Digging Errors in NMT: Evaluating and Understanding Model Errors from Hypothesis Distribution

Anonymous ACL submission

Abstract

Sound evaluation of a neural machine translation (NMT) model is key to its understanding and improvement. Current evaluation of an NMT system is usually built upon a heuristic decoding algorithm (e.g., beam search) and an evaluation metric assessing similarity between the translation and golden reference (e.g., BLEU). However, this system-level evaluation framework is prone to its evaluation over only one best hypothesis and search errors brought by heuristic decoding algorithms. To better understand NMT models, we propose a novel evaluation protocol, which defines model errors with hypothesis distribution. In particular, we first propose an exact top-k decoding algorithm, which finds top-ranked hypotheses in the whole hypothesis space and avoids search errors. Then, we evaluate NMT model errors with the distance between hypothesis distribution with the ideal distribution, aiming for a comprehensive interpretation. We apply our evaluation on various NMT benchmarks and model architectures to provide an in-depth understanding of how NMT models work. We show that the state-of-the-art Transformer models are facing serious ranking errors and do not even outperform the random chance level. We further provide several interesting findings over data-augmentation techniques, dropouts, and deep/wide models. Additionally, we analyze beam search's lucky biases and regularization terms. Interestingly, we find these lucky biases decrease when increasing model capacity.

1 Introduction

Recent sequence-to-sequence (Seq2Seq) models (Sutskever et al., 2014; Vaswani et al., 2017) have shown promising results in neural machine translation (NMT), where methods typically frame a conditional probability distribution from a source sentence to a target sentence.

One key to the booming of neural machine translation is the sound evaluation, which shows the trajectory to a better model design and architecture. The commonly used evaluation protocol of an NMT system comprises two main components: a search algorithm and an evaluation metric. The algorithm is responsible for decoding a translated sentence, and the metric computes the discrepancy between the generated translation and the reference.

The above evaluation paradigm is widely used in NMT. It assumes that the gap between an NMT model and the ideal model can be depicted by the gap between decoded translations and references. However, this assumption does not always hold. Recent literature (Stahlberg and Byrne, 2019; Meister et al., 2020) points out that search errors brought by heuristic decoding methods would hide huge flaws of NMT models (model errors), such as empty string is commonly scored with highest probability among model's probabilities over all hypotheses, i.e., hypothesis distribution. It is essential to evaluate model errors without being interfered by search errors.

Those approaches perform well, but they only take the *mode*[1] hypothesis of hypothesis distribution to evaluate the term of model errors, which is not comprehensive in only considering one hypothesis. In contrast, we want to understand model errors in the hypothesis distribution-level:

- *Q1:What are the most crucial errors that a model's hypothesis distribution is facing?*
- *Q2:How do these model errors connect with current search algorithms?*

To answer these questions, we introduce a new *distribution-level evaluation* of model errors. The decoding and evaluation of model errors need to fit the requirements of the *distribution-level evaluation* of an NMT model. For the decoding algorithm, it should be both exact (not affected by search errors) and able to access the top-ranked hypotheses

[1]Mode is the hypothesis with the highest probability in a distribution.

given by hypothesis distribution. For the evaluation, it is essential to identify how good or bad these top hypotheses are quantitatively. Particularly, we propose an exact top-k decoding algorithm that not only avoids search errors but can access the top-ranked region of the whole hypothesis space. Furthermore, we provide formal definitions of *distribution-level evaluation*, and, to deal with infinite search space, present a computationally viable approach to evaluate an NMT model's hypothesis-ranking (HR) ability, based on our exact top-k decoding.

Extensive experiments are conducted over three different machine translation benchmarks with small, medium, and large sizes. We find that the state-of-the-art Transformer models have weaker hypothesis ranking abilities than that of the random chance level. We further provide several interesting findings over data-augmentation techniques, dropouts, and deep/wide models. In addition, we connect model errors with two crucial factors of beam search algorithms: beam search lucky biases and regularization terms. Interestingly, we find those lucky biases decrease with the increase of model capacity.[2]

Our contributions can be summarized as follows:

- To the best of our knowledge, we are the first to propose and provide formal definitions of distribution-level evaluation protocol without the interfere of search errors.
- We provide a realization of distribution-level model errors with a newly proposed exact top-k decoding method and hypothesis-ranking (HR) based evaluation.
- On errors that NMT models are facing (Q1), we conduct in-depth analysis over various NMT techniques and find that the state-of-the-art Transformer models face severe hypothesis-ranking problems with abilities weaker than the random chance level.
- On connection with search algorithms (Q2), we analyze the search algorithm in lucky biases and regularization terms. Interestingly, we find these lucky biases decrease when increasing model capacity.

2 Definitions

In this section, we introduce the formal definitions of *system-level*, *mode-level* and *distribution-level evaluations*.

[2]Codes will be released upon acceptance.

2.1 NMT Model and Hypothesis Space

Give an NMT model M, a source sentence x and a reference sentence $\hat{y}$. Most of the NMT models are auto-regressive models, which define a conditional distribution for a hypothesis y_i as:

$$P(y_i|x) = \prod_{t\in(1,T)} P(y_i^t|x; y_i^{1:t-1}), \\ = M(x, y_i), \quad (1)$$

where t represents the time step on target side and T is the total length of y_i.

The hypothesis space of M is defined as the set of all hypotheses given by M,

$$\mathcal{Y} = \{y,\ \forall P(y_i|x) > 0\}, \quad (2)$$

and we refer to $\mathcal{Y}$ as M's **hypothesis space**. We have the hypothesis distribution of an NMT model over its hypothesis space as,

$$P(\mathcal{Y}|x) = \{P(y_i|x),\ \forall y_i \in \mathcal{Y}\}. \quad (3)$$

2.2 System-level Evaluation

Given a decoding algorithm F and an evaluation metric like BLEU (Papineni et al., 2002), the system-level evaluation of NMT system usually proceeds by first decoding a hypothesis y' from the hypothesis space.

$$y' = F(\mathcal{Y}, P(\mathcal{Y})), \quad (4)$$

where F usually selects one or a few translation(s) with the highest step-by-step conditional probabilities from hypothesis space because of the auto-regressive computation order. Next, we evaluate the similarity measure for y' and reference $\hat{y}$.

$$S_{\text{system}} = \text{Score}(\hat{y}, y'). \quad (5)$$

2.3 Mode-level Evaluation

It is recognized in previous literature (Niehues et al., 2017; Stahlberg et al., 2018; Stahlberg and Byrne, 2019; Meister et al., 2020) that evaluating an NMT model and the decoding method as a whole system hinders our understanding of true NMT model errors. Therefore, Stahlberg and Byrne (2019) proposes an exact decoding method that finds the top-1 hypothesis y_m over hypothesis distribution (mode) to evaluate model errors.

$$y_m = \text{argmax}_{y\in\mathcal{Y}}(P(\mathcal{Y})), S_{\text{me}} = \text{Score}(\hat{y}, y_m). \quad (6)$$

They find empty strings usually appear to be the modes of distributions and use the empty rate of modes to quantify the model errors.

