# Twist Decoding: Diverse Generators Guide Each Other

**Jungo Kasai**♡♣ **Keisuke Sakaguchi**♢* **Ronan Le Bras**♣ **Hao Peng**♣*
**Ximing Lu**♡♣ **Dragomir Radev**♠ **Yejin Choi**♡♣ **Noah A. Smith**♡♣

♡Paul G. Allen School of Computer Science & Engineering, University of Washington
♣Allen Institute for AI ♢Tohoku University
♠Department of Computer Science, Yale University
{jkasai,lux32,yejin,nasmith}@cs.washington.edu keisuke.sakaguchi@tohoku.ac.jp
{ronanlb,haop}@allenai.org dragomir.radev@yale.edu

## Abstract

Many language generation models are now available for a wide range of generation tasks, including machine translation and summarization. Combining such diverse models may lead to further progress, but ensembling generation models is challenging during inference: conventional ensembling methods (e.g., shallow fusion) require that the models share vocabulary/tokenization schemes. We introduce TWIST decoding, a simple and general text generation algorithm that benefits from diverse models at inference time. Our method does not assume the vocabulary, tokenization or even generation order is shared. Our extensive evaluations on machine translation and scientific paper summarization demonstrate that TWIST decoding substantially outperforms each model decoded in isolation over various scenarios, including cases where domain-specific and general-purpose models are both available. TWIST decoding also consistently outperforms the popular reranking heuristic where output candidates from one model are rescored by another. We hope that our work will encourage researchers and practitioners to examine generation models collectively, not just independently, and to seek out models with complementary strengths to the currently available models.[1]

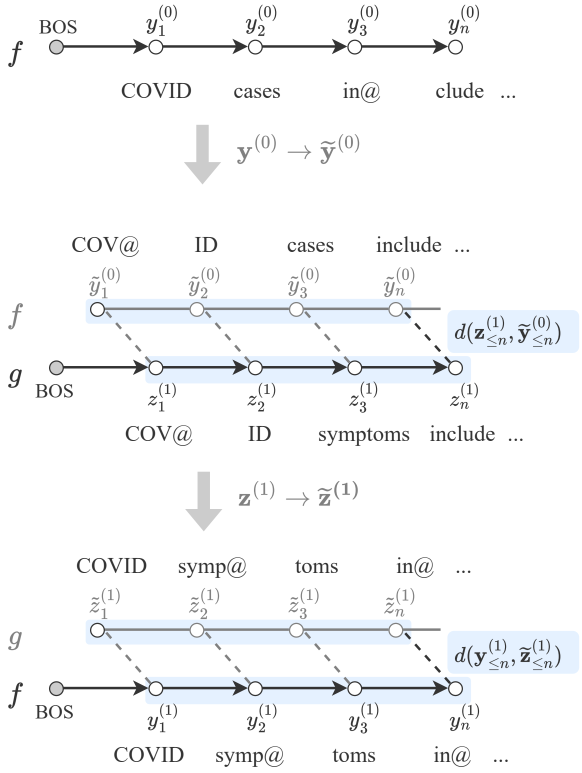

Figure 1: TWIST decoding of two generation models, $f$ and $g$, that does not assume a shared vocabulary, tokenization, or generation order. Beam search is first applied to $f$ to generate $\mathbf{y}^{(0)}$, followed by output mapping to $\tilde{\mathbf{y}}^{(0)}$ (e.g., $f$'s detokenization and $g$'s tokenization or sequence reversal). $g$ is then decoded with beam search augmented with distances from the set of previously-generated outputs (here only one sequence $\mathbf{y}$ is shown): $d(\mathbf{z}_{\leq n}^{(1)}, \tilde{\mathbf{y}}_{\leq n}^{(0)})$. Subsequently, $f$ is similarly decoded with $g$'s guidance. Here we show one iteration that already achieves substantial improvements (§4). @ indicates the BPE separator.

## 1 Introduction

Natural language generation is an important building block for many applications, such as machine translation, summarization, and question answering (Ng et al., 2019; Lewis et al., 2020; Raffel et al., 2020; Brown et al., 2020; Asai et al., 2021, *inter alia*). Researchers have recently explored and advanced models for generation in various aspects, including model architecture (Bahdanau et al., 2015; Vaswani et al., 2017), domain adaptation (Chu and Wang, 2018; Bapna and Firat, 2019), prompting (Brown et al., 2020), and even generation order (Gu et al., 2018). The resulting generation models are diverse, trained on different data, with different

[*] This work was done while Keisuke Sakaguchi was at the Allen Institute for AI and Hao Peng was at the University of Washington.

[1] Our code is available at https://github.com/jungokasai/twist_decoding.

*Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4909 - 4923
December 7-11, 2022 ©2022 Association for Computational Linguistics

assumptions, at different times. We hypothesize that diverse generation models may achieve better results through ensembling, if the various approaches have complementary strengths. Given the high cost of unifying approaches during training time (Strubell et al., 2019; Schwartz et al., 2019), inference-time combination of existing models is an attractive alternative.

One well-established ensembling technique is "shallow fusion" (Sutskever et al., 2014; Gulcehre et al., 2015; Firat et al., 2016, *inter alia*), which aggregates models' scores during beam search. This approach requires, however, that the models use the same vocabulary/tokenization scheme and organize the search in the same way (e.g., autoregressive, left-to-right factorization).

We introduce a new inference algorithm, Twist decoding (Fig. 1), that enables more diverse generators to guide each other. Twist decoding can combine generators with different vocabularies, (de)tokenization, and even generation order without any additional training or finetuning. Our method decodes a model by standard beam search, but the scores at every step incorporate a simple function that measures the distance from outputs of the other model. We run this procedure on each generation model in turn, so that both can benefit from each other.

We present extensive experiments on machine translation and scientific paper summarization and show that Twist decoding can improve performance over each model decoded in isolation across several scenarios: combining 1) generic and domain-specific models, 2) left-to-right and right-to-left generation models, and 3) models that generate using different conditioning inputs. Our results show consistent performance gains from combining generic and domain-specific translation models over a wide range of domains, including medical and legal translation. Applications in these domains require particularly high accuracy, and Twist decoding is a desirable alternative to standard beam search on a single model. Interestingly, we find that Twist decoding between generic and domain models is effective even when parallel data from the domain are scarce and the domain model yields poor performance by itself, suggesting complementary strengths of diverse generators (§3.4).

