# Continual Learning for Neural Machine Translation

**Yue Cao,**[1,2,3] **Hao-Ran Wei,**[4] **Boxing Chen,**[4] **Xiaojun Wan**[1,2,3]
[1]Wangxuan Institute of Computer Technology, Peking University
[2]Center for Data Science, Peking University
[3]The MOE Key Laboratory of Computational Linguistics, Peking University
[4]Alibaba Group
{yuecao,wanxiaojun}@pku.edu.cn, {funan.whr,boxing.cbx}@alibaba-inc.com

## Abstract

Neural machine translation (NMT) models are data-driven and require large-scale training corpus. In practical applications, NMT models are usually trained on a general domain corpus and then fine-tuned by continuing training on the in-domain corpus. However, this bears the risk of catastrophic forgetting that the performance on the general domain is decreased drastically. In this work, we propose a new continual learning framework for NMT models. We consider a scenario where the training is comprised of multiple stages and propose a dynamic knowledge distillation technique to alleviate the problem of catastrophic forgetting systematically. We also find that the bias exists in the output linear projection when fine-tuning on the in-domain corpus, and propose a bias-correction module to eliminate the bias. We conduct experiments on three representative settings of NMT application. Experimental results show that the proposed method achieves superior performance compared to baseline models in all settings.[1]

## 1 Introduction

Continual learning, which is also referred to as incremental learning or lifelong learning, is a learning paradigm that allows the agent to continuously learn from new knowledge without forgetting previously learned knowledge. Humans naturally have the ability to continually acquire knowledge while preserving old knowledge throughout their lifespan.

In real-world applications, data is usually given in a continuous stream form, and only part of the data is available at the beginning of training. Therefore, the ability to learn from continuous streams of information is crucial for artificial intelligence systems. However, continual learning remains a big challenge for artificial intelligence systems and models since they suffer from the problem of **catastrophic forgetting** (French, 1993), i.e., the learning of new tasks may cause the model to forget the knowledge learned from previous tasks. This phenomenon typically leads to a significant performance decrease in previously learned tasks. One trivial solution to avoid catastrophic forgetting is to retrain from scratch by combining old and new tasks. However, this methodology is computationally inefficient and needs to store old data all the time.

Recently, continual learning has received increasing attention in the artificial intelligence filed. Most of existing works focus on computer vision tasks (Zenke et al., 2017; Aljundi et al., 2017; Triki et al., 2017; Hou et al., 2018; Aljundi et al., 2018; Hou et al., 2019; Wu et al., 2019).

In the natural language processing area, several methods have been proposed to alleviate the problem of catastrophic forgetting for **Neural Machine Translation (NMT)** models. For example, Freitag and Al-Onaizan (2016) propose to ensemble models trained on different domains. However, this brings a storage issue: as the number of domains increases, the number of stored models also increases. Saunders et al. (2019) and Thompson et al. (2019) add an L2 or EWC regularization to each parameter to prevent the model's parameters from changing too much. However, for those transformer models with more than 100 million parameters, the time and space cost for computing L2 or EWC regularization is expensive. Khayrallah et al. (2018) propose a regularized training objective that minimizes the cross-entropy between in domain model's output distribution and that of the out-of-domain model. This method can essentially be regarded as a kind of knowledge distillation.

The above works assume that the training is divided into two stages, i.e., out-of-domain training and in-domain fine-tuning. In this work, we extend

[1]This work was done when Yue Cao was an intern at Alibaba. Codes are available at *https://github.com/caoy1996/CL-NMT.*

these works and propose a new continual learning framework for NMT models. We consider a more general scenario where the training is comprised of multiple stages. We propose a dynamic knowledge distillation-based method to alleviate the problem of catastrophic forgetting in a systematic and principled way.

We also find that when fine-tuning on new data, there exists a strong bias towards the new words in the output embedding layer (i.e. the linear projection before the last softmax layer) of the decoder, which results in the bias in the generation that favors words from new data. To address this issue, we incorporate the model with a bias-correction module that normalizes the weights in the projection layer. The bias-correction module can effectively eliminate the bias of significant differences in magnitudes.

We consider three continual learning scenarios: (1) *in-domain multi-stage training*, where $m$ streams of data from the same domain are fed to the model sequentially, (2) *domain-incremental training*, where $m$ streams of data from different domains are fed to the model sequentially, and (3) *time-incremental training*, where $m$ streams of data from different time are fed to the model sequentially. Experimental results show that the proposed method can effectively address the catastrophic forgetting issue and balance the weights in the projection layer, thus achieving superior results compared to the competitive models.

In summary, the prime contributions of this paper are as follows:

- We propose a novel continual learning framework for neural machine translation. Compared with existing works, we consider a more general scenario where the training is comprised of multiple stages.

- We propose a novel method to alleviate the problem of catastrophic forgetting in a systematic way. We also find the existence of bias in the output embedding layer and propose a bias-correction module to address this issue.

- Experimental results in three different settings all show that the proposed method obtains superior performance compared to competitive models.[2]

[2]Codes and data will be released once this paper gets accepted.