Concretely, the rate of empty modes is defined by checking $y_m =$ "<EOS>". We call this paradigm the mode-level evaluation in the following sections.

2.4 Distribution-level Evaluation

Evidently, selecting only one hypothesis in the whole hypothesis space loses much information of the hypothesis distribution and makes the evaluation biased. Alternatively, we define the distribution-level evaluation which directly deals with the NMT model's hypothesis distribution over $\mathcal{Y}$, by computing the distance between $P(\mathcal{Y})$ and ideal distribution $P_{\text{ideal}}(\mathcal{Y})$:

$$S_{\text{dist}} = \text{Score}(P(\mathcal{Y}), P_{\text{ideal}}(\mathcal{Y})). \quad (7)$$

Providing a sound definition to the ideal distribution of an NMT model is non-trivial. Here we mainly model one key attribute of the ideal distribution, which we call the *hypothesis ranking ability*. Intuitively, the ideal model's distribution over hypothesis space should align with the translation qualities over the hypothesis space. In particular, if the translation quality of a specific hypothesis translation y_i is better than that of y_j, the model's probability over y_i should also be higher than that over y_j.

$$P(y_i|x) > P(y_j|x) \text{ if } Q(y_i) > Q(y_j) \quad \forall y_i, y_j \in \mathcal{Y}, \quad (8)$$

where $Q(y_i)$ is the translation quality function (e.g., BLEU), and short for $Q(\hat{y}, y_i)$.

Hence, by extending such ability from pairwise to all hypotheses of a source sentence x, we define a proxy for ideal distribution using the perfectly ordered hypothesis array of which the indices are sorted by translation quality. Formally, we define a perfect hypothesis-level ranking (HR) array $\boldsymbol{\mathcal{Y}}_{\text{HR}}$ over the hypothesis space $\mathcal{Y}$ with,

$$\boldsymbol{\mathcal{Y}}_{\text{HR}} = [y_{I^0_{\text{HR}}}, y_{I^1_{\text{HR}}}, \cdots, y_{I^n_{\text{HR}}}]; \quad (9)$$

$$I_{\text{HR}} = \text{argsort}([Q(y_1), \cdots, Q(y_n)]). \quad (10)$$

The model's array produced by hypothesis distribution can be collected by sorting probabilities,

$$\boldsymbol{\mathcal{Y}}_{\text{M}} = [y_{I^0_{\text{M}}}, y_{I^1_{\text{M}}}, \cdots, y_{I^n_{\text{M}}}]; \quad (11)$$

$$I_{\text{M}} = \text{argsort}([P(y_1|x), \cdots, P(y_n|x)]). \quad (12)$$

Thus, we can now define the distribution-level model errors with the distance between these two sorted arrays,

$$S_{\text{dist}} = \text{D}(\boldsymbol{\mathcal{Y}}_{\text{HR}}, \boldsymbol{\mathcal{Y}}_{\text{M}}), \quad (13)$$

where D is a certain distance function.

2.5 Relationship between Mode and Distribution

In this section, we discuss the relationship between mode-level and distribution-level model error evaluation. We show that the empty rate is a special case of our distribution-level evaluation.

On the one hand, given any matching-based translation quality function Q, the empty output should always be scored 0 and ranked to the last of $\boldsymbol{\mathcal{Y}}_{\text{HR}}$. On the other hand, y_m is the mode of hypothesis distribution and should rank the first in $\boldsymbol{\mathcal{Y}}_{\text{M}}$.

$$y_m = \boldsymbol{\mathcal{Y}}_{\text{M}}[0]; \; y_{emp} = \boldsymbol{\mathcal{Y}}_{\text{HR}}[n-1]. \quad (14)$$

Then, the corresponding distance function D in (13) is:

$$\text{D}_{\text{empty}} = \begin{cases} 1 & \boldsymbol{\mathcal{Y}}_{\text{M}}[0] = \boldsymbol{\mathcal{Y}}_{\text{HR}}[n-1] \\ 0 & \text{else} \end{cases}, \quad (15)$$

As shown, the rate of empty modes is a special case of our distributional modeling.

3 Our Proposed Evaluation

Previous distribution-level definitions are powerful and promising. Yet, the realization of distributional evaluation is non-trivial due to the exponentially large search space of an NMT model. In the following sections, we provide our exact top-k decoding method, which helps find the topmost hypotheses over the whole hypothesis space, and describe how to perform such distribution-level evaluation based on the exact decoding algorithm.

3.1 Exact Top-k Decoding

As mentioned above, obtaining whole hypothesis space is intractable due to the exponentially large search space. Therefore, one reasonable approximation is to focus more on hypotheses with the highest probabilities. Luckily, we care about these hypotheses the most in real-world NMT applications.

We extended the exact decoding algorithm (Stahlberg and Byrne, 2019) and propose a top-k DFS-based exact decoding algorithm (Algorithm

1). Our decoding method is guaranteed to find the exact top-k hypotheses from the model's hypothesis distribution. Particularly, we traverse the search space of an NMT model in depth-first order. We enumerate all tokens in the vocabulary at each search step and concatenate them with the current history as the next possible translation prefixes. During the search process, we keep track of the current top-k hypotheses that we find. Specifically, a minimum heap is used to maintain current top-k hypotheses during the search procedure. The hypothesis with the lowest score in the minimum heap dynamically update our lower bound during searching: Once we find a newly finished hypothesis (i.e., ended with </s>), we push the hypothesis into the heap and make adjustments to retain the heap size equals k. Then, we update the lower bound and truncate decoding paths. Finally, the hypotheses stayed in the minimum heap are returned. We use beam search result as the initial bound of the search space and sort the vocabulary before enumeration for a faster update of lower bounds.

It is not easy to make top-k exact search tractable under the condition of modern CPU/GPU architectures. We devote many efforts to implementation for making the search time under the acceptable line, and sometimes our method is even faster than the original DFS algorithm implementation (Stahlberg and Byrne, 2019). The implementation details and computational cost analysis can be found in Appendix B.

ALGORITHM 1: DFS-based Top-k Exact Search.

