Importance of a Search Strategy in Neural Dialogue Modelling

Ilya Kulikov
New York University
kulikov@cs.nyu.edu

Alexander H. Miller
Facebook AI Research

Kyunghyun Cho
New York University
Facebook AI Research
CIFAR Azrieli Global Scholar

Jason Weston
Facebook AI Research
New York University

Abstract

Search strategies for generating a response from a neural dialogue model have received relatively little attention compared to improving network architectures and learning algorithms in recent years. In this paper, we consider a standard neural dialogue model based on recurrent networks with an attention mechanism, and focus on evaluating the impact of the search strategy. We compare four search strategies: greedy search, beam search, iterative beam search and iterative beam search followed by selection scoring. We evaluate these strategies using human evaluation of full conversations and compare them using automatic metrics including log-probabilities, scores and diversity metrics. We observe a significant gap between greedy search and the proposed iterative beam search augmented with selection scoring, demonstrating the importance of the search algorithm in neural dialogue generation.

1 Introduction

There are three high-level steps to building a neural autoregressive sequence model for dialog modelling, of the kind inspired by the successful work of Vinyals and Le (2015). First, decide on a specific network architecture which will consume both previous utterances as well as any extra information such as speaker identifiers. Second, select a suitable learning strategy. Finally, decide on your search algorithm, as neural autoregressive sequence models do not admit a tractable, exact approach for generating the most likely response.

Recent research in neural dialogue modelling has often focused on the first two aspects. A number of variants of sequence-to-sequence models (Sutskever et al., 2014; Cho et al., 2014; Kalchbrenner and Blunsom, 2013) have been proposed for dialogue modelling in recent years, including hierarchical models (Serban et al., 2016) and transformers (Mazaré et al., 2018; Yang et al., 2018). These advances in network architectures have often been accompanied by advanced learning algorithms. Serban et al. (2017) introduce latent variables to their earlier hierarchical model and train it to maximize the variational lower bound, similar to Zhao et al. (2017) who propose to build a neural dialogue model as a conditional variational autoencoder. Xu et al. (2017) and Li et al. (2017b) train a neural dialogue model as conditional generative adversarial networks (Mirza and Osindero, 2014). These two learning algorithms, varitional lower-bound maximization and adversarial learning, have been combined into a single model by Shen et al. (2018), which has been followed by Gu et al. (2018).

Despite abundant endeavors on modelling and learning, search has received only a little attention. Most of the work on search has focused on training an additional neural network that provides a supplementary score to guide either greedy or beam search; we refer to this as the selection strategy. Li et al. (2015) propose a maximum mutual information criterion for decoding using a reverse model. This has been extended by Li et al. (2017a), where an extra neural network is trained to predict an arbitrary reward given a partial hypothesis and used during decoding. Similarly, Zemlyanskiy and Sha (2018) train a neural network that predicts the other participant's personality given a partial conversation and use its predictability as an auxiliary score for re-ranking a set of candidate responses. None of these approaches study how the choice of the underlying search algorithm, rather than its scoring function, affects the quality of the neural dialogue model.

In this paper, we investigate the effects of varying search and selection strategies on the quality of generated dialogue utterances. We start with a straightforward modeling approach using an attention-based sequence-to-sequence model (Bahdanau et al., 2014) trained on the recently-released PersonaChat dataset (Zhang et al., 2018). We evaluate three search algorithms: greedy search, beam search and iterative beam search, the last of which is designed by us based on earlier works by Batra et al. (2012). These algorithms are qualitatively different from each other in the size of subspace over which they search for the best response. Furthermore, we investigate the effect of learning an additional scoring function to select a response from those returned by the search function, observing additional improvements by thus separating search and selection. All of these alternatives are compared using human evaluation of multi-turn conversations.

We observed high variance in the human evaluation distribution due to the bias in individual workers and propose an algorithm to reduce it using Bayesian inference, which aims to approximate the posterior distribution of the scores using a latent worker bias variable.

Our experiments reveal that a significant improvement can be achieved by simply choosing a better search strategy, with the best strategy being the combination of the proposed iterative beam search and a sequence selection function. Human annotators favoured conversations with the same model when the best search strategy was used, and the diversity of generated responses, measured in terms of the numbers of distinct bi-/trigrams within each conversation, was higher. These observations strongly suggest the importance of search in neural dialogue modelling, and that any comparison of neural dialogue models must be done after selecting the best search strategy for each model.

