Learning to Decode for Future Success

Jiwei Li, Will Monroe and Dan Jurafsky
Computer Science Department, Stanford University, Stanford, CA, USA
jiweil,wmonroe4,jurafsky@stanford.edu

Abstract

We introduce a simple, general strategy to manipulate the behavior of a neural decoder that enables it to generate outputs that have specific properties of interest (e.g., sequences of a pre-specified length). The model can be thought of as a simple version of the actor-critic model that uses an interpolation of the actor (the MLE-based token generation policy) and the critic (a value function that estimates the future values of the desired property) for decision making. We demonstrate that the approach is able to incorporate a variety of properties that cannot be handled by standard neural sequence decoders, such as sequence length and backward probability (probability of sources given targets), in addition to yielding consistent improvements in abstractive summarization and machine translation when the property to be optimized is BLEU or ROUGE scores.

1 Introduction

Neural generation models (Sutskever et al., 2014; Bahdanau et al., 2015; Cho et al., 2014; Kalchbrenner and Blunsom, 2013) learn to map source to target sequences in applications such as machine translation (Sennrich et al., 2015; Gulcehre et al., 2015), conversational response generation (Vinyals and Le, 2015; Sordoni et al., 2015), abstractive summarization (Nallapati et al., 2016; Rush et al., 2015).

Neural generation models are standardly trained by maximizing the likelihood of target sequences given source sequences in a training dataset. At test time, a decoder incrementally generates a sequence with the highest probability using search strategies such as beam search. This locally incremental nature of the decoding model leads to the following issueL Decoders cannot be tailored to generate target sequences with specific properties of interest, such as pre-specified length constraints (Shao et al., 2017; Shi et al., 2016), which might be useful in tasks like conversational response generation or non-factoid question answering, and cannot deal with important objectives, such as the mutual information between sources and targets (Li et al., 2016a), that require knowing the full target sequence in advance.

To address this issue, we propose a general strategy that allows the decoder to incrementally generate output sequences that, when complete, will have specific properties of interest. Such properties can take various forms, such as length, diversity, mutual information between sources and targets, and BLEU/ROUGE scores. The proposed framework integrates two models: the standard seq2seq model, trained to incrementally predict the next token, and a future output estimation model, trained to estimate future properties solely from a prefix string (or the representation associated with this string), and incorporated into the decoder to encourage it to make decisions that lead to better long-term future outcomes.

Making decoding decisions based on future success resembles the central idea of reinforcement learning (RL), that of training a policy that leads to better long-term reward. Our work is thus related to a variety of recent work inspired by or using reinforcement learning (e.g., REINFORCE or actor-critic models) for sequence generation (Wiseman and Rush, 2016; Shen et al., 2015; Bahdanau et al., 2016; Ranzato et al., 2016). The proposed model can be viewed as a simpler but more effective version of the actor-critic RL model (**?**) in sequence generation: it does not rely on the critic to update the policy (the actor), but rather, uses a linear interpolation of the actor (the policy) and the critic (the value function) to make final decisions. Such a strategy comes with the following benefits: (1) It naturally avoids the known problems such as large variance and instability with the use of reinforcement learning in tasks with enormous search spaces like sequence generation. As will be shown in the experiment sections, the simplified take on reinforcement without policy updates yields consistent improvements, not only outperforming standard SEQ2SEQ models, but also the RL models themselves in a wide range of sequence generation tasks; (2) training RL-based generation models using specific features like sequence length as rewards not only increases the model's instability but may also lead to suboptimal generated utterances, for example, sequences that satisfy a length constraint but are irrelevant, incoherent or even ungrammatical.[1]

We study how to incorporate different properties into the decoder different properties of the future output sequence: (1) sequence length: the approach provides the flexibility of controlling the output length, which in turns addresses sequence models' bias towards generating short sequences (Sountsov and Sarawagi, 2016); (2) mutual information between sources and targets: the approach enables modeling the bidirectional dependency between sources and targets at each decoding timestep, significantly improving response quality on a task of conversational response generation and (3) the properties can also take the form of the BLEU and ROUGE scores, yielding consistent improvements in machine translation and summarization, yielding the state-of-the-art result on the IWSLT German-English translation task.

2 Model Overview

In this section, we first review the basics of training and decoding in standard neural generation models. Then we give a sketch of the proposed model.

2.1 Basics

Neural sequence-to-sequence (SEQ2SEQ) generation models aim to generate a sequence of tokens Y given input sequence X. Using recurrent nets, LSTMs (Hochreiter and Schmidhuber, 1997) or CNNs (Krizhevsky et al., 2012; Kim, 2014), X is first mapped to a vector representation, which is then used as the initial input to the decoder. A neural generation model defines a distribution over outputs by sequentially predicting tokens using a softmax function:

$$p(Y|X) = \prod_{t=1}^{n_Y} p(y_t|X, y_{1:t-1})$$

Decoding typically seeks to find the maximum-probability sequence Y^* given input X:

$$Y^* = \arg\max_{Y} p(Y|X) \tag{1}$$

The softmax function that computes $p(y_t|X, y_{1:t-1})$ takes as input the hidden representation at time step $t-1$, denoted by h_{t-1}. The hidden representation h_{t-1} is computed using a recurrent net that combines the previously built representation h_{t-2} and the word representation e_{t-1} for word y_{t-1}. It is infeasible to enumerate the large space of possible sequence outputs, so beam search is normally employed to find an approximately optimal solution. Given a partially generated sequence $y_{1:t-1}$, the score for choosing token y_t (denoted by $S(y_t)$) is thus given by