Input : x: Source sentence, y: Translation prefix (default: []), p: $\log P(y|x)$ (default 0.0), k: Top-k hypotheses to output, V: Vocabulary.
, Output : List l contains top-k hypotheses with log-probabilities.

```
global minHeap
global γ ← −inf
Function dfsTopK(x, y, p):
    if y[|y| − 1] =< /s > then
        push(minHeap, (p, y))
        if len(minHeap) > k then
            pop(minHeap)
        end
        if len(minHeap) = k then
            γ ← minHeap[0][0]
        end
    end
    for v ∈ V do
        p′ ← p + log P(v|x, y)
        if p′ ≥ γ then
            dfsTopK(x, [y; v], p′)
        end
    end
    return minHeap
return dfsTopK(x, [], 0.0)
```

3.2 HR-based Model Errors

Here, we present our HR-based model evaluation. As discussed above, getting the complete HR ranked array $\mathcal{Y}_{\text{HR}}$ and model ranked array $\mathcal{Y}_{\text{M}}$ are both intractable. Our evaluation has to rely on approximation.[3] Luckily, in real-world NMT models, we prioritize top-ranked hypotheses. Thus we define our evaluation over these top-ranked hypotheses found by our exact top-k. Formally, we define a truncated model array:

$$\tilde{\mathcal{Y}_{\text{M}}} = \mathcal{Y}_{\text{M}}[0:k];\ \tilde{I}_{\text{M}} = I_{\text{M}}[0:k], \tag{16}$$

where k denotes how many top-ranked hypotheses we consider. For $\mathcal{Y}_{\text{HR}}$, we select hypotheses appeared in $\mathcal{Y}_{\text{M}}$ to form a local HR array $\tilde{\boldsymbol{Y}_{\text{HR}}}$,

$$\tilde{\mathcal{Y}_{\text{HR}}} = [y_{\tilde{I}^0_{\text{HR}}}, y_{\tilde{I}^1_{\text{HR}}}, \cdots, y_{\tilde{I}^k_{\text{HR}}}], \tag{17}$$

$$\tilde{I}_{\text{HR}} = \text{argsort}([Q(y_{\tilde{I}^0_{\text{M}}}), \cdots, Q(y_{\tilde{I}^k_{\text{M}}})]). \tag{18}$$

For ranking distance D, we provide two distance functions here. Firstly, we propose an extended version of nDCG (Järvelin and Kekäläinen, 2002), Top**k** **R**anked **G**ains (**kRG**):

$$\text{kRG}(\mathcal{Y}_{\text{HR}}, \tilde{\mathcal{Y}_{\text{M}}}) = \frac{\text{DCG}(\tilde{\mathcal{Y}_{\text{M}}})}{\text{DCG}(\tilde{\mathcal{Y}_{\text{HR}}})}, \tag{19}$$

$$\text{DCG}_k(\mathcal{Y}) = \sum_{y_j \in \mathcal{Y}[0:k]} \frac{2^{f(y_j)}}{\log_2(j+1)}, \tag{20}$$

$$f(y_j) = k - \text{Rank}(y_j, \mathcal{Y}), \tag{21}$$

where $f(y_j)$ denotes the relevance score of a certain ranked hypothesis. kRG directly measures the ranking of a model's top-k hypotheses, where 0 means a completely wrong ranking and 1 means a perfect ranking.

Next, in concern of translation quality of selected top-k hypotheses, we further propose Top**k** **Q**uality-based **R**anked **G**ains (**kQRG**):

$$\text{kQRG}(\mathcal{Y}_{\text{HR}}, \tilde{\mathcal{Y}_{\text{M}}}) = \frac{\text{DCG}_{qk}(\tilde{\mathcal{Y}_{\text{M}}})}{\text{DCG}_{qk}(\mathcal{Y}_{\text{HR}})} \tag{22}$$

$$\text{DCG}_{qk}(\mathcal{Y}) = \sum_{y_j \in \mathcal{Y}[0:k]} 2^{\text{Q}(y_j)} D(j), \tag{23}$$

[3] We provide another interesting approach with edit-distance as the quality function in the Appendix.

Name	Train	Dev	Test	BPE
NIST Zh-En	1.2M	1664	5105	40K/30K
WMT'14 En-De	4.5M	3000	3003	32K
WMT'14 En-Fr	35.7M	6003	3003	40K

Table 1: Statistics of Datasets

where we replace relevance score with translation quality $Q \in [0, 1]$ and normalize over $\mathcal{Y}_{\text{HR}}$. We approximate $\text{DCG}_{qk}(\mathcal{Y}_{\text{HR}})$ with its upper-bound:

$$\text{DCG}_{qk}(\mathcal{Y}_{\text{HR}}) = \sum_{y_j \in \mathcal{Y}_{\text{HR}}[0:k]} 2^{\text{Q}(y_j)} D(j) \quad (24)$$

$$<= \sum_{j \in [0:k]} 2^{1.0} D(j). \quad (25)$$

kQRG consider both how the topmost hypotheses ranked and whether these hypotheses have good translation quality, where 0 represents a bad ranking with totally wrong translation, and 1 represents a perfect ranking and a perfect collection of top-ranked hypotheses.

4 Experiments and Findings

Setups. All experiments are conducted over three commonly used NMT benchmarks, NIST Chinese-English, WMT'14 English-German, and WMT'14 English-French with small, medium, and large sizes. The statistics of datasets are presented in Table 1. Detailed pre-processing steps can be found in Appendix A.

Training and Evaluation. Our models are trained using the *fairseq* toolkit[4]. We train each of our Transformer models for 100k/300k/300k steps for three datasets and evaluate each model for 5000 steps. The default label smoothing is 0.1. The dropout rates for different Transformer models range from 0.1 to 0.4. The batch sizes are 8k/64k/64k tokens for three datasets. All our Transformer models are pre-norm models. Other hyper-parameter settings are the same as in (Vaswani et al., 2017). For evaluation, we report case-sensitive tokenized BLEU scores using *multi-bleu.perl*[5] for both WMT'14 En-De and En-Fr, and case-insensitive tokenized BLEU scores for NIST Chinese-English. We select the best checkpoint on the validation set and report its performance on the test set. All reported results are averaged over all sentences in the test set. For results with beam search, the beam size is 5, and the length penalty is 0.6. For all results with distribution-level metrics, our translation quality function is the sentence-bleu method provided by sacrebleu (Post, 2018), and we use floor smooth with 0.01 as floor value. Other metrics like METEOR can also be used. By default, we use top-10 hypotheses for approximation in experiments.

[4] https://github.com/pytorch/fairseq

[5] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/generic/multi-bleu.perl

Method	System	Mode	HR-Based	
	BLEU	# Emp	kRG	kQRG
Transformer	27.22	64.70	50.33	10.50
w/o LS	26.76	34.85	51.62	13.44
w/ para BT	27.36	27.26	51.70	14.05
w/ para FT	28.06	**0.93**	**53.76**	**18.70**
w/ dropout 0.2	27.81	61.74	50.30	11.48
w/ dropout 0.3	27.25	58.81	50.59	11.10
w/ 12-layer Enc	27.75	58.11	50.84	11.30
w/ 18-layer Enc	28.03	53.58	51.55	12.02
w/ Dim 768	28.00	50.18	51.56	11.77
w/ Dim 1024	**28.49**	44.72	51.58	12.81

Table 2: Model errors of different models in WMT'14 En-De task. All numbers range in [0, 100%]. 'para BT' and 'para FT' denote back-translation and forward-translation over parallel golden data, and LS denotes label smoothing.