We share trained model, code and human evaluation transcripts with readers for any further analysis. [1]

2 Neural dialogue modelling

Since (Vinyals and Le, 2015), a neural autoregressive sequence model based on sequence-to-sequence models (Sutskever et al., 2014; Cho et al., 2014) has become one of the most widely studied approaches to dialogue modelling (see, e.g., Serban et al., 2016, 2017; Zhao et al., 2017; Xu et al., 2017; Li et al., 2016, 2017a,b; Zemlyanskiy and Sha, 2018; Zhang et al., 2018; Miller et al., 2017; Shen et al., 2018; Gu et al., 2018). In this approach, a neural sequence model is used to model a conditional distribution over responses given a context which consists of previous utterances by both itself and a partner in the conversation as well as any other information such as features of the speaker.

2.1 Neural autoregressive sequence modeling

A neural autoregressive sequence model learns the conditional distribution over all possible responses given the context X, and the conditional probability of a response y is factorized into a product of next-token probabilities:

$$p(y|X) = \prod_{t=1}^{T} p(y_t|y_{<t}, X). \tag{1}$$

Each conditional distribution on the r.h.s above is then modelled by a neural network, and popular choices include recurrent neural networks (Mikolov et al., 2010; Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014), convolutional networks (Dauphin et al., 2016; Gehring et al., 2017) and self-attention (Sukhbaatar et al., 2015; Vaswani et al., 2017). Our goal is to assess the impact of the search algorithm, so we fix the model to a standard neural autoregressive sequence model: see the appendix for a full list of hyperparameter values.

Learning: Maximum Likelihood Each example in a training set D consists of auxiliary information or context U (such as a persona profile or external knowledge context) and a sequence of utterances, each of which is marked with a speaker tag, i.e.,

$$C = (U, (Y_1^a, Y_1^b, \dots, Y_L^a, Y_L^b) \in D,$$

where Y_l^s is the utterance from the l-th turn by a speaker s. The conditional log-probability assigned to this example given by a neural sequence model is then written as

$$\log p(C) = \sum_{s \in \{a,b\}} \sum_{l=1}^{L} \log p(Y_l^s | Y_{<l}^s, Y_{\leq l}^{\bar{s}}, U), \tag{2}$$

where $\bar{s} = a$ if $s = b$ and otherwise $\bar{s} = b$. Each term inside the summation above is mapped to the autoregressive distribution in Eq. (1) by considering $\{Y_{<l}^s, Y_{<l}^{\bar{s}}, U\}$ as X.

[1] https://beamdream.github.io/

Learning maximizes the log-probabilities of all the conversations in the training set:

$$L = \frac{1}{|D|} \sum_{C \in D} \log p(C). \tag{3}$$

This is done often using stochastic gradient descent with backpropagation (Rumelhart et al., 1985).

2.2 Inference/generation

In this paper, we generate a response to the current state of the conversation (but do not attempt to plan ahead to future exchanges), maximizing

$$\begin{aligned}&\log p(Y|Y^s_{<l}, Y^{\bar{s}}_{<l}, U)\\ &= \sum_{t=1}^{T} \log p(y_t|y_{<t}, Y^s_{<l}, Y^{\bar{s}}_{<l}, U).\end{aligned}$$

Unfortunately, it is intractable to solve this problem due to the exponentially-growing space of all possible responses w.r.t. the maximum length T. It is thus necessary to resort to approximate search algorithms. We describe here the two most widely used search algorithms for neural autoregressive sequence models.

Greedy search Greedy search has been the search algorithm of choice among the recent papers on neural dialogue modelling (Gu et al., 2018; Zhao et al., 2017; Xu et al., 2017; Weston et al., 2018; Zhang et al., 2018).

This algorithm moves from left to right selecting one token at a time, simply choosing the most likely token at the current time step:

$$\hat{y}_t = \arg\max_{v \in V} \log p(y_t = v|\hat{y}_{<t}, Y^s_{<l}, Y^{\bar{s}}_{<l}, U).$$

Greedy search has been found significantly suboptimal within the field of machine translation (see, e.g., Table 1 in Chen et al., 2018), where similar neural sequence models are frequently used.

Beam search Instead of maintaining a single hypothesis at a time, as in greedy search above, beam search maintains K hypothesis:

$$\mathcal{H}_t = \{(y^1_1, \ldots, y^1_t), \ldots, (y^K_1, \ldots, y^K_t)\} \tag{4}$$

Each hypothesis is expanded with all possible next tokens v to form candidate hypotheses, each of which is in the form of

$$\tilde{h}^i_v = (y^i_1, \ldots, y^i_t, v), \tag{5}$$

where $v \in V$. Each candidate is associated with its score:

$$s(\tilde{h}^i_v) = \sum_{t'=1}^{t} \log p(y^i_{t'}|y^i_{<t'}) + \log p(v|y^i_{\leq t}). \tag{6}$$

The new hypothesis set of K hypotheses is then constructed as

$$\mathcal{H}_{t+1} = \underset{i,v}{\text{arg-top-}k}\ s(\tilde{h}^i_v)$$

When all the hypotheses in the new hypothesis set have terminated, i.e., $y^i_{t+1} = \langle\text{eos}\rangle$ for all i, beam search terminates, and the hypothesis with the highest score (6) is returned.

