**Abstract**

Recent advances in natural language processing have led to the availability of large pre-trained language models (LMs), with rich generative capabilities. Although these models are able to produce fluent and coherent text, it remains a challenge to control various attributes of the generation, including sentiment, formality, topic and many others. We propose a Beam Reweighing (BEAMR) method, building on top of standard beam search, in order to control different attributes. **BEAMR** combines any generative LM with an attribute discriminator, offering full flexibility of generation style and attribute, while the beam search backbone maintains fluency across different domains. Notably, **BEAMR** allows practitioners to leverage pre-trained models without the need to train generative LMs together with discriminators. We evaluate **BEAMR** in two diverse tasks: sentiment steering, and machine translation formality. Our results show that **BEAMR** performs on par with or better than existing state-of-the-art approaches (including fine-tuned methods), and highlight the flexibility of **BEAMR** in both causal and seq2seq language modeling tasks.

**1 Introduction**

Text generation has improved significantly in recent years due to architectural advances in deep learning (namely, the transformer architecture and attention mechanism ([Vaswani et al., 2017]) and training paradigms, allowing practitioners to train large language models on vast, unlabelled corpora, and transfer knowledge between various domains.

Controllable text generation involves generating text according to specific requirements, which may include a specific topic ([Baheti et al., 2018]), attribute ([Goswamy et al., 2020]), reward signal ([Tambwekar et al., 2019]), or other potential constraints. This task presents significant challenges, as large, unlabelled corpora are unlikely to be sufficient for learning domain-specific, controllable characteristics, and thus transferring knowledge becomes substantially more difficult. Moreover, due to the growing size of recent language models it is also less feasible to train and finetune them for many different controllable dimensions.

Recent work in controllable text generation involves various ways of incorporating desired attributes into the text generated by the base LM. Many approaches ([Yang and Klein, 2021; Liu et al., 2021; Ghazvininejad et al., 2017]) rely directly on decoding-time strategies in order to steer the generation towards a desired attribute. However, these approaches typically rely on token-level decoding which can result in various disfluencies in the output (e.g., repetition) ([Holtzman et al., 2020]) or limited generalizability due to tight coupling between the generation and attribute models. Several works ([Dathathri et al., 2020; Keskar et al., 2019; Krause et al., 2020; Zeldes et al., 2020; Khalifa et al., 2021]) attempt to tune a portion of the base LM in order to steer it towards a desired attribute. This tuning is either performed directly on the LM (i.e. via a fine-tuning stage), or using an auxiliary attribute model and applying gradient perturbations to LM latent states.

In this work, we propose a simple and robust decoding-based approach to controllable text generation, allowing practitioners to leverage existing, pre-trained, generative LMs and existing attribute models. Our method first uses the beam search algorithm to propose fluent and relevant candidates for a given input prompt from a generative language model. Subsequently, the candidates are scored by a discriminative model trained for a particular attribute (e.g., sentiment analysis, emotion detection, or topic classification). The candidate scores produced by beam search are combined with the scores from the discriminative model to produce a distribution over the candidates. We then sample a single candidate generation from this distribution. Our method solves some of the existing is-
sues in controllable text generation approaches, by (1) leveraging beam search to produce more fluent and relevant candidates, (2) expanding the generalizability of controllable generation via a custom similarity measure that can be selected based on the discriminative model, and (3) eliminating the need for tight coupling between the generative and discriminative models by reweighing at the natural language level, agnostic to the tokenization scheme, thereby allowing practitioners to leverage strong models for generation and scoring.

We perform several experiments with our approach, compare to several state-of-the-art methods for controllable text generation and show that BEAMR is generalizable to various LMs and target applications. First, we experiment with controlling the sentiment of generations using an attribute model finetuned for sentiment analysis. We then highlight the generalizability of the BEAMR method by applying it to the sequence-to-sequence task of adjusting the formality of text translated from Spanish to English. In sentiment steering experiments, BEAMR outperforms the SOTA DExPerts model (Liu et al., 2021) in positive steering, and offers good control ability in negative steering, such as sentiment, formality, emotion or topic. Our method first decodes a set of $K$ candidates, using diverse beam search (Vijayakumar et al., 2018) to improve variety among the candidates. The candidates are then scored using an attribute model. We then reweigh their likelihoods $\ell(b)$ with the attribute scores $s$ and apply a softmax transformation to produce a reweighed candidate distribution $\tilde{p}$, encoding fluency and attribute characteristics. The reweighed distribution is used to sample a single candidate.