## 2 Related Works

### 2.1 Neural Machine Translation

The task of machine translation is to automatically translate a written text from one natural language into another. Early machine translation systems are mostly built upon statistical learning techniques, which mainly rely on various count-based features (Brown et al., 1990; Och, 2003; Koehn et al., 2007). Recently, statistical machine translation (SMT) has largely been superseded by neural machine translation (NMT), which tackles machine translation with deep neural networks (Luong et al., 2015; Vaswani et al., 2017). Most NMT models either use LSTM (Luong et al., 2015) or Transformer (Vaswani et al., 2017) architectures.

NMT systems are sensitive to the data distributions (Stahlberg, 2019). To improve the performance of NMT models in low-resource domains, a widely-used technique is to train the model on a general domain corpus, and then fine-tune it on the in-domain corpus via continual training (Sennrich et al., 2016; Luong and Manning, 2015). However, this suffers from the problem of catastrophic forgetting (French, 1993) that the performance of the model on the general domain has decreased drastically. In this work, we aim to mitigate the catastrophic forgetting for NMT models.

As for the bias in NMT systems, Michel and Neubig (2018) 2018 adapt the bias of the output softmax to build a personalized NMT model. Different from their work, we propose to elinamate the bias in the output layer.

### 2.2 Continual Learning

Most of continual learning models are proposed for computer vision tasks. These models mainly fall into parameter-based methods (Aljundi et al., 2018; Kirkpatrick et al., 2016; Zenke et al., 2017) and distillation-based methods (Aljundi et al., 2017; Triki et al., 2017; Hou et al., 2018, 2019; Wu et al., 2019). The parameter-based methods estimate the importance of each parameter and penalize the model once it updates the important parameters. The distillation-based methods transfer important knowledge from an old model to a new model through a teacher-student framework. Usually, a modified cross-entropy loss is adopted to preserve the knowledge of the old model.

In the field of natural language processing, there are some researches on solving catastrophic forgetting problem in lifelong learning (Freitag and

Al-Onaizan, 2016; Khayrallah et al., 2018; Saunders et al., 2019; Thompson et al., 2019). However, these works only consider the scenario of one-stage incremental training. To the best of our knowledge, there is no previous work that takes into account the scenario in which the training consists of multiple stages.

Domain adaptation learning (or transfer learning) is a task similar to continual learning. The difference is that domain adaptation learning only cares about the performance of in-domain data, while continual learning cares about not only the performance on in-domain data, but also the performance on out-of-domain data.

## 3 Methods

### 3.1 Overall

Given a bilingual translation pair $(x, y)$, the NMT model $g$ learns the parameter $\xi$ and $\theta$ to maximize the conditional log-likelihood $\log P(y|x, \xi, \theta)$. Generally, the probability of generating $i$-th word is computed as

$$p(y_i|y_{1:i-1}, x) = \frac{\exp\{\theta^\top \phi(x_i, y_{1:i-1}, \xi)\}}{\sum_j \exp\{\theta^\top \phi(x_j, y_{1:i-1}, \xi)\}} \quad (1)$$

where $x_i$, $y_i$ is the $i$-th word in $x$ and $y$, $\phi(\cdot, \xi)$ is a nonlinear function that maps an input $x$ into a dense representation. The linear projection parameterized by $\theta$ maps the dense representation to the word distributions, followed by a softmax activation to output the probability of generating each word. For NMT models, the nonlinear function $\phi(\cdot, \xi)$ is usually chosen as the encoder-decoder framework. In the following text, for the convenience of narration, we use $w$ to refer to $\xi$ and $\theta$, i.e., $w = \xi \cup \theta$.

Under the continual training setting, the encoder and decoder $\phi(\cdot, \xi)$ is trained on data of different domains successively. When fine-tuning on new data, the learned parameters $\xi$ may overfit new data and degrade the performance on old data, which is known as the problem of **catastrophic forgetting**. On the other hand, when fine-tuning on a new domain corpus, we need to add new words from the new domain to the vocabulary, so we need to expand the projection matrix in the linear projection. At this stage, the model always samples new words to generate, and the ground truths for those old words are always 0. After several epochs, the model may mistakenly believe that the old words are no longer used and thus reduce the probability of old words to be 0 for all samples. This causes the **biased weights** issue.

Our goal is twofold: (1) for the parameters $\xi$ in the encoder and decoder $\phi(\cdot, \xi)$, we aim to alleviate the catastrophic forgetting problem, and (2) for the linear projection $\theta$, we aim to eliminate the bias generated during continuous training. For the former, we propose a dynamic knowledge distillation-based technique to alleviate the catastrophic forgetting problem during multi-stage continual training (Section 3.2). For the latter, we incorporate the model with a bias-correction module that eliminates the bias of projection weights (Section 3.3).

### 3.2 Alleviate the Catastrophic Forgetting Issue

As discussed above, we propose to alleviate the catastrophic forgetting in the encoder and decoder under the continual training setting.

#### 3.2.1 Definition

We consider the scenario where the training is comprised of $m$ stages, denoted by $k = 1, \cdots, m$. At $k$-th stage, a subset of data $\{x_k^{(i)}, y_k^{(i)}\}_{i=1}^{T_k}$ are fed to the model, where $T_k$ refers to the number of samples at $k$-th stage, $x_k^{(i)}$ refers to $i$-th sample at $k$-th stage.