Twist decoding can be seen as a generalization of reranking heuristics that have proven effective in syntactic parsing (Shen and Joshi, 2003; Charniak and Johnson, 2005; Collins and Koo, 2005), speech recognition (Collins et al., 2005), and machine translation (Shen et al., 2004; Och et al., 2004): one model generates candidate sequences, followed by rescoring from another model. We present extensive comparisons with reranking baselines and demonstrate that Twist decoding outperforms reranking consistently. We also observe that since the encoder computations on two models can be parallelized, the inference time required for Twist decoding is much shorter than the sum of the two models, resulting only in a 50% increase, relative to decoding of a single model in isolation (§4). Twist decoding is therefore a viable alternative to standard beam search on a single model and the widespread reranking heuristic.

```
Twist Decoding
$g$ generates $\mathbf{z}$ with guidance from $f$ at iteration $t$
$k$: beam size.  $M$: maximum length.
$\mathcal{V}_g$: vocabulary of $g$.  $g(\cdot)$: scoring function.
$\mathcal{Y}^{(t-1)}$: set of output sequences from $f$.  $\mathcal{Z}^{(t)}$: new outputs.
$B_n$: beam of continuing sequences.
$H$: expanded hypotheses before beam selection.
$d(\cdot, \cdot)$: distance between partial sequences.
$\lambda_f$: scalar coefficient for the distance.
$\widetilde{\mathcal{Y}}^{(t-1)} = \left\{\widetilde{\mathbf{y}} = \text{map\_output}(\mathbf{y}) \mid \mathbf{y} \in \mathcal{Y}^{(t-1)}\right\}$
$B_0 \leftarrow \{\text{BOS}\}$,  $\mathcal{Z}^{(t)} \leftarrow \varnothing$
for $n \in \{1, \ldots, M\}$ :
    $H \leftarrow \varnothing$
    for $\mathbf{z} \in B_{n-1}$ :    # Expansion.
        for $z \in \mathcal{V}_g$ :
            $s \leftarrow g(\mathbf{z} \circ z) - \lambda_f \min_{\widetilde{\mathbf{y}} \in \widetilde{\mathcal{Y}}^{(t-1)}} d\left(\mathbf{z} \circ z, \widetilde{\mathbf{y}}_{\leq n}\right)$
            $H.\text{add}(\langle s, \mathbf{z} \circ z \rangle)$
    $B_n \leftarrow \text{topk}(H)$,  $\mathcal{Z}^{(t)}.\text{add}\left(\text{finished}(H)\right)$
return $\mathcal{Z}^{(t)}$
```

Figure 2: Twist decoding when $g$ is guided by $f$. Swap $f$ and $g$ and $\mathbf{y}$ and $\mathbf{z}$ to obtain $\mathcal{Y}^{(t)}$. map_output converts outputs from $f$ to $g$; e.g., $f$'s detokenization followed by $g$'s tokenization. It would also include sequence reversal if $f$ or $g$ is a right-to-left model. The highlighted line is the *only* modification that Twist decoding introduces to standard beam search. The input sequence to $g$ is omitted. See also Kasai et al. (2022b) for the stopping criterion and implementation details (the *first come, first served* heuristic).

## 2 Twist Decoding

We propose Twist decoding, a general decoding algorithm that generates text from diverse models without assumptions of a shared vocabulary, tokenization, or generation order. At the core of the algorithm is a simple modification in standard beam search (highlighted in Fig. 2); we incorporate into a scoring function the distance from outputs that are previously generated by another model.

### 2.1 Initial Decoding

Let us assume that we have two generation models: $f$ and $g$.[2] Both $f$ and $g$ assign scores to output sequences.[3] For example, $f$ can be a domain-specific translation model and $g$ a generic one. $f$ and $g$ perform their own pre/postprocessing (e.g., (de)tokenization) and factorization (e.g., left-to-right or right-to-left factorization). Here we suppress for brevity the conditional dependence on the input (e.g., machine translation input from the source language). Standard beam search with a beam size $k$ is first applied to $f$ to produce a set of $k$ output sequences: $\mathcal{Y}^{(0)}$. This approximately solves $\mathrm{topk}_{\mathbf{y}}\, f\left(\mathbf{y}\right)$ by pruned breadth-first search, and often returns higher-quality outputs than the exact search counterpart (Stahlberg and Byrne, 2019; Meister et al., 2020a).

### 2.2 Mutually-Guided Decoding

Once $\mathcal{Y}^{(0)}$ is obtained, we proceed with decoding generators with mutual guidance (Fig. 2; $t \geq 1$).

**Output Sequence Mapping.** The commonly-used technique of ensembling (Sutskever et al., 2014) or shallow fusion (Gulcehre et al., 2015; Stahlberg et al., 2018) adds scores from $f$ and $g$ at every step and executes the same search algorithm to approximately solve $\mathrm{topk}_{\mathbf{y}}\, f(\mathbf{y}) + g(\mathbf{y})$. This method thus necessitates a shared vocabulary, tokenization, and generation order (Imamura and Sumita, 2017). We relax this assumption and first map the candidates in $\mathcal{Y}^{(t-1)}$ to output sequences for $g$: $\widetilde{\mathcal{Y}}^{(t-1)}$ (Line 1 in Fig. 2). This mapping (map_output) typically involves deterministic operations of $f$'s detokenization followed by $g$'s tokenization. Sequence reversal is also performed if $f$ and $g$ generate in the opposite order. For example, if $g$ uses byte-pair encoding (Sennrich et al., 2016b), but $f$ does not, we might have $\mathbf{y} =$*John does n't like Mary* mapped to $\widetilde{\mathbf{y}} =$*Jo@ hn doesn't like Mar@ y*, where @ denotes subword separation.

**Decoding with Distance Terms.** We then decode $g$ with guidance from $\widetilde{\mathcal{Y}}^{(t-1)}$. Specifically, we perform beam search with a simple modification in scoring (Line 7). In this work, we use a simple distance measure that adds binary distances at all positions (i.e., the Hamming distance):

$$d\left(\mathbf{z}_{\leq n}, \widetilde{\mathbf{y}}_{\leq n}\right) = \sum_{i \leq n} \mathbb{1}\left\{z_i \neq \widetilde{y}_i\right\}$$

We also explored using the distance between (sub)word embeddings from the model: $\sum_{i \leq n} \| e(z_i) - e(\widetilde{y}_i) \|_2$, but this did not bring improvements (§4). Note also that when $i$ exceeds the length of $\widetilde{\mathbf{y}}$, we assume $\widetilde{y}_i = \textsc{eos}$. The overall distance term is then

$$\min_{\widetilde{\mathbf{y}} \in \widetilde{\mathcal{Y}}^{(t-1)}} d\left(\mathbf{z}_{\leq n}, \widetilde{\mathbf{y}}_{\leq n}\right)$$

Here we minimize over the output sequences to compute the distance to the closest candidate. These candidates from $\widetilde{\mathcal{Y}}^{(t-1)}$ can be equally good outputs but differ only by one word; in such cases, this minimization operation avoids overestimation of the distances. The new score at step $n$ in beam search is now computed by:

$$g(\mathbf{z}_{\leq n}) - \lambda_f \min_{\widetilde{\mathbf{y}} \in \widetilde{\mathcal{Y}}^{(t-1)}} d\left(\mathbf{z}_{\leq n}, \widetilde{\mathbf{y}}_{\leq n}\right),$$

where $\lambda_f$ is a scalar coefficient for the distance term that controls the importance of $f$ relative to $g$. We tune $\lambda_f \in \{0.1, 0.3, 1.0, 3.0\}$ during development. After this beam search, we obtain a new candidate set, $\mathcal{Z}^{(t)}$. We then run the same beam search (Fig. 2) with the roles of $f$, $\mathcal{Y}$ and $g$, $\mathcal{Z}$ swapped.[4] Namely, we decode $f$ with distance terms from $\mathcal{Z}^{(t)}$ at each step of beam search:

$$f(\mathbf{y}_{\leq n}) - \lambda_g \min_{\widetilde{\mathbf{z}} \in \widetilde{\mathcal{Z}}^{(t)}} d\left(\mathbf{y}_{\leq n}, \widetilde{\mathbf{z}}_{\leq n}\right)$$

Finally, the highest-scoring sequence from $\mathcal{Y}^{(t)}$ is output. This process of mutually-guided decoding can be repeated multiple times. We observe, however, that one iteration ($t = 1$) suffices to bring performance gains (§4). We also present detailed sensitivity analysis over varying $\lambda_f$ and $\lambda_g$ and find that TWIST decoding is particularly effective when $\lambda_g > \lambda_f$ (i.e., initial exploration by $g$ is encouraged with relatively little guidance from $f$'s original outputs; see §4).

