Self-Guided Curriculum Learning for Neural Machine Translation

Lei Zhou*
Nagoya University
zhou.lei@a.mbox.nagoya-u.ac.jp

Liang Ding
The University of Sydney
ldin3097@sydney.edu.au

Kevin Duh
Johns Hopkins University
kevinduh@cs.jhu.edu

Shinji Watanabe
Carnegie Mellon University
shinjiw@ieee.org

Ryohei Sasano
Nagoya University
sasano@i.nagoya-u.ac.jp

Koichi Takeda
Nagoya University
takedasu@i.nagoya-u.ac.jp

Abstract

In supervised learning, a well-trained model should be able to recover ground truth accurately, i.e. the predicted labels are expected to resemble the ground truth labels as much as possible. Inspired by this, we formulate a difficulty criterion based on the recovery degrees of training examples. Motivated by the intuition that after skimming through the training corpus, the neural machine translation (NMT) model "knows" how to schedule a suitable curriculum according to learning difficulty, we propose a self-guided curriculum learning strategy that encourages the NMT model to learn from easy to hard on the basis of recovery degrees. Specifically, we adopt sentence-level BLEU score as the proxy of recovery degree. Experimental results on translation benchmarks including WMT14 English⇒German and WMT17 Chinese⇒English demonstrate that our proposed method considerably improves the recovery degree, thus consistently improving the translation performance.

1 Introduction

Inspired by the learning behavior of humans, Curriculum Learning (CL) for neural network training starts from a basic idea of "starting small", namely better to start from easier aspects of a task and then progress towards aspects with increasing level of difficulty (Elman, 1993). Bengio et al. (2009) achieves significant performance boost on several tasks by forcing models to learn training examples following an order from "easy" to "difficult". They further explain CL method with two important constituents: *how to rank training examples* by learning difficulty and *how to schedule the presentation of training examples* based on that rank.

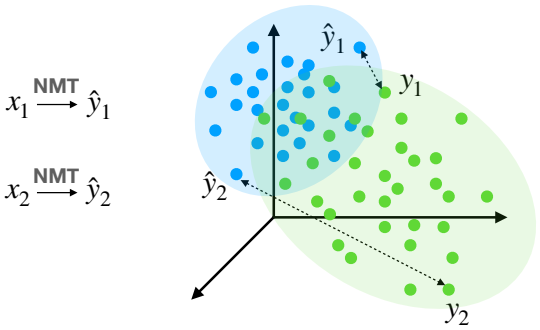


Figure 1: The NMT model is well-trained on parallel corpus $\mathbb{D}$, $\{(x_1, y_1), (x_2, y_2)\} \in \mathbb{D}$. $\hat{y}_i$ is translated from x_i. The distance between the *ground truth* y_i and the *NMT generated hypothesis* $\hat{y}_i$ represents the recovery degree (dashed arrows), which is computed by sentence-level BLEU in our case. Blue- and green-colored examples represent the NMT learned distribution and the empirical distribution, respectively. Taking x_1 and x_2 as the input, the training example (x_1, y_1) shows a better *recovery degree*, which means it's easier to be mastered than (x_2, y_2).

In the field of neural machine translation (NMT), empirical studies have shown that CL strategies contribute to both convergence speed and model performance (Zhang et al., 2018; Platanios et al., 2019; Zhang et al., 2019; Liu et al., 2020; Zhan et al., 2021; Ruiter et al., 2020). These CL strategies vary by difficulty criteria and curriculum schedules. Early difficulty criterion depends on manually crafted features and prior knowledge such as sentence length and word rarity (Kocmi and Bojar, 2017). The drawback lies in the fact that humans understand learning difficulty differently from NMT models. Recent works choose to derive difficulty criteria based on the probability distribution of training examples to approximate the perspective of NMT models. For instance, Platanios et al. (2019) turn discrete numerical difficulty scores into relative probabilities and then construct

*Part of this work was done when the first author visited CLSP, JHU.

Proceedings of the 18th International Conference on Spoken Language Translation, pages 206–214
Bangkok, Thailand (Online), August 5–6, 2021.

the difficulty criterion, while others derive difficulty criterion from independently trained language model (Zhang et al., 2019; Dou et al., 2020; Liu et al., 2020) and word embedding model (Zhou et al., 2020b). Xu et al. (2020) derive difficulty criterion from the NMT model in the training process. And these difficulty criteria are applied to either fixed curriculum schedule (Cirik et al., 2016) or dynamic one (Platanios et al., 2019; Liu et al., 2020; Xu et al., 2020; Zhou et al., 2020b).