Assuming that $u_k(\cdot)$ is a gold function sampled from an unknown distribution $\mathbb{P}_y$ that maps each $x_k^{(i)}$ to $y_k^{(i)}$ at stage $k$, i.e., $y_k^{(i)} = u_k(x_k^{(i)})$. Under the continual learning setting, our goal is to learn a deep neural model $g(\cdot; w)$ parameterized by $w$, such that $g(\cdot; w)$ not only fits well to $u_k(\cdot)$, but also $u_{k-1}(\cdot), u_{k-2}(\cdot), \cdots, u_1(\cdot)$ received in early stages to alleviate the catastrophic forgetting.

#### 3.2.2 Formulation

Considering that in some cases, recent data is more important than early data, we set a *discount* (Sutton and Barto, 1998) $\alpha^s$ to $u_{k-s}(\cdot)$, and minimize the cross-entropy loss between model output $g(\cdot; w)$ and weighted sum of $u_k(\cdot)$:

$$\min \mathcal{L}_k(w_k) \triangleq -\sum_{i=1}^{T_k} z_k(x_k^{(i)}) \times \log g(x_k^{(i)}; w_k) \quad (2)$$

where $z_k(x)$ is the normalized sum of $u_k(\cdot)$:

$$z_k(x) = \frac{1-\alpha}{1-\alpha^k} \sum_{s=0}^{k-1} \alpha^s u_{k-s}(x) \quad (3)$$

Notice that with $\alpha$ close to 1, minimizing $\mathcal{L}_k(w_k)$ is closely related to minimizing

$\sum_{i=1}^{T_k} \left(\mathrm{E}_{u\sim\mathbb{P}_y} u(x)\right) \times \log g(x^{(i)}; w_k)$. In our experiments, we set $\alpha = 0.999$ for the case which the data from different stages have no priority.

For an input $x$ in stage $k$, the computation of $z_k(x)$ needs us to get the value of $\{u_s(x)\}_{s=1}^{k}$ first. A simple but inefficient way is to store the outputs or a learned approximation of $u_s(x)$ of every stages, which means that we need to store $m$ models if the training is comprised of $m$ stages. To reduce the space overhead, we rewrite Eq. 3 as

$$\begin{aligned} z_k(x) &= \frac{1-\alpha}{1-\alpha^k}[u_k(x) + \alpha u_{k-1}(x) + \cdots + \alpha^{k-1} u_1(x)] \\ &= \frac{1-\alpha}{1-\alpha^k}\left[u_k(x) + \alpha\left(u_{k-1}(x) + \cdots + \alpha^{k-2} u_1(x)\right)\right] \\ &= \frac{1-\alpha}{1-\alpha^k}\left[u_k(x) + \alpha \frac{1-\alpha^{k-1}}{1-\alpha} z_{k-1}(x)\right] \\ &= \frac{1-\alpha}{1-\alpha^k} u_k(x) + \alpha \frac{1-\alpha^{k-1}}{1-\alpha^k} z_{k-1}(x) \end{aligned} \tag{4}$$

Let $\lambda_k = \alpha\frac{1-\alpha^{k-1}}{1-\alpha^k}$, notice that $\frac{1-\alpha}{1-\alpha^k} + \alpha\frac{1-\alpha^{k-1}}{1-\alpha^k} = 1$, we have:

$$z_k(x) = (1-\lambda_k)u_k(x) + \lambda_k z_{k-1}(x) \tag{5}$$

Eq. 5 reveals that $z_k(x)$ can be derived from $z_{k-1}(x)$ and $u_k(x)$, so we can instead seek to calculate $z_{k-1}(x)$ to avoid storing too many sub-models.

Since in the last stage, we make the distribution of $g(x; w_{k-1})$ be as similar to $z_{k-1}(x)$ as possible by minimizing their cross-entropy. Therefore, in $k$-th stage, we use $g(x; w_{k-1})$ to approximate $z_{k-1}(x)$.

The training objective of our model at $k$-th stage can be written as:

$$\begin{aligned} \min\ \hat{\mathcal{L}}_k(w_k) \triangleq - \sum_{i=1}^{T_k} &\left[(1-\lambda_k)u_k(x_k^{(i)}) + \lambda_k g(x_k^{(i)}; w_{k-1})\right] \\ &\times \log g(x_k^{(i)}; w_k) \end{aligned} \tag{6}$$

### 3.2.3 Relevance to Knowledge Distillation

The proposed method can also be regarded as a special kind of knowledge distillation. To explain this, we rewrite Eq. 6 as

$$\begin{aligned} \hat{\mathcal{L}}_k(w_k) &= -\sum_{x_k} z_k(x_k) \times \log g(x_k; w_k) \\ &= -\sum_{x_k} [(1-\lambda_k)u_k(x_k) + \lambda_k z_{k-1}(x)] \times \log g(x_k; w_k) \\ &= -(1-\lambda_k)\sum_{x_k} u_k(x_k) \times \log g(x_k; w_k) \\ &\quad - \lambda_k \sum_{x_k} z_{k-1}(x_k) \times \log g(x_k; w_k) \end{aligned} \tag{7}$$

The first term in Eq. 7 minimizes a cross-entropy loss between gold label $y_k = u_k(x_k)$ and model output $g(x_k; w_k)$, which is a standard translation loss. The second term in Eq. 7 minimizes the cross-entropy between the model's output of last stage $g(x; w_{k-1})$ and current stage $g(x; w_k)$. If we consider the trained model of last stage as the “teacher", and the model of current stage as the “student", then this is a standard knowledge distillation loss.