**Reranking Heuristic as a Special Case.** Notice that as $\lambda_f \to \infty$, $g$'s generation falls back to a reranking heuristic: top $k$ sequences from the initial $f$ decoding are reranked according to

[2] The algorithm can be readily extended to three or more generators. We also abuse $f$ or $g$ to mean both the generator and its scoring function.

[3] They typically assign log-probabilities, but it is not necessary to assume the scores form a valid probability distribution.

[4] We can stop inference with $\mathcal{Z}^{(t)}$, but we found that led to performance degradation in preliminary development.

$g$. This reranking heuristic has proven successful in a wide range of sequence generation tasks, including machine translation (Shen et al., 2004), syntactic parsing (Collins and Koo, 2005), and speech recognition (Collins et al., 2005). Reranking is performed in many strong machine translation systems to use a right-to-left model to improve a left-to-right model; e.g., top-performing systems in recent WMT competitions (Ng et al., 2019; Kiyono et al., 2020; Wang et al., 2021; Akhbardeh et al., 2021). In our experiments, we extensively compare performances of TWIST decoding and reranking and demonstrate that the former consistently outperforms the latter.

## 3 Experiments

We present experiments across three scenarios: combining domain and generic models for machine translation (§3.1), left-to-right and right-to-left machine translation models (§3.2), and scientific paper summarization models that take as input different parts of the paper (§3.3). We empirically compare TWIST decoding with decoding in isolation and the widely-adopted reranking baselines, illustrating that TWIST decoding offers performance improvements in various situations *without* any change to the trained models.

### 3.1 Domain and Generic Models

Machine translation has now been used for many domains, ranging from everyday conversations to medical documents. Machine translation models are often trained on large amounts of parallel data, such as the Europarl corpus (Koehn, 2005) and the OPUS data (Tiedemann, 2012). Applying these models to out-of-domain data remains a challenge (Koehn and Knowles, 2017; Chu and Wang, 2018), and users for some of these domains require high accuracy in translation (e.g., medical and legal documents). We will demonstrate that TWIST decoding between general-purpose and domain-specific models is a viable approach to tackle this problem.

**Setups.** We use machine translation datasets over diverse domains from prior work (Koehn and Knowles, 2017; Hu et al., 2019): German→English over medical (1.1M training sentence pairs), legal (720K pairs), Koran (religious text, 480K pairs), and subtitles (14M pairs) domains.[5] For the domain-specific models, we train a base-sized transformer model (Vaswani et al., 2017) with a 6-layer encoder and a 6-layer decoder on the training data of each domain. The top-performing German→English system from WMT19 (Barrault et al., 2019; Ng et al., 2019)[6] is used as the generic model. This generic model is a large-sized transformer trained on a concatenation of publicly available parallel data, including the Europarl (Koehn, 2005) and UN (Ziemski et al., 2016) corpora with the backtranslation technique (Sennrich et al., 2016a). We follow (de)tokenization (Koehn et al., 2007) and byte-pair encoding (Sennrich et al., 2016b) of previous work (Koehn and Knowles, 2017; Hu et al., 2019).[7]

For every domain, we evaluate a total of six configurations: decoding of the generic and domain models each in isolation; the reranking baseline and TWIST decoding with $f$ being the generic model and $g$ being the domain model, as well as the versions where $f$ and $g$ are swapped to see the effect of the two roles. In all cases, we use beam size 5 (Freitag and Al-Onaizan, 2017) and length penalty 1 (Wu et al., 2016) and conduct all experiments using the `fairseq` library (Ott et al., 2019). All performance is measured with the COMET score (Rei et al., 2020a,b) and the SACREBLEU implementation (Post, 2018) of the BLEU score (Papineni et al., 2002). Note that COMET is based on crosslingual contextual representations (Conneau et al., 2020), and recent work showed that it achieves significantly higher correlation with expert human judgment than BLEU and other n-gram-based metrics (Kasai et al., 2022a,c). More experimental details are described in Appendix §A.1.

**Results.** Seen in Table 1 are the results from our experiments over various domains. Firstly, given two translation models $f$ and $g$, TWIST decoding outperforms the reranking baseline in all configurations (indicated in blue) with only one exception (a small drop in BLEU in the subtitles domain). Particularly noteworthy are the gains in the medical domain: TWIST decoding outperforms the reranking heuristic by 5.8 COMET and 1.4 BLEU points when $f$ is the domain model and $g$ is the generic model. TWIST decoding is thus an effective generalization over the reranking heuristic commonly used in the literature across domains.

Comparing the performance of decoding in iso-

[5] We excluded the IT domain because we found significant overlap between training and dev./test data.

[6] https://github.com/pytorch/fairseq/tree/main/examples/wmt19.

[7] https://github.com/JunjieHu/dali.

| German→English | | | Medicine | | Law | | Koran | | Subtitles | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | $f$ | $g$ | COMET | BLEU | COMET | BLEU | COMET | BLEU | COMET | BLEU |
| Isolation | Generic | – | 44.5 | 41.2 | 30.6 | 34.8 | 14.4 | 16.6 | 34.4 | 31.3 |
| Isolation | Domain | – | 80.7 | 48.3 | 60.7 | 40.9 | 8.7 | 17.0 | 32.3 | 29.0 |
| Rerank | Generic | Domain | 59.6 | 43.5 | 56.4 | 36.1 | 14.7 | 17.0 | 40.3 | **32.3** |
| TWIST | Generic | Domain | 71.6 | 47.5 | 61.4 | 40.2 | **16.5** | 18.5 | **41.0** | 32.2 |
| $\Delta$ (TWIST− Rerank) | | | +12.0 | +4.0 | +5.0 | +4.1 | +1.8 | +1.5 | +0.7 | −0.1 |
| Rerank | Domain | Generic | 75.8 | 48.7 | 59.9 | 40.8 | 12.3 | 18.1 | 36.5 | 30.3 |
| TWIST | Domain | Generic | **81.6** | **50.1** | **61.6** | **41.3** | 15.3 | **18.7** | 37.3 | 31.0 |
| $\Delta$ (TWIST− Rerank) | | | +5.8 | +1.4 | +1.7 | +0.5 | +3.0 | +0.6 | +0.8 | +0.7 |

Table 1: Combination of generic and domain-specific translation models. The generic model is the top-performing translation model in WMT19 (Ng et al., 2019) that is trained on a collection of parallel corpora, such as the Europarl and the UN corpora. Two settings are considered for the reranking baseline and TWIST decoding: $f$ is the generic model and $g$ is the domain model or the reverse. The best scores are in **bold**. COMET (Rei et al., 2020a,b) uses crosslingual contextual representations (Conneau et al., 2020) and achieves significantly higher correlation with expert human judgment than BLEU (Papineni et al., 2002) and other alternative metrics (Kasai et al., 2022a,c).

lation and TWIST decoding, we observe that the best score from TWIST decoding substantially outperforms each individual model over all domains: e.g., 81.6 vs. 80.7 (domain model) and 81.6 vs. 44.5 (generic model) COMET points in the medical domain. In both medical and legal domains, the generic model underperforms the domain model by a large margin. Nonetheless, TWIST decoding between the two improves over the domain model, suggesting that TWIST decoding makes use of their complementary strengths. Finally, we see a consistent pattern regarding $f$ and $g$: both TWIST decoding and the reranking baseline perform better when the higher-performing model is chosen as $f$. (e.g., the domain model performs better in medicine and law, and vice versa in subtitles.) This is expected because $f$ is used both for initial decoding and final decoding with $g$'s guidance (Fig. 1).