A well-trained NMT model estimates the optimal probability distribution mapping from the source language to the target language, which is assumed to be able to recover the ground truth translations accurately (Liu et al., 2021). However, if we perform inference on the training set, many of the predictions are inconsistent with the references. It reflects the **distribution shift** between the NMT model leaned distribution and the empirical distribution of training corpus, as Figure 1 illustrated. For a training example, a high recovery degree between prediction and ground-truth target sentence means it's easier to be mastered by the NMT model while a lower recovery degree means it's more difficult (Ding and Tao, 2019; Wu et al., 2020b). To this end, we employ this recovery degree as the difficulty criterion, where the recovery degree is computed by the sentence-level BLEU. We put forward an analogy of this method that humans can schedule a personal and effective curriculum after skimming over a textbook, namely *self-guided curriculum*.

In this work, we cast the recovery degree of each training example as its learning difficulty, enforcing the NMT model to learn from examples with higher recovery degrees to those with lower degrees. Also, we implement our proposed recovery-based difficulty criterion with fixed and dynamic curriculum schedules. Experimental results on two machine translation benchmarks, i.e., WMT14 En-De and WMT17 Zh-En, demonstrate that our proposed self-guided CL can alleviate the distribution shift problem in vanilla NMT models, thus consistently boosting the performance.

2 Problem Definition

For a better interpretation of curriculum learning for neural machine translation, we put the discussion of various CL strategies into a probabilistic perspective. Such perspective also motivates us to derive this recovery-based difficulty criterion.

2.1 Neural Machine Translation

Let $\mathcal{S}$ and $\mathcal{T}$ represent the probability distributions over all possible sequences of tokens in source and target languages, respectively. We denote the distribution of a random source sentence $\mathbf{x}$ and $\mathbf{y}$ as $P_{\mathcal{S}}(\mathbf{x})$ and $P_{\mathcal{T}}(\mathbf{y})$. NMT model is to learn a conditional distribution $P_{\mathcal{S},\mathcal{T}}(\mathbf{y}|\mathbf{x})$ with a probabilistic model $P(y|x;\theta)$ parameterized by θ, where θ is estimated by minimizing the objective:

$$J(\theta) = -\mathbb{E}_{x,y\sim P_{\mathcal{S},\mathcal{T}}(\mathbf{x},\mathbf{y})} \log P(y|x;\theta) \quad (1)$$

2.2 Curriculum Learning for Neural Machine Translation

CL methods decompose the NMT model training into K phases, enforcing the optimization trajectory in parameter space to visit a series of points $\theta^1, \ldots, \theta^K$. Each training phase can be viewed as a sub-optimal process, optimized on a subset $\mathbb{D}_k$ of the training corpus $\mathbb{D}$:

$$J(\theta^k) = -\mathbb{E}_{x,y\sim \hat{P}_{\mathbb{D}_k}} \log P(y|x;\theta^k) \quad (2)$$

where $\hat{P}_{\mathbb{D}_k}$ is the empirical distribution of $\mathbb{D}_k$. According to the definition of curriculum learning, the optimization difficulty increases from $J(\theta^1)$ to $J(\theta^K)$ (Bengio et al., 2009). In practice, it's achieved by grouping training examples into subsets in ascending order of learning difficulty. The process splitting $\mathbb{D}$ into K subsets can be formulated as follows:

- score $\leftarrow d(z^n)$, $z^n \in \mathbb{D}$, where $d(\cdot)$ is a difficulty criterion
- For $k = 1, \ldots, K$ do; $\mathbb{D}_k \leftarrow \{z^n | \text{Constraint}(d(z^n), k)\}$

z represents examples in $\mathbb{D}$, $\mathbb{D} = \{z^n\}_{n=1}^N$, $z^n = (x^n, y^n)$. Training corpus $\mathbb{D}$ is split into K subsets $\{\mathbb{D}_1, \ldots, \mathbb{D}_K\}$, that $\bigcup_{k\in K} \mathbb{D}_k = \mathbb{D}$.

With these notations, we review the DIFFICULTY CRITERIA in existing CL methods from a probabilistic perspective as these methods generally derive difficulty criteria from a probabilistic distribution. For example:

Explicit Feature $d(x^n) = P_{\mathbb{D}}(\text{Feature}(x^n))$, where $\text{Feature}(\cdot)$ is handcrafted features and linguistic prior knowledge such as sentence length and word rarity. With the cumulative density function (CDF), numerical scores are mapped into a relative probability distribution over all training

examples (Platanios et al., 2019). Only features of source sentences are taken into consideration in their practice.