One can increase the size of the subspace over which beam search searches for a response by simply increasing the size of the beam K. While beam search is currently the method of choice in many applications, it is known to suffer from the problem that most of the hypotheses discovered are near each other in the response space (Li et al., 2016, 2015). For tasks like dialogue which are much more open-ended than e.g. machine translation, this is particularly troublesome as many high quality responses may be missing in the beam.

Avoiding repeating n-grams Although this has not been reported in a formal publication in the context of neural dialogue modelling to our knowledge, OpenNMT-py (Klein et al., 2017) implements so-called n-gram blocking. In n-gram blocking, a hypothesis in a beam $\mathcal{H}_t$ is discarded if there is an n-gram that appears more than once within it. This feature is especially useful in dialogue modelling, since it is unlikely for any n-gram to repeat within a single utterance.

3 Uncovering hidden responses

We now propose an improved search strategy, which consists of a more diverse *search* exposing more high quality responses by considering an iterative beam search, followed by *selection* from that set with a learnt scoring function.

Search To address the locality issue in beam search, we propose an iterative beam search to radically increase the size of search space without introducing much computational overhead, inspired by earlier work on diverse beam search (Vijayakumar et al., 2018; Batra et al., 2012; Li et al., 2016).

Selection Search strategies that rely on the conditional log-probability of the sequence for selection tend to prefer syntactically well-formed responses due to the word-based maximum likelihood training. We thus introduce a parameterized sequence selection scoring function which aims to select the best candidate among the given set of possible hypotheses.

3.1 Iterative beam search

The search space over which beam search has operated can be characterized by the union of all partial hypothesis sets $\mathcal{H}_t$ in Eq. (4):

$$\mathcal{S}_0 = \cup_{t=1}^{T} \mathcal{H}_t,$$

where we use the subscript 0 to indicate that beam search has been done without any other constraint. Re-running beam search with an increased beam width K would result in the search space that overlaps significantly with $\mathcal{S}_0$, and would not give us much of a benefit with respect to the increase in computation.

Instead, we keep the beam size K constant but run multiple iterations of beam search while ensuring that any previously explored space

$$\bar{\mathcal{S}}_{<l} = \cup_{l'=0}^{l-1} \mathcal{S}_{l'}$$

is *not* included in a subsequent iteration of beam search. This is done by setting the score of each candidate hypothesis $s(\tilde{h}^i_{t+1})$ in Eq. (6) to negative infinity, when this candidate is included in $\bar{\mathcal{S}}_{<l}$. We relax this inclusion criterion by using a non-binary similarity metric, and say that the candidate is included in $\bar{\mathcal{S}}_{<l}$, if

$$\min_{h \in \bar{\mathcal{S}}_{<l}} \Delta(\tilde{h}^i_{t+1}, h) < \epsilon, \tag{7}$$

where Δ is a string similarity measure, such as Hamming distance as used in this work, and ϵ is a similarity threshold.

This procedure ensures that a new partial hypothesis set of beam search in the l-th iteration does not overlap at all with any part of the search space explored earlier during the first $l-1$ iterations of beam search. By running this iteration multiple times, we end up with a set of top hypothesis from each iteration of beam search, from which the best one is selected according to for instance the log-probability assigned by the model.

Practical implementation A major issue with iterative beam search in its naive form is that it requires running beam search multiple times, when even a single run of beam search can be prohibitively slow in an interactive environment, such as in dialogue generation.

We address this computational issue by performing these many iterations of beam search in parallel simultaneously. At each time step in the search, we create sets of candidate hypotheses for all iterations in parallel, and go through these candidate sets in sequence from the $l = 0$-th iteration down to the last iteration, while eliminating those candidates that satisfy the criterion in Eq. (7). We justify this parallelized approach by defining the similarity measure Δ to be always larger than the threshold ϵ when the previous hypothesis $\tilde{h}^i_{t+1}$ is longer than h in Eq. (7).

3.2 Learning to select sequences

When training a sequence-to-sequence model by maximizing the log-likelihood in Eq. (3), each and every token in the response side is treated equally. This encourages the model to focus more on frequent tokens than less frequent ones, which consequently makes the model more syntax-oriented than semantics-oriented, as discussed earlier by Collobert et al. (2011). Although this behaviour is desirable when we use the model to generate a well-formed response, it is not necessarily desirable for *selecting* a semantically meaningful response given the context.