Method	BLEU	# Emp	kRG	kQRG
Transformer	40.78	46.75	52.50	18.27
w/o LS	40.70	**19.51**	54.17	23.47
w/ para FT	40.95	27.26	**55.76**	**26.99**
w/ 12-layer Enc	41.28	44.99	52.81	18.84
w/ 18-layer Enc	41.74	53.58	52.93	19.18
w/ Dim 768	41.73	46.12	53.08	19.07
w/ Dim 1024	**42.35**	40.42	53.24	20.01

Table 3: Model errors of different models in WMT'14 En-Fr task. All numbers range in [0, 100%].

4.1 Findings on NMT Techniques

Table 2, 3, 4 demonstrate the results for different Transformer-based models in WMT'14 En-De, WMT'14 En-Fr and NIST Zh-En tasks respectively. We make following observations:

Method	BLEU	# Emp	kRG	kQRG
Transformer	42.47	41.14	51.31	11.33
w/o LS	42.44	**14.59**	**53.56**	16.51
w/ para FT	42.17	17.52	52.99	**18.09**
w/ 12-layer Enc	43.38	36.24	51.49	12.48
w/ 18-layer Enc	43.81	43.11	51.41	11.79
w/ Dim 768	42.88	40.76	51.61	13.23
w/ Dim 1024	**43.43**	34.03	52.13	14.05

Table 4: Model errors of different models in NIST Chinese-English task. All numbers range in [0, 100%].

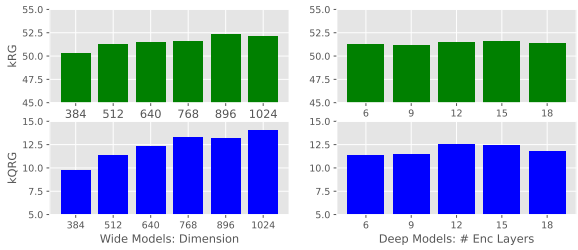


Figure 1: Results on NIST with wide and deep models.

I. Failure of mode evaluation. Let us take a look at the empty rates, the evaluation for model errors proposed in previous literature. We find that removing label smoothing, adding pseudo-parallel data will drastically decrease the number of empty rates, even close to 0 ("*para FT*"), indicating an almost perfect model with tiny model errors. However, it is not the case. Our kRG and kQRG results indicate that the model still has much to improve (to 100%). These demonstrate that mode-level evaluation collapses when evaluating certain models. In contrast, our evaluation can consistently evaluate these models and is consistent with human judgments (See Appendix D), showing the superiority of our evaluation.

II. The State-of-the-art Transformer models face serious ranking problem. Among all models, our kRG results range from about $50\% \sim 56\%$. To further interpret these results, we take 100k trails of random permutation using the same scoring function, and the kRG is **58.72%**. It indicates that even though our Transformer models show some ranking ability, SOTA Transformer models perform worse than the random chance level in terms of the ranking abilities. When including the translation quality into accounts, i.e., kQRG, we find the distribution-level scores only range from $10\% \sim 20\%$ for En-De, $18\% \sim 27\%$ for En-Fr, and $11\% \sim 18\%$ for Zh-En. It shows that Transformer models' hypothesis distributions are deficient. The models give high probabilities to low-quality translations and face severe model errors.

III. Widening models are more effective in reducing model errors. Recently, many interests have been drawn for using deeper models instead of wider models to increase model capacity. However, as shown in Figure 1 and tables, widening models is still more effective in improving both kRG and kQRG, demonstrating better abilities of ranking and selection.

IV. Different dropout values affect model errors slightly. NMT models tend to be sensitive to hyper-parameter settings like dropout values in system-level evaluation. Here, we evaluate models with dropout $\{0.1, 0.2, 0.3\}$ in Table 2 and find that kRG and kQRG are only slightly affected by various dropouts. A potential reason is that some empirically better choices in hyper-parameters, e.g., suitable dropout values, may only be over-fittings to the search process (e.g., beam search).

V. Model confidence may be crucial to boosting the model's distribution-level performance. Results show that the models trained using parallel and forward translation data show impressive improvements in both HR metrics, e.g., +3.43 and +8.20 for En-De. Nonetheless, it has similar BLEU performance with beam search compared with our deeper model (*w/ 18-layer Encoder*) and wider model *w/ Dim 768*. In this case, system-level evaluation fails to capture decent improvements over the model's distribution over candidate space. As forward-translation training and disabling label-smoothing generally enhance the model confidence over beam search candidates, we conjecture that model errors are highly related to model confidence and leave the exploration as future work.

4.2 Connection with Beam Search

4.2.1 Lucky Bias and Model Errors

As pointed out in recent work (Meister et al., 2020), beam search seems to bring a lucky bias that covers some of the model errors. In this section, we utilize the HR-based evaluation to understand the bias brought by beam search.

Concretely, we use HR-based evaluation to evaluate the errors from both exact top-k and beam

Method	Distribution		Beam Search	
	kRG	kQRG	kRG	kQRG
6-layer	51.31	11.33	61.36$^{(+10.05)}$	23.58$^{(+12.25)}$
9-layer	51.11	11.49	61.73$^{(+10.62)}$	23.84$^{(+12.36)}$
12-layer	51.49	12.48	61.69$^{(+10.20)}$	24.12$^{(+11.64)}$
15-layer	51.56	12.41	61.57$^{(+10.00)}$	24.37$^{(+11.96)}$
18-layer	51.41	11.79	61.11$^{(+9.70)}$	24.29$^{(+12.49)}$

Table 5: HR-based evaluation over distributional top-10 outputs and beam top-10 outputs when increasing number of encoder layers.