Language Model $d(x^n) = -\frac{1}{I}\log P_{\mathrm{LM}}(w_1^n, \ldots, w_I^n)$, where a language model is adopted to estimate the perplexity of each sentence $x = w_1, \ldots, w_I$. Language models trained on source and target side can be used jointly, e.g., $d(x^n) + d(y^n)$ (Zhou et al., 2020b). In other works (Zhang et al., 2019; Dou et al., 2020), language models in different domains are adopted to compute the cross-entropy difference of each sentence, indicating its difficulty for domain adaptation.

Word Embedding $d(x^n) = \sum_{i=1}^{I} \|\mathbf{w}_i^n\|$, where $\mathbf{w}_1, \ldots, \mathbf{w}_I$ is a distributed representation of source sentence x mapped through a independent word embedding model. In the case of Liu et al. (2020), the norm of word vector on the source side is used as the difficulty criterion. They also use the CDF function to assure the difficulty scores are within $[0, 1]$.

NMT Model $d(z^n; \theta^k) = \frac{l(z^n;\theta^k) - l(z^n;\theta^{k-1})}{l(z^n;\theta^{k-1})}$, $l(z^n; \theta^k) = -\log P(y^n|x^n; \theta^k)$, where θ^k represents the NMT model parameters at the kth training phase. The decline of loss is defined as the difficulty criterion in Xu et al. (2020). Besides, the score of cross-lingual patterns may also be a proper difficulty criterion for NMT (Ding et al., 2020a; Zhou et al., 2020a; Wu et al., 2021), which we leave as the future work.

We now turn to CURRICULUM SCHEDULING. There are two controlling factors, extraction of training set and training phase duration. In other words, how to split training corpus into subsets and when to load them. Given K mutual exclusive subsets $\{\mathbb{D}_1, \ldots, \mathbb{D}_K\} \subseteq \mathbb{D}$, there are two general regimens loading them as training progresses: one pass and baby steps. In *one pass* regimen, k subsets $\mathbb{D}_k$ are loaded as training set one by one, while in *baby steps* regimen, these subsets are merged into the current training set one by one (Cirik et al., 2016). According to Cirik et al. (2016), baby steps outperforms one pass. Later approaches generally take the idea of baby steps in that easy examples are not cast aside while the probability increases for difficulty examples to be batched.

On top of baby steps, we can summarize existing works into two schedule settings: fixed schedule and dynamic schedule. In *fixed schedule*, both training set extraction and training phase duration are fixed (Cirik et al., 2016; Zhang et al., 2019). The size of the training set scales up by a certain proportion of the total training examples, usually $|\mathbb{D}_k| = N/K$ at the beginning of a new training phase. And each training phase spends a fixed number of training steps. In *dynamic schedule*, either training set extraction or training phase duration is dynamic. Depending on which controlling factor is dynamic, we group existing dynamic schedules into two types: the competence type and the self-paced type. Competence-based CL method is proposed by (Platanios et al., 2019). In *competence* type of dynamic schedule, training set extraction is dynamic while the training phase duration is fixed. At the beginning of a training phase, the CL algorithm compute the model competence c at the moment, then extract examples with difficulty scores lower than c as the training set for the current phase, $\{z^n | d(z^n) \leq c, z^n \in \mathbb{D}\}$. For K training phases, the competence-based schedule is to determine $(K-1)$ upper limits with a scale factor within range of $d(z^n)$, which is $[0, 1]$. Platanios et al. (2019) take training steps $1, \ldots, t, \ldots, T$ as the scale factor, thus the general form of competence function is : $c(t) = \min\left(1, \sqrt[p]{t\frac{1-c_0^p}{T} + c_0^p}\right)$. Recent works develop model competence by introducing different scale factors, such as the norm of the source embedding of the NMT model (Liu et al., 2020) and BLEU score on validation set (Xu et al., 2020). Another type of dynamic schedule is the *self-paced* one (Jiang et al., 2015; Zhou et al., 2020b), in which training set extraction is fixed while the training phase duration is dynamic. After a training phase begins, it goes on until convergence or until meeting certain conditions. For example in Zhou et al. (2020b), model training will progress to the next phase if the model uncertainty stops decline.