We thus propose to augment the underlying sequence-to-sequence model with a selection scoring function $R(y|x) = s \in \mathbb{R}$, where y is a response and R computes its score given the final decoder state of the underlying model. We train this scoring function using a pairwise ranking loss and a set of negative examples, following (Collobert et al., 2011):

$$L_R = \frac{1}{|Y^-|} \sum_{y^- \in Y^-} \max(0, m - R(y^*|x) + R(y^-|x)), \tag{8}$$

where Y^- is a set of negative responses and y^* is a ground-truth response, given the context x. We choose negative responses from the training set uniformly at random.

We thus train a sequence-to-sequence model by minimizing the weighted sum of the negative log-likelihood (3) and ranking loss (8):

$$L = \alpha L + \beta L_R. \tag{9}$$

We start with $\alpha = 1$ and $\beta = 0$ until the model converges and fine-tune it with α set to 0.1 and β set to 1, which is equivalent to pretraining the model with maximum likelihood only first and finetuning it with both near the end of learning.

4 Dialogue evaluation

Broadly there are two ways to evaluate a neural dialogue model. The first approach is to use a set of (often human generated) reference responses and compare a generated response against them (Serban et al., 2015; Liu et al., 2016). There are two sub-approaches; (1) measure the perplexity of reference responses using the neural dialogue model, and (2) compute a string match-based metric of a generated response against reference responses. Neither of these approaches however captures the effectiveness of a neural sequence model in conducting a full conversation, as during this evaluation the model responses are computed given a full reference context from the dataset, i.e. it does not see its own responses in the dialogue history, but gold responses instead. This constraint is necessary as a reference response is valid only when placed within a given context, and any deviation in the context from the collected context easily, if not always, invalidates it as a reference response.

We thus take the second approach, where a neural dialogue model has a full conversation with a human partner (or annotator) (Zhang et al., 2018; Zemlyanskiy and Sha, 2018; Weston et al., 2018). Unlike the first approach, it requires active human interaction, as a conversation almost always deviates from a previously collected conversation even with the same auxiliary information (U in Eq. (2)). This evaluation strategy reflects both how well a neural dialogue model generates a response given a correct context as well as how well it adapts to a dynamic conversation—the latter was not measured by the first strategy. In the rest of this section, we describe our approach to human evaluation of a full conversation and propose Bayesian calibration to address the annotator bias.

4.1 Human evaluation of a full conversation

We ask a human annotator to have a conversation with a randomly selected bot (characterized by its choice of search algorithm) for at least five turns. At the end of the conversation, we ask the annotator three sets of questions:[2]

1. Overall score ($\{1, 2, 3, 4, 5\}$)
2. Marking of each *good* utterance-pair ($\{0, 1\}$)
3. Marking of each *bad* utterance-pair ($\{0, 1\}$)

The first overall score allows us to draw a conclusion on which algorithm makes a better conversation overall. The latter two are collected in addition to the overall score to investigate the relationship between the overall impression and the quality of each utterance-pair.

4.2 Bayesian calibration

Although human evaluation is desirable, raw scores collected by human annotators are difficult to use directly due to annotator bias. Some annotators are more generous while others are quite harsh, leading the naive average score to have very high variance; for example, as recently reported in (Zhang et al., 2018; Zemlyanskiy and Sha, 2018). It is necessary to calibrate raw scores so as to remove these annotator biases, and we propose to use Bayesian inference here as a framework for removing such biases. We describe two instances of this framework.

1-5 star rating of a conversation We treat both the unobserved score M_i of each model we are comparing, in our case each search algorithm, and the unobserved bias B_j of each annotator as latent variables. The score of the i-th model follows the following distribution:

$$\mu_i \sim \mathcal{U}(1, 5), \text{ and } M_i \sim \mathcal{N}(\mu_i, 1^2),$$

where $\mathcal{U}$ and $\mathcal{N}$ are uniform and normal distributions. It states that *a priori* each model is likely to be uniformly good or bad. The annotator bias B_j then follows

$$B_j \sim \mathcal{N}(0, 1^2),$$

where we are stating that each annotator does not have any bias *a priori*.

Given the model score M_i and annotator bias B_j, the conditional distribution over an observed score S_{ij} given by the j-th annotator to the i-th model is then:

$$S_{ij} \sim \mathcal{N}(M_i + B_j, 1^2).$$

[2] We provide the detailed descriptions of the questions in the appendix.

Of course, due to the nature of human evaluation, only a few of S_{ij}'s are observed.

The goal of inference in this case is to infer the posterior mean and the variance:

$$\begin{aligned}&\mathbb{E}[M_i|\,\{S_{ij}|S_{ij}\in\mathcal{O}\}], \qquad (10)\\ &\mathbb{V}[M_i|\,\{S_{ij}|S_{ij}\in\mathcal{O}\}],\end{aligned}$$

where $\mathcal{O}$ is a set of observed scores.