Method	Distribution		Beam Search	
	kRG	kQRG	kRG	kQRG
D384	50.28	9.78	61.85$^{(+11.57)}$	22.97$^{(+13.19)}$
D512	51.31	11.33	61.36$^{(+10.05)}$	23.58$^{(+12.25)}$
D640	51.50	12.29	61.90$^{(+10.40)}$	23.90$^{(+11.61)}$
D768	51.61	13.23	61.35$^{(+9.74)}$	23.85$^{(+10.62)}$
D896	52.33	13.15	60.07$^{(+7.74)}$	23.83$^{(+10.68)}$
D1024	52.13	14.05	61.66$^{(+9.53)}$	24.10$^{(+10.05)}$

Table 6: HR-based evaluation over distributional top-10 outputs and beam top-10 outputs when increasing model's dimension size.

search k outputs. In this way, the gap between two errors provides a quantitative method to evaluate the lucky bias brought by beam search. Experiments are conducted in NIST Zh-En and results are shown in Table 5 and 6. We find that beam search indeed provides the inductive bias that improves models by reducing model errors. For different models, the beam search outputs improves **kRG** with $7\% \sim 11\%$ and **kQRG** with $10\% \sim 13\%$, which shows a better hypothesis ranking ability. Since models do not change, these findings prove that the inductive bias of beam search filters out some hypotheses with low quality or assigned wrong probabilities by NMT models. Additionally, we notice one interesting fact that **with the increases of model capacity, the lucky bias brought by beam search decreases**. This provides a clue about when these lucky biases would disappear.

4.2.2 Understanding Search Regularization

Recent research has reported that regularization terms are essential in the success of beam search algorithms. Regularization terms, like length penalty (Bahdanau et al., 2014; Wu et al., 2016), UID penalties (Meister et al., 2020), typically modify the log-probability produced by model. In this section, we study how these regularization terms (or called penalties) affect model errors.

#	Method	kRG	kQRG
	Transformer(base)	50.33	10.50
1	w/ length normalization	**67.27**	**11.83**
2	w/ max regularization	62.88	11.79
3	w/ square regularization	63.85	11.82
4	w/ variance regularization	64.53	**11.83**
5	w/ greedy regularization	63.14	11.78

Table 7: Model errors computed by top-100 hypotheses in hypothesis space reranked using regularization terms, in WMT'14 En-De. All numbers range in [0, 100%.]

These regularizations of beam search are commonly considered as inductive biases to help beam search avoid errors. As a result, these regularization terms are commonly regarded as fixes for the beam search algorithm. Nonetheless, we argue that these regularization terms substantially improve the model's ranking capability and can be evaluated using the proposed HR-based model error metrics. Recall our definition of the model's ranking array defined in Equation 11. Then, a specific regularization term $\mathcal{R}(y)$ changes model rankings to:

$$\mathcal{Y}_{\text{MR}} = [y_{I^0_{\text{MR}}}, y_{I^1_{\text{MR}}}, \cdots, y_{I^n_{\text{MR}}}], \quad (26)$$

$$I_{\text{MR}} = \text{argsort}([\log P(y_1|x) + \mathcal{R}(y_1), \cdots, \log P(y_n|x)] + \mathcal{R}(y_n)), \quad (27)$$

Accordingly, we can evaluate search penalties with kRG and kQRG. In particular, we use search penalties to rerank the exact search outputs for the Transformer model, to see how search penalties change rankings and overall model errors. For search penalties, we choose length normalization (Wu et al., 2016) and UID regularizations (Meister et al., 2020).

As shown in Table 7, all regularization terms substantially improve the model errors by a strong margin, from 50.33 to 62+ in kRG and from 10.50 to 11.79+ in kQRG. We find that length normalization (1) performs the best among all terms among all penalties, which proves length bias is an important issue in model errors. The UID terms (2-5) have lower performance in the ranking ability (kRG) but get similar results in kQRG, compared with length normalization, demonstrating that UID terms improve in an orthogonal direction compared with length norm term.

5 Related Work

Decoding Methods. Most decoding methods in NMT aims to find the hypothesis with the highest conditional probability. This is called the maximum-a-posterior (MAP) decoding algorithm. Among all MAP decoding methods, beam search is the most widely applied method in the modern NMT systems for evaluation, which utilizes a fixed beam size for each decoding step. Naive beam search with log probabilities has several known drawbacks, such as favoring short translations and its monotonic constraint. Hence, many regularization/rescoring methods (Bahdanau et al., 2014; Wu et al., 2016; He et al., 2016; Yang et al., 2018; Murray and Chiang, 2018) or beam search variants (Freitag and Al-Onaizan, 2017; Shu and Nakayama, 2018) are proposed to improve the actual performance. Other than beam search, one promising MAP decoding technique for evaluation is the DFS-based exact search (Stahlberg and Byrne, 2019), which is designed to find the mode of model distributions. Despite its high computational cost, it reveals important information about model distributions. We follow this approach and present a top-k exact search method, which can access the top-region of hypothesis distribution.

In addition, there are some non-MAP decoding algorithms. A typical one is the stochastic sampling-based decoding methods (Ackley et al., 1985; Holtzman et al., 2019), which randomly choose candidates from each step's output distribution. Eikema and Aziz (2020) introduces a Minimum Bayesian Risk decoding method based on sampling. Leblond et al. (2021) propose a metric-driven search approach via Monte-Carlo Tree Search (MCTS). These approaches are promising and may incorporate with our distribution-level evaluation in future directions.

Error Evaluation. Evaluation of NMT errors focuses on studying the gap between machine-translated results and human-translated references. Statistical matching metrics, such as BLEU, METEOR (Papineni et al., 2002; Banerjee and Lavie, 2005; Koehn et al., 2007; Denkowski and Lavie, 2014; Guo and Hu, 2019), are dominant in evaluating errors. These metrics prove that linguistic similarity between references and machine translations correlates the human evaluation well. However, to the best of our knowledge, these statistical metrics evaluate one best hypothesis decoded from heuristic decoding algorithm (i.e., system-level evaluation), which incorporate huge search errors and bias understanding of NMT models.

Recent efforts (Niehues et al., 2017; Stahlberg et al., 2018; Stahlberg and Byrne, 2019; Meister et al., 2020; Eikema and Aziz, 2020) are devoted to analyzing model errors without search errors and provide meaningful conclusions. Nonetheless, these approaches still evaluate over one hypothesis in hypothesis distribution except with the one with highest probability. This is incomprehensive due to neglecting errors inside the whole hypothesis distribution. In contrast, we dig into model errors over distribution-level and provide a comprehensive evaluation. In addition, we provide various interesting findings over model errors with regards to NMT techniques and search algorithms.

6 Conclusion and Future Work

This paper presents a novel distribution-level evaluation protocol for model errors. Specifically, we propose a new exact top-k decoding algorithm and evaluate NMT model errors with the distance between hypothesis distribution and ideal distribution. Our evaluation protocol helps understand current NMT systems from two perspectives: 1) various NMT techniques, and 2) search algorithms, which together comprise the backbone of NMT systems. With experiments over various NMT benchmarks and architectures, we prove the effectiveness of our evaluation on model errors, and provide understandings over commonly used NMT techniques such as data augmentation, dropouts, increasing model capacity, etc. In addition, we demonstrate that current NMT models underperform in terms of hypothesis ranking ability. Regarding beam search, we mainly investigate lucky biases and search penalties and show that the lucky biases decrease when the model capacity increases. Furthermore, the experimental results provide a clue about when these biases would disappear. Finally, we prove that search penalties can help rank hypotheses correctly.