More formally, let $B_j = \{b^1, \ldots, b^K\}$ denote the set of candidates for iteration $j$ of BEAMR and $b^k \in B_j$ denote the $k$th candidate, with likelihood $\ell(b^k)$. Let $g_\phi : \mathcal{P}(\mathcal{V}) \rightarrow \mathbb{R}^m$ represent a discriminator for an $m$-dimensional attribute. Given a target attribute vector $a \in \mathbb{R}^m$, we compute a score for candidate $b^k$:

$$d_k = D(g_\phi(b^k), a)$$

where $D : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$ is an appropriate similarity measure and $\gamma > 0$ is a scaling hyperparameter. Note that Eqn. (3) ensures that the scores

Greedy. In this approach, tokens are generated by iteratively choosing the most likely token from $p(x \mid c_t)$, and updating the prompt $c$.

Beam Search. In this approach a set of most likely candidates are maintained at each timestep. First, $K$ possible tokens are sampled or selected from $p(x \mid c_t)$. At each subsequent step, beam search expands the search space to $K^2$ possible hypotheses, before pruning back down to $K$ based on the likelihood of the candidates. For a given candidate $b_t = \{b_1, b_2, \ldots, b_t\}$, the likelihood is computed as

$$\ell(b_t) = \sum_{j \leq t} \log p(b_j \mid b_{<j})$$

Diverse Beam Search. Vijayakumar et al. (2018) proposed a modified version of beam search in order to produce more diverse candidates. They divide the set of all candidates into $G$ disjoint groups, and incorporate a group dissimilarity metric into the likelihood calculation.

3 Beam Reweighing

We propose to modify the beam search algorithm by reweighing the candidate likelihoods in order to control a diverse set of attributes of the text, such as sentiment, formality, emotion or topic. Our method first decodes a set of $K$ candidates, using diverse beam search (Vijayakumar et al., 2018) to improve variety among the candidates. The candidates are then scored using an attribute model. We then reweigh their likelihoods $\ell(b)$ with the attribute scores $s$ and apply a softmax transformation to produce a reweighed candidate distribution $\tilde{p}$, encoding fluency and attribute characteristics. The reweighed distribution is used to sample a single candidate.
are an increasing function of $\gamma$, by transforming the
output of $D$ so that $\mathbb{R} \rightarrow [1, \infty)$ without changing
the ranking order.

Combining the attribute score $s_k = s(b^k, a)$
with the likelihood $\ell(b^k)$ gives us a reweighed dis-
tribution $\tilde{p}$ over $B_j$:

$$\tilde{p}_k = \text{softmax}(\ell(b^k) + s_k) \quad (4)$$

A candidate can then be sampled from this dis-
tribution, $b \sim \tilde{p}$. This formulation is akin to a
product of experts model (Hinton, 2002; Welling,
2007) treating the LM $f_\theta$ as a linguistic expert and
the discriminator $g_\phi$ as an attribute expert. Figure
1 presents a diagram of the BEAMR procedure.

### 3.1 Generalizability of Beam Reweighing

Our formulation of BEAMR is flexible enough to
accommodate a variety of possible attributes and
discriminator models, including both continuous
and categorical attributes. This can be achieved via
the choice of the similarity measure $D$.

**Continuous Attribute.** The simplest case of a
continuous attribute is $m = 1$, where $y = g_\phi(b)$ is
a regression score, such as a sentiment between $-1$
(negative) and $1$ (positive). In this case we can take
$D$ to be a standard similarity measure on $\mathbb{R}$, such as
the inverse of $L_1$ or $L_2$ metrics, namely, $D(y, a) = \frac{|y - a|}{1}$ or $D(y, a) = \frac{2}{1} \cdot |y - a|^2$, where $a$ is the
target attribute score.

**Categorical Attribute.** For categorical at-
tributes with $m > 1$, such as emotion classes (e.g.,
joy, anger, fear and surprise), $g_\phi(b)$ produces a
vector of logits $y \in \mathbb{R}^m$. In this case $a$ is a one-hot
encoding of the target class $c \in \{1, \ldots, m\}$, and
so we can take $D$ to be negative cross-entropy,

$$D(y, a) = \log \left( \frac{\exp(y_c)}{\sum_{i=1}^{m} \exp(y_i)} \right) \quad (5)$$

**Multiple Attributes.** In the case that we want
to control the generated text according to multiple
attributes, for example, joy and surprise, we can re-
frame the problem as a multi-label prediction prob-
lem. Given a classifier $g_\phi$ that produces a vector
of independent logits $y \in \mathbb{R}^m$, and a target binary
vector $a \in \{0, 1\}^m$ such that $a_i = 1$ ($1 \leq i \leq m$)
for the desired attributes, we can take $D$ to be the
average of negative binary cross-entropy across the
attributes,

$$D(y, a) = \frac{1}{m} \sum_{i=1}^{m} a_i \log \sigma(y_i)$$

$$+ (1 - a_i) \log(1 - \sigma(y_i)) \quad (6)$$

where $\sigma(\cdot)$ is the sigmoid function.