Binary rating of an utterance When an annotator labels pairs of utterances from the conversation with a binary score $\{0,1\}$ (such as whether that pair was a "good" exchange), we need to further take into account the turn bias T_k:

$$T_k \sim \mathcal{N}(0,1^2).$$

Also, as we will use a Bernoulli distribution for each observed score, we modify the priors of the model scores and annotator biases:

$$M_i \sim \mathcal{N}(0,1^2) \text{ and } B_j \sim \mathcal{N}(0,1^2)$$

The distribution of an observed utterance-pair score S_{ijk} is then

$$S_{ijk} \sim \mathcal{B}(\text{sigmoid}(M_i + B_j + T_k)),$$

where $\mathcal{B}$ is a Bernoulli distribution.

The goal of inference is is to compute

$$\begin{aligned}&\mathbb{E}_{M_i|\{S_{ijk}|S_{ijk}\in\mathcal{O}\}}\left[\text{sigmoid}(M_i)\right], \qquad (11)\\ &\mathbb{V}_{M_i|\{S_{ijk}|S_{ijk}\in\mathcal{O}\}}\left[\text{sigmoid}(M_i)\right],\end{aligned}$$

which estimate the average number of positively labelled utterance-pairs given the i-th model and the uncertainty in this estimate, respectively.

Inference We use no-u-turn (NUTS) sampler (Hoffman and Gelman, 2014) for posterior inference. We use Pyro (Bingham et al., 2018) in both of the cases.

5 Experiment Settings

5.1 Data: Persona-Chat

We exclusively use PersonaChat, released recently by Zhang et al. (2018) and the main dataset for the Conversational Intelligence Challenge 2 (ConvAI2),[3] to train a neural dialogue model. The dataset contains dialogues between pairs of speakers each randomly assigned personas from a set of 1155, each consisting of 4-5 lines of description about the part they should play, e.g. *"I have two dogs"* or *"I like taking trips to Mexico"*. The training set consists of 9,907 such dialogues where the partners play their roles, and a validation set of 1,000 dialogues. The ConvAI2 test set has not been released. Each dialogue is tokenized into words, resulting in a vocabulary of 19,262 unique tokens. We refer the reader to (Zhang et al., 2018) for more details.

Model (Learning)	PPL↓	Hits@1↑
Seq2Seq §2.1	24.84	0.047
Seq2Seq+Scorer §3.2	26.26	0.557

Table 1: Validation Accuracy on PersonaChat of our compared models. Although the perplexity (PPL), which reflects the well-formedness of a response, is lower with the model trained with maximum likelihood, our MLE+Ranking model is better at retrieving a semantically-relevant response (Hits@1) when fine-tuned and used together with a sequence selection scoring function.

5.2 Neural dialogue modelling

Model We closely follow (Bahdanau et al., 2014) in building an attention-based neural autoregressive sequence model. The encoder has two bidirectional layers of 512 long short-term memory (LSTM, Hochreiter and Schmidhuber, 1997) units each direction-layer, and the decoder has two layers of 512 LSTM units each. We use global general attention as described by Luong et al. (2015). We use the same word embedding matrix on both the encoder and decoder, which is initialized from 300-dimensional pretrained GloVe vectors (Pennington et al., 2014). We allow word embedding weights to be updated during the training.

The sequence selection scoring function used in our proposed search strategy is a multi-layer perceptron with four layers of 512 $\tanh$ units each. It outputs a scalar score at the end, and takes as input the LSTM cell from the final step in the decoder.

Learning We use Adam (Kingma and Ba, 2014) with the initial learning rate set to 0.001. We apply dropout (Srivastava et al., 2014) between the LSTM layers with the dropout rate of 0.5 to prevent overfitting. We train the neural dialogue model until it early-stops on the validation set,[4]

[3] http://convai.io/

[4] When the validation loss (3) does not improve for twelve epochs, we early-stop.

then fine-tune it together with the scoring function. We set α and β in Eq. (9) to 0.1 and 1, respectively, during finetuning.

We show in Table 1 the quality of the trained model before and after finetuning. We use the metrics used by the ConvAI2 competition, which are perplexity (PPL) and hits@1. Compared to the leaderboard,[5] the finetuned model is reasonable, and we believe, serves well as an underlying system for investigating the effect of search algorithms. Furthermore, this agrees well with the intuition behind introducing the sequence selection scoring function and training it with the pair-wise ranking loss; that is, the model can focus better on semantics rather than on syntax while selecting the best candidate. For all search experiments, we use the finetuned model.