From our point of view, it is essential to understand NMT models without search errors, and we believe that is one of the future directions of NMT. Researchers can save a large amount of time in tuning models where beam search blessing does not necessarily exist. It is valuable if the decoding algorithms are exact as well, to align with the evaluation properly. We plan to investigate a faster exact decoding algorithm or approximation algorithm that can be deployed in NMT systems.

References

David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169.

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

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pages 376–380.

Bryan Eikema and Wilker Aziz. 2020. Is map decoding all you need? the inadequacy of the mode in neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4506–4520.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 56–60.

Yinuo Guo and Junfeng Hu. 2019. Meteor++ 2.0: Adopt syntactic level paraphrase knowledge into machine translation evaluation. In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 501–506.

Wei He, Zhongjun He, Hua Wu, and Haifeng Wang. 2016. Improved neural machine translation with smt features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Michael D Hendy and David Penny. 1982. Branch and bound algorithms to determine minimal evolutionary trees. *Mathematical Biosciences*, 59(2):277–290.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–180.

Rémi Leblond, Jean-Baptiste Alayrac, Laurent Sifre, Miruna Pislar, Jean-Baptiste Lespiau, Ioannis Antonoglou, Karen Simonyan, and Oriol Vinyals. 2021. Machine translation decoding beyond beam search. *arXiv preprint arXiv:2104.05336*.

Alan Mackworth. 2013. Lecture notes in introduction to artificial intelligence.

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020. If beam search is the answer, what was the question? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2173–2185, Online.

Kenton Murray and David Chiang. 2018. Correcting length bias in neural machine translation. *arXiv preprint arXiv:1808.10006*.

Jan Niehues, Eunah Cho, Thanh-Le Ha, and Alex Waibel. 2017. Analyzing neural MT search and model performance. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 11–17, Vancouver.

Mohammad Norouzi, Samy Bengio, Navdeep Jaitly, Mike Schuster, Yonghui Wu, Dale Schuurmans, et al. 2016. Reward augmented maximum likelihood for neural structured prediction. *Advances In Neural Information Processing Systems*, 29:1723–1731.

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

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany.

Raphael Shu and Hideki Nakayama. 2018. Improving beam search by removing monotonic constraint for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 339–344.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proceedings of the 2019 Conference on Empirical*

Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 3356–3362, Hong Kong, China.

Felix Stahlberg, Danielle Saunders, Gonzalo Iglesias, and Bill Byrne. 2018. Why not be versatile? applications of the SGNMT decoder for machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 208–216, Boston, MA. Association for Machine Translation in the Americas.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27:3104–3112.

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.

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*.

Yilin Yang, Liang Huang, and Mingbo Ma. 2018. Breaking the beam search curse: A study of (re-) scoring methods and stopping criteria for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3054–3059.

A Experimental Details

A.1 Detailed Descriptions of Datasets

For our WMT'14 En-De/En-Fr tasks, we use 4.5M / 35.7M preprocessed data, which is tokenized and split using byte pair encoded (BPE) (Sennrich et al., 2016) with 32K/40K merge operations and a shared vocabulary for source and target sides. For En-De, we use *newstest2013* as the validation set and *newstest2014* as the test set. For En-Fr, we use the combination of *newstest2012* and *newstest2013* as our validation set and *newstest2014* as the test set.

For the NIST Zh-En task, we use 1.25M sentences extracted from LDC corpora[6]. To validate the performance of our model, we use the NIST 2006 (MT06) test set with 1664 sentences as our validation set. Then, the NIST 2002 (MT02), 2004 (MT04), 2005 (MT05), 2008 (MT08) test sets are used as our test sets, which contain 878, 1788, 1082, and 1357 sentences, respectively. All reported results are averaged over different test sets.

B Implementation Details of Exact Top-k

Here we explain the implementation details of our exact top-k algorithm. The detailed algorithm is shown in Algorithm 2. Our implementation is built upon *uid-decoding*[7] and *sgnmt*[8] projects, and is compatible with the models trained with *fairseq*. The original implementation of exact top-1 decoding heavily relies on CPU operations. In contrast, our top-k version moves a number of computations to GPU, and improves several implementation details as follows.

- **Optimizing the iterating process.** As defined the 13-th line of our Algorithm 2, we need to iterate through all words in the vocabulary. However, the order of iterations significantly influences the speed because of the lower bounds. Empirically, we find that iterating the vocabulary greedily substantially reduces the run time.

- **Batching the hypotheses for each time step.** As stated at the 14-th line of Algorithm 2, we iterate one word and perform one forward model inference at a time. However, the GPU utilization of this scheme is extremely low. Thus, we use batch technique and batch $\mathbf{b}$ different words for one model forward pass, which efficiently increases the GPU utilization.

- **Good lower bounds facilitate the search process.** We observe that better lower bounds vastly reduce the search time. In our implementation, we use the top n-best list output from the beam search with larger beam sizes than n as our lower bounds.

As a result, the speed is improved significantly.

B.1 Worst-case Analysis for Exact Search Algorithm

This section analyzes the worst-case behaviors of exact search algorithms. First, let us discuss a simple case when the exact search does not use lower bounds. Given a target sentence set $Y_l = \{y|\text{len}(y) = l\}$ where all hypotheses in that set have the same length l, it is obvious that the search operations needed for exact top-1 and exact top-k algorithms are the same, i.e., $N_l = |Y_l| = |V|^l$. Thus, the total search operations for all lengths[9] $l \in [1, l_{\max}]$ can be computed by $N = \sum_{l\in[1,l_{\max}]} N_l$.

Next, we consider the case with lower bounds. Since lower bounds help trim the search space, the worst case happens when the search algorithm finds the hypotheses in a reversed order. In that case, lower bounds could not trim any search space and have to iterate all hypotheses. Hence, the numbers of search operations needed for both top-1 and top-k algorithms are identical, i.e., $N = \sum_{l\in[1,l_{\max}]} N_l$ operations. On the other hand, both the top-1 and our top-k algorithms are similar to Branch&Bound algorithm (Hendy and Penny, 1982), which cannot lower the time complexity in the worst case, and its time complexity is the same as the one of depth-first-search (DFS) algorithm (Mackworth, 2013). However, it is practically useful because it is proved to be able to improve the search speed significantly.

C Empirical Computational Cost

C.1 Compared with Different Decoding Algorithms

This section provides several empirical results to show how different decoding methods perform in

[6]The corpora include LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

[7]https://github.com/rycolab/uid-decoding

[8]https://github.com/ucam-smt/sgnmt

[9]We do not use the length constraint in our implementation. Here, we add the max length constraint for the clarity of our proof.