$$S(y_t) = \log p(y_t|h_{t-1}) \tag{2}$$

2.2 The Value Function Q

The core of the proposed architecture is to train a future outcome prediction function (or value function) Q, which estimates the future outcome of taking an action (choosing a token) y_t in the present. The function Q is then incorporated into $S(y_t)$ at each decoding step to push the model to generate outputs that lead to future success. This yields the following definition for the score $S(y_t)$ of taking action y_t:

$$S(y_t) = \log p(y_t|h_{t-1}) + \gamma Q(X, y_{1:t}) \tag{3}$$

where γ denotes the hyperparameter controlling the trade-off between the local probability prediction $p(y_t|h_{t-1})$ and the value function $Q(X, y_{1:t})$. The input to Q can take various forms, such as the vector representation of the decoding step after y_t has been considered (i.e., h_t) or the raw strings (X and $y_{1:t}$).[2]

Q can be trained either jointly with or independently of the SEQ2SEQ model. When training Q, we provide it with source-target pairs (X, Y), where Y is a full sequence. $Y = \{y_1, y_2, ..., y_N\}$ can either be sampled or decoded using a trained model (making Q dependent on the pre-trained

[1]A workaround is to use the linear interpolation of the MLE-based policy and the value function for a specific property as a reward for RL training. This strategy comes with the following disadvantages: it requires training different models for different interpolation weights, and again, suffers from large training variance.

[2]One can think of h_t as the output of a function that takes as input X and $y_{1:t}$.

SEQ2SEQ model) or can be taken from the training set (making Q independent of the SEQ2SEQ model). However, Y must always be a full sequence. The future outcome of generating each of the tokens of Y (y_1, y_2, ..., y_N) is the feature score (BLEU, length, mutual information, etc.) associated with the full sequence Y, denoted by $q(Y)$. The future outcome function Q is trained to predict $q(Y)$ from $(X, y_{1:t})$, where $1 \leq t \leq N$.

Q estimates the long-term outcome of taking an action y_t. It is thus similar to the value function in Q-learning, the role of the critic in actor-critic reinforcement learning (Sutton, 1988; Grondman et al., 2012), the value network for position evaluation in the Monte-Carlo tree search of AlphaGo (Silver et al., 2016), or the h* function in A^* search. (Och et al., 2001).

In this paper, *value function* and *future outcome prediction function* and Q are interchangeable

In the sections below, we will describe how to adapt this general framework to various features with different properties and different kinds of input to the future outcome prediction function.

3 Q for Controlling Sequence Length

For tasks like machine translation, abstractive summarization and image caption generation, the information required to generate the target sequences is already embedded in the input. Usually we don't have to worry about the length of targets, since the model can figure it out itself; this is a known, desirable property of neural generation models (Shi et al., 2016).

However, for tasks like conversational response generation and non-factoid question answering, in which there is no single correct answer, it is useful to be able to control the length of the targets. Additionally, in tasks like conversational response generation, SEQ2SEQ models have a strong bias towards generating short sequences (Sountsov and Sarawagi, 2016). This is because the standard search algorithm at decoding time can only afford to explore very a small action space. As decoding proceeds, only a small number of hypotheses can be maintained. By Zipf's law, short sequences are significantly more frequent than longer ones. Therefore, the prefixes of shorter responses are usually assigned higher probability by the model. This makes prefixes of longer sequences fall off the beam after a few decoding steps, leaving only short sequences.

One can still force the model to keep generating tokens (simply by prohibiting the *EOS* token and forcing the decoding to proceed). However, since the previous decoded tokens were chosen with a shorter sentence in mind, artificially lengthening the response this way will result in low-quality responses. In particular, problems arise with repetition ("*no, no, no, no, no, no*") or incoherence (*"i like fish, but i don't like fish but I do like fish"*).

3.1 Training Q for Sequence Length

Shao et al. (2017) give one efficient method of generating long sequences, consisting of a stochastic search algorithm and segment-by-segment reranking of hypotheses. The fundamental idea is to keep a diverse list of hypotheses on the beam and remove those that are similar to each other, so as to explore the space more adequately. While more adequately exploring the search space can increase the likelihood of generating long sequences, since the beam is more likely to include a prefix of a long sequence, this method doesn't offer direct control over sequence length. Length information seems to be embedded in the hidden representations of neural models in some implicit way (Shi et al., 2016). We therefore build another neural model to expose this length information and use it to estimate the number of words left to be generated.

Given a pre-trained sequence-to-sequence model, an input sequence X, and a target $Y = \{y_1, y_2, ..., y_N\}$, where N denotes the length of y, we first run a forward pass to compute the hidden representation h_t associated with each time step on the target side ($1 \leq t \leq N$). Then we build a regression model $Q(h_t)$, which takes as input h_t to predict the length of the remaining sequence, i.e., $N - t$. The model first passes h_t to two non-linear layers, on top of which is a linear regression model which outputs the predicted number of tokens left to decode. The regression model is optimized by minimizing the mean squared loss between the predicted sequence length and the gold-standard length $N - t$ on source–target pairs taken from the training set.