### 3.2 Left-to-Right and Right-to-Left Models

Language generation models usually factorize sequences autoregressively in a left-to-right order, but previous work showed that left-to-right (L2R) models can be improved by reranking their outputs with a separate right-to-left (R2L) model (Imamura and Sumita, 2017; Ng et al., 2019; Kiyono et al., 2020, *inter alia*). TWIST decoding can be readily applied to such scenarios since it does not assume shared generation order between models.

**Setups.** We experiment with two language pairs from the WMT 2020 news translation task (Barrault et al., 2020): Chinese→English (**WMT20 ZH-EN**, 48M training sentence pairs) and English→German (**WMT20 EN-DE**, 48M pairs). Submissions for these language pairs to the shared task have human evaluations from professional translators (Freitag et al., 2021), and the correlation between automatic metrics and the human ratings are studied in subsequent work (Kasai et al., 2022a); COMET (Rei et al., 2020b,a) achieves the highest correlation out of the 15+ metrics.

Similar to the previous experiments, we measure all performance using COMET and BLEU scores. Note that we use two reference translations per instance for WMT20 ZH-EN and three for WMT20 EN-DE, following Kasai et al. (2022a). They both have reference translations from two different services, and WMT20 EN-DE has an additional translation created by linguists who are asked to paraphrase the two translations as much as possible. These paraphrased translations are shown to increase correlation with human judgments by mitigating the translationese effect (Graham et al., 2020) and diversifying the reference (Freitag et al., 2020). On each dataset, we follow the preprocessing and tokenization (Koehn et al., 2007; Sennrich et al., 2016b) from Kasai et al. (2022a)[8] and train a large-sized transformer model for left-to-right and right-to-left translation, in which the output English/German sequences are reversed after tokenization. We implement all models and decoding with `fairseq` and apply beam search with beam size 5 and length penalty 1. We again consider a total of six settings: reranking and TWIST decoding with L2R as $f$ and R2L as $g$ or the reverse, as well as the individual models. Further details can

[8] https://github.com/jungokasai/billboard/tree/master/baselines.

be found in Appendix §A.2.

| WMT20 Test | | | ZH→EN | | EN→DE | |
|---|---|---|---|---|---|---|
| **Method** | $f$ | $g$ | COMET | BLEU | COMET | BLEU |
| Isolation | L2R | – | 40.8 | 35.5 | 42.9 | 45.5 |
| Isolation | R2L | – | 40.4 | 35.0 | 43.3 | 44.9 |
| Rerank | L2R | R2L | 41.4 | 36.1 | 43.7 | 46.0 |
| TWIST | L2R | R2L | 42.8 | **36.8** | **45.4** | **46.7** |
| Δ (TWIST− Rerank) | | | +1.4 | +0.7 | +1.7 | +0.7 |
| Rerank | R2L | L2R | 41.2 | 35.4 | 44.7 | 45.2 |
| TWIST | R2L | L2R | **43.1** | **36.8** | 44.8 | 46.0 |
| Δ (TWIST− Rerank) | | | +1.9 | +1.4 | +0.1 | +0.8 |

Table 2: Combination of left-to-right (L2R) and right-to-left (R2L) transformer translation models. ZH: Chinese. DE: German. Two settings are considered for reranking and our TWIST decoding each: L2R or R2L as $f$. The best scores are in **bold**.

**Results.** Table 2 shows the results from L2R and R2L translation models. TWIST decoding again outperforms the reranking counterpart by a considerable margin in COMET and BLEU on both language pairs; e.g., 43.1 vs. 41.2 COMET points on WMT20 ZH-EN when $f$ is R2L and $g$ is L2R. The best performance is achieved by TWIST decoding on both datasets and improves over the individual models by more than 1 BLEU point. The reranking baseline, on the other hand, does not outperform L2R in BLEU when $f$ is R2L: 35.4 vs. 35.5 (ZH-EN) and 45.2 vs. 45.5 (EN-DE). This result illustrates that TWIST decoding is a more effective approach to combine models with different generation order than the popular reranking.

### 3.3 Summarization with Different Input

We also experiment with strong models on a highly abstractive scientific paper summarization task: **SciTLDR** (Cachola et al., 2020). Specifically, we use two BART-based models from prior work (Cachola et al., 2020) that differ in input type: one that only takes as input the paper abstract (Abst.) and the other a concatenation of the abstract, introduction, and conclusion (AIC).[9]

**Setups.** We use the train/dev./test split from Cachola et al. (2020). Again following Cachola et al. (2020), we use all human-written summaries (written either by authors or undergraduate computer science students) as the reference and evaluate performance in terms of the ROUGE score (Lin, 2004).[10] We average the instance-level scores from the Python rouge-score implementation.[11] Similar to our previous experiments, we use beam size 5 and length penalty 1. See more detail in Appendix A.3.

**Results.** Table 3 presents our results. TWIST decoding substantially outperforms the reranking baseline when $f$ is the AIC model (e.g., +0.5 ROUGE-L points), but they yield (almost) the same performance when $f$ is the Abst. model. Nonetheless, TWIST decoding achieves the best performance out of all configurations. Our small improvements might be attributed to the fact that the input to the Abst. model is a strict subset of the AIC model and there are only limited benefits from combining them.

| SciTLDR Summ. Test | | | ROUGE | | |
|---|---|---|---|---|---|
| **Method** | $f$ | $g$ | R-1 | R-2 | R-L |
| Isolation | Abst. | – | 39.9 | 21.1 | 34.5 |
| Isolation | AIC | – | 40.2 | 21.3 | 34.9 |
| Rerank | Abst. | AIC | 40.5 | 21.7 | 35.1 |
| TWIST | Abst. | AIC | 40.5 | 21.7 | 35.0 |
| Δ (TWIST− Rerank) | | | 0.0 | 0.0 | −0.1 |
| Rerank | AIC | Abst. | 40.1 | 21.2 | 34.8 |
| TWIST | AIC | Abst. | **40.7** | **22.1** | **35.3** |
| Δ (TWIST− Rerank) | | | +0.6 | +0.9 | +0.5 |

Table 3: Combination of scientific paper summarization models. Both models are BART-based models from prior work (Cachola et al., 2020) with different input: abstract only (Abst.) or abstract, introduction, and conclusion (AIC). The best scores are in **bold**. The ROUGE scores (R-1, R-2, and R-L) are computed by averaging instance-level scores from the Python rouge-score implementation.

### 3.4 Low-Resource Scenarios

In our experiments over four diverse domains (§3.1), we assumed that plenty of parallel data is available in every domain, and the domain model generally outperformed the generic model. Concretely, we used 1.1M and 720K training sentence pairs for the medical and legal domains, based on the data splits from previous work (Koehn and Knowles, 2017; Hu et al., 2019). In real-world applications, however, these domain-specific translation data are often scarce since they need to be annotated by bilingual speakers with expertise in

[9] https://github.com/allenai/scitldr.

[10] We release our models and their outputs, so other metrics can be readily used as well in the future.

[11] https://pypi.org/project/rouge-score/.

those domains. The question arises: *can a domain model trained on small parallel data still help the generic model by its complementary strengths?* To simulate such low-resource scenarios, we randomly sample {10k, 20k, 40k, 80k} sentence pairs and conduct the same evaluations with the generic and domain models as $f$ and $g$, respectively.

Fig. 3 plots COMET scores of various decoding methods on the medical and legal domains. The score from the generic model is constant because we only change the domain training data. There is a striking trend: even though the domain model performs poorly by itself, it improves the generic model through TWIST decoding over varying sizes. Reranking also helps the generic model as the data size increases, but the improvement is less pronounced than that of TWIST decoding.