3 Methodology

As mentioned above, due to the distribution shift problem, predictions made by a well-trained vanilla NMT model can be inconsistent with the references when performing inference on the training set. Training examples with higher recovery degrees are easier to be masted by the NMT model while those with lower recovery degrees are likely to be more difficult. Table 1 shows a comparison of two training examples with distant recovery de-

High Recovery Degree (BLEU 77.01)	
Source	该动议如被通过, 提案或修正案中后被核准的各部分应合成整体再付表决。
Reference	If the motion for division is carried, those parts of the proposal or of the amendment which are subsequently approved shall be put to the vote as a whole.
Prediction	If the motion for division is carried , those parts of fm draft resolution or of the amendment that are subsequently approved shall be put to the vote as a whole.
Low Recovery Degree (BLEU 5.19)	
Source	并且慢慢地, 非常缓慢地把头抬到它的眼睛正好可以直视哈利的位置便停了下来。它朝哈利使了一下眼色。
Reference	Slowly, very slowly, it raised its head until its eyes were on a level with Harry's. It winked.
Prediction	Slowly and very slowly – thinking his head up, still adding to poster him gladly stare to stopped Harry's face alone, and then blurted it out to Harry like a stop.

Table 1: Examples from WMT17 Chinese⇒English with distant recovery degrees measured by sentence-level BLEU score. We mark prediction errors with red underline.

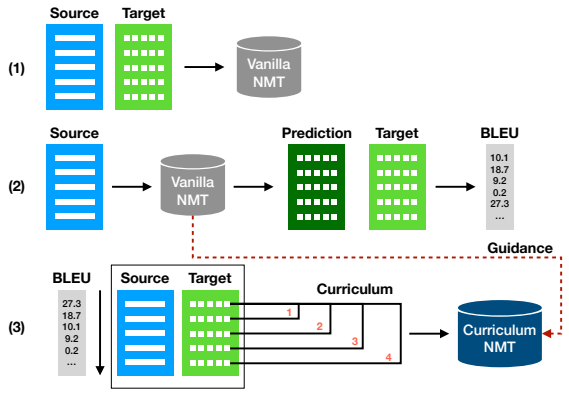


Figure 2: Workflow of self-guided CL strategy

grees.

In this section, we first introduce our recovery-based difficulty criterion and then propose to implement this criterion with fixed and dynamic curriculum schedules. The workflow of our proposed self-guided curriculum learning strategy is illustrated in Figure 2.

3.1 Difficulty Criterion

The objective function of the vanilla model can be written as an average distribution over the training corpus $\mathbb{D}$:

$$J(\varphi) = \mathbb{E}_{x,y\sim\hat{p}_{\mathbb{D}}} L(f(x^n;\varphi), y^n) \qquad (3)$$

where $f(x^n;\varphi)$ represents model's prediction and L is the loss function. As noted in Section 2, curriculum learning minimizes the objective $J(\theta)$ with a set of sub-optimal processes from easy to difficult. Examples that better fit into the average distribution learned by the vanilla model with parameter φ get higher recovery degrees. Starting curriculum learning on a set of examples with higher recovery degrees is to start optimizing $J(\theta)$ from a smaller parameter space in the neighborhood of parameter φ. In the machine translation scenario, we care more about model performance in terms of translation quality. So we choose BLEU score, the de facto automatic metric for MT, to measure the recovery degree. The difficulty criterion based on sentence-level BLEU score is as follows:

$$d(z^n) = -\text{BLEU}(f(x^n;\varphi), y^n) \qquad (4)$$

Other reference-based automatic metrics for MT are applicable in this difficulty criterion as well.

3.2 Curriculum Scheduling

Following basic operations of the baby steps regimen, we first split training corpus $\mathbb{D}$ into K mutual exclusive subsets $\{\mathbb{D}_1, \ldots, \mathbb{D}_K\}$, corresponding to K training phases. With difficulty criterion $d(\cdot)$, we define the corpus splitting function g:

$$\begin{aligned} &g(d(\cdot)) : \mathbb{D} \longrightarrow \{\mathbb{D}_1, \ldots, \mathbb{D}_K\}, \\ &\mid \forall a \in \mathbb{D}_k, \forall b \in \mathbb{D}_{k+1}, d(a) \leq d(b) \end{aligned} \qquad (5)$$

Then we explore both fixed and dynamic schedules:

Fixed In fixed schedule, the training duration of each training phase is predefined. At the beginning

Algorithm 1: Fixed Scheduling

Input: Parallel corpus $\mathbb{D} = \{z^n\}_{n=1}^N$, $z^n = (x^n, y^n)$

Train vanilla model φ on $\mathbb{D}$
Compute difficulty score $d(z^n), z^n \in \mathbb{D}$ with φ by Eq. 4
Split $\mathbb{D}$ into subsets $\{\mathbb{D}_1, \ldots, \mathbb{D}_K\}$ by Eq. 5
$\mathbb{D}_{\text{train}} = \varnothing$
for $k = 1, \ldots, K$ **do**
    $\mathbb{D}_{\text{train}} = \mathbb{D}_{\text{train}} \cup \mathbb{D}_k$
    for *training steps* $t = 1, \ldots, T$ **do**
        Train CL model θ^k on $\mathbb{D}_{\text{train}}$