3.2 Decoding

Given an input X, suppose that we wish to generate a target sequence of a pre-specified length N. At decoding time step $t - 1$, we first obtain the vector representation h_{t-1} for the current time step. The score used to rank choices for the next token y_t is a linear combination of the log probability outputted from the sequence model and the mean square loss

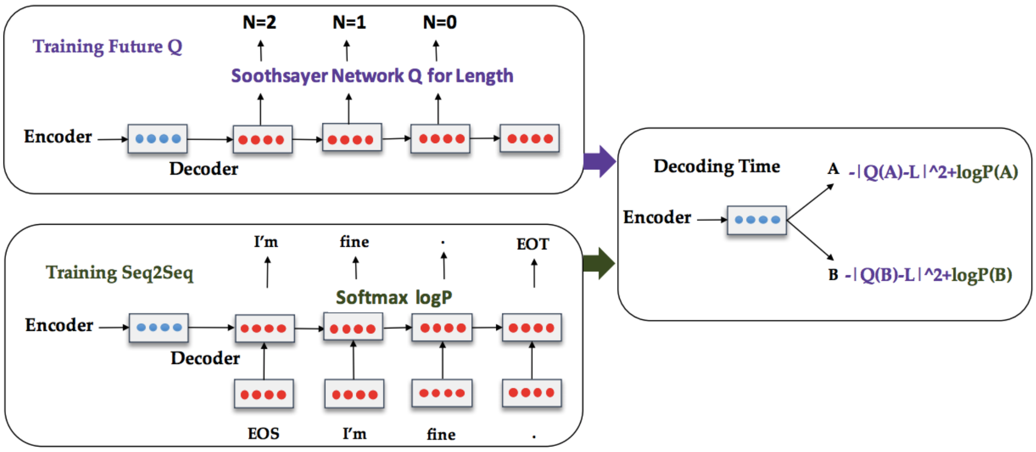

Figure 1: An illustration of the proposed future length prediction model. **N** denotes the number of words left to generate and L denotes the pre-specified sequence length.

between the number of words left to generate ($N - t$) and the output from $Q(h_t)$:

$$y_t = \arg\max_{y} \log p(y_{1:t}|X) - \lambda||(N-t) - Q(h_t)||^2 \quad (4)$$

where $y_{1:t}$ is the concatenation of y_t and previously decoded (given) sequence $y_{1:t-1}$, and h_t is obtained from the sequence model by combining h_{t-1} and the word representation of y_t as if it were the true next token. λ is a hyperparameter controlling the influence of the future length estimator.

3.3 Experiments

We evaluate the proposed model on the task of open-domain conversation response generation (Vinyals and Le, 2015; Sordoni et al., 2015; Serban et al., 2015a; Mei et al., 2016; Serban et al., 2015b), in which a model must predict the next turn of a dialogue given the preceding ones. We use the OpenSubtitles (OSDb) dataset (Tiedemann, 2009).

We compare the proposed model with the standard SEQ2SEQ beam search (SBS) decoder. We first group test dialogue pairs by target length and decode each group. At decoding time, for an input with a gold target of length L, we force the model to generate an output of length L. This can be achieved by selecting a hypothesis that predicts an *EOS* token at time step $L + 1$.[3] If no *EOS* is predicted at time step $L + 1$, we continue decoding and stop once an *EOS* token is generated. We report BLEU scores on the concatenation of the outputs from each length cluster.[4]

Model	SBS	Length prediction Q
BLEU	1.45	1.64
AdverSuc	0.034	0.040
machine-vs-random	0.923	0.939

Table 2: Comparison of the proposed algorithm with length prediction and the standard beam search algorithm.

We also report adversarial success (*AdverSuc*) and *machine-vs-random* accuracy, evaluation metrics proposed in Li et al. (2016c). Adversarial success refers to the percentage of machine-generated responses that are able to fool a trained evaluator model into believing that they are generated by a human; *machine-vs-random* accuracy denotes the accuracy of a (different) trained evaluator model at distinguishing between machine-generated responses and randomly-sampled responses.[5] Higher values of adversarial success and *machine-vs-random* accuracy indicate the superiority of a model. We refer readers to Li et al. (2016c) for more details. Table 2 presents the quantitative results: adding predictor rankers increases the general quality of generated responses.

Sampled responses (from a random batch, without cherry-picking) are shown in Table 1, with more examples shown in Table 7 in the Appendix. We force the decoding model to generate 20 tokens using the strategy described above. We can

[3] If multiple hypotheses satisfy this requirement, we pick the one with the largest likelihood.

[4] In this setup, both algorithms are allowed to know the length of the gold-standard targets. The results from different models are thus comparable. This is to remove the effect of target length on the evaluation metrics (all metrics employed are sensitive to target length).

[5] The estimators for *AdverSuc* and *machine-vs-random* accuracy are trained using a hierarchical network (Serban et al., 2016) See Li et al. (2016c) for details.

Input	it s an 8 symbol address .
SBS	i don 't know where it came from but it s not the same as it used to be .
FP(λ=1)	you know i think i ve seen it once and i ve seen it in the past too ! .
FP(λ=5)	but we don t know who she was at the scene but that s a good idea ! ? !
Input	come on pollyanna let 's go .
SBS	come on let 's go let 's go go go go go go go go go go go go go
FP(λ=1)	you want me to take you to your house and come back and get me some ice cream ? !
FP(λ=5)	you want me to take the stairs and go with you ... and get the hell out of here ! !
Input	that 's hardly a winning attitude .
SBS	the only way that we could find out is that we 're in the middle of the game ! ! !
FP(λ=1)	but i think the most important thing is that we 're both in the same class together . . .
FP(λ=5)	but the other thing is i think you can use it as an excuse for your own personal gain .
Input	i see and you know why ?
SBS	because i don 't know who i am or where i am or where i am or where i am
FP(λ=1)	why would the people think i 'm the only guy who could get a job in the middle east ?
FP(λ=5)	i know the guy i saw at the scene ... and he said it was my father 's birthday !
Input	but venus was no easy target .
SBS	it was the only way to make it to the top of the island . . . . . . but it wasn 't
FP(λ=1)	i think we have the right to be in the middle of some sort of a trap . . .
FP(λ=5)	the only reason i left here to save you ... was because i didn 't care who they are !
Input	i 'm not afraid of her .
SBS	i 'm afraid she 's afraid of the dark ... ... but i 'm afraid she 's afraid of me .
FP(λ=1)	you don 't like to tell people you 're just a child and you don 't know her ? !
FP(λ=5)	i 'll be in a very awkward moment of her time and i 'm afraid she 'll hurt us again