Therefore, the proposed method can also be seen as optimizing a weighted sum of translation and distillation loss, which is similar to Khayrallah et al. (2018). The difference is that Khayrallah et al. (2018) only consider the case where the training is comprised of two stages, and thus they use a fixed $\lambda = 0.1$ in Eq. 7, i.e.,

$$\begin{aligned} \hat{\mathcal{L}}'(w_k) = &-(1-\lambda)\sum_{x_k} u_k(x_k) \times \log g(x_k; w_k) \\ &- \lambda \sum_{x_k} z_{k-1}(x_k) \times \log g(x_k; w_k) \end{aligned} \tag{8}$$

When applying Eq. 8 to multi-stage incremental training, it is easy to deduce that they actually fit a $z'_k(x) = \sum_{s=0}^{k-1} \lambda^s(1-\lambda)u_{k-s}(x)$ at the $k$-th stage, which means that the weights of old knowledge are always lower. When $\lambda < 1$, the model will always pay more attention to new data and decay the weights of old knowledge at an exponential rate. Under this case, the model will quickly forget the general knowledge learned from earliest stage and overfit the new data. On the other hand, if choose $\lambda$ close to 1, the model hardly learns new knowledge as the weight of translation loss close to 0. During experiments, we find that $\lambda = 0.7$ works well for this method, so we set $\lambda = 0.7$ in the following experiments.

Our method adjusts the weight $\lambda_k$ dynamically and gradually increases the weight of distilled loss ($\lambda_k = \alpha\frac{1-\alpha^{k-1}}{1-\alpha^k}$). Therefore, our model can balance the learning of new knowledge and memorization of old knowledge. We name the proposed method as **“dynamic knowledge distillation"**.

## 3.3 Eliminate the Bias in Linear Projection

### 3.3.1 Biased Weights In the Linear Projection

To reveal the bias weights phenomenon in the linear projection in continual training, we conduct a test that first trains an English-German NMT model on an IT-related corpus, and then fine-tunes it on law-related corpus.[3] We find that after fine-tuning on law-related data, the model will no longer generate IT-specific words even we feed an IT-related

[3]The number of training samples for IT and law corpus are 232K and 205K respectively.

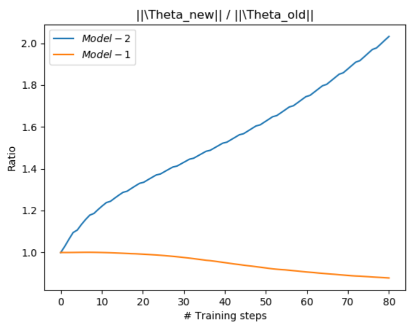

Figure 1: The changes of $\eta$ with the training of Model-1 and Model-2. This figure shows that directly fine-tuning the model on new data will cause the biased weights problem.

source sentence to the model. As a consequence, the model performs extremely poorly on the IT test set.

We hypothesize that the model reduces the old words' probability by shrinking their corresponding weights in the last linear projection $\Theta$. To verify this, we train two models simultaneously: one is trained on combined IT-related and law-related corpus (referred to as Model-1), and the other is trained on IT-related corpus first, and then fine-tuned on the law-related corpus (referred as Model-2).

Denote $\eta$ as the ratio of new words weights and old words weights in the last linear projection:

$$\eta = \left(\frac{1}{n_{new}} \sum_{\theta \in \Theta_{new}} ||\theta||\right) / \left(\frac{1}{n_{old}} \sum_{\theta \in \Theta_{old}} ||\theta||\right) \tag{9}$$

We calculate the changes of $\eta$ with the training of Model-1 and Model-2 respectively and plot the results in Fig. 1. Since Model-1 can achieve good performance on both IT and law test sets, we consider its weights' ratio as the "ground truth".

Fig. 1 shows that compared to Model-1, Model-2's norm of the weights for new words is much higher than those for old words as the training goes by. In Eq. 1, if $i$-th word should be picked out, then $\theta_i^\top \phi(x, w)$ should be a positive number.[4] In this case, decreasing $||\theta_i||$ will reduce the probability of generating $i$-th word. This results in the bias in the generation that favors new words.

[4]In transformer model, $\theta_i^\top \phi(x, w)$ will $\approx 0$ for most words, but for those words that are likely generated, $\theta_i^\top \phi(x, w)$ will $> 0$.

### 3.3.2 Weight Normalization for Bias Correction

Based on the above observation, we propose to add a *weight normalization* module similar to Nguyen and Chiang (2018) in the linear projection.

