated dev set. We first set a relatively large max value (100) and get 20 evenly spaced numbers inside the interval $(0, 100)$. Then we calculate model PPL on the dev set with $s^{\text{offset}}$ set to these different offsets. Then we do another search in the interval that has lowest PPL. We repeat this iteration multiple times, and stops when PPL change is smaller than $1e^{-3}$. The final tuned value for Blenderbot is around 3.02, and around 0.17 for BART.

## B Evaluation Data Collection

To improve the quality of collected dataset, we design a qualification test, which the turkers need to pass before they can work on real assignments. The test is designed to help turkers understand our task better. For PersonaChat, we give turkers two dialogue samples with pre-selected highlights, and ask them to choose the appropriate response that not only continues the dialogue, but also is relevant to the highlights. For CNN/Dailymail , the turkers are shown two example articles and the corresponding reference summaries. We have already picked some highlights in the article, but there is one highlight missing. And the turker is required to pick the missing highlight. The interface for the PersonaChat qualification test is shown in Figure 5.

We also add multiple checks in our script to prevent trivial answers. We ban trivial copy&paste from the given context. A time check is added that requires turker to spend at least 60 seconds on a single HIT. For the two assignments in PersonaChat, we add a content check that prevents duplicate highlights or response. We show our interface for PersonaChat in Figure 6. Despite these checks and the qualification tests, there still exist a small number of misbehaving turkers who attempt to cheat. Therefore we also manually monitor the incoming submissions, and ban misbehaving turkers and filter out their submissions.

More examples of our interface and instructions can be found in our uploaded data samples.

## C Auxiliary Results and Examples

In Figure 4, we provide a simple visualization of the trained attention vectors of BART. To make the

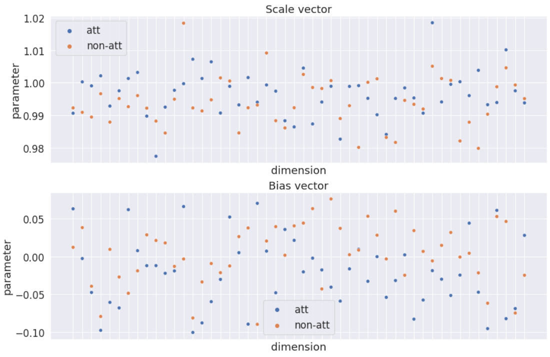
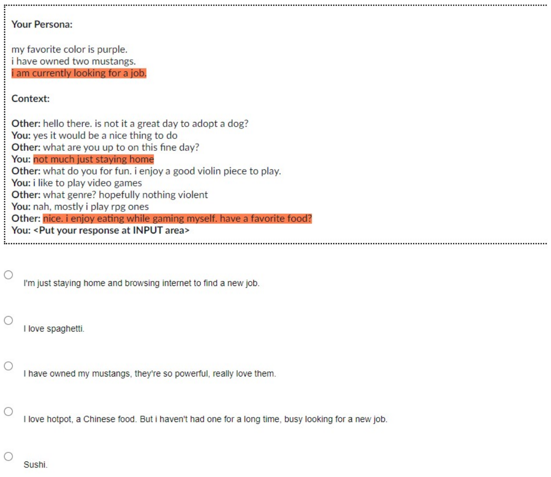


Figure 4: 50 random dimensions of the trained attention vector on first encoder layer of the BART model.



**Your Persona:**

my favorite color is purple.
i have owned two mustangs.
i am currently looking for a job.

**Context:**

**Other:** hello there. is not it a great day to adopt a dog?
**You:** yes it would be a nice thing to do
**Other:** what are you up to on this fine day?
**You:** not much just staying home
**Other:** what do you for fun. i enjoy a good violin piece to play.
**You:** i like to play video games
**Other:** what genre? hopefully nothing violent
**You:** nah, mostly i play rpg ones
**Other:** nice. i enjoy eating while gaming myself. have a favorite food?
**You: <Put your response at INPUT area>**

○ I'm just staying home and browsing internet to find a new job.