### 4 Evaluation

We conduct several experiments in order to evalu-
ate BEAMR against SOTA controllable generation
approaches, in various applications. We focus on
(1) a sentiment steering task, whereby we generate
positive or negative continuations to a variety of
prompts (including positive, negative and neutral
prompts), and (2) a machine translation formality
task, whereby input sentences are translated to En-
lish and the translations are adjusted in order to
improve the formality of the text, whilst maintain-
ing the original meaning. We detail the relevant
datasets, baselines and metrics for each experiment.
We also conduct an analysis of hyperparameter se-
lection for both tasks.

#### 4.1 Sentiment Steering

We focus on the task of controlling the sentiment
(positive or negative) of generated text, given a
short prompt as input. For this experiment, we
closely follow the experimental setup outlined in
Liu et al. (2021). We evaluate two variants of
BEAMR: (1) using the base GPT-2 large model and
(2) using the appropriate finetuned expert model
from DExperts (Liu et al., 2021).

#### 4.1.1 Datasets

We use the prompts dataset provided by Liu et al.
(2021), originally collected from OpenWebText
Corpus (OWT) (Gokaslan and Cohen, 2019). We
use the same selections of 250 positive, 250 nega-
tive and 500 neutral prompts from Liu et al. (2021)
as in their PPLM evaluation. For each prompt, we
generate 25 continuations and score them using the
default DistilBERT sentiment classifier.

#### 4.1.2 Baselines

We consider the same baselines as outlined in
Liu et al. (2021). GPT-2 (Radford et al., 2019) is
used without any steering towards a particular sen-
timent. PPLM (Dathathri et al., 2020) is used toget-
er with a sentiment classifier trained on SST-
5 (Socher et al., 2013). CTRL (Keskar et al., 2019)
is used by providing “Reviews” as the control code
combined with a rating of 1.0 for negative steering and 5.0 for positive steering. CTRL’s original training included examples from Amazon Reviews. GeDi (Krause et al., 2020) is used with the original sentiment-conditioned LMs, originally trained on IMDb movie reviews. DExperts using both positive and negative expert LMs is used. We present the results from the large version of DExperts.

4.1.3 Metrics

In our evaluation, we focus on several key metrics: steering ability, fluency, diversity and relevance.

Automated Evaluation. We use the DistilBERT sentiment classifier to evaluate the steering ability by computing the proportion of continuations for each type of prompt that succeed in generating the desired sentiment. We evaluate the fluency of the generations by computing the average perplexity under a base GPT2-XL model. We evaluate the diversity by computing the number of unique n-grams (Dist-1, 2 and 3 scores) (Li et al., 2016) across the generations of each prompt.

Human Evaluation. Although automated evaluation is easy to perform, it may not accurately reflect human judgments, especially for fluency and relevance metrics (Hashimoto et al., 2019; Liu et al., 2017). To that end, we design a human evaluation study to evaluate steering ability, fluency and relevance. We separately evaluate positive and negative steering. We randomly sample 10 neutral and 10 positive/negative prompts for each experiment. For each pair of models for comparison (i.e. BEAMR paired with another baseline, such as GPT-2, CTRL, DExperts, etc.), we sample 3 generations per model. We conduct human evaluations on the Amazon Mechanical Turk (MTurk) platform, with 5 MTurk workers answering 3 questions about each pair of generations:

1. Which generation is more positive (resp. negative)?
2. Which generation is more fluent?
3. Which generation is more relevant to the prompt?

For each question workers may choose one of the models in the pair, or report that both models equally exhibit the characteristic in question. We compute 95% simultaneous confidence intervals (Goodman, 1965) for all three multinomial proportions for each pair of models and each question. We also perform a Z-test on the difference in proportions between the models in each pair.

4.1.4 Results

Automated Evaluation. Tables 1a and 1b show the results of the sentiment-based steering task for positive and negative steering, respectively. BEAMR scores in the top 2 models in terms of steering ability in all but one experiment, and outperforms DEx-
Table 1: Results of sentiment steering experiment. Given a neutral, negative or positive prompt, the models are tasked with producing positive or negative generations. **% Positive Sentiment** is computed as the average percentage of positive generations out of 25 total generations for each prompt. **Perplexity** is the average conditional perplexity of generations given the prompt, using a GPT2-XL model. **Diversity** is measured using the average number of distinct uni/bi/tri-grams in the generations for each prompt. Top 2 results are bolded.