Output: Trained CL model θ

of the kth training phase, subset $\mathbb{D}_k$ is merged into the current training set. After finished with T steps, the training progresses to the next phase $k+1$, see Algorithm 1:

Dynamic We follow the *self-paced* type of dynamic schedule as described in Section 2, in which training duration is dynamic while training set extraction is done before training starts. We define the condition of training phase progressing by the model recovery degree. In training phase k, if the CL model constantly demonstrates recovery degrees higher than the vanilla model on the newly merged subset $\mathbb{D}_k$, the CL model training will advance to the training phase $k+1$. For easier operation, we randomly sub-sample $\mathbb{D}'_k$ from $\mathbb{D}_k$ for model recovery validation. Based on the performance on $\{x^n, y^n\} \in \mathbb{D}'_k$, which is measured by corpus-level BLEU score, we compute model recovery degree of the CL model at current training phase k by:

$$o_c(k) = \text{BLEU}(f(x^n; \theta^k), y^n) \tag{6}$$

Similarly, with the same additional validation set $\mathbb{D}'_k$, we compute model recovery degree of the vanilla model by:

$$o_v(k) = \text{BLEU}(f(x^n; \varphi), y^n) \tag{7}$$

If $o_c > o_v$, training phase will progress to the next one. Otherwise, the current training phase will go on until it reaches the predefined maximum time steps T, and then moves to the next phase. The training process is as described in Algorithm 2.

Algorithm 2: Dynamic Scheduling

Input: Parallel corpus $\mathbb{D} = \{z^n\}_{n=1}^N$, $z^n = (x^n, y^n)$

Train vanilla model φ on $\mathbb{D}$
Compute difficulty score $d(z^n), z^n \in \mathbb{D}$ with φ Eq. 4
Split $\mathbb{D}$ into subsets $\{\mathbb{D}_1, \ldots, \mathbb{D}_K\}$ by Eq. 5
$\mathbb{D}_{\text{train}} = \varnothing$
for $k = 1, \ldots, K$ **do**
    $\mathbb{D}_{\text{train}} = \mathbb{D}_{\text{train}} \cup \mathbb{D}_k$
    for *training steps* $t = 1, \ldots, T$ **do**
        Train CL model θ^k on $\mathbb{D}_{\text{train}}$
        Compute model recovery degree o_c and o_v, Eq.6,7
        if $o_c > o_v$ **then**
            Stop and move to the next phase

Output: Trained CL model θ

4 Experiments

4.1 Datasets

We conduct experiments on two machine translation benchmarks: WMT'14 English⇒German (En-De) and WMT'17 Chinese⇒English (Zh-En). For En-De, the training set consists of 4.5 million sentence pairs. We use newstest2012 as the validation set and report test results on both newstest2014 and newtest2016 for fair comparison with existing approaches. For Zh-En, we follow (Hassan et al., 2018) to extract 20 million sentence pairs as the training set. We use newsdev2017 as the validation set and newstest2017 as the test set. Chinese sentences are segmented with a word segmentation toolkit Jieba[1]. Sentences in other languages are tokenized with Moses[2]. We learn Byte-Pair Encoding(BPE) (Sennrich et al., 2016) with 32k merge operations. And we learn BPE with a shared vocabulary for En-De. We use BLEU (Papineni et al., 2002) as the automatic metrics for computing recovery degree and evaluating model performance with statistical significance test (Collins et al., 2005).

4.2 Model Settings

We perform proposed CL method with the FAIRSEQ[3] (Ott et al., 2019) implementation of the

[1] https://github.com/fxshy/jieba
[2] https://github.com/mosesdecoder
[3] https://github.com/pytorch/fairseq

#	Systems	WMT14 EnDe		WMT16 EnDe		WMT17 ZhEn	
		BLEU	Δ	BLEU	Δ	BLEU	Δ
1	Transformer BASE	27.30	-	32.76‡	-	23.69†	-
2	w/ Competence-based CL	28.19†	-	32.84‡	-	24.30†	-
3	w/ Norm-based CL	28.81†	-	-	-	25.25†	-
4	w/ Uncertainty-aware CL	-	-	33.93‡	-	25.02‡	-
	This work						
5	Transformer BASE	27.63	-	33.03	-	23.78	-
6	w/ SGCL Fixed	28.16↑	0.53	33.55↑	0.52	24.65↑	0.87
7	w/ SGCL Dynamic	**28.62**⇑	0.99	**34.07**⇑	1.04	**25.34**⇑	1.56