Concretly, we normalize the weights for all words by:

$$\hat{\theta}_i = \theta_i / ||\theta_i|| \tag{10}$$

and compute the probability of generating each word as:

$$\hat{p}_i(x) = \frac{\exp\{\gamma \cdot \hat{\theta}_i^\top \phi(x, w)\}}{\sum_j \exp\{\gamma \cdot \hat{\theta}_j^\top \phi(x, w)\}} \tag{11}$$

where $\gamma$ is a (learnable) scaling scalar. The introduction of $\gamma$ is to control the peakiness of the softmax distribution.

Notice that since the encoder and decoder are shared and always used for data from different domains, they do not suffer the biased weights problem.

## 4 Experiments

### 4.1 Experiment Settings

We consider the following three representative training scenarios in NMT systems:

- **In-domain incremental training:** We split the training data in the same domain into $m$ sets, and fed one set of data to the model at each stage. We share the same validation and test sets among different stages in this setting.

  Notice that since the data in different stages are from the same domain, we do not incorporate the bias-correction module under this setting.

- **Domain-incremental training:** We first train the model on a large-scale general domain corpus[5], and then fine-tune it on $m$ new domains successively. We calculate the model's performance on the test sets of general and the new domains at each stage.

- **Time-incremental training:** Time-incremental training is a special case of in-domain incremental training, where the training data come from different time and are fed to the model in chronological order. We set this scenario to simulate the training of NMT model on real-world time streaming data.

[5]WMT14 News Commentary.

| IWSLT2013 | 20% | 40% | 60% | 80% | 100% |
|---|---|---|---|---|---|
| Combined | 24.98 | 32.18 | 35.19 | 37.00 | 37.72 |
| Fine-tuning | 24.98 | 29.25* | 32.57* | 34.09* | 34.51* |
| + knowledge distill. | 24.98 | 30.69 (+1.44) | 33.46 (+0.89) | 34.61 (+0.52)* | 34.85 (+0.34)* |
| + EWC reg. | 24.98 | 29.65 (+0.40)* | 32.75 (+0.18)* | 34.13 (+0.04)* | 34.43 (-0.08)* |
| Ours | 24.98 | **30.94 (+1.69)** | **33.49 (+0.92)** | **34.96 (+0.87)** | **35.20 (+0.69)** |
| **WMT14** | **20%** | **40%** | **60%** | **80%** | **100%** |
| Combined | 16.75 | 19.91 | 23.82 | 25.36 | 27.14 |
| Fine-tuning | 16.75 | 18.67* | 22.07* | 23.36* | 25.25* |
| + knowledge distill. | 16.75 | 19.19 (+0.52)* | 22.82 (+0.75) | 23.88 (+0.52)* | 25.90 (+0.65)* |
| + EWC reg. | 16.75 | 18.70 (+0.03)* | 22.41 (+0.34)* | 23.84 (+0.48)* | 25.54 (+0.29)* |
| Ours | 16.75 | **19.44 (+0.77)** | **23.02 (+0.95)** | **24.17 (+0.81)** | **26.22 (+0.97)** |

Table 1: Experiment results of different models under in-domain incremental training setting on IWSLT2013 and WMT14 datasets. Best results are highlighted in bold. Statistically significant improvements ($p < 0.1$) over our method are marked with *.

| Method | + It | + Koran | + Law | + Medical | + Subtitles |
|---|---|---|---|---|---|
| Fine-tuning | 44.38 | 23.41 | **57.71** | **54.65** | **30.02** |
| + knowledge distill. | 44.36 (-0.02) | 23.50 (+0.09) | 57.54 (-0.17) | 54.49 (-0.16) | 29.91 (-0.11) |
| + EWC reg. | 44.12 (-0.26) | 22.94 (-0.47) | 57.10 (-0.61) | 54.03 (-0.62) | 29.51 (-0.51) |
| Ours | **44.41 (+0.03)** | **23.49 (+0.08)** | 57.52 (-0.19) | 54.58 (-0.07) | 29.87 (-0.15) |
| w/o Dynamic KD. | 44.09 (-0.29) | 23.09 (-0.32) | 57.24 (-0.47) | 54.03 (-0.62) | 29.43 (-0.59) |
| w/o BiC. | 44.34 (-0.04) | 23.36 (-0.05) | 57.46 (-0.25) | 54.43 (-0.22) | 29.73 (-0.29) |

Table 2: Experiment results of different models under domain-incremental training setting. The best results are highlighted in bold.

In our experiments, we set $m = 5$. Following previous works on lifelong learning (Aljundi et al., 2017; Triki et al., 2017; Aljundi et al., 2018; Hou et al., 2019; Wu et al., 2019), we use a memory with fixed capacity to reserve the training examples sampled from old data. The data stored in the memory and the new data are together fed to the model at each stage. The memory size is set to 50, 000 in our experiments.