Table 1: Sample of responses generated by standard beam search (denoted by *SBS*) and the future length prediction (*FP*) algorithm with two different values of λ. We force each decoding algorithm to generate responses with length 20. More examples are shown in Table 7 (Appendix).

clearly identify problems with the standard beam search algorithm: the decoder produces tokens that are optimal for shorter sequences, eliminating candidates from the beam that would lead to longer possibilities. Once the length reaches the point where the intended shorter sequence would naturally conclude, it has no option but to fill space with repetitions of tokens (e.g., *"go, go, go"* or strings of punctuation) and phrases (e.g., *i don 't know who i am or where i am or where i am or where i am*), or addenda that are sometimes contradictory (e.g., *it was the only way to make it to the top of the island . . . . . . but it wasn 't*). This issue is alleviated by the proposed length-prediction algorithm, which plans ahead and chooses tokens that lead to meaningful sequences with the desired length. More coherent responses are observed when the hyperparameter λ is set to 1 than when it is set to 5, as expected, since the decoding algorithm deviates more from the pre-trained model when λ takes larger values.

4 Q for Mutual Information

4.1 Background

Maximum mutual information (MMI) has been shown to be better than maximum likelihood estimation (MLE) as an decoding objective for conversational response generation tasks (Li et al., 2016a). The mutual information between source X and target Y is given by $\log[p(X,Y)/p(X)p(Y)]$, which measures bidirectional dependency between sources and targets, as opposed to the unidirectional dependency of targets on sources in the maximum likelihood objective. Modeling the bidirectional dependency between sources and targets reduces the prevalence of generic responses and leads to more diverse and interesting conversations.[6] Maximizing a weighted generalization of mutual information between the source and target can be shown using Bayes' rule to be equivalent to maximizing a linear combination of the forward probability $\log p(Y|X)$ (the standard objective function for SEQ2SEQ models) and the backward probability $\log p(X|Y)$:[7]

$$Y^* = \arg\max_Y \log p(Y|X) + \lambda \log p(X|Y) \quad (5)$$

Unfortunately, direct decoding using Eq.5 is infeasible, since it requires completion of target generation before $p(X|Y)$ can be effectively computed, and the enormous search space for target y prevents exploring all possibilities. An approximation approach is commonly adopted, in which an N-best list is first generated based on $p(Y|X)$ and then reranked by adding $p(X|Y)$. The problem with this reranking strategy is that the beam search

[6] This is because although it is easy to produce a sensible generic response Y regardless of the input sequence X, it is much harder to guess X given Y if Y is generic.

[7] When using this objective, $p(Y|X)$ and $p(X|Y)$ are separately trained models with different sets of parameters.

step gives higher priority to optimizing the forward probability, resulting in solutions that are not globally optimal. Since hypotheses in beam search are known to lack diversity (Li et al., 2016b; Vijayakumar et al., 2016; Gimpel et al., 2013), after decoding is finished, it is sometimes too late for the reranking model to have significant impact. Shao et al. (2017) confirm this problem and show that the reranking approach helps for short sequences but not longer ones.

4.2 Training Q for Mutual Information

The first term of Eq. 5 is the same as standard SEQ2SEQ decoding. We thus focus our attention on the second term, $\log p(X|Y)$. To incorporate the backward probability into intermediate decoding steps, we use a model to estimate the future value of $p(X|Y)$ when generating each token y_t.

For example, suppose that we have a source-target pair with source X = *"what 's your name"* and target Y = *"my name is john"*. The future backward probability of the partial sequences *"my"*, *"my name"*, *"my name is"* is thus $p(X|Y)$. Again, we use $Q(y_t)$ to denote the function that maps a partially generated sequence to its future backward probability, and we can factorize Eq. 5 as follows:

$$y_t = \arg\max_y \ \log p(y_{1:t-1}, y|X) + \lambda Q(y) \quad (6)$$

We propose two ways to obtain the future backward-probability estimation function $Q(y_t)$.

(1) As in the strategies described in Sections 5 and 3, we first pretrain a SEQ2SEQ model for both $p(Y|X)$ and $p(X|Y)$. The training of the latter is the same as a standard SEQ2SEQ model but with sources and targets swapped. Then we train an additional future backward-probability estimation function $Q(X, y_{1:t})$, which takes as inputs the hidden representation of intermediate decoding steps (i.e., h_t) from the forward probability model and predicts the backward probability for the entire target sequence Y using the pretrained backward SEQ2SEQ model (i.e., $\log p(X|Y)$ with Y being the full target).