5.3 Search Strategies

We test four search strategies; **greedy** and **beam** search algorithms from Sec. 2.2, iterative beam search (**iter-beam**) from Sec. 3.1, and iterative beam search combined with final sequence selection (**iter-beam+scorer**) from Sec. 3.2.

Beam search maintains five hypotheses throughout search. Beam search performs final hypothesis score adjustment using a length penalty as described by Wu et al. (2016). In iterative beam search, beam search runs 15 times with beam size 5 each, generating 15 top-hypotheses. **iter-beam** selects the best response among these using the conditional log-probability (2), while **iter-beam+scorer** uses the score R. We use n-gram blocking in Sec. 2.2 for any variant of beam search (**beam**, **iter-beam**, **iter-beam+scorer**) with n up to 7 as this improved results for all methods.

5.4 Evaluation

Human evaluation We use ParlAI (Miller et al., 2017) which provides seamless integration with Amazon Mechanical Turk (MTurk) for human evaluation. A human annotator is paired with a model with a specific search strategy, and both are randomly assigned personas out of a set of 1155, and are asked to make a conversation of at least either five or six turns (randomly decided). We allow each annotator to participate in at most six conversations per search strategy and collect approximately 50 conversations per search strategy.[6] Each conversation is given three scores by the annotator, as described in Sec. 4.1.

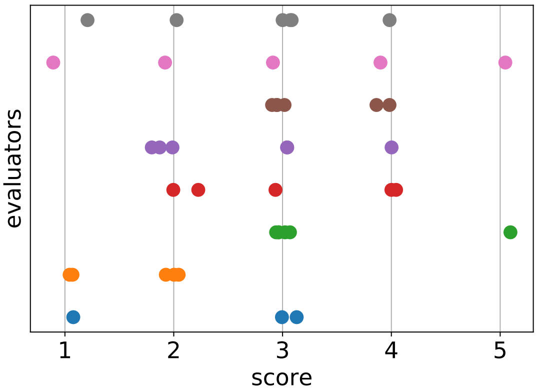

Figure 1: The distribution of scores given by annotators to greedy search. Each row (in an arbitrary order) plots scores given by a single annotator over multiple conversations. This points to the existence of annotator bias. See for instance the bottom three annotators.

Bayesian calibration In order to remove annotator bias, or inter-annotator variability, we use Bayesian calibration from Sec. 4.2. We take 50 warm-up steps and collect 150 samples using NUTS sampler for inferring the posterior mean and variance of the overall score in Eq. (10), while we use 30 warm-up steps and 50 samples for inferring the mean and variance of the average number of positively (or negatively) labelled utterance-pairs in Eq. (11).

Automatic metrics In addition to human evaluation, we also compute a variety of automatic metrics to quantitatively characterize each search algorithm and its impact. First, we report the **log-p**robability of a generated response assigned by the model which is a direct indicator of the quality of a search algorithm. Second, we compute the average number of unique n-grams generated per conversation normalized by the number of generated tokens in the conversation, called **distinct-**n from (Li et al., 2015), with $n = 2$ and $n = 3$. This metric measures the diversity of generated responses, which is considered to correlate well with how engaging a neural dialogue system is (or conversely, anticorrelated with its rate

[5] http://convai.io/#leaderboard

[6] Some conversations were dropped due to technical errors during human evaluation, resulting in total 52, 52, 50 and 48 conversations for **greedy**, **beam**, **iter-beam** and **iter-beam+scorer**, respectively.

of producing boring, meaningless response like "I don't know").

5.5 Result and Analysis

Annotator bias In Fig. 1, we plot the scores provided by the human annotators for one search strategy (**greedy**), where each row corresponds to each annotator. Consider the three annotators in the bottom of the plot. Their spreads are similar, spanning three points, but their means are clearly separated from each other, which points to the existence of annotator bias. This observation supports the necessity of the Bayesian calibration described in Sec. 4.2, we thus analyze results with calibrated scores (while reporting both calibrated and uncalibrated versions).

Another property of annotators' scores in Fig. 1 is that each annotator has their own distinct "spread" as well. This spread is not modelled in the current version of the Bayesian calibration, and we leave incorporating it for the future.

Human evaluation In Table 2, we present the scores from human evaluation. A major observation we make is that greedy search, which has been the search algorithm of choice in neural dialogue modelling, significantly lags behind the variants of beam search in all metrics. This stark difference is worth our attention, as this difference is *solely* due to the choice of a search algorithm and is not the result of different network architectures nor learning algorithms. In fact, this cannot even be attributed to different parameter initialization, as we use only *one* trained model for all of these results.

The three beam-search variants are however more difficult to distinguish from each other in terms of human evaluation. We conjecture that this indistinguishability may be due to the coarse-grained nature of human evaluation based on a scalar score (or three scalar scores.) These variants are qualitatively different from each other in other aspects, as we see below.