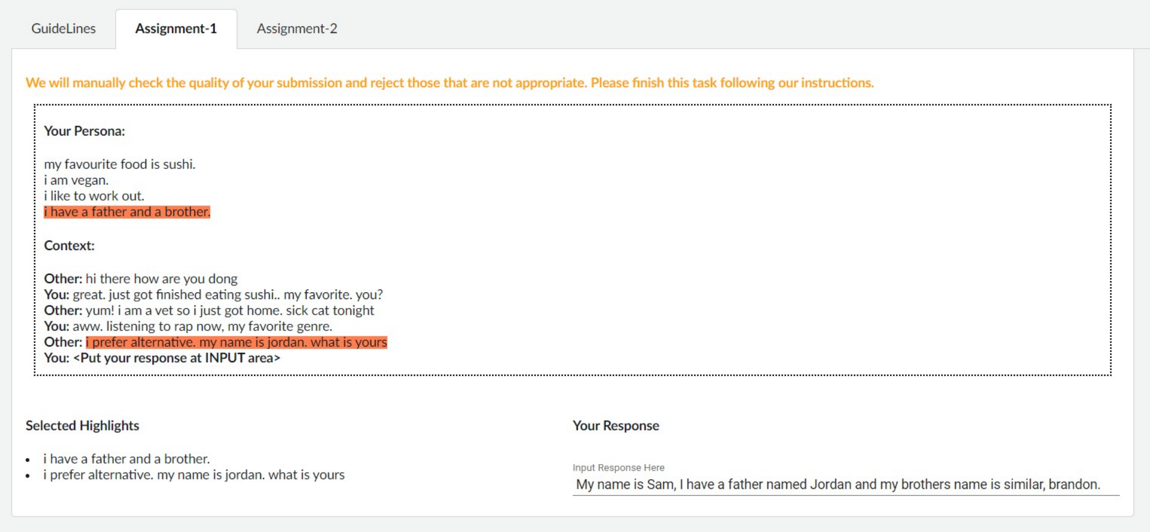
○ I love spaghetti.

○ I have owned my mustangs, they're so powerful, really love them.

○ I love hotpot, a Chinese food. But i haven't had one for a long time, busy looking for a new job.

○ Sushi.

Figure 5: An example of our AMT qualification test for PersonaChat. We have chosen the highlights in the context, and the turker is supposed to choose a response that not only continues the dialogue, but also is relevant to the highlights.



We will manually check the quality of your submission and reject those that are not appropriate. Please finish this task following our instructions.

**Your Persona:**

my favourite food is sushi.
i am vegan.
i like to work out.
i have a father and a brother.

**Context:**

**Other:** hi there how are you dong
**You:** great. just got finished eating sushi.. my favorite. you?
**Other:** yum! i am a vet so i just got home. sick cat tonight
**You:** aww. listening to rap now, my favorite genre.
**Other:** i prefer alternative. my name is jordan. what is yours
**You: <Put your response at INPUT area>**

**Selected Highlights**

- i have a father and a brother.
- i prefer alternative. my name is jordan. what is yours

**Your Response**

Input Response Here
My name is Sam, I have a father named Jordan and my brothers name is similar, brandon.

Figure 6: An example of our AMT interface on PersonaChat. The highlights and a response are labeled by a turker.

figure easy to grasp, we randomly sample 50 dimensions (out of 768) of trained att-vec parameters. In the figure only the trained parameters for the first encoder layer is shown, and we find that the patterns for the other layers are similar.

We first find that the learned scaling parameters are close to 1 and the bias parameters are close to zero. This implies that the transformation defined by the att-vec is not drastic, and a relatively small tweak could be enough to steer the model's focus.

An interesting pattern can be observed for the scaling vectors. $\theta_{\text{scale-nonatt}}$ tends to down-scale the embedding, comparing to $\theta_{\text{scale-att}}$. This matches our intuition that the embeddings on the non-highlighted positions should be down-played.

| | | |
|---|---|---|
| Persona: | My parents did not want me. **It's a dead end job so i am looking for something different.** I was raised by my uncle and aunt. | My parents did not want me. It's a dead end job so i am looking for something different. **I was raised by my uncle and aunt.** |
| Context: | [P-1]: Hello how are you doing?<br>[P-2]: Pretty good. how are you?<br>[P-1]: I am making dinner getting ready for bed.<br>[P-2]: I just got off work.<br>[P-1]: **I am looking for something different to do.**<br>[P-2]: [TO ANSWER] | [P-1]: Hello how are you doing?<br>[P-2]: Pretty good. how are you?<br>[P-1]: I am making dinner getting ready for bed.<br>[P-2]: I just got off work.<br>[P-1]: **I am looking for something different to do.**<br>[P-2]: [TO ANSWER] |
| reference: | Do you know anywhere that's hiring work from home? I'm looking for a new job. | I grow up with my aunt and we always had scripture study after dinner, you can try it. |
| vanilla: | My parents wanted me to be a doctor like them, but i do not want to do that. | My parents wanted me to be a doctor like them, but i do not want to do that. |
| att-offset: | What do you do for a living? I have been looking for a new job myself. | Do you have any relatives you can ask for help with? my family was always there for me. |
| att-vec: | What do you do for a living? I am unemployed right now looking for a job. | What do you do for a living? I work with my aunt at her law firm. |