ALGORITHM 2: DFS-based Top-k Exact Search.

Input : x: Source sentence, y: Translation prefix (default: []), p: $\log P(y|x)$ (default 0.0), k: Top-k hypotheses to output

Output : List l contains top-k hypotheses with log-probabilities.

```
global minHeap
global γ ← −inf
Function dfsTopK(x, y, p):
    if y[|y| − 1] =< /s > then
        push(minHeap, (p, y))
        if len(minHeap) > k then
            pop(minHeap)
        end
        if len(minHeap) = k then
            γ ← minHeap[0][0]
        end
    end
    for v ∈ V do
        p' ← p + log P(v|x, y)
        if p' ≥ γ then
            dfsTopK(x, [y; v], p')
        end
    end
    return minHeap
return dfsTopK(x, [], 0.0)
```

terms of computational time. We randomly sample 100 sentences in WMT'14 En-De *newstest2014* and report the corresponding run time as well as the number of expansion operations. The expansion operation, i.e., model's forward pass, is the most time-consuming operation in the exact search algorithm and is linear to the number of computation flops. We report the computational costs for three different algorithms, including *Beam Search*, *Exact Top-1* and *Exact Top-5*. Each reported number is the average over four runs with different samples as inputs.

The results are shown in Table 8. First, we can see that *Beam Search* is about ten to twenty times faster than exact search algorithms. This is consistent with results in previous literature. Second, compared with previous Exact Search implementation, our implementation of top-5 search has almost the same time cost as top-1, which demonstrates the effectiveness and efficiency of our proposed approach.

By taking the number of expansions into account, we notice two more interesting facts – On the one hand, the number of expansions is not linear to k. Our top-k algorithm explores only about five times the search space compared with top-1 algorithm. On the other hand, our algorithm is significantly more efficient than the original implementation, with four times faster in terms of the number of expansions and only about two times in terms of the computational cost.

C.2 Compared with Different k

We also report results with different values of k, shown in Table 9. The computational time and the number of expansions grow as k increases. When we enlarge the number of k from 5 to 10, the time costs grow by about 1.9 times ($^{15,916.2}/_{8,914.4}$), which denotes an almost linear time cost with regard to k. Compared to (Stahlberg and Byrne, 2019), our algorithms are more efficient – Our top-5 algorithm operates two times of expansions and performs comparably with their algorithms in terms of computational time.

In the main content of our paper, we mainly use top-10 results for our evaluation method for the trade-off between efficiency and effectiveness.

D Human Evaluation

This section presents the human judgements on our kQRG results. We sample 100 sentences from NIST Zh-En test sets and provide top-5 exact decoding results for comparison. We ask two professional Chinese-English annotators to rank the top-5 results of three different systems for each source sentence, and average the ranks. We choose to rank systems rather than directly score them since the absolute score of each system's top-5 given by humans may differ drastically. In contrast, comparing against each other is more reliable. The system gives the best top-5 results among systems are scored with 3, and the one with worst top-5 results are scored with 1. Note that these ranks are not easy to judge by humans. Through annotation, we find the hypotheses given by NMT models are sometimes very similar, with only a few different tokens. Therefore, if a human annotator cannot distinguish the difference between two hypotheses, it is allowed to give tied scores, e.g., $[3, 2, 2]$.

The results are shown in Figure 2. We can see that the trend of human judgements is closely aligned with the ranks given by kQRG. Human results are higher than kQRG results due to the tied scores. This proves that the differences in kQRG are consistent with the ones of human judgements.

E Edit-Distance-based Evaluation

In this section, we provide an approach for computing model errors, which do not rely on approximation for $\mathcal{Y}_{\text{HR}}$. The challenge is to score the

Method	Time Cost (seconds)	Num Expansions
Beam Search	453.0	-
Stahlberg and Byrne (2019)	8,064.0	2,769.6
Exact Top-5 w/ BS lower bounds	8,914.4	6,029.4

Table 8: Time cost and number of expansions for exact search algorithms with 4 sampled runs on 100 test sentences.

Method	Time Cost (seconds)	Num Expansions
Stahlberg and Byrne (2019)	8,064.0	2,769.6
Exact Top-5 w/ BS lower bounds	8,914.4	6,029.4
Exact Top-10 w/ BS lower bounds	15,916.2	10,865.9
Exact Top-20 w/ BS lower bounds	28,313.9	19,155.8

Table 9: Computational time and expansions for exact search algorithms when k increases.

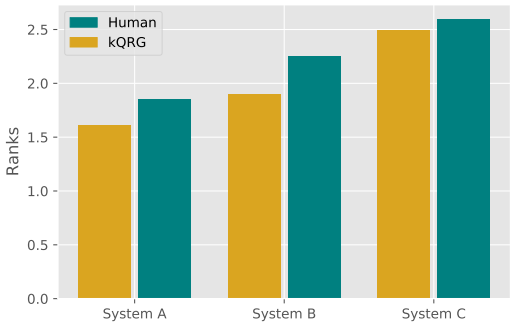


Figure 2: Human evaluated ranks versus kQRG ranks for three different systems.

unbounded hypothesis space . It would be helpful if a HR array with a certain translation quality function Q can be queried without actually constructing the array.

Therefore, inspired by (Norouzi et al., 2016), we model $\mathcal{Y}_{\text{HR}}$ with an edit-distance based ranked array, where each hypothesis is ranked by its edit-distance to the references,

$$Q(y_i, \hat{y}, x) = -\text{Edit\_Distance}(y_i, \hat{y}), \quad (28)$$

where a larger edit distance means a lower rank in the HR array. Then, given a hypothesis y_i with e edits to reference, we can predict the number of hypotheses with e edits from $\hat{y}$ by,

$$c(e, T) = \sum_{s=0}^{T} \binom{T}{s} \binom{T+e-2s}{e-s} |V|^e, \quad (29)$$

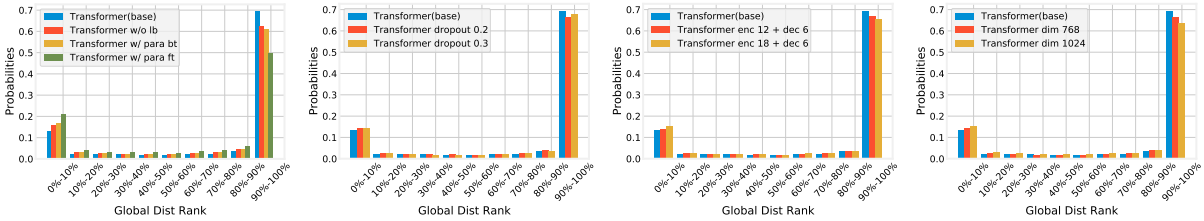
where s is the number of substitution operations and $|V|$ is the vocabulary size. For more details, we refer our readers to (Norouzi et al., 2016). Then we can estimate the rank of a hypothesis y_i with the sum of numbers of hypotheses with edit distance lower than e,

$$\text{Rank}(y) = \sum_{e' \in [0,e)} c(e', T). \quad (30)$$

The visualization of edit-distance rankings is shown in Figure 3, which illustrates how the model error changes when using different architectures and techniques.