(a) Positive Steering

<table>
<thead>
<tr>
<th>Model</th>
<th>% Positive Sentiment ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Neutral Prompts</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>GPT-2</td>
<td>50.03</td>
</tr>
<tr>
<td>PPLM</td>
<td>52.69</td>
</tr>
<tr>
<td>CTRL</td>
<td>60.77</td>
</tr>
<tr>
<td>GeDi</td>
<td>85.61</td>
</tr>
<tr>
<td>DExperts</td>
<td>94.79</td>
</tr>
<tr>
<td>BeamR</td>
<td>95.26</td>
</tr>
<tr>
<td>BeamR + Positive Expert</td>
<td>98.87</td>
</tr>
</tbody>
</table>

(b) Negative Steering

<table>
<thead>
<tr>
<th>Model</th>
<th>% Positive Sentiment ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Neutral Prompts</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>GPT-2</td>
<td>50.03</td>
</tr>
<tr>
<td>PPLM</td>
<td>39.05</td>
</tr>
<tr>
<td>CTRL</td>
<td>37.94</td>
</tr>
<tr>
<td>GeDi</td>
<td>9.06</td>
</tr>
<tr>
<td>DExperts</td>
<td><strong>3.27</strong></td>
</tr>
<tr>
<td>BeamR</td>
<td>5.86</td>
</tr>
<tr>
<td>BeamR + Negative Expert</td>
<td>1.99</td>
</tr>
</tbody>
</table>

We conducted additional experiments to quantify the effect of diversity, suggesting that it is able to produce varied generations that on the whole achieve the correct sentiment. Overall, these results highlight that BeamR can achieve a good balance between generating the correct sentiment and producing fluent text.

**Human Evaluation.** Figure 2 presents the results of human evaluation on the sentiment steering task. We see that BeamR significantly outperforms PPLM, GeDi and DExperts in fluency for negative steering, and otherwise performs on par with other models. BeamR significantly outperforms PPLM, GPT-2 and CTRL in both negative and positive steering ability. On the other hand, GeDi and DExperts outperform BeamR in steering ability, particularly in the negative steering experiment, which may support our earlier observations. BeamR performs on par with other models in terms of relevance.

**Effects of Hyperparameters.** We conducted additional experiments to quantify the effect of diversity in producing positive generations for neutral prompts. Noticeably, BeamR struggles to achieve the steering ability of DExperts when tasked to produce negative generations for positive prompts. This may be explained by the fact that DExperts better incorporates negative tokens into its generation via its negative expert, whereas BeamR is less likely to sample negative tokens from the base generation LM. In order to confirm this intuition, we also present results from BeamR using the negative and positive experts as the generation model. We see that combining BeamR with an expert model finetuned on the appropriate sentiment greatly improves performance and outperforms DExperts in both types of steering.

We also see that BeamR outperforms all other models in terms of perplexity. The low perplexity of BeamR compared to other methods may be explained by the fact that it utilizes beam search and reweighs candidate sequences of tokens, rather than reweighing individual tokens. Previous work (Holtzman et al., 2020) has shown that beam search leads to lower perplexity, although it tends to degenerate to repetition. BeamR avoids repetition by performing separate iterations of beam search with shorter candidate lengths and introducing additional variability by utilizing a diversity measure (Vijayakumar et al., 2018) and sampling from the candidate distribution. It is important to note that combining BeamR with a finetuned expert model increases the perplexity of the generations, likely due to a shift in the language distribution between the finetuned expert model and the base GPT-2 model. BeamR also performs competitively in terms of diversity, suggesting that it is able to produce varied generations that on the whole achieve the correct sentiment.
Which generation is more relevant to the prompt?

Which generation is more fluent?

ns: $p \leq 1.00$

*: $1 \times 10^{-2} < p \leq 5 \times 10^{-2}$

**: $1 \times 10^{-3} < p \leq 1 \times 10^{-2}$

***: $1 \times 10^{-4} < p \leq 1 \times 10^{-3}$

****: $p \leq 1 \times 10^{-4}$

Figure 2: Results of human evaluations in sentiment steering experiment. For clarity, responses from options ‘Equally positive/negative/fluent/relevant’ are not shown. 95% simultaneous confidence intervals for multinomial proportion estimates are shown in black. Significance results from $Z$-test of the difference between multinomial proportions are shown at the edges of the plot, with corresponding legend below plot.

The scaling hyperparameter $\gamma$ and beam length $T$ on both positive and negative steering, in terms of steering ability and fluency. Figure 3 in the Appendix Section A.3.1 presents the plots of % Positive Generations vs. Perplexity for varying settings of $\gamma$ and $T$. As we might expect, increasing $\gamma$ allows BEAMR to reach the desired sentiment in a higher proportion of generations. Moreover, increasing the beam length $T$ leads to a lower perplexity, signifying more fluent generations.

4.2 Machine Translation Formality

In this set of experiments we focus on the task of controlling the formality of English text that has been translated from Spanish. Unlike the sentiment steering task in Section 4.1 where BEAMR was applied to a causal language model, this involves applying BEAMR to a seq2seq translation model, thus further exhibiting the generalizability of our method. We follow the experimental setup outlined in Yang and Klein (2021).