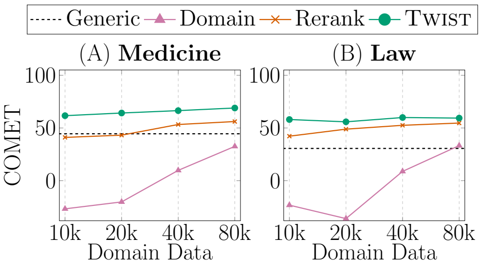

Figure 3: Results when parallel data are scarce in the target domain. Both TWIST decoding and reranking use the generic model as $f$ and the domain model as $g$. COMET (Rei et al., 2020a) is a regression-based metric that can take negative values. $\lambda$s are tuned in each case, and we found that as the domain model ($g$) gets stronger, $\lambda_g$ increases, relative to $\lambda_f$. This observation is aligned with the intuition that $\lambda_g$ indicates the relative importance of g's guidance.

## 4 Analysis

**Iterations.** So far, we have only applied one iteration of TWIST decoding, but Fig. 4 plots performance over multiple iterations. Iteration 0 signifies $f$'s initial decoding ($\mathbf{y}^{(0)}$ in Fig. 1), and every iteration involves $g$'s decoding with $f$'s guidance ($\mathbf{z}^{(t)}$) and its reverse ($\mathbf{y}^{(t)}$). We observe that the first iteration brings most of the performance gains. This makes TWIST decoding practically appealing, as it improves performance without much increase in the computation or inference time (see below).

**Inference Time.** Table 4 reports the runtime of each decoding method, relative to $f$'s decoding in isolation. We use batch size 1 on the same single A100-SXM GPU and measure the wall-clock time from when all models are loaded until all outputs are obtained. As expected, TWIST decoding results in a slowdown compared to decoding in isolation, but the increase in time is only 50%. The inference time for TWIST decoding is much shorter than the sum of $f$ and $g$ in isolation (1.4× vs. 2.1× on medical translation) because 1) the encoder computation for $f$ and $g$ can be parallelized and 2) the encoder computation for $f$ is done only once while we need two runs of $f$'s decoder. We leave it to future work to further speed up TWIST decoding; since the slowdown of TWIST decoding primarily comes from the decoder, it can be sped up by best-first beam search (Meister et al., 2020b), a deep-encoder, shallow-decoder strategy (Kasai et al., 2021a), or a fast, linear-complexity variant of the transformer decoder (Peng et al., 2021; Kasai et al., 2021b) that is shown to retain the performance of the standard encoder-decoder transformer. Another approach could be sequence-level knowledge distillation (Kim and Rush, 2016), which has proven successful in speeding up an ensemble translation model (Freitag et al., 2017).

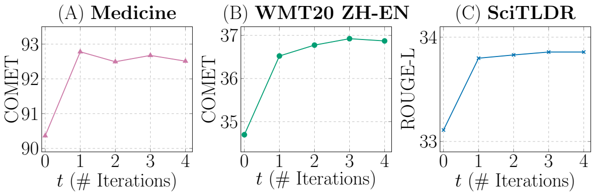

Figure 4: Effects of iterations on dev. performance. Iteration 0 refers to the initial decoding from $f$. Every iteration consists of $g$'s decoding with $f$'s guidance followed by $f$'s decoding with $g$'s guidance. The values of $\lambda$s are kept the same over all iterations for simplicity. Initially, we explored gradually increasing the $\lambda$s as f and g's outputs become closer, but we found no substantial performance gain.

| Inference | Medicine | | | WMT20 ZH→EN | | |
|---|---|---|---|---|---|---|
| Method | $f$ | $g$ | Time | $f$ | $g$ | Time |
| Isolation | Domain | – | 1.0× | R2L | – | 1.0× |
| Isolation | Generic | – | 1.1× | L2R | – | 1.0× |
| Rerank | Domain | Generic | 1.0× | R2L | L2R | 1.0× |
| TWIST | Domain | Generic | 1.4× | R2L | L2R | 1.5× |

Table 4: Inference time relative to a single model decoded in isolation. It is measured on the same single Nvidia A100-SXM GPU with batch size 1. We measure the wall-clock time from when the models are loaded until the last sentence is translated on the test data.

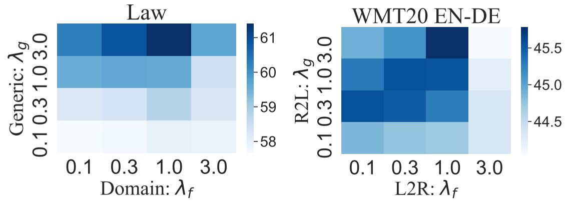

Figure 5: Dev. set performance measured in the COMET score (Rei et al., 2020a,b) with varying $\lambda_f$ and $\lambda_g$. See Appendix §B for other configurations.

| **Dev. Set Results** | **Medicine** | | **WMT20 ZH-EN** | |
|---|---|---|---|---|
| **Distance Function** | COMET | BLEU | COMET | BLEU |
| Original | 92.8 | 58.1 | **36.5** | **26.4** |
| One Candidate | **93.2** | **58.2** | 35.4 | 25.8 |
| Embed. Distance | 92.8 | 57.9 | **36.5** | 26.2 |

Table 5: Variants of the distance function in TWIST decoding. $f$ is the domain model and $g$ is the generic model for medical translation (German→English). $f$ is R2L and $g$ is L2R for WMT20 Chinese→English.

**Sensitivity Analysis on Distance Coefficients.** As discussed in §2.2, $\lambda_f$ and $\lambda_g$ weight the distance terms *from* $f$ and $g$ respectively. We tuned $\lambda_f$ and $\lambda_g$ on the dev. set from the range of $\{0.1, 0.3, 1.0, 3.0\}$. Fig. 5 visualizes how they affect the overall performance on the dev. sets. $\lambda_g > \lambda_f$ generally yields good performance, suggesting the effectiveness of the initial exploration by $g$ with relatively weaker guidance from $f$.

**Variants of Distance Functions.** We experiment with two variants of distance terms (Table 5): 1) *one candidate*, which measures the distance from the 1-best candidate from the other model (vs. minimization over multiple candidates; §2.2) and 2) *embed. distance*, which calculates the distance based on the Euclidean distance between the embeddings. Here the embeddings are taken from the output layer of the decoder. Overall, both variants yield similar performance to the original distance function, but the *one candidate* method has a substantial performance drop on WMT20 ZH-EN. Note also that the *embed. distance* method necessitates additional distance computations between the token embeddings. This result illustrates that our original distance function is a simple yet effective design choice.

**Examples.** Seen in Table 6 are example German→English translations from the medical domain. The left section presents a case where the domain model translates the technical term, *Spätdyskinesie*, into the corresponding English term: tardive dyskinesia. The generic model, on the other hand, generates a literal translation: late dyskinesia. In the right section, the domain model fails to handle the coordinate structure: 12.1% and 3.2% with aripiprazole vs. 12.1% with aripiprazole and 3.2% with placebo. Further, the final output has wording closer to the reference translation: trials vs. studies and bipolar patients vs. bipolar disorder. These examples illustrate that TWIST decoding benefits from the complementary strengths of the domain and generic models.