### 4.1.1 Data Preparation

We use the IWSLT2013 de-en translation data[6] and WMT14 de-en translation data[7] for in-domain incremental training. The number of training samples of IWLST2013 dataset is 206,122 in total, and we use 41,224 samples to train the model at each stage. The validation and test sets are shared among all stages, and the numbers of validation and test samples are 3,000. The number of training samples of WMT14 dataset is 4,500,000 in total.

We use the new data split of OPUS multi-domain dataset released by Aharoni and Goldberg[8] for domain-incremental training. This dataset contains de-en data from IT, koran, law, medical, and subtitles fields. The numbers of training samples for these domains are 222,927, 17,982, 467,309, 248,099 and 500,000, respectively. The numbers of validation and test samples are 2,000 for each domain.

We use WMT news-commentary 2015-2019 de-en translation data[9] for time-incremental training. The WMT news-commentary data was first built in 2015 and some new data was added in each subsequent year. News-commentary 2015 contains 216,897 training samples, and 26,576, 27,999, 12,774 and 54,038 new samples are added in 2016-2019, respectively. The test sets contain 3,000 samples for each year. Notice that each year's test set may contain test samples from previous years. For example, the 2017 test set contains both new test samples from 2017 and some old test samples from 2015 and 2016.

## 4.2 Competitive Methods

We use the following competitive models for comparison in experiments:

- **Fine-tuning** This model is directly fine-tuned on new data.

[6]http://workshop2013.iwslt.org/59.php
[7]http://www.statmt.org/wmt14/
[8]https://github.com/roeeaharoni/unsupervised-domain-clusters
[9]http://www.statmt.org/

- **Combined** This model is trained on combined new data and old data from scratch, which is considered the **upper bound** in the field of continual learning.

- **Knowledge Distillation (KD) (Khayrallah et al., 2018)** When fine-tuning on current set of data, this model optimizes a weighted sum of NLL loss and regularization term: $\mathcal{L}(w) = (1-\alpha)\mathcal{L}_{nll}(w) + \alpha\mathcal{L}_{reg}(\theta)$. The regularization term is formulated in the spirit of knowledge distillation that minimizes the cross-entropy between in-domain (teacher) model's output distribution and that of the out-of-domain (student) model. The value of $\alpha$ is fixed at every stage.

- **Elastic Weight Consolidation (EWC) (Saunders et al., 2019; Thompson et al., 2019)** This model optimizes a weighted sum of NLL loss and EWC term. We recommend readers refer to their papers for more details.

For the convenience of narration, we refer to the knowledge distillation, elastic weight consolidation, and our proposed method as **"learning-without-forgetting (LWF)"**-based methods. To study the effectiveness of different components of our proposed method, we also test the following variants of our model:

- **w/o dynamic knowledge distillation** It removes the dynamic knowledge distillation module from the proposed model.

- **w/o bias correction** It removes the bias correction module from the proposed model.

### 4.3 Implementation Details

We use the Fairseq toolkit (Ott et al., 2019) to implement the proposed model. We process the text into subword units by using the subword-nmt toolkit[10].

We adopt the transformer (Vaswani et al., 2017) as the model architecture. We set the model's hidden size, feed-forward hidden size to 512, 2048, and set the number of layers and the number of heads to 6 and 8, respectively. We use the same configuration for all encoders and decoders.

For training and inference, we use Adam optimizer (Kingma and Ba, 2014) and use the same parameters and learning rate schedule as previous work (Vaswani et al., 2017). We use warm-up learning rate (Goyal et al., 2017) for the first 3,000 steps, and the initial warm-up learning rate is set to $1e$-7. We use the dropout technique and set the dropout rate to $0.4$. We use beam search for inference, and the beam size is set to $5$.

The max update steps of each model are different, depending on when they converge.

## 5 Results and Analysis

### 5.1 In-Domain Incremental Training

The experimental results of in-domain incremental training are shown in Table 1. Notice that the combined model is trained on all data observed so far, and it serves as the upper bound in this setting and will not participate in the comparison.

It first can be seen that there is a gap between the fine-tuning model and combined model, which suggests that there is some amount of general knowledge that has been forgotten by the model during fine-tuning. The performance improved when incorporating knowledge distillation, EWC regularization, or the proposed dynamic knowledge distillation techniques into the fine-tuning process, which shows that learning-without-forgetting strategies can help the model remember the general knowledge and benefit the fine-tuning. The improvement is less significant for the EWC-based model.

By comparing results of our model with the knowledge distillation-based and EWC regularization-based methods, we can see that our model outperforms them in all cases. The proposed model achieves an average improvement of 0.3 and 0.8 BLEU scores compared to the knowledge distillation-based and EWC regularization-based methods, respectively.

The above results confirm the finding of prior works that the learning-without-forgetting strategies can benefit the continual training, and demonstrate that the proposed method adds more gains.

We also study the effect of $\alpha$ in Eq. 3. A small value of $\alpha$ indicates that the model will pay more attention to new data, and penalize less for forgetting old knowledge. The detailed experiment results are shown in Table 3. We can observe that when $\alpha$ is larger than 0.5, the proposed method can achieve good performance, and the model achieves the best BLEU scores when $\alpha = 0.5$ or $\alpha = 0.7$.