4.2.1 Datasets

We use the Fisher and CALLHOME corpus (Post et al., 2013) of Spanish and English transcribed conversations, using the Spanish sentences as input to the Marian Spanish-to-English machine translation model (Junczys-Dowmunt et al., 2018). We leverage the pretrained formality classifier provided by Yang and Klein (2021) as the attribute model for BEAMR. The classifier was trained on the Entertainment/Music portion of the GYAFc formality corpus (Rao and Tetreault, 2018). For this experiment, BEAMR uses the Marian model as the generative LM ($f_\theta$) and the pretrained FUDGE classifier as the attribute model ($g_\phi$).

4.2.2 Baselines

We consider the same baselines as in Yang and Klein (2021). MarianMT base model is used to generate translation, without any steering towards more formal text. T5 style transfer model (Rafail et al., 2020) is finetuned on the GYAFc corpus (Entertainment/Music portion) and applied post-hoc to the output of MarianMT translations. FUDGE classifier is used to guide the translations of MarianMT in a token-by-token manner.

4.2.3 Metrics

For evaluation, we consider two important criteria: translation accuracy and formality. We evaluate the translation accuracy by computing the BLEU score between the generations and the gold-standard translations provided in the Fisher/CALLHOME corpus (Post et al., 2013). We evaluate the formality using a pretrained formality classifier provided by Yang and Klein (2021) that has been trained on the Family/Relationships portion of GYAFc (Rao and Tetreault, 2018).

4.2.4 Results

Table 2 presents the results of the translation formality experiment. Notably, combining an unfinetuned Marian model and FUDGE with BEAMR, we achieve a higher BLEU score and a higher formality score than FUDGE, signifying more formal translations which are closer to the gold standard. Similarly, with a Marian model that was finetuned...
on the Fisher training set, we see that BEAMR can reach FUDGE’s BLEU score while also achieving a higher formality score.

<table>
<thead>
<tr>
<th>Model</th>
<th>Unfinetuned BLEU ↑</th>
<th>Form. ↑</th>
<th>Finetuned BLEU ↑</th>
<th>Form. ↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>Marian</td>
<td>16.98</td>
<td>0.45</td>
<td>22.03</td>
<td>0.41</td>
</tr>
<tr>
<td>+ T5</td>
<td>7.87</td>
<td>0.96</td>
<td>9.63</td>
<td>0.97</td>
</tr>
<tr>
<td>+ FUDGE</td>
<td>17.96</td>
<td>0.51</td>
<td>22.18</td>
<td>0.48</td>
</tr>
<tr>
<td>+ BeamR</td>
<td>18.47</td>
<td>0.63</td>
<td>21.74</td>
<td>0.63</td>
</tr>
</tbody>
</table>

Table 2: Results for the machine translation formality task. Given a sentence in Spanish, the models are tasked to produce a form English translation. BLEU measures the accuracy of translation via n-gram precision. Form. is the average formality score provided by the FUDGE classifier trained on the Family/Relationships portion of the GYAF dataset. Top results are bolded.

**Effects of Hyperparameters.** We conducted additional experiments to understand the effects of varying scaling hyperparameter $\gamma$ and beam length $T$ on the quality and formality of translations. Figures 4a and 4b in the Appendix Section A.3.2 present BLEU vs. formality score with varying $T$ and $\gamma$, respectively.

We can see that varying $\gamma$ allows for a trade-off between formality and translation accuracy. Namely, increasing $\gamma$ improves formality score but decreases BLEU score. We also see trends in formality and translation accuracy when changing $T$. For shorter beam lengths, BEAMR makes locally optimal choices for formality, but suffers a significant decrease in BLEU score when considering the full translation. This hints at a similar behaviour as observed in sentiment steering (Section 4.1), namely that leveraging beam search can improve the quality of generation while leaving ample room for control.

**5 Related Work**

Recent methods in controllable text generation (Weng, 2021) may be categorized under decoding methods and tuning methods. Roughly speaking, decoding methods apply controllable characteristics only at the output distribution of a LM, while tuning methods additionally attempt to encode controllable characteristics into the generative LM itself, by tuning either some or all of its parameters.

**5.1 Decoding methods**

Decoding methods are applied to produce text output from an autoregressive generative language model. We first outline several general approaches to decoding from language models.