Note that we use exact top-100 hypotheses of model distribution for this visualization. We expect the hypotheses are located among the top $0 \sim 10\%$ of edit-distance rankings. Interestingly, across all models, the probability mass of these top-ranking hypotheses lies either in the top 10% or the last 10% of edit-distance ranks. About 70% of the hypotheses are located in the last 10%. It indicates model errors are severe, where the model prefers both good hypotheses (ranked $0 \sim 10\%$) and bad hypotheses (ranked $90\% \sim 100\%$). In addition, we make several observations:

- A model without label smoothing strongly decreases the number of top hypotheses in the last 10% of ranks. (Figure 3(a))
- Pseudo data generation techniques decrease the model errors. The model with forward-translated data significantly decreases model errors. (Figure 3(a))
- Increasing the dropout rate does not necessarily improve model errors. (Figure 3(b))



(a) Loss and pseudo data. (b) Different dropouts. (c) Deeper architectures. (d) Wider architectures.

Figure 3: The edit-distance ranking visualization of Transformer models with different techniques. Here, we use top-10 hypotheses of exact search. **para FT** and **para BT** denote the model trained with both golden dataset and forward/backward generated pseudo dataset.

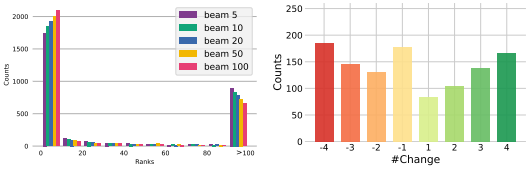
- Increasing model capacity, like deepening or widening model architectures, generally reduces model errors monotonically. (Figure 3(c, d))

We use edit-distance metric because the commonly used metrics in NMT (e.g., BLEU or METOR) are difficult to estimate the ranks of hypotheses, therefore cannot provide the above analysis.

F Investigations on Beam Search Curse

Our proposed exact top-k decoding algorithm enables us to investigate the relationship between the beam search algorithm and the top-k decoding algorithm. First, we conduct an experiment to help understand to what extent these two algorithms disagree. We first run the exact decoding algorithm using $k = 100$ and beam search with beam sizes of 5, 10, 20, 50, 100 on the WMT' 14 English-German testset. We plot the ranked positions of the beam search results over the exact outputs in Figure 4a, which demonstrates that a larger beam size leads to fewer search errors since the ranked positions of those hypotheses are higher. However, the BLEU scores of the beam search results are inversely proportional to the beam size, which are $27.28\%, 27.27\%, 26.98\%, 26.03\%, 23.87\%$ respectively. This is called *beam search curse* (Yang et al., 2018) - translation quality degrades as beam sizes increases.

We further investigate the trends in terms of BLEU scores from sentence-level when using a larger beam size. For each source sentence, we track the BLEU score of its translated sentence from a small beam size to a large beam size. As a result, we can obtain two properties for each sentence. One is the number of getting a better translation. Conversely, the other is the number of getting a worse translation. We plot the results



(a) (b)

Figure 4: The experimental results of comparing top-k exact decoding with beam search decoding. (a) The numbers of ranked positions of different beam search results. (b) Sentence-level BLEU trends when increasing the beam size. The horizontal axis is the number of changes (positive values means getting better, vice versa). The vertical axis is the counts of changes.

in Figure 4b. The results show that the trend is random in sentence-level, and not consistent with the facts we observe from the whole dataset. This reveals that the NMT model has a large number of ranking errors, i.e., two very close hypotheses may be ordered falsely by the model.

Based on the above experimental results, we discover that *beam search curse* is closely related to the **hypothesis ranking** of an NMT model. This finding motivates us to study how to evaluate the hypothesis distribution of an NMT system. If an NMT model can successfully order all hypotheses by a certain scoring function, the *beam search curse* will disappear.

G Case Study

This section provides a case study for English-German translation outputs for our Exact Top-k decoding algorithm. Table 10 shows the generated hypotheses, their corresponding log probabilities, and BLEU scores.

There are several problems of models' generated outputs based on the example: First, the ranking problem we argue in the main content apparently exists, which is demonstrated in our provided ex-

Rank	LogProb	BLEU	hypothesis
Ref	-	100.00	**Zwei Anlagen so nah beieinander: Absicht oder Schildbürgerstreich? <EOS>**
1	-9.04	00.00	<EOS>
2	-10.13	20.45	Zwei Leuchten so nah beieinander: absichtlich oder einfach nur ein dummer Fehler? <EOS>
3	-10.40	07.47	Zwei Leuchten so nahe beieinander: absichtlich oder einfach nur ein dummer Fehler? <EOS>
4	-10.56	22.24	Zwei Leuchten so nah beieinander: absichtlich oder nur ein dummer Fehler? <EOS>
5	-10.92	08.13	Zwei Leuchten so nahe beieinander: absichtlich oder nur ein dummer Fehler? <EOS>
6	-10.94	05.89	Zwei Leuchten so nahe beieinander? <EOS>
7	-11.10	22.24	Zwei Leuchten so nah beieinander: absichtlich oder einfach ein dummer Fehler? <EOS>
8	-11.15	37.60	Zwei Leuchten so nah beieinander: Absicht oder einfach nur ein dummer Fehler? <EOS>
9	-11.21	17.63	Zwei Leuchten so nah beieinander? <EOS>
10	-11.39	40.90	Zwei Leuchten so nah beieinander: Absicht oder nur ein dummer Fehler? <EOS>

Table 10: The generated translations with top-10 decoding. The source sentence is *"Two sets of lights so close to one another: intentional or just a silly error?"*

ample. For instance, the model gives the highest score to an empty hypothesis (only <EOS>), which ranks the model's mode hypothesis the worst in the hypothesis space. Second, the model ranks some sub-optimal hypotheses in the top-10 rankings, like 2-nd, 4-th, 7-th, 10-th. However, the best hypothesis is ranked only at the 10-th position. It can also prove the existence of the ranking problem. Third, the model favors shorter hypotheses. The hypotheses at rank positions 1-st, 6-th, and 9-th are much shorter than the others. The short hypotheses have roughly similar scores compared with the longer ones. Furthermore, most of the hypotheses share a similar prefix, which is similar to the reference, demonstrating that the model can find proper translations with incorrect log probabilities. Those problems indicate the existence of an under-confidence problem, which is in line with our findings in Section 4.1.