[10]https://github.com/rsennrich/subword-nmt

| **Method** | **20%** | **40%** | **60%** | **80%** | **100%** |
|---|---|---|---|---|---|
| Upper bound | 24.98 | 32.18 | 35.19 | 37.00 | 37.72 |
| Baseline | 24.98 | 29.25 | 32.57 | 34.09 | 34.51 |
| Ours ($\alpha = 0.999$) | 24.98 | 30.59 | 33.22 | 34.45 | 35.04 |
| Ours ($\alpha = 0.9$) | 24.98 | 30.71 | 33.19 | 34.51 | 35.12 |
| Ours ($\alpha = 0.7$) | 24.98 | **31.04** | **33.52** | 34.91 | 35.12 |
| Ours ($\alpha = 0.5$) | **24.98** | **30.96** | 33.49 | **34.96** | **35.20** |
| Ours ($\alpha = 0.3$) | 24.98 | 30.23 | 33.12 | 34.82 | 34.88 |
| Ours ($\alpha = 0.1$) | 24.98 | 30.18 | 32.99 | 34.29 | 34.34 |

Table 3: The effect of $\alpha$ in the dynamic knowledge distillation module. The proposed method can achieve good results when $\alpha > 0.5$.

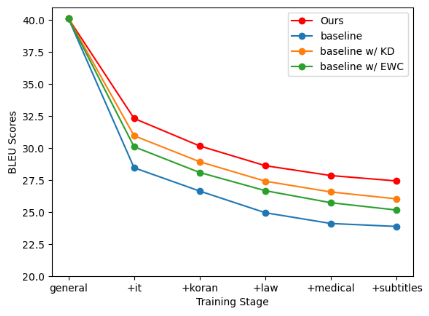

Figure 2: The performance of our model and competitive models on general test set after each stage. Baseline is the model that directly fine-tuned on new data. The proposed method significantly outperform competitive methods in all cases.

## 5.2 Domain-Incremental Training

In this setting, we first train a general NMT model on the large-scale WMT16 de-en dataset, and then fine-tune the model on IT, koran, law, medical, and subtitles domain sequentially. Considering that these domains have no priority to each other, so we set $\alpha = 0.999$ (approximate 1) in Eq. 3.

To explore the degree to which the model forgets old knowledge during incremental training, after each incremental training phase, we report the results of the models on the general domain (WMT16 de-en) test set. We present the experimental results of this part in Fig. 2, and we also present the results of the ablation study in Fig. 3. Due to the forgetting of old knowledge, the result is a descending curve of the BLEU score after each phase.

We can see from Fig. 2 that our model outperforms all competitive models at any stage. Incorporating the proposed method to the fine-tuning can bring an improvement of 3-4 BLEU scores in the general domain, indicating that our proposed method can effectively alleviate the catastrophic forgetting issue, and maintain the performance of the model on old data.

It seems that the largest drop in performance happens at the first training step. This is because the "private knowledge" of the general domain will be covered by the new knowledge mostly at the first training step, while the few remaining knowledge will be gradually covered in the later steps. The results also show that when fine-tuning on the new domain that contains more training samples, the occurrence of catastrophic forgetting would be more obvious, and our method can gain more improvements.

The knowledge distillation-based method can also improve the results on the general domain, but the improvement is lower than ours. This is because the underlying thought of Eq. 8 is to attenuate old knowledge at an exponential rate (when $k = 5$, the coefficient of $u_1(x)$ is 0.072). Thus after several stages, the model will focus more on new data and neglect old data.

We also analyze the representations of sentences in different stages and investigate how they evolve over time. For this purpose, we compute the average sentence representation $s$ in general domain, and compute the ratio of changes $\|s_{t+1} - s_t\|/\|s_t\|$ at each stage. We find that our method lead to fewer changes compared to baseline model (0.16 vs. 0.21), indicating that our method is better at preserving previously learned knowledge.

We also study whether the introduction of these "learning-without-forgetting" strategies will harm the domain transfer, i.e., decreasing the results of the model on the current/new domain. Therefore, we also report the results of the model on the current domain. These results are shown in Table 2. Due to the imbalanced training data in different domains, the combined model performs poorly in some domains, especially those with small training samples, so we do not report the results of the combined model under this setting.

The results in Table 2 show that our model performs slightly better or at least comparable to the model that is directly fine-tuned on new data. We hypothesize that this is because the proposed method reserves general knowledge learned from the general domain corpus, such as the basic grammar and word semantics, to the continual training model when fine-tuned on new data. Therefore encouraging the model to remember this knowledge can better help the model leverage general knowledge to improve performance on new do-

| Method | 2015 | + 2016 | + 2017 | + 2018 | + 2019 |
|---|---|---|---|---|---|
| Combined (Upper Bound) | 29.03 | 32.41 | 37.69 | 46.22 | 35.38 |
| Fine-tuning | 29.03 | 31.97 | 37.07 | 45.34 | 34.51 |
| + knowledge distill. | 29.03 | **32.29 (+0.32)** | 37.43 (+0.36) | 45.83 (+0.49) | 34.81 (+0.30) |
| + EWC reg. | 29.03 | 32.07 (+0.10) | 37.38 (+0.31) | 45.69 (+0.35) | 34.67 (+0.16) |
| Ours | 29.03 | 32.27 (+0.30) | **37.55 (+0.48)** | **46.08 (+0.74)** | **35.06 (+0.55)** |
| w/o Dynamic KD. | 29.03 | 32.04 (+0.07) | 37.19 (+0.12) | 45.51 (+0.17) | 34.63 (+0.12) |
| w/o BiC. | 29.03 | 32.19 (+0.22) | 37.45 (+0.38) | 46.06 (+0.72) | 34.91 (+0.40) |