Search quality: log-p Better search algorithms find responses with higher log-probability according to the model, as shown in Table 3. This is a natural consequence from exploring a larger subset of the search space.

The higher log-probability does not correspond with increases in sequence selection score according to the learnt scoring function, demonstrating the fundamental disconnect between maximum likelihood and sequence ranking as discussed in Sec. 3.2. By using the scoring function to select the best response among a diverse set of hypotheses, we get responses with a significantly higher selection score (**iter-beam+scorer**).

A notable observation from Table 3 is that the neural sequence model assigns very low log-probabilities and scores to human responses (collected from the validation set).

This suggests that there is more room to improve the models and learning algorithms to place a high probability on human responses.

Diversity: distinct-2 **and distinct-**3 Although the beam-search variants (**beam**, **iter-beam** and **iter-beam+scorer**) were not significantly different in their human ratings, the diversity of generated responses in Table 4 clearly separates them. The proposed iterative beam search combined with selection via the learnt scoring function generates significantly more unique bi- and trigrams than all the other search strategies, indicating this model will be more engaging for longer-term interactions than the competing approaches.

We still observe a significant gap between the best search strategy and humans in these metrics, similar to what we observed with log-probabilities and scores above. This leaves open room for improving network architectures, learning algorithms and/or search strategies even further.

6 Conclusion

In this paper, we have empirically validated the importance of search algorithms in neural dialogue modelling by evaluating four search strategies on one trained model. Extensive evaluation revealed that greedy search, which has been the search algorithm of choice in neural dialogue modelling, significantly lags behind more sophisticated search strategies, such as beam search and its iterative variant. Using human evaluation and measuring the diversity of generated responses, we found the novel strategy of iterative beam search followed by final selection using a scoring function trained with a ranking loss to be the best among the four strategies we compared in this paper. This strategy was deemed an equally good conversationalist as the other beam-search variants by human annotators, while maintaining a higher level of diversity.

Search	Overall Score (1-5)↑		% Good Pairs↑		% Bad Pairs↓	
strategy	Raw	Calibrated	Raw	Calibrated	Raw	Calibrated
greedy	2.79±1.05	2.78±0.24	0.43	0.37±0.08	0.43	0.48±0.08
beam	2.98±1.06	3.00±0.25	0.54	**0.49±0.08**	0.36	0.45±0.08
iter-beam	2.88±1.24	2.86±0.24	0.55	**0.49±0.08**	0.34	**0.35±0.07**
iter-beam+scorer	3.25±1.26	**3.09±0.25**	0.52	0.48±0.07	0.33	0.41±0.07

Table 2: Human evaluation result (average±standard deviation). We observe a significant gap between greedy search and beam search variants, while most of the variants of beam search are within one standard deviation from each other. We bold-face the best strategies per calibrated metric and underline all the strategies within one standard deviation of the best strategy. The standard deviations of the raw scores and the calibrated scores are not compatible, as the latter reflects the uncertainty in posterior inference rather than the spread of scores.

Search strategy	log-p↑	scorer $R(y\|x)$↑
greedy	-10.15±2.9	-3.04±1.36
beam	-7.62±2.25	-3.47±1.53
iter-beam	**-6.62**±1.71	-3.18±2.22
iter-beam+scorer	-12.40±3.00	**-2.68**±1.28
human	-43.58±15.9	-4.79±1.67

Table 3: The average conditional log-probabilities and scores assigned to generated responses. Better search algorithms (**iter-beam**) find responses with a higher log-probability according to the model, but do not necessarily correspond to a higher score from the scoring function (**iter-beam+scorer**). The low log-probability and score assigned to human responses suggest that search is not the sole remedy but better modeling and learning must be investigated further.

Search	distinct-n ↑	
strategy	$n = 2$	$n = 3$
greedy	0.6142	0.5988
beam	0.6739	0.59304
iter-beam	0.6908	0.5953
iter-beam+scorer	**0.7526**	**0.6812**
human	0.86675	0.8147

Table 4: The diversity of generated responses measured by distinct-n clearly indicates that a better search strategy generates more diverse responses, although the best performer is still significantly less diverse than humans. distinct-n's for humans were measured using the conversations from the validation set.

Our observation clearly emphasizes the importance of a good search strategy in neural dialogue modelling, which has thus far been given less attention. With this finding, we encourage authors of future papers on neural dialogue modelling to clearly state which search algorithm has been used, why such choice has been made and the details of its implementation and hyperparameters, in order for readers and the research community to correctly assess the impact of any newly proposed neural dialogue model. Lastly, we believe our observations here raise the question of how many new network architectures and learning algorithms have been proposed, abandoned, or compared favourably or unfairly to existing approaches due to the lack of extensive investigation on search strategies.