Table 2: Experiment results on WMT14 En⇒De with newstest2014 and newstest2016, and WMT17 Zh⇒En. For baseline and existing CL methods, Row 1-4, "†" marks the results from Liu et al. (2020), and "‡" marks the results from Zhou et al. (2020b). Since Platanios et al. (2019) only report their results on En⇒De newstest2016, up to 30.16, which is lower than later implementations, we show the implemented results of the Competence-based CL method from Liu et al. (2020) and Zhou et al. (2020b) instead. For the results of our proposed methods, "⇑/↑" indicates significant difference ($p < 0.01/0.05$) from Transformer BASE.

Transformer BASE (Vaswani et al., 2017). For regularization, we use the dropout of 0.3 and 0.1 for En-De and Zh-En respectively, with label smoothing ϵ = 0.1. We train the model with a batch size of approximately 128K tokens. We use Adam (Kingma and Ba, 2015) optimizer. The learning rate warms up to 5×10^{-4} in the first 16K steps and then decays with the inverse square-root schedule. We evaluate the translation performance on an ensemble of the top 5 checkpoints to avoid stochasticity. We use shared embeddings for En-De experiments. All our experiments are conducted with 4 NVIDIA Quadro GV100 GPUs.

4.3 Curriculum Learning Settings

The vanilla model and the CL model share the same Transformer BASE setting. For the recovery degree, we let the trained vanilla model make predictions of source sentences in the training corpus with beam size set to 1 for we only need to reveal the recovery feature at the moment. Then we evaluate the predictions with sentence-level BLEU score. Specifically, we use `fairseq-score` to get sentence-level BLEU score, which implements smoothing method 3, i.e., NIST smoothing method (Chen and Cherry, 2014) by default. According to Zhou et al. (2020b), 4 baby steps is superior to those with larger baby steps, so we choose to decompose the CL training into 4 training phases. Implementing the proposed difficulty criterion, we investigate the performance of two curriculum schedules:

- **SGCL Fixed** represents self-guided curriculum learning with fixed schedule.
- **SGCL Dynamic** represents self-guided curriculum learning with dynamic schedule.

5 Results

Table 2 summarises our experimental results together with existing CL methods. Row 1 shows the results of the standard Transformer BASE on these benchmarks. Row 2-4 demonstrate results from existing curriculum learning approaches. Row 5 shows the results of our Transformer BASE implementation, and row 6-7 are the results of our proposed CL models. For En-De, if existing works report results on one of newstest2014 and newstest2017, then only the reported one is shown. We report results on them both for fair comparison.

We train our implemented baseline of Transformer BASE and proposed CL models for 300k steps. For both SGCL Fixed and SGCL Dynamic methods, we observe superior performances over the strong baseline on all three test sets of two benchmarks, which agree with existing approaches that curriculum learning can facilitate the NMT model. And if we compare the two scheduling methods, SGCL Dynamic outperforms SGCL Fixed. A possible reason is that the dynamic schedule encourages the CL model to spend more steps on the more difficult subset. Encouragingly, we observe considerable gains over other curriculum learning counterparts.

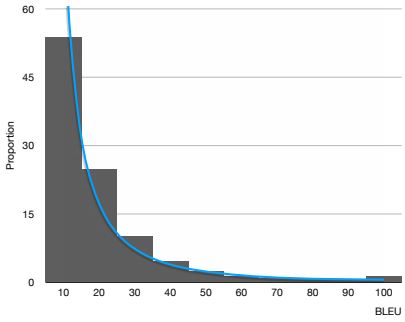
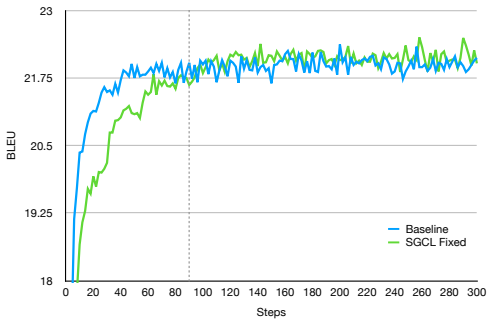


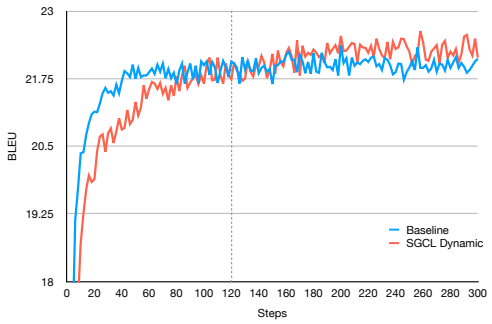
Figure 3: Recovery degree (sentence-level BLEU) distribution of the training set.