(2) We can directly train models to calculate $Q(y_t) = p(X|y_{1:t})$, i.e., the probability of generating a full source given a partial target. To do this, we first break y into a series of partial sequences, i.e., $y_{1:1}$, $y_{1:2}$, ..., $y_{1:N}$, which is {*"i"*, *"i am"*, *"i am john"*} in the example above. Then we pair each partial sequence $y_{1:t}$ ($1 \leq t \leq N$) with the source and use each pair $(y_{1:t}, X)$ as a training example to a train SEQ2SEQ model, with $y_{1:t}$ as the source and X as the target. Since we are increasing the size of the training set by roughly a factor of 10 (the average target length is about 10), training is extremely computation-intensive. We reduce the training time by grouping $y_{1:t}$ by length and training a separate SEQ2SEQ model for each length group. At decoding time, we use the score from the model corresponding to the length of the current partially decoded target to generate the next token. Since SEQ2SEQ models for different target lengths are independent, they can be trained in parallel.

We find that option (2) generally outperforms option (1), but option (2) requires training in parallel on a large number of machines.

	Q for MMI	MMI	SBS
BLEU	1.87	1.72	1.45
AdverSuc	0.068	0.057	0.043
Distinct-1	0.019	0.010	0.005
Distinct-2	0.058	0.030	0.014

(a) Full dataset.

	Q for MMI	MMI	SBS
BLEU	2.13	2.10	1.58
AdverSuc	0.099	0.093	0.074
Distinct-1	0.024	0.014	0.007
Distinct-2	0.065	0.033	0.017

(b) Set with short targets.

	Q for MMI	MMI	SBS
BLEU	1.52	1.34	1.58
AdverSuc	0.042	0.029	0.022
Distinct-1	0.017	0.008	0.004
Distinct-2	0.054	0.027	0.012

(c) Set with long targets.

Table 3: Comparison of the proposed future prediction model with MMI-based reranking (*MMI*) and MLE-based standard beam search (*SBS*).

4.3 Experimental Results

We compare the results for the approach with standard beam search using the MLE objective and the MMI reranking approach of Li et al. (2016a), which performs reranking only after decoding is finished. We report BLEU scores and *AdverSuc* scores[8] on the test set. We also report diversity scores (denoted by *Distinct-1* and *Distinct-2*); these are defined as in Li et al. (2016a) to be the the number of distinct unigrams and bigrams (respectively) in generated responses, divided by the total number of generated tokens (to avoid favoring long sentences). Addi-

[8] The *machine-vs-random* scores for the three models are very similar, respectively 0.947, 0.939, 0.935.

Input	it s an 8 symbol address .
SBS	i don 't know where it came from .
MMI	i don t know where it came from .
Q(MMI) (λ=1)	i don 't know the exact address .
Q(MMI) (λ=5)	it 's a very old house in paris
Input	that 's hardly a winning attitude .
SBS	that 's the way it 's always been .
MMI	it 's the only way to win the race .
Q(MMI) (λ=1)	it 's just a game .
Q(MMI) (λ=5)	it 's just a game of golf . take it easy .
Input	what is up with those two ?
SBS	i don 't know what 's going on .
MMI	i don 't know what the hell 's going on here !
Q(MMI) (λ=1)	why don 't you go back and have a look ?
Q(MMI) (λ=5)	why don 't we go up there now and check ?
Input	i got to get them in .
SBS	i don 't have time for this .
MMI	i don 't have time for them .
Q(MMI) (λ=1)	you don 't have to worry about them .
Q(MMI) (λ=5)	i gotta talk to them all .
Input	i see and you know why ?
SBS	i don 't know what you talking about .
MMI	because i 'm in love with you .
Q(MMI) (λ=1)	because i don 't know what the hell i 'm talking about !
Q(MMI) (λ=5)	because it 's because of you

Table 4: Sample of responses generated by (1) standard beam search (*SBS*); (2) the MMI reranking approach of Li et al. (2016a), which performs reranking only after decoding is complete (denoted by *MMI*); and (3) the future prediction model Q(MMI) with different values of future prediction weight λ.

tionally, we split the dev set into a subset containing longer targets (with length larger than 8) and a subset containing shorter ones (smaller than 8). During decoding, we force the model to generate targets of the same length as the gold standard targets using the strategy described in Section 3.3.

Table 3 presents quantitative results for the different decoding strategies. On the full test set, the future backward-probability prediction model outperforms the approach of reranking when decoding is fully finished. Specifically, a larger performance improvement is observed on examples with longer targets than on those with shorter ones. This effect is consistent with the intuition that for short responses, due to the relatively smaller search space, doing reranking at the end of decoding is sufficient, whereas this is not the case with longer sequences: as beam search proceeds, a small number of prefixes gradually start to dominate, with hypotheses differing only in punctuation or minor morphological variations. Incorporating mutual information in the early stages of decoding maintains diverse hypotheses, leading to better final results.

Table 4 presents sampled outputs from each strategy, with more results shown in the Table 8 (Appendix). As can be seen, the results from reranking are generally better than those from MLE, but sometimes both approaches still generate the same generic outputs. This is due to the fact that reranking is performed only after more interesting outputs have fallen off the beam. Using smaller values of λ, the future backward-probability prediction approach generally yields better results than reranking. When using larger values of λ, the algorithm tends to produce more diverse and interesting outputs but has a greater risk of generating irrelevant responses.

5 Q for BLEU/ROUGE

The future outcome function Q can be trained to predict arbitrary features. These features include BLEU (Papineni et al., 2002) or ROUGE (Lin, 2004) scores. We thus train Q to directly predict future BLEU or ROUGE values. In this situation, the future prediction function is able to reduce the discrepancy between training (using maximum likelihood objective) and testing (using BLEU or ROUGE) (Wiseman and Rush, 2016; Shen et al., 2015; Ranzato et al., 2016).