Table 4: Experiment results of different models in time-incremental training setting. Best results are highlighted in bold. The combined model serves as the upper bound in this setting and will not participate in the comparison.

mains. This observation is consistent with some previous work (Khayrallah et al., 2018).

The results of the ablation study in Fig. 3 show that both the dynamic knowledge distillation and bias correction module contribute to the improvement of the results. Although the bias correction module is simple, it plays a very important role in the proposed model. After removing the bias correction module, the result of the model drops by 0.9-2.1 BLEU scores.

## 5.3 Time-Incremental Training

Table 4 shows the results of different models in time-incremental training setting. Since the test set of each year is a combination of old and new test samples, we directly report the results of different models on current year's test set. The combined model serves as the upper bound and will not participate in the comparison.

As expected, the proposed model outperforms competitive models in most cases. There is an improvement of 0.3-0.8 BLEU scores over the fine-tuned model, 0-0.3 BLEU scores over the knowledge distillation-based model, and 0.2-0.5 BLEU scores over the EWC regularization-based model. These results show that the proposed method for continual training is effective.

The results of ablation study show that the bias correction module is less beneficial for the model under this setting as the removal of bias correction module only results in a decrease of 0.1-0.2 BLEU score to the performance. We hypothesize that this is because the domain variation among test sets from 2015 to 2019 is smaller than that in domain-incremental experiments. Therefore, the biased weights phenomenon is slighter in this case.

## 6 Conclusion

In this paper, we propose a new continual learning framework for neural machine translation. We first propose a dynamic knowledge distillation-based method to alleviate the problem of catastrophic forgetting in a multi-stage view, and then propose a bias-correction module to address the biased weights issue. To verify the effectiveness of the proposed method, we conduct experiments in three different settings: in-domain incremental training, time-incremental training, and domain-incremental training. Experimental results show that the proposed method can obtain superior performance compared to competitive models.

In the future, we will apply the proposed method to other NLP tasks to test its robustness.

## Acknowledgments

This work was partially supported by National Natural Science Foundation of China (61772036), Beijing Academy of Artificial Intelligence (BAAI) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We thank the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

## References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7747–7763. Association for Computational Linguistics.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 144–161. Springer.

Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. 2017. Expert gate: Lifelong learning with a network of experts. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 7120–7129. IEEE Computer Society.

Peter F. Brown, John Cocke, Stephen Della Pietra, Vincent J. Della Pietra, Frederick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Comput. Linguistics*, 16(2):79–85.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *CoRR*, abs/1612.06897.

Robert M. French. 1993. Catastrophic interference in connectionist networks: Can it be predicted, can it be prevented? In *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, pages 1176–1177. Morgan Kaufmann.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. 2017. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. 2018. Lifelong learning via progressive distillation and retrospection. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 452–467. Springer.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. 2019. Learning a unified classifier incrementally via rebalancing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 831–839. Computer Vision Foundation / IEEE.

Huda Khayrallah, Brian Thompson, Kevin Duh, and Philipp Koehn. 2018. Regularized training objective for continued training for domain adaptation in neural machine translation. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, NMT@ACL 2018, Melbourne, Australia, July 20, 2018*, pages 36–44. Association for Computational Linguistics.

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

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. The Association for Computational Linguistics.

Paul Michel and Graham Neubig. 2018. Extreme adaptation for personalized neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 312–318. Association for Computational Linguistics.

Toan Q. Nguyen and David Chiang. 2018. Improving lexical choice in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 334–343. Association for Computational Linguistics.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, 7-12 July 2003, Sapporo Convention Center, Sapporo, Japan*, pages 160–167. ACL.

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 *Proceedings of the NAACL-HLT*, pages 48–53.

Danielle Saunders, Felix Stahlberg, Adrià de Gispert, and Bill Byrne. 2019. Domain adaptive inference for neural machine translation. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 222–228. Association for Computational Linguistics.

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

*54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Felix Stahlberg. 2019. Neural machine translation: A review. *CoRR*, abs/1912.02047.

Richard S. Sutton and Andrew G. Barto. 1998. Reinforcement learning: An introduction. *IEEE Trans. Neural Networks*, 9(5):1054–1054.

Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. 2019. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2062–2068. Association for Computational Linguistics.

Amal Rannen Triki, Rahaf Aljundi, Matthew B. Blaschko, and Tinne Tuytelaars. 2017. Encoder based lifelong learning. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1329–1337. IEEE Computer Society.

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 *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. 2019. Large scale incremental learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 374–382. Computer Vision Foundation / IEEE.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995. PMLR.