Subset	Range	Average
$\mathbb{D}_1$	17.72 - 100.00	35.62
$\mathbb{D}_2$	9.18 - 17.72	12.77
$\mathbb{D}_3$	5.16 - 9.18	6.97
$\mathbb{D}_4$	0.00 - 5.16	3.35

Table 3: Range and average of recovery degree (sentence-level BLEU) in subsets $\{\mathbb{D}_1, \mathbb{D}_2, \mathbb{D}_3, \mathbb{D}_4\}$



(a) Baseline vs. SGCL Fixed



(b) Baseline vs. SGCL Dynamic

Figure 4: Learning curves w.r.t BLEU scores.

6 Analysis

6.1 Recovery Degree

We conduct experiments on En-De for further analysis of the proposed CL methods.

As described in Section 3, we adopt sentence-level BLEU score to measure the recovery degrees of all examples in the training corpus with a vanilla NMT model. When making predictions with the vanilla model, we set the beam size to 1 for simplicity. So the recovery degrees could be lower than test results of a strong baseline. If we look at the distribution in terms of BLEU score on all training examples, as Figure 3 illustrated, the distribution is very dense in the region with lower scores. Specifically, more than 53.9% training examples get a recovery degree lower than 10. It reflects the distribution shift problem of well-trained vanilla NMT mode, that the model learned distribution and empirical distribution on training corpus are inconsistent.

In our case, the training corpus is split into 4 subsets with about equal size, $\{\mathbb{D}_1, \mathbb{D}_2, \mathbb{D}_3, \mathbb{D}_4\}$. Table 3 is the range and average of recovery degrees of each subset, revealing the learning difficulty of each subset merges into training set as training phase progress. We also look at the average lengths of source sentences in these 4 subsets, which are 22.40, 23.84, 25.33, 29.35, reflecting a gentle increase. As a comparison, if we sort the training examples by lengths of source sentences and split them into 4 subsets, the average lengths become 10.96, 18.66, 27.00, 44.30. So we can infer that the recovery degree is related to but not fully depend on sentence length, indicating that shorter sentences are not always easier to be masted by the NMT model.

6.2 Learning Curves

Figure 4 demonstrates the learning curves of baseline vs. SGCL Fixed and baseline vs. SGCL Dynamic. As illustrated, the baseline converges faster at the beginning but stays at a lower level as training progresses, while proposed CL methods show constant improvements and outperform the baseline in the later training process. A possible reason that the CL models don't outperform the baseline at the beginning might be, they boost their performance after all training examples are merged into the training set. After all training examples are included, CL models are able to maintain better growth momentum than the baseline.

We also observe that the SGCL Dynamic gains more significant improvements over the baseline than the SGCL Fixed. Given 300k training steps, different curriculum schedules suggest different

Source	然而, 就在大部分互联网医疗企业挣扎在A轮或B轮的融资路上的时候, 有几家细分领域领先企业仍能获得资本热捧。
Reference	However, just as the majority of internet medical companies struggle on the way of a round or b round of financing, several segment-leading enterprises can still be favored by investors.
Vanilla (8.61)	However, even as most internet healthcare companies struggle to raise money in a or b rounds, a few of the leading segments still enjoy the capital boom.
SGCL (27.45)	However, even as most internet health companies struggle with a round or b round of financing, several segments leading business still enjoy the capital boom.

Table 4: Predictions made by the Vanilla model and the SGCL Dynamic model with a same input sentence. We mark the errors with red underline. The number in parentheses, e.g. (8.61) are sentence-level BLEU scores.

ways of splitting the training steps. For the SGCL Fixed, we empirically define the training steps spent on phase 1 to phase 4 as 30k, 30k, 30k, 210k. That is to say, after 90k steps, the model is training with all examples in the training corpus. For SGCL Dynamic, as mentioned in Section 3, if the CL model outperforms the vanilla model on the newly merged subset, training progresses to the next phase. In practice, after new examples merge into the training set, we first train for 20k steps and then check the performance of the CL model every 10k steps. As a result, the model starts to train with all training examples after 120k steps and tends to spend more time steps in later training phases, consistent with other existing dynamic scheduling methods.

6.3 Case Study

Figure 4 presents a case study in Zh-En. It indicates that our approach achieves a performance boost because of better lexical choice. To better understand how our approach alleviates the low-recovery problem, we conduct statistic analysis on the sentence-level BLEU scores of predictions made by the vanilla model and the CL model on the test set. It shows that the proportion of predictions with a BLEU score under 10 is 10.0% with the vanilla model and is down to 8.1% with the CL one.