5.1 Model

Given a pre-trained sequence generation model, an input sequence X, and a partially decoded sequence $y_{1:t-1}$, we want to estimate the future reward for taking the action of choosing word y_t for the current time-step. We denote this estimate $Q(\{y_t, y_{1:t-1}, X\})$, abbreviated $Q(y_t)$ where possible.

The future prediction network is trained as follows: we first sample y_t from the distribution $p(y_t|X, y_{1:t-1})$, then decode the remainder of the sequence Y using beam search. The future outcome for the action y_t is thus the score of the final decoded sequence, $q(Y)$. Having obtained pairs $(q(Y), \{X, y_{1:t}\})$, we train a neural network model that takes as input X and $y_{1:t}$ to predict $q(Y)$. The network first maps the input sequence X and the partially decoded sequence $y_{1:t}$ to vector representations using LSTMs, and then uses another network that takes the concatenation of the two vectors to output the final outcome $q(Y)$. The future prediction network is optimized by minimizing the mean squared loss between the predicted value and the real $q(Y)$ during training.

At decoding time, $Q(y_t)$ is incorporated into the decoding model to push the model to take actions that lead to better future outcomes. An action y_t is thus evaluated by the following function:

$$y_t = \arg\max_y \ \log p(y_{1:t-1}, y|X) + \lambda Q(y) \quad (7)$$

λ is a hyperparameter that is tuned on the development set.

5.2 Experiments

We evaluate the decoding model on two sequence generation tasks, machine translation and abstractive summarization.

Machine Translation We use the German-English machine translation track of the IWSLT 2014 (Cettolo et al., 2014), which consists of sentence-aligned subtitles of TED and TEDx talks. For fair comparison, we followed exactly the data processing protocols defined in Ranzato et al. (2016), which have also been adopted by Bahdanau et al. (2016) and Wiseman and Rush (2016). The training data consists of roughly 150K sentence pairs, in which the average English sentence is 17.5 words long and the average German sentence is 18.5 words long. The test set is a concatenation of dev2010, dev2012, tst2010, tst2011 and tst2012, consisting of 6750 sentence pairs. The English dictionary has 22822 words, while the German has 32009 words.

We train two models, a vanilla LSTM (Sutskever et al., 2014) and an attention-based model (Bahdanau et al., 2015). For the attention model, we use the *input-feeding* model described in Luong et al. (2015) with one minor modification: the weighted attention vectors that are used in the softmax token predictions and those fed to the recurrent net at the next step use different sets of parameters. Their values can therefore be different, unlike in Luong et al. (2015). We find that this small modification significantly improves the capacity of attention models, yielding more than a +1.0 BLEU score improvement. We use structure similar to that of Wiseman and Rush (2016), a single-layer sequence-to-sequence model with 256 units for each layer. We use beam size 7 for both standard beam search (SBS) and future outcome prediction.

Results are shown in Table 5, with *SBS* standing for the standard beam search model and *future func* as the proposed future prediction model. Baselines employed include the REINFORCE model described in Ranzato et al. (2016), the actor-critic RL model described in Bahdanau et al. (2016) and the beam-search training scheme described in Wiseman and Rush (2016). Results are reprinted from the best setting in the corresponding paper.

REINFORCE (Ranzato et al., 2016)	20.7
Actor-Critic (Bahdanau et al., 2016)	22.4
Wiseman and Rush (2016)	26.3
vanilla LSTM + SBS	18.9
vanilla LSTM + Q(BLEU)	19.7 (+0.8)
attention+ SBS	27.9
attention + Q(BLEU)	**28.3** (+0.4)

Table 5: BLEU scores for different systems. Baseline scores are best scores reprinted from corresponding papers. SBS denotes standard beam search.

attention + SBS	12.2
attention + Q(ROUGE)	**13.2** (+1.0)

Table 6: ROUGE-2 for abstractive summarization. SBS denotes standard beam search.

Our implementation of the attention model itself already achieves state-of-the-art performance on this benchmark. The proposed future outcome model adds +0.4 BLEU, pushing the SOTA performance up to 28.3. Since the trained SEQ2SEQ model is already quite strong, there is less room for improvement. For the vanilla LSTM, however, due to its relative inferiority, we observe a more significant improvement from the future outcome prediction approach.

Abstractive Summarization We follow the protocols described in Rush et al. (2015), in which the source input is the first sentence of a new article and the target output is the headline. Our training dataset consists of 2M pairs. We train a two-layer word-level attention model with 512 units for each layer. Experimental results are shown in Table 6. We observe a +1.0 ROUGE performance improvement from the proposed model over standard beam search.

6 Conclusion

In this paper, we propose a general strategy that enables a neural decoder to generate outputs that have specific properties of interest. We show how to use a model Q to optimize three useful properties of the output—sequence length, mutual information and BLEU/ROUGE scores —and investigate the effects of different designs for the predictor model and decoding algorithm. Our model provides a general and easy-to-implement way to control neural generation models to meet their specific needs, while improving results on a variety of generation tasks.

References

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086* .

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT evaluation campaign, IWSLT 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325* .

Kyunghyun Cho. 2016. Noisy parallel approximate decoding for conditional recurrent language model. *arXiv preprint arXiv:1605.03835* .

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* .

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL-HLT*.

Kevin Gimpel, Dhruv Batra, Chris Dyer, and Gregory Shakhnarovich. 2013. A systematic exploration of diversity in machine translation. In *EMNLP*.

Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42(6):1291–1307.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535* .

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

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *EMNLP*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A diversity-promoting objective function for neural conversation models.