## 5 Further Related Work

**Decoding from Multiple Models.** Much early work proposed methods to generate text from multiple models especially for machine translation (often called *consensus-based decoding*; Bangalore et al., 2001, 2002; Matusov et al., 2006; Rosti et al., 2007; Sim et al., 2007; Hildebrand and Vogel, 2008). Most of these methods limit their search space to n-best candidates from individual translation models (Li et al., 2009), contrasting with our TWIST decoding where one model can update its translation outputs under the guidance of another model. *Collaborative decoding* (Li et al., 2009) trains a separate feature-based scorer that measures the consensus between phrase-based Chinese-to-English translation models. Several recent works proposed inference algorithms for decoding from multiple generators for specific tasks, such as detoxification and abductive reasoning (West et al., 2021; Liu et al., 2021).

**Alternatives to Left-to-Right Decoding.** We showed that TWIST decoding can be used to benefit from models with diverging generation order. Several prior works proposed approaches for generating text in a different fashion than the standard left-to-right order. For example, much recent work explored non-autoregressive generation (Gu et al., 2018; Lee et al., 2018; Mansimov et al., 2019; Ghazvininejad et al., 2019; Kasai et al., 2020, *inter alia*) primarily to parallelize and speed up inference. More specifically, several works introduced training and/or inference algorithms that combine left-to-right and right-to-left models for machine translation (Zhou et al., 2019) and commonsense inference (Zaidi et al., 2020). Qin et al. (2020) incorporated right (future) context into a left-to-right language model by iterative gradient-based updates

| Medicine | Domain (Isolation) 👍 Generic (Isolation) 👎 | Domain (Isolation) 👎 Generic (Isolation) 👍 |
|---|---|---|
| **Reference** | If signs and symptoms of tardive dyskinesia appear in a patient on ABILIFY, dose reduction or discontinuation should be considered. | In placebo-controlled trials, the incidence of akathisia in bipolar patients was 12.1% with aripiprazole and 3.2% with placebo. |
| **Domain** | If signs and symptoms of tardive dyskinesia appear in one patient on ABILIFY, a dose reduction or discontinuation should be considered. | In placebo-controlled trials, the incidence of akathisia in bipolar disorder was 12.1% and 3.2% with aripiprazole. |
| **Generic** | If a patient treated with ABILIFY shows signs and symptoms of late dyskinesia, it should be considered to reduce the dose or stop treatment. | In placebo-controlled studies, the incidence of akathisia in bipolar patients was 12.1% with aripiprazole and 3.2% with placebo. |
| **Twist**<br>$f$: Domain<br>$g$: Generic | If signs and symptoms of tardive dyskinesia appear in one patient on ABILIFY, a dose reduction or discontinuation should be considered. | In placebo-controlled trials, the incidence of akathisia in bipolar patients was 12.1% with aripiprazole and 3.2% with placebo. |

Table 6: Example outputs from machine translation on the medical domain. For Twist decoding, $f$ is the domain model, and $g$ is the generic model. In the left section, the generic model fails to capture technical terminology (late dyskinesia vs. tardive dyskinesia for the German term, *Spätdyskinesie*), and Twist decoding chooses the correct term of tardive dyskinesia from the domain model. In the right example, the domain model has a problem in coordination (12.1% and 3.2% with aripirazole vs. 12.1% with aripirazole and 3.2% with placebo), and Twist decoding successfully benefits from the accurate translation of the generic model.

on the output representations. Those algorithms are designed specifically for the combination of left-to-right and right-to-left generation and cannot be easily extended to more general situations, such as diverging tokenization and vocabularies where Twist decoding has been shown effective.

## 6 Conclusion

We presented Twist decoding, a general inference algorithm that generates text from diverse models without the assumption of a shared vocabulary, tokenization, or generation order. Our method enables diverse models to guide each other, thereby outperforming individual models over various scenarios, even when one of the models is much weaker because of limited data. We also demonstrated that Twist decoding can be viewed as a generalization and improvement of the commonly-adopted reranking heuristic. As it only requires a small change in code, we hope that researchers and practitioners will explore complementary strengths of diverse generation models through Twist decoding.

## Limitations

We evaluated our decoding method that combines generation models both on machine translation and scientific paper summarization over several scenarios: combining 1) generic and domain-specific models, 2) left-to-right and right-to-left generation models, and 3) models that generate using different conditioning inputs. Our machine translation experiments span diverse domains, including medical and legal text. We also presented results from recent English-to-German and Chinese-to-English WMT data. Nonetheless, our domain translation experiments are limited to German-to-English, and we only dealt with scientific papers written in English, mainly due to availability of data. There are also many other language generation tasks for which our method can be useful. Since we open-source our codebase built on top of a popular library, we hope that practitioners will use it for applications of their interest and further assess our decoding algorithm in many application scenarios.

Evaluating language generation remains a challenging research problem. We carefully set up our experiments to mitigate potential evaluation issues. The WMT 2020 test data consist only of news text written in the original language, in contrast to the test data from WMT 2018 (Bojar et al., 2018) or earlier. The WMT 2020 EN→DE and DE→EN test data that we used thus come from completely different documents. This avoids the translationese effect that would overestimate the translation performance due to the simplicity of translated text (Graham et al., 2020). Moreover, the WMT 2020 test data for English-to-German and Chinese-to-English translation have multiple reference translations per instance, which increases the correlation of reference-based, automatic evaluations with human judgment (Kasai et al., 2022a). We presented results using automatic metrics from recent work (Rei et al., 2020b) as well as conventional, n-gram overlap metrics (Papineni et al., 2002; Lin, 2004). Recent automatic metrics have shown to have higher correlation with human judgments, but human judgments are sometimes inconsistent, especially when crowdsourced (Clark et al., 2021; Kasai et al., 2022c). Since our decoding method is a simple modification of the widely-used beam search algorithm, we hope that it will be tested and used in real-world systems of language generation.

## Acknowledgements

We thank Hila Gonen, Phillip Keung, the ARK group at the UW, and the Mosaic team at the Allen Institute for AI for their helpful feedback on this work. This work was supported in part by the DARPA MCS program through NIWC Pacific (N66001-19-2-4031) and Google Cloud Compute. Hao Peng was supported by a Google Ph.D. Fellowship.

## References

Farhad Akhbardeh, Arkady Arkhangorodsky, Magdalena Biesialska, Ondřej Bojar, Rajen Chatterjee, Vishrav Chaudhary, Marta R. Costa-jussa, Cristina España-Bonet, Angela Fan, Christian Federmann, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Leonie Harter, Kenneth Heafield, Christopher Homan, Matthias Huck, Kwabena Amponsah-Kaakyire, Jungo Kasai, Daniel Khashabi, Kevin Knight, Tom Kocmi, Philipp Koehn, Nicholas Lourie, Christof Monz, Makoto Morishita, Masaaki Nagata, Ajay Nagesh, Toshiaki Nakazawa, Matteo Negri, Santanu Pal, Allahsera Auguste Tapo, Marco Turchi, Valentin Vydrin, and Marcos Zampieri. 2021. Findings of the 2021 conference on machine translation (WMT21). In *Proc. of WMT*.

Akari Asai, Xinyan Yu, Jungo Kasai, and Hannaneh Hajishirzi. 2021. One question answering model for many languages with cross-lingual dense passage retrieval. In *Proc. of NeurIPS*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Srinivas Bangalore, German Bordel, and Giuseppe Riccardi. 2001. Computing consensus translation from multiple machine translation systems. In *Proc. of ASRU*.

Srinivas Bangalore, Vanessa Murdock, and Giuseppe Riccardi. 2002. Bootstrapping bilingual data using consensus translation for a multilingual instant messaging system. In *Proc. of COLING*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proc. of EMNLP*.

Loïc Barrault, Magdalena Biesialska, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Matthias Huck, Eric Joanis, Tom Kocmi, Philipp Koehn, Chi-kiu Lo, Nikola Ljubešić, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Santanu Pal, Matt Post, and Marcos Zampieri. 2020. Findings of the 2020 conference on machine translation (WMT20). In *Proc. of WMT*.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proc. of WMT*.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proc. of WMT*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proc. of NeurIPS*.