Table 5: Generation samples from different attention control approaches for PersonaChat. The inputs on both sides are identical, except a different piece of persona is highlighted, which leads to different generations from att-offset and att-vec.

| | |
|---|---|
| Article: | England World Cup hopeful Elliot Daly has been named the Aviva Premiership's player of the month for March.[1] **The uncapped Wasps centre, 22, scored two tries in March as the Coventry-based club maintained on course for a Champions Cup spot next season**.[2] **Equally comfortable at full back, Daly's form at centre has him pushing incumbent England No 13 Jonathan Joseph hard for his place**.[3] Elliot Daly is handed his player of the month award by Wasps director of rugby Dai Young.[4] He looks certain to be included in Stuart Lancaster's England's World Cup training squad named next month.[5] **Wasps director of rugby Dai Young said: 'If England are going to go out and win the Rugby World Cup, he's the type of player you've got to have in the team so I'd certainly like to think he's pushing for a place in the squad.'**.[6] BT Sport analyst Austin Healey added: 'For a good couple of years now us rugby watchers have known of the potential of Elliot Daly.'[7] $\cdots$ (LEAVE OUT UNIMPORTANT SENTECES) |
| reference: | Elliot Daly was in fine form at outside centre for Wasps in March. Daly, 22, has not yet been capped by England. Wasps head coach Dai Young says England must pick him for World Cup. |
| vanilla: | England World Cup hopeful Elliot Daly has been named Aviva Premiership's player of the month for March. The 22-year-old scored two tries in March as Coventry-based club maintained on course for a Champions Cup spot next season. Daly's form at centre has him pushing incumbent England No 13 Jonathan Joseph hard for his place. |
| att-offset: | England World Cup hopeful Elliot Daly has been named Aviva Premiership's player of the month for March. The uncapped Wasps centre scored two tries in March as Coventry-based club maintained on course for a Champions Cup spot next season. Daly's form at centre has him pushing incumbent England No 13 Jonathan Joseph hard for his place. |
| att-vec: | The uncapped Wasps centre scored two tries in March. Daly's form at centre has him pushing incumbent England No 13 Jonathan Joseph hard for his place. Wasps director of rugby Dai Young said: 'If England are going to go out and win the Rugby World Cup, he's the type of player you've got to have in the team.' |

Table 6: Generation samples of different attention control approaches on CNN/Dailymail dataset. The highlighted setences are marked in bold. The span marked in red in the reference summary is captured by the generation from att-vec, but not by att-offset. The sentence number marked in the input corresponds the sentence index in Figure 3.

| | |
|---|---|
| Article: | **Kabul, Afghanistan (CNN)A suicide bomber detonated his explosives near a group of protesters in eastern Afghanistan on Thursday, killing 17 people and wounding dozens more, police said**.[1] **"An Afghan lawmaker taking part in the protests in the city of Khost was among the 64 people wounded,"**[2] **said Faizullah Ghairat**, **the provincial police chief Taliban spokesman Zabiullah Mujahid denied his group was responsible for the attack**.[3] No other organization has so far claimed responsibility.[4] Humayoon Humayoon, an Afghan member of parliament for Khost province, and the other protesters were on their way to join a larger rally against the provincial governor, according to Zahir Jan, an eyewitness.[5] The suicide attack hit the group around 10 a.m. local time, police said.[6] · · · (LEAVE OUT UNIMPORTANT SENTENCES) |
| reference: | An Afghan lawmaker is among 64 people wounded in the attack, police say. Taliban spokesman denies his group was responsible for the attack. |
| vanilla: | A suicide bomber detonates his explosives near a group of protesters, killing 17 people and wounding dozens more. The Taliban spokesman denies his group was responsible for the attack. No other organization has so far claimed responsibility for the attack. The suicide attack hit the group around 10 a.m. local time. |
| att-offset: | A suicide bomber detonates his explosives near a group of protesters, killing 17 people and wounding dozens more. The Taliban spokesman denies his group was responsible for the attack. No other organization has so far claimed responsibility for the attack. The suicide attack hit the group around 10 a.m. local time. |
| att-vec: | A suicide bomber detonates his explosives near a group of protesters, killing 17 people. An Afghan lawmaker is among the 64 people wounded, police say. Taliban spokesman Zabiullah Mujahid denies his group was responsible for the attack. No other organization has so far claimed responsibility. |

Table 7: Generation samples of different attention control approaches on CNN/Dailymail dataset. The span marked in red in the reference summary is captured by the generation from att-vec, but not by att-offset.