7 Conclusion

In this work, we propose a self-guided CL strategy for neural machine translation. The intuition behind it is that after skimming through all training examples, the NMT model naturally learns how to schedule a curriculum for itself. We discuss existing difficulty criteria for curriculum learning from a probabilistic perspective, which also explains our motivation for deriving a difficulty criterion based on recovery degree. Moreover, we corporate this recovery-based difficulty criterion with both fixed and dynamic curriculum schedules. Empirical results show that with a self-guided CL strategy, the NMT model achieves better performance over the strong baseline on translation benchmarks. In the future, we will corporate recovery-based difficulty criterion with other dynamic scheduling methods. Also, it will be interesting to apply our proposed CL strategy to different scenarios, e.g., non-autoregressive generation (Gu et al., 2018; Wu et al., 2020a; Ding et al., 2020b).

Acknowledgments

Lei Zhou is supported by a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO). The authors wish to thank the anonymous reviewers for their insightful comments and suggestions.

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ACM*.

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level bleu. In *WMT*.

Volkan Cirik, Eduard Hovy, and Louis-Philippe Morency. 2016. Visualizing and understanding curriculum learning for long short-term memory networks. *CoRR*.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *ACL*.

Liang Ding and Dacheng Tao. 2019. The University of Sydney's machine translation system for WMT19. In *WMT*.

Liang Ding, Longyue Wang, and Dacheng Tao. 2020a. Self-attention with cross-lingual position representation. In *ACL*.

Liang Ding, Longyue Wang, Di Wu, Dacheng Tao, and Zhaopeng Tu. 2020b. Context-aware cross-attention for non-autoregressive translation. In *COLING*.

Zi-Yi Dou, Antonios Anastasopoulos, and Graham Neubig. 2020. Dynamic data selection and weighting for iterative back-translation. In *EMNLP*.

Jeffrey L Elman. 1993. Learning and development in neural networks: The importance of starting small. *Cognition*.

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

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, et al. 2018. Achieving human parity on automatic chinese to english news translation. *arXiv*.

Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. 2015. Self-paced curriculum learning. In *AAAI*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In *RANLP*.

Xuebo Liu, Houtim Lai, Derek F. Wong, and Lidia S. Chao. 2020. Norm-based curriculum learning for neural machine translation. In *ACL*.

Xuebo Liu, Longyue Wang, Derek F. Wong, Liang Ding, Lidia S. Chao, and Zhaopeng Tu. 2021. Understanding and improving encoder layer fusion in sequence-to-sequence learning. *ICLR*.

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-HLT*.

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

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. In *NAACL-HTL*.

Dana Ruiter, Josef van Genabith, and Cristina España-Bonet. 2020. Self-induced curriculum learning in self-supervised neural machine translation. In *EMNLP*.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Neural IPS*.

Di Wu, Liang Ding, Fan Lu, and Jian Xie. 2020a. SlotRefine: A fast non-autoregressive model for joint intent detection and slot filling. In *EMNLP*.

Di Wu, Liang Ding, Shuo Yang, and Dacheng Tao. 2021. Slua: A super lightweight unsupervised word alignment model via cross-lingual contrastive learning. *ArXiv*.

Shuangzhi Wu, Xing Wang, Longyue Wang, Fangxu Liu, Jun Xie, Zhaopeng Tu, Shuming Shi, and Mu Li. 2020b. Tencent neural machine translation systems for the WMT20 news translation task. In *WMT*.

Chen Xu, Bojie Hu, Yufan Jiang, Kai Feng, Zeyang Wang, Shen Huang, Qi Ju, Tong Xiao, and Jingbo Zhu. 2020. Dynamic curriculum learning for low-resource neural machine translation. In *COLING*.

Runzhe Zhan, Xuebo Liu, Derek F Wong, and Lidia S Chao. 2021. Meta-curriculum learning for domain adaptation in neural machine translation. In *AAAI*.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat. 2018. An empirical exploration of curriculum learning for neural machine translation. *arXiv*.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. 2019. Curriculum learning for domain adaptation in neural machine translation. In *NAACL*.

Lei Zhou, Liang Ding, and Koichi Takeda. 2020a. Zero-shot translation quality estimation with explicit cross-lingual patterns. In *WMT*.

Yikai Zhou, Baosong Yang, Derek F. Wong, Yu Wan, and Lidia S. Chao. 2020b. Uncertainty-aware curriculum learning for neural machine translation. In *ACL*.