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel Weld. 2020. TLDR: Extreme summarization of scientific documents. In *Findings of EMNLP*.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proc. of ACL*.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proc. of COLING*.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proc. of ACL*.

Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *CL*.

Michael Collins, Brian Roark, and Murat Saraclar. 2005. Discriminative syntactic language modeling for speech recognition. In *Proc. of ACL*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proc. of ACL*.

Orhan Firat, Baskaran Sankaran, Yaser Al-onaizan, Fatos T. Yarman Vural, and Kyunghyun Cho. 2016. Zero-resource translation with multi-lingual neural machine translation. In *Proc. of EMNLP*.

Markus Freitag and Yaser Al-Onaizan. 2017. Beam search strategies for neural machine translation. In *Proc. of WMT*.

Markus Freitag, Yaser Al-Onaizan, and Baskaran Sankaran. 2017. Ensemble distillation for neural machine translation.

Markus Freitag, George Foster, David Grangier, Viresh Ratnakar, Qijun Tan, and Wolfgang Macherey. 2021. Experts, errors, and context: A large-scale study of human evaluation for machine translation. *TACL*.

Markus Freitag, David Grangier, and Isaac Caswell. 2020. BLEU might be guilty but references are not innocent. In *Proc. of EMNLP*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke S. Zettlemoyer. 2019. Mask-predict: Parallel decoding of conditional masked language models. In *Proc. of EMNLP*.

Yvette Graham, Barry Haddow, and Philipp Koehn. 2020. Translationese in machine translation evaluation.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O. K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *Proc. of ICLR*.

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

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mengnan Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving human parity on automatic Chinese to English news translation.

Almut Silja Hildebrand and Stephan Vogel. 2008. Combination of machine translation systems via hypothesis selection from combined n-best lists. In *Proc. of AMTA: Student Research Workshop*.

Junjie Hu, Mengzhou Xia, Graham Neubig, and Jaime Carbonell. 2019. Domain adaptation of neural machine translation by lexicon induction. In *Proc. of ACL*.

Kenji Imamura and Eiichiro Sumita. 2017. Ensemble and reranking: Using multiple models in the NICT-2 neural machine translation system at WAT2017. In *Proc. of WAT*.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2017. Tying word vectors and word classifiers: A loss framework for language modeling. In *Proc. of ICLR*.

Jungo Kasai, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020. Non-autoregressive machine translation with disentangled context transformer. In *Proc. of ICML*.

Jungo Kasai, Nikolaos Pappas, Hao Peng, James Cross, and Noah A. Smith. 2021a. Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation. In *Proc. of ICLR*.

Jungo Kasai, Hao Peng, Yizhe Zhang, Dani Yogatama, Gabriel Ilharco, Nikolaos Pappas, Yi Mao, Weizhu Chen, and Noah A. Smith. 2021b. Finetuning pretrained transformers into RNNs. In *Proc. of EMNLP*.

Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Lavinia Dunagan, Jacob Morrison, Alexander R. Fabbri, Yejin Choi, and Noah A. Smith. 2022a. Bidimensional leaderboards: Generate and evaluate language hand in hand. In *Proc. of NAACL*.

Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Dragomir Radev, Yejin Choi, and Noah A. Smith. 2022b. Beam decoding with controlled patience.

Jungo Kasai, Keisuke Sakaguchi, Lavinia Dunagan, Jacob Morrison, Ronan Le Bras, Yejin Choi, and Noah A. Smith. 2022c. Transparent human evaluation for image captioning. In *Proc. of NAACL*.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *Proc. of EMNLP*.

Shun Kiyono, Takumi Ito, Ryuto Konno, Makoto Morishita, and Jun Suzuki. 2020. Tohoku-AIP-NTT at WMT 2020 news translation task. In *Proc. of WMT*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proc. of MT Summit*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proc. of ACL Demo and Poster Sessions*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proc. of NGT*.

Jason D. Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proc. of EMNLP*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020.

BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proc. of ACL*.

Mu Li, Nan Duan, Dongdong Zhang, Chi-Ho Li, and Ming Zhou. 2009. Collaborative decoding: Partial hypothesis re-ranking using translation consensus between decoders. In *Proc. of ACL*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proc. of Text Summarization Branches Out*.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proc. of ACL*.

Elman Mansimov, Alex Wang, and Kyunghyun Cho. 2019. A generalized framework of sequence generation with application to undirected sequence models.

Evgeny Matusov, Nicola Ueffing, and Hermann Ney. 2006. Computing consensus translation for multiple machine translation systems using enhanced hypothesis alignment. In *Proc. of EACL*.

Clara Meister, Ryan Cotterell, and Tim Vieira. 2020a. If beam search is the answer, what was the question? In *Proc. of EMNLP*.

Clara Meister, Tim Vieira, and Ryan Cotterell. 2020b. Best-first beam search. *TACL*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proc. of EMNLP*.

Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. In *Proc. of WMT*.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proc. of NAACL*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL Demonstrations*.

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

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. 2021. Random feature attention. In *Proc. of ICLR*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proc. of WMT*.

Ofir Press and Lior Wolf. 2017. Using the output embedding to improve language models. In *Proc. of EACL*.

Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. 2020. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In *Proc. of EMNLP*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *JLMR*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020a. COMET: A neural framework for MT evaluation. In *Proc. of EMNLP*.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020b. Unbabel's participation in the WMT20 metrics shared task. In *Proc. of WMT*.

Antti-Veikko Rosti, Necip Fazil Ayan, Bing Xiang, Spyros Matsoukas, Richard Schwartz, and Bonnie Dorr. 2007. Combining outputs from multiple machine translation systems. In *Proc. of NAACL*.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. Green AI. *CACM*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proc. of ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proc. of ACL*.

Libin Shen and Aravind K. Joshi. 2003. An SVM-based voting algorithm with application to parse reranking. In *Proc. of NAACL*.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *Proc. of NAACL*.

Khe Chai Sim, William J. Byrne, Mark J. F. Gales, Hichem Sahbi, and Philip C. Woodland. 2007. Consensus network decoding for statistical machine translation system combination. In *Proc. of ICASSP*.

Felix Stahlberg and Bill Byrne. 2019. On NMT search errors and model errors: Cat got your tongue? In *Proc. of EMNLP*.

Felix Stahlberg, James Cross, and Veselin Stoyanov. 2018. Simple fusion: Return of the language model. In *Proc. of WMT*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proc. of ACL*.

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

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proc. of LREC*.

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 *Proc. of NeurIPS*.

Longyue Wang, Mu Li, Fangxu Liu, Shuming Shi, Zhaopeng Tu, Xing Wang, Shuangzhi Wu, Jiali Zeng, and Wen Zhang. 2021. Tencent translation system for the WMT21 news translation task. In *Proc. of WMT*.

Peter West, Ximing Lu, Ari Holtzman, Chandra Bhagavatula, Jena D. Hwang, and Yejin Choi. 2021. Reflective decoding: Beyond unidirectional generation with off-the-shelf language models. In *Proc. of ACL*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Najam Zaidi, Trevor Cohn, and Gholamreza Haffari. 2020. Decoding as dynamic programming for recurrent autoregressive models. In *Proc. of ICLR*.

Long Zhou, Jiajun Zhang, and Chengqing Zong. 2019. Synchronous bidirectional neural machine translation. *TACL*.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. 2016. The United Nations parallel corpus v1.0. In *Proc. of LREC*.