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

Jiwei Li, Will Monroe, Tianlin Shi, and Dan Jurafsky. 2016c. Adversarial reinforcement learning for neural dialogue generation.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*.

Yi Luan, Yangfeng Ji, and Mari Ostendorf. 2016. LSTM based conversation models. *arXiv preprint arXiv:1603.09457* .

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Hongyuan Mei, Mohit Bansal, and Matthew R Walter. 2016. Coherent dialogue with attention-based language models. *arXiv preprint arXiv:1611.06997* .

Ramesh Nallapati, Bowen Zhou, Çağlar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *CoNLL*.

Franz Josef Och, Nicola Ueffing, and Hermann Ney. 2001. An efficient A* search algorithm for statistical machine translation. In *Proceedings of the Workshop on Data-Driven Methods in Machine Translation*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *ICLR*.

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

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909* .

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015a. Building end-to-end dialogue systems using generative hierarchical neural network models. *arXiv preprint arXiv:1507.04808* .

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015b. Hierarchical neural network generative models for movie dialogues. *arXiv preprint arXiv:1507.04808* .

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16)*.

Louis Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating long and diverse responses with neural conversational models. *arXiv preprint arXiv:1701.03185* .

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2015. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433* .

Xing Shi, Kevin Knight, and Deniz Yuret. 2016. Why neural translations are the right length. In *EMNLP*.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *Nature* 529(7587):484–489.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. *arXiv preprint arXiv:1506.06714* .

Pavel Sountsov and Sunita Sarawagi. 2016. Length bias in encoder decoder models and a case for global conditioning. *arXiv preprint arXiv:1606.03402* .

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Richard S Sutton. 1988. Learning to predict by the methods of temporal differences. *Machine learning* 3(1):9–44.

Jörg Tiedemann. 2009. News from OPUS – A collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing*. volume 5, pages 237–248.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424* .

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. In *Proceedings of the ICML Deep Learning Workshop*.

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960* .

Appendix

Input	one of those american robot cars .
SBS	i don 't know what the hell you 're talking about ... ... but it 's the only solution .
FP(λ=1)	you know i don t know if you 've got a lot of good ideas on the subject sir !
FP(λ=5)	the first time they 've ever had a chance to know ... ... who got them killed ? ! !
Input	the comanches mate their women early .
SBS	they don 't even know who they 're dealing with but they don 't want to see their children .
FP(λ=1)	and now it seems to have a new owner who has a name for a new friend in america .
FP(λ=5)	the only reason that you don 't have a problem is that there 's no more . . . .
Input	but she is still our innocent .
SBS	she 's not the only one who knows what she 's doing to her own life . . . .
FP(λ=1)	and now we can all find out what happened in the last place that she was in love with isn
FP(λ=5)	and the only reason we 've taken the case ... is because she has no intention of leaving her .
Input	oh must i the whole sentence ?
SBS	i don 't know if i 'm guilty or not guilty ... ... but i 'm guilty of murder .
FP(λ=1)	i mean i 'll have you in my quarters when i 'm through with the crime scene . ! ?
FP(λ=5)	well if you 'd mind saying i 'd rather be the one who 's on my own now ! !
Input	because that 's a real guy .
SBS	that 's why he 's in love with a guy who doesn 't know what he 's talking about .
FP(λ=1)	but i think we all know who he was and why we came from the real world ! ? !
FP(λ=5)	i mean you know who i think that 's the guy who lives in the real world right now .
Input	that 's what this job is .
SBS	you don 't have to worry about the money ... ... or the money or the money or anything .
FP(λ=1)	and i 'll tell you that i 'm not gonna let you in on this one okay ? ! ?
FP(λ=5)	it means we have to go to jail because we have to go through the whole thing all right sir
Input	supervisor tang they are starting to quarrel
SBS	i don 't want to be late for the meeting . . . . . . but i can 't
FP(λ=1)	but the boss doesn 't even care about his family and he 's the only one in this family .
FP(λ=5)	and if we do not succeed ... ... the next time we 'll be back together . . . !
Input	we 're almost out of time .
SBS	we 've got to get back to the ship before the sun hits the moon and the moon will rise .
FP(λ=1)	and the next day the next time you go into the city you 'll go home to bed again !
FP(λ=5)	the last time i checked out ... ... i had to be in the wrong place . ! ? !
Input	get your fat ass over there !
SBS	get the fuck out of my way ! ! ? ! ? ! ? ! ? ! ? ! ?
FP(λ=1)	don 't you think i 've got a better idea than to get my ass kicked out here ? !
FP(λ=5)	you know i 'm going to have to go over and see the other two kids ? ! ? !
Input	i gotta have the whole beehive .
SBS	i 'm in the middle of the street and i 'm in the middle of the street and i say
FP(λ=1)	he says i gotta go to bed and i 'll get the car and i 'll make it easy !
FP(λ=5)	the other night when the party starts going down ... ... there 's no other place for them ! ?
Input	what are you trying to hide ?
SBS	i don 't know what you 're talking about . . . . . . but it 's the truth .
FP(λ=1)	i thought he said that i 'd find a way to make sure he didn 't kill me . !
FP(λ=5)	i thought he said you were gonna go to a meeting ... ... or a party ? ! ? !
Input	oh my that looks yummy lily .
SBS	i don 't think she 's gonna wear it in the first place but she 's gonna be okay .
FP(λ=1)	you know i don 't think you 've ever been in love with a girl like me ever before !
FP(λ=5)	you know i don 't know how much longer that 's gonna be in your body . okay ? .
Input	the man said nothin at first .
SBS	he said he didn 't know what he was talkin about but he didn 't say nothin about it .
FP(λ=1)	he said that the first thing that happened to him was the first time he ever came here again !
FP(λ=5)	and then the second and third ... ... i went to the bathroom and there 's nothin wrong ? !
Input	we get in a big hand .
SBS	we don 't know what we 're talking about but we 're in the middle of a big fight .
FP(λ=1)	the other day when we came here ... ... we had to go to the police right away . !
FP(λ=5)	you know we 'll see you around the block when we get there and get your ass outta here !
Input	i have an announcement to make .
SBS	i want you to meet the president of the united states ... ... and i want to thank you .
FP(λ=1)	you know that i was the first person i ever worked with in my entire career . . . !
FP(λ=5)	you know that i was the only man in my crew who would be interested in this . . .
Input	the unknown order is chaos man .
SBS	it 's the only thing we need to do right now right now is that we 're in the middle .
FP(λ=1)	you know what we 'll find on the planet when you 're in the middle of the war ? !
FP(λ=5)	i know the right way to get rid of it but i can 't handle it . ! ? !