Acknowledgments

We thank Théo Matussière for his work on the human evaluation code and procedure, in particular for proposing the good/bad utterance-pair evaluation technique. KC was partly supported by Samsung Electronics (Improving Deep Learning using Latent Structure) and thanks support by AdeptMind, eBay, TenCent, NVIDIA and CIFAR.

References

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

Dhruv Batra, Payman Yadollahpour, Abner Guzman-Rivera, and Gregory Shakhnarovich. 2012. Diverse m-best solutions in markov random fields. In *European Conference on Computer Vision*, pages 1–16. Springer.

Eli Bingham, Jonathan P. Chen, Martin Jankowiak, Fritz Obermeyer, Neeraj Pradhan, Theofanis Karaletsos, Rohit Singh, Paul Szerlip, Paul Horsfall, and Noah D. Goodman. 2018. Pyro: Deep Universal Probabilistic Programming. *arXiv preprint arXiv:1810.09538*.

Yun Chen, Victor OK Li, Kyunghyun Cho, and Samuel R Bowman. 2018. A stable and effective learning strategy for trainable greedy decoding. *arXiv preprint arXiv:1804.07915*.

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

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language modeling with gated convolutional networks. *arXiv preprint arXiv:1612.08083*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

Xiaodong Gu, Kyunghyun Cho, Jungwoo Ha, and Sunghun Kim. 2018. Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. *arXiv preprint arXiv:1805.12352*.

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

Matthew D Hoffman and Andrew Gelman. 2014. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(1):1593–1623.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.

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

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proc. ACL*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2017a. Learning to decode for future success. *arXiv preprint arXiv:1701.06549*.

Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. 2017b. Adversarial learning for neural dialogue generation. *arXiv preprint arXiv:1701.06547*.

Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023*.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. *arXiv preprint arXiv:1809.01984*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

Alexander H Miller, Will Feng, Adam Fisch, Jiasen Lu, Dhruv Batra, Antoine Bordes, Devi Parikh, and Jason Weston. 2017. ParlAI: A dialog research software platform. *arXiv preprint arXiv:1705.06476*.

Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1985. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. 2015. A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742*.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*, pages 3295–3301.

Xiaoyu Shen, Hui Su, Shuzi Niu, and Vera Demberg. 2018. Improving variational encoder-decoders in dialogue generation. *arXiv preprint arXiv:1802.02032*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

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

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

Ashwin K Vijayakumar, Michael Cogswell, Ramprasaath R Selvaraju, Qing Sun, Stefan Lee, David J Crandall, and Dhruv Batra. 2018. Diverse beam search for improved description of complex scenes. In *AAAI*.

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

Jason Weston, Emily Dinan, and Alexander H Miller. 2018. Retrieve and refine: Improved sequence generation models for dialogue. *arXiv preprint arXiv:1808.04776*.

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

Zhen Xu, Bingquan Liu, Baoxun Wang, Chengjie SUN, Xiaolong Wang, Zhuoran Wang, and Chao Qi. 2017. Neural response generation via gan with an approximate embedding layer. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 617–626. Association for Computational Linguistics.

Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning semantic textual similarity from conversations. *arXiv preprint arXiv:1804.07754*.

Yury Zemlyanskiy and Fei Sha. 2018. Aiming to know you better perhaps makes me a more engaging dialogue partner. *arXiv preprint arXiv:1808.07104*.

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? *arXiv preprint arXiv:1801.07243*.

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*.

A Human evaluation questionnaire

A.1 Overall scoring question

Right after the end of dialogue system asks worker the following question:

Now the conversation is completed! Please evaluate the conversation by clicking a button with score from [1, 2, 3, 4, 5] below, this score should reflect how you liked this conversation (1 means you did not like it at all, and 5 means it was an engaging conversation).

A.2 Good/bad pairs selection

After the first question system asks following questions:

Now please select every interaction pair which you consider as a **good**, natural pair of messages. Do not compare them between each other, try to use your life experience now.

Now please select every interaction pair which you consider as a **bad**, some examples of bad partner response are: not answering your question, answering different question, random content, contradicts previous statements etc.

B Model/Training hyperparameters

RNN type	LSTM
RNN layers	2
hidden dim	512
embedding dim	300
dropout rate	50%
attention type	global general
bidirectional encoder	True
shared weights	encoder/decoder embeddings
negative samples	5
margin for ranking loss	1.0
fine-tuning rank weight	1.0
fine-tuning generation weight	0.1
batch size	64
optimizer	Adam
starting learning rate	0.001
gradient clip threshold	0.1
embedding pretraining	glove 840B
validation every...	0.5 epochs
validation metric	hits@1
max valid patience	12 epochs