# Appendices

## A Hyperparameters and Settings

We provide training and implementation details for easy replication of our work.

### A.1 Domain Machine Translation

We generally follow the preprocessing and subword tokenization from Koehn and Knowles (2017); Hu et al. (2019). Table 7 lists the hyperparameters and setting on `fairseq` that we use for all domain-specific translation models. All embeddings are shared (Press and Wolf, 2017; Inan et al., 2017). We choose the checkpoint that achieved the best loss on the validation data.

| **Hyperparameter** | **Value** |
|---|---|
| label smoothing | 0.1 |
| # max tokens | 8192 |
| dropout rate | 0.1 |
| encoder embedding dim | 512 |
| encoder ffn dim | 2048 |
| # encoder attn heads | 8 |
| decoder embedding dim | 512 |
| decoder ffn dim | 2048 |
| # decoder attn heads | 8 |
| max source positions | 1024 |
| max target positions | 1024 |
| Adam lrate | $5 \times 10^{-4}$ |
| Adam $\beta_1$ | 0.9 |
| Adam $\beta_2$ | 0.98 |
| lr-scheduler | inverse square |
| warm-up lr | $1 \times 10^{-7}$ |
| # warmup updates | 4000 |
| # max updates | 600K |
| # GPUs | 8 |
| length penalty | 0.6 |

Table 7: Domain translation `fairseq` hyperparameters and setting. We generally follow the base-sized configuration from Vaswani et al. (2017).

### A.2 Left-to-Right and Right-to-Left

**WMT20 ZH-EN** Table 8 lists the hyperprameters and setting on `fairseq` that we use for left-to-right and right-to-left models on the WMT20 ZH-EN dataset. We generally follow the preprocessing and tokenization from Kasai et al. (2022a). We use `newstest-2019` as the dev. set and the official training data.[12] We apply Moses tokenization (Koehn et al., 2007) and BPE with 32K operations (Sennrich et al., 2016b) to English text. We tokenize Chinese text with the Jieba package,[13] following Hassan et al. (2018). Separately from English, BPE with 32K operations is then applied to Chinese. The decoder input and output embeddings are tied.

**WMT20 EN-DE** The same hyperparameters are chosen as in WMT20 ZH-EN (Table 8). We again follow Kasai et al. (2022a) and preprocess both English and German text by the Moses tokenizer and *joint* BPE with 32K operations. All embeddings are shared.

| **Hyperparameter** | **Value** |
|---|---|
| label smoothing | 0.1 |
| # max tokens | 4096 |
| dropout rate | 0.1 |
| encoder embedding dim | 1024 |
| encoder ffn dim | 4096 |
| # encoder attn heads | 16 |
| decoder embedding dim | 1024 |
| decoder ffn dim | 4096 |
| # decoder attn heads | 16 |
| max source positions | 1024 |
| max target positions | 1024 |
| Adam lrate | $5 \times 10^{-4}$ |
| Adam $\beta_1$ | 0.9 |
| Adam $\beta_2$ | 0.98 |
| lr-scheduler | inverse square |
| warm-up lr | $1 \times 10^{-7}$ |
| # warmup updates | 4000 |
| # max updates | 600K |
| # GPUs | 8 |
| length penalty | 0.6 |

Table 8: L2R and R2L translation `fairseq` hyperparameters and setting. We generally follow the large-sized configuration from Vaswani et al. (2017).

### A.3 SciTLDR

We use two BART-based pretrained models from Cachola et al. (2020): the abstract-only version of BART and the AIC version of CATTS$_{\text{XSUM}}$.[14] These two models are both BART-based models; CATTS$_{\text{XSUM}}$ is obtained by finetuning BART on the XSUM dataset (Narayan et al., 2018) with multitask scaffolding (Cachola et al., 2020).

### A.4 $\lambda$ Tuning

We tune $\lambda_f$ and $\lambda_g$ from $\{0.1, 0.3, 1.0, 3.0\}$, based on the dev. BLEU/ROUGE-L score on machine translation and paper summarization, respectively. Table 9 reports the selected $\lambda$ values in all scenarios.

[12] `http://www.statmt.org/wmt20/translation-task.html`.

[13] `https://github.com/fxsjy/jieba`.

[14] They are both available at `https://github.com/allenai/scitldr`.

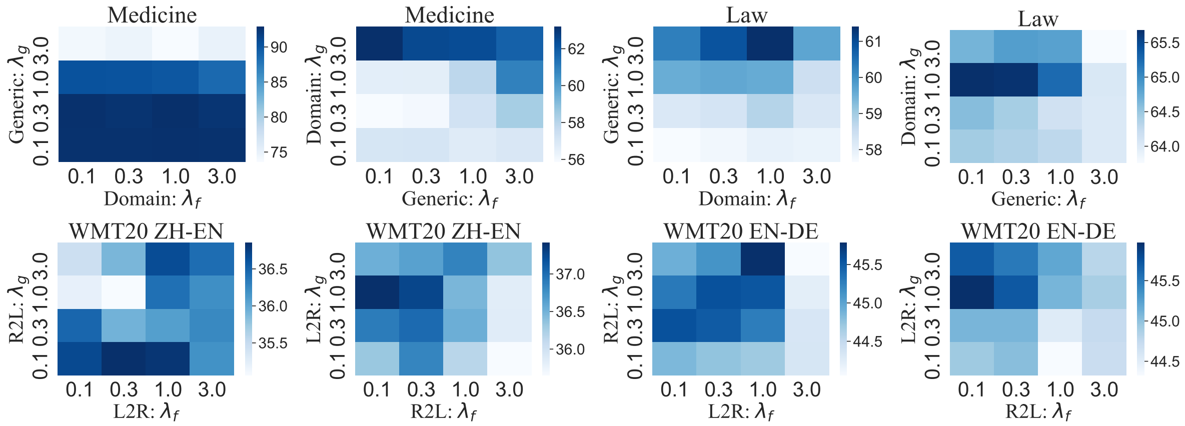

Figure 6: Dev. set performance measured in the COMET score (Rei et al., 2020a,b) with varying $\lambda_f$ and $\lambda_g$.

| | | | **Tuned** $\lambda$ | |
|---|---|---|---|---|
| **Dataset** | $f$ | $g$ | $\lambda_f$ | $\lambda_g$ |
| Medicine | Domain | Generic | 0.1 | 0.3 |
| | Generic | Domain | 0.1 | 3.0 |
| Law | Domain | Generic | 1.0 | 0.1 |
| | Generic | Domain | 0.1 | 3.0 |
| Koran | Domain | Generic | 1.0 | 3.0 |
| | Generic | Domain | 0.3 | 3.0 |
| Subtitles | Domain | Generic | 1.0 | 1.0 |
| | Generic | Domain | 1.0 | 1.0 |
| WMT20 ZH-EN | L2R | R2L | 1.0 | 3.0 |
| | R2L | L2R | 0.1 | 3.0 |
| WMT20 EN-DE | L2R | R2L | 0.3 | 0.3 |
| | R2L | L2R | 0.1 | 0.3 |
| SciTLDR | Abst. | AIC | 1.0 | 3.0 |
| | AIC | Abst. | 0.3 | 3.0 |

Table 9: Selected $\lambda_f$ and $\lambda_g$ values.

## B Sensitivity Analysis on $\lambda$

Fig. 6 presents the sensitivity analysis in the COMET score over many scenarios. Apart from a few exceptions, $\lambda_g > \lambda_f$ tends to yield good performance, suggesting the effectiveness of the initial exploration by $g$ with relatively weaker guidance from $f$.