Typical decoding is done by sampling from the next token distribution, or picking the most likely token. However, these approaches lead to undesired output (Holtzman et al., 2020): sampling may lead to the model producing gibberish while greedy decoding often leads to repetitions. Several basic approaches have been proposed to tackle these issues, including top-k sampling (Fan et al., 2018), top-p sampling (Holtzman et al., 2020) and repetition-penalized sampling (Keskar et al., 2019). An alternative approach is the beam search algorithm (Graves, 2012) which maintains a collection of k best sequences at each time step. In order to promote more diversity in the generated candidates, Vijayakumar et al. (2018) proposed a diverse beam search algorithm, which splits the candidates into separate groups and enforces a dissimilarity metric across the groups.

Several approaches have been explored to guide decoding according to a particular attribute. Ghazvininejad et al. (2017) modify the beam search algorithm to incorporate weighted feature functions during each step. They use several manually designed feature functions including custom wordlists, repetition penalty, and alliteration metrics for the problem of poetry generation. More recently Liu et al. (2021); Yang and Klein (2021) have proposed leveraging multiple language models to re-rank hypotheses according to a particular attribute. Liu et al. (2021) achieves this by fine-tuning generative language models on appropriate subsets of a dataset (e.g., training experts on toxic and non-toxic subsets of a dataset) and combining token-level distributions from the original language model and expert models. The downside of this approach is that it requires annotated data and additional training of the expert models, which may not be available for resource-constrained scenarios and domains. Yang and Klein (2021) propose to use a binary classifier trained for a particular task, to reweigh the token-level distribution produced by a generative LM. They highlight the flexibility of their approach in a variety of experiments, including couplet generation and topic control. Our method differs from and improves on FUDGE in several key ways, by:

- Applying reweighing to beam-level decoding thereby avoiding typical disfluency and repetition issues from token-level decoding men-
tioned in Holtzman et al. (2020)

- Allowing for the choice of a custom similarity measure \( D \) appropriate for the discriminator (e.g., regressor, classifier), thereby offering precise control of the desired target attribute value

- Removing the requirement of shared tokenization between the generative LM and the discriminator and instead reweighing natural language hypotheses, thereby improving generalizability to different LMs

5.2 Tuning methods

The majority of recent work on controllable text generation has focused on fine-tuning some or all of the parameters of a generative language model. Keskar et al. (2019) train a transformer model (CTRL) to learn a conditional distribution over the data. By prepending different control codes (for instance, "Wikipedia" or "Reviews") to raw text from different sources (Wikipedia, or Amazon Reviews, respectively), it learns to associate certain types of text with the control codes. At inference time, CTRL interprets the first token in the prompt to be a control code, and can thus generate text in the corresponding style.

Dathathri et al. (2020) proposed Plug-and-Play Language Models (PPLM), a method to steer a subset of the parameters of a generative language model according to a lightweight auxiliary attribute model. They achieve this via backpropagation of the attribute model loss gradient into the past attention key-value pairs of a transformer-based language model. They experiment with simple attribute models consisting of a bag-of-words to encourage the LM to use words from the bag, as well as simple classifiers (e.g., sentiment) trained on top of the generative LM representations.

Zeldes et al. (2020) briefly describe a method to shift the output distribution of a generative language model using an auxiliary model. They combine the logits of both models and train them in tandem to maximize the likelihood of a certain attribute.

Our method is inspired by PPLM and also resembles a decoding method (Zeldes et al., 2020), whereby we similarly propose to control the output distribution of the generative language model. However, unlike those methods, we do not require that the generative and auxiliary models be trained together. In fact, our method is flexible and robust to the choice of the generative and auxiliary attribute models and can leverage pre-trained models, avoiding the need to re-train one or both of the models.

6 Conclusion

We present a simple and modular decoding-based approach to controllable generation, BEAMR. BEAMR combines a generative LM with an attribute discriminator and leverages beam search decoding in order to steer generated text to the desired target attribute. We show the results of BEAMR in two diverse tasks: sentiment-based steering, and machine translation formality steering. Our results from automated evaluations show that BEAMR outperforms strong baselines for both tasks, and human evaluations for sentiment steering further support this.

Noticeably, BEAMR struggles with negative sentiment steering, especially when compared to GeDi and DExperts. We hypothesize this may be due to GeDi and DExperts having direct access to class-conditioned distributions in their generation. Namely, GeDi trains a class-conditioned LM using control codes and anti-control codes (including \(<\text{negative}>\)) and DExperts trains separate expert and anti-expert LMs on subsets of the data (including an anti-expert trained on negative-only text). Future work on BEAMR may incorporate additional sources of language and attribute information to address this shortcoming.

BEAMR offers a great deal of flexibility by allowing us to plug different and independent generative LMs and attribute discriminators (with potentially different tokenization schemes). Moreover, BEAMR generalizes beyond classification tasks to any type of discriminator by appropriately selecting a similarity measure. Leveraging beam search for text decoding from a LM, BEAMR’s generations avoid some of the typical problems with token-based decoding (such as repetition or disfluencies). Our work highlights that strong controllable text generation can be achieved by mixing together large pre-trained generative and discriminative models, with a flexible backbone offered by BEAMR, without sacrificing fluency.