Table 7: Sampled examples from responses generated by standard beam search (denoted by *SBS*) and the proposed Length-Predictor algorithm in Section 4 with different weight λ. We force each decoding algorithm to generated responses with length 20. We randomly take one batch from the dev set and the examples are not cherry picked.

Input	we 're not at war sir .		
MLE	we 're in the middle of a war .	**MMI**	we 're in the middle of a war .
FP (λ=1)	we don 't know where we are .	**FP** (λ=5)	we 're not at peace captain we 'll do what 's best for them
Input	the agency doesn 't think so .		
MLE	you don 't know what you 're talking about .	**MMI**	oh , really ?
FP (λ=1)	i don 't think it 's funny .	**FP** (λ=5)	i think it 's because they don 't know who he is .
Input	he just stole her from me .		
MLE	i don 't know who he is .	**MMI**	i don 't know who he is .
FP (λ=1)	calm down ! !	**FP** (λ=5)	oh, what 's your plan then ?
Input	the comanches mate their women early .		
MLE	that 's the way it 's gonna be .	**MMI**	i don 't know what the hell they 're talking about but they don 't care about women .
FP (λ=1)	that 's why they come here .	**FP** (λ=5)	they 're a bunch of wild savages ...
Input	there 's no tape inside anyway !		
MLE	no no no no no no no no no no no no no no no no no .	**MMI**	no no no no no no no no no no no no no no no no no !
FP (λ=1)	where did he hide the tape ?	**FP** (λ=5)	i don 't think i should take you to the movies !
Input	supervisor tang they are starting to quarrel		
MLE	why don 't you come with me ?	**MMI**	why don 't you go back to your room and have a chat with them
FP (λ=1)	this is ridiculous !	**FP** (λ=5)	i don 't care !
Input	we 're almost out of time .		
MLE	there 's no need to worry about it .	**MMI**	i 'll be right back
FP (λ=1)	there 's no need to panic .	**FP** (λ=5)	we 've got to get back to the ship before it 's too late ..
Input	get your fat ass over there !		
MLE	get the fuck out of here !	**MMI**	get the fuck out of my face !
FP (λ=1)	get the fuck out of my face !	**FP** (λ=5)	don 't you get tired of this man 's fucking shit !
Input	oh my that looks yummy lily .		
MLE	i don 't think it 's funny .	**MMI**	i don 't think it 's a good idea to wear a wedding dress .
FP (λ=1)	that 's a nice dress .	**FP** (λ=5)	i thought we agreed to have dinner together with your wife and your daughter and ...
Input	our scout ships have reached UNknown		
MLE	you know what you doing, right ?	**MMI**	there 's no sign of the rebel fleet ..
FP (λ=1)	the rebel ships are still in orbit .	**FP** (λ=5)	they 're still in orbit in the alpha quarter of space ...
Input	the UNknown order is chaos man .		
MLE	it 's the only way to get rid of it .	**MMI**	it 's the only way to get rid of it .
FP (λ=1)	it 's just the beginning .	**FP** (λ=5)	it 's the rules of the country
Input	but you are not a mistake .		
MLE	i don 't know who you are or what you are.	**MMI**	i don 't know who you are or what you are or what you are .
FP (λ=1)	i don 't know who you are .	**FP** (λ=5)	you have no right to give your life for the cause of the mistake of our enemy ...
Input	heaven and earth must i remember ?		
MLE	i don't know what you are talking about	**MMI**	there 's no such thing as ghosts .
FP (λ=1)	there 's no need to be afraid of the dark .	**FP** (λ=5)	there 's only one place where we can see it
Input	we got a murder down here ...	**SBS**	i don 't know what the hell 's goin on here .
MMI	i don 't know what the hell 's goin on here .	**FP** (λ=1)	i don 't know who he is ...
FP (λ=5)	we gotta do something about this ...		
Input	but venus was no easy target .		
SBS	there 's no need to worry about it .	**MMI**	it was the only way to survive .
FP (λ=1)	i don 't know what the hell you 're talking about ...	**FP** (λ=5)	there 's a whole bunch of them in the middle east

Table 8: Sampled examples from responses generated by (1) standard beam search using MLE objective (denoted by *MLE*); (2) the MMI reranking model (denoted by *MMI*) in (Li et al., 2016a) that perform reranking only after decoding is fully done; (3) the proposed partial-seq-MMI model (denoted by *FP*) with different values of future prediction weight λ.