7 Ethics of Controllable Text Generation

Usage of large language model for text generation can pose various risks, including producing harm-
ful content or misinformation (Sheng et al., 2020; Gehman et al., 2020; Wallace et al., 2021). Controllable text generation may create additional risks if used maliciously. However, it can also help researchers and practitioners avoid the biases learned by large language models and reduce the aforementioned risks (Liu et al., 2021; Dathathri et al., 2020). Therefore, we believe advancing research in controllable text generation is valuable in order to understand the pitfalls of large language models and develop strong measures to prevent harmful content generation.

Human evaluation experiments were conducted on the Amazon Mechanical Turk platform, and evaluators were compensated above the federal minimum wage in the country of residence (United States).

References


Matt Post, Gaurav Kumar, Adam Lopez, Damianos Karakos, Chris Callison-Burch, and Sanjeev Khudanpur. 2013. Improved speech-to-text translation with the Fisher and Callhome Spanish-English speech


A Appendix

A.1 Implementation Details
All experiments were conducted on a single NVidia Tesla T4 GPU. Transformers package (Wolf et al., 2020) version 4.8 was used to implement all algorithms and experiments. Table 3 presents average amount of time to run each experiment.

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Avg. Time (in minutes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentiment Steering</td>
<td>1.19 per batch of 8</td>
</tr>
<tr>
<td>Machine Translation Formality</td>
<td>0.0033 per 1 generation</td>
</tr>
<tr>
<td>Machine Translation Formality (Training)</td>
<td>1.22 per epoch (20 epochs)</td>
</tr>
</tbody>
</table>

1 Corresponds to training of the FUDGE classifier on the Entertainment/Music portion of the GYAFC formality corpus (Rao and Tetreault, 2018)

Table 3: Average time taken (per example or per epoch) to run each experiment in Section 4.

A.2 Hyperparameters

A.2.1 Sentiment Steering
Table 4 presents the full hyperparameter configurations for the sentiment steering task in Section 4.1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generation Model</td>
<td>GPT2-Large (774M params.)</td>
</tr>
<tr>
<td>Discriminator Model</td>
<td>DistilBERT (66M params.)</td>
</tr>
<tr>
<td>Generation Length</td>
<td>20</td>
</tr>
<tr>
<td>Temperature</td>
<td>1.0</td>
</tr>
<tr>
<td>Diversity Penalty</td>
<td>10.0</td>
</tr>
<tr>
<td>Scaling ($\gamma$)</td>
<td>${1, 2, 3}$</td>
</tr>
<tr>
<td>Beam Length ($T$)</td>
<td>${1, 3, 5, 7, 10}$</td>
</tr>
<tr>
<td>Number of Candidates ($K$)</td>
<td>5</td>
</tr>
<tr>
<td>Beam Length Penalty</td>
<td>1.0</td>
</tr>
<tr>
<td>Batch Size</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 4: Models and hyperparameters used for sentiment steering experiments with BEAMR. Best-found hyperparameters are bolded, where applicable.

A.2.2 Machine Translation Formality
Table 5 presents the full hyperparameter configurations for the machine translation formality task in Section 4.2.

A.3 Additional Experiments
This section contains additional results for the experiments in Sections 4.1 and 4.2.

A.3.1 Sentiment Steering Hyperparameters
Figure 3 shows the results of hyperparameter experiments from 4.1.

A.3.2 Machine Translation Formality Hyperparameters
Figures 4a and 4b show the results of beam length ($T$) and scaling hyperparameter ($\gamma$) experiments (resp.) from 4.2.

A.3.3 Visualization of Reweighing
In order to better understand the effects of the reweighing step in Eqn. (4), we selected a prompt from the sentiment steering task, and ran BEAMR to get 15 generations for each set of hyperparameters ($\gamma, T \in \{0.1, 0.3, 1, 3\} \times \{3, 5, 7, 15\}$).

Figure 5 shows the average candidate, attribute and reweighed distributions across 15 generations, from a single step in the BeamR algorithm. We see that for small values of $\gamma < 1$, the reweighed distributions closely resemble the original candidate distributions while the attribute distribution is almost flat. When $\gamma \geq 1$, we see the reweighed distributions take the shape of the attribute distributions, signifying a stronger effect of the attribute score. We also see some effect of the beam length hyperparameter on the reweighing. In particular, for small $T$, the reweighed distributions closely match the attribute distributions, however as $T$ increases, there is a larger gap between the distributions. This gap is offset by increasing the value of $\gamma$.

A.4 Qualitative Examples

A.4.1 Sentiment Steering
Tables 6a and 6b show some qualitative examples from positive and negative steering (resp.) comparing BEAMR and baseline models.

A.4.2 Machine Translation Formality
Table 7 shows some qualitative examples comparing BEAMR and FUDGE with reference translations (Salesky et al., 2019).
A.5 Human Evaluation

Figure 6 shows an example screenshot of the human evaluation instructions from MTurk.

A.6 Dataset Details

Table 8 presents the size of datasets used in our experiments in Section 4.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Label</th>
<th>Number of examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentiment Prompts</td>
<td>Positive</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td>Neutral</td>
<td>500</td>
</tr>
<tr>
<td></td>
<td>Negative</td>
<td>250</td>
</tr>
<tr>
<td>GYAFC Inference</td>
<td>Formal</td>
<td>50967/1019/1000</td>
</tr>
<tr>
<td></td>
<td>Informal</td>
<td>50967/1332/1000</td>
</tr>
<tr>
<td>GYAFC Evaluation</td>
<td>Formal</td>
<td>51595/1082/1000</td>
</tr>
<tr>
<td></td>
<td>Informal</td>
<td>51595/1416/1000</td>
</tr>
</tbody>
</table>

Table 8: Dataset sizes used for experiments in Section 4.
Figure 5: Comparison of differences between candidate, attribute and reweighed distributions from a single iteration of BEAMR for different combinations of scaling and beam length hyperparameters. Each point corresponds to the average density across 15 generations for a selected prompt.
Prompt: Until then, we should start

<table>
<thead>
<tr>
<th>Model</th>
<th>Generation</th>
<th>(a) Positive Steering</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPT-2</td>
<td>Until then, we should start collaborating. I’d love it if you could set up a foundation with the common interests of digital creators. If you want to know what makes one better than another,</td>
<td></td>
</tr>
<tr>
<td>PPLM</td>
<td>Until then, we should start thinking of this awkwardness as largely just an blessings-and-worries life being shaped out.</td>
<td></td>
</tr>
<tr>
<td>CTRL</td>
<td>Until then, we should start by looking at the overall performance of both systems. If you want to know what makes one better than another,</td>
<td></td>
</tr>
<tr>
<td>GeDi</td>
<td>Until then, we should start by enjoying each other very much!</td>
<td></td>
</tr>
<tr>
<td>DEExperts</td>
<td>Until then, we should start by appreciating the powerful sights and sounds of Japan’s ancient forests and take pleasure in sampling traditional food</td>
<td></td>
</tr>
<tr>
<td>BeamR</td>
<td>Until then, we should start the new year with a simple message of gratitude for the world we have. As we open our hearts</td>
<td></td>
</tr>
<tr>
<td>BeamR + Positive Expert</td>
<td>Until then, we should start by acknowledging that the remarkable success of former boxing champion Mike Tyson’s trilogy with Evander Holyfield</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Example generations from sentiment steering task, comparing BEAMR with baseline methods.

| Model            | Generation                                                                 |
|------------------|-----------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| GPT-2            | Until then, we should start collaborating. I’d love it if you could set up a foundation with the common interests of digital creators. If you want to know what makes one better than another, |
| PPLM             | Until then, we should start thinking of this awkwardness as largely just an blessings-and-worries life being shaped out. |
| CTRL             | Until then, we should start by looking at the overall performance of both systems. If you want to know what makes one better than another, |
| GeDi             | Until then, we should start by enjoying each other very much! |
| DEExperts        | Until then, we should start by appreciating the powerful sights and sounds of Japan’s ancient forests and take pleasure in sampling traditional food |
| BeamR            | Until then, we should start the new year with a simple message of gratitude for the world we have. As we open our hearts |
| BeamR + Negative Expert | Until then, we should start by thinking of this as just another lame, uninspired, and under-the-radar procedural, |

(b) Negative Steering

Table 7: Example translations from machine translation formality task, comparing FUDGE and BEAMR with reference translations (Salesky et al., 2019).
Two text generation models were provided the following prompt:

**Prompt:** The negative income tax is much more politically

Carefully read each of the generations below.

For each question, choose the generation that best reflects the desired property. If both generations equally reflect the property, please select the third option.

**Generation A:** The negative income tax is much more politically feasible than VAT,” said Mr. Wijhuis, who co-founded the Fijian

**Generation B:** The negative income tax is much more politically lucrative than the positive income tax. But there is a certain logic to it, if you’re thinking.

Which generation is more positive?

- A
- B
- Equally positive

Which generation is more fluent?

- A
- B
- Equally fluent

Which generation is more relevant to the prompt?

- A
- B
- Equally relevant

Submit

Figure 6: Example of human evaluation instructions from MTurk experiments. For negative steering, the first question is phrased: “Which generation is more negative?”