Overcoming Catastrophic Forgetting During Domain Adaptation of Seq2seq Language Generation

Dingcheng Li, Zheng Chen, Eunah Cho, Jie Hao,
Xiaohu Liu, Xing Fan, Chenlei Guo, Yang Liu
Amazon Alexa AI
Seattle, Washington, USA
zgchen,eunahch,jieha,derekliu
fanxing,guochenl,yangliud@amazon.com

Abstract

Seq2seq language generation models that are trained offline with multiple domains in a sequential fashion often suffer from catastrophic forgetting. Lifelong learning has been proposed to handle this problem. However, existing work such as experience replay or elastic weighted consolidation requires incremental memory space. In this work, we propose an innovative framework, RMR_DSE that leverages a recall optimization mechanism to selectively memorize important parameters of previous tasks via regularization, and uses a domain drift estimation algorithm to compensate for the drift between different domains in the embedding space. These designs enable the model to be trained on the current task while keeping the memory of previous tasks, and avoid much additional data storage. Furthermore, RMR_DSE can be combined with existing lifelong learning approaches. Our experiments on two seq2seq language generation tasks, paraphrase and dialog response generation, show that RMR_DSE outperforms state-of-the-art models by a considerable margin and greatly reduces forgetting.

1 Introduction

Seq2seq language generation is the essential framework for many tasks such as machine translation, summarization, paraphrase, question answering, dialog response generation. In these applications, models are typically trained offline using annotated data from a fixed set of domains. However, in real-world applications, it is desirable for the system to expand its knowledge to new domains and functionalities, that is, it has the capability of human-like lifelong learning (LLL) (Ring et al., 1994; Chaudhry et al., 2019) of acquiring new utterance patterns without forgetting what it has already learned. Neural networks struggle to learn continuously and experience catastrophic forgetting (CF) when optimized on a sequence of learning problems (McCloskey and Cohen, 1989; French, 1999). Some past work in LLL has demonstrated that discriminative models can be incrementally learnt for a sequence of tasks (Kirkpatrick et al., 2017; Chen et al., 2020; Wang et al., 2019). In contrast, under generative settings there has been limited research. Recent work in this area includes (Mi et al., 2020; Madotto et al., 2020; He et al., 2021; Shin et al., 2017).

Existing work in LLL adopts the *replay based methods* (Pellegrini et al., 2019), such as Latent Replay, or *regularization based methods* (Huszár, 2018; Li and Hoiem, 2018), such as Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017), Synaptic Intelligence (Zenke et al., 2017). Although they can reduce CF, they have some limitations. The replay-based methods require storing samples from previous tasks, and regularization methods often view all the model parameters as equally important and regularize them to the same extent. In addition, those approaches do not explicitly address the data distribution shift that causes the CF problem. The semantic gap between the embedding spaces of two domains is a leading reason of CF (Wang et al., 2021b). As illustrated in Figure 1, each data point and their cluster centers trained in Task 1 are shifted after training for Task 2. Yu et al. (2020) proposed to compensate this gap without using any exemplars via domain shift. However, that study focused on classification tasks.

In this work, we propose a novel method, regularized memory recall mechanism with additional domain shift estimation (RMR_DSE), to alleviate CF in continuous seq2seq language generation. The first RMR component improves the regularization-based method through adaptive regularization. We convert fisher information matrix deployed in EWC to a tunable hyperparameter constrained by a vocabulary-related hyperparameter. Further, we add a regularizer derived from the gradients of the generative function to tune model parameters. The second DSE component compen-

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 5441 - 5454

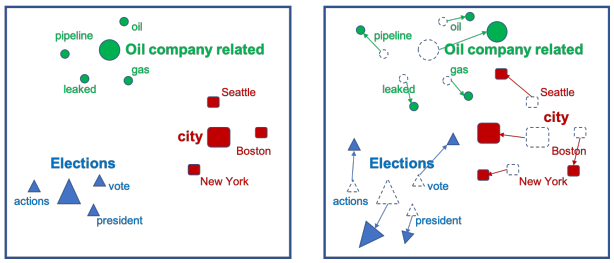

Figure 1: Illustration of Domain Shift: (a) Data with three relevant topic/cluster in the embedding space after model trained on task 1. (b) Data with previous topics in the embedding space after the model trained on task 2, the arrow indicates the domain shift between two tasks, which is what our DSE aims to estimate.

sates the representation difference from the two domains by estimating the semantic gap between them. We obtain embeddings for the current task's data using the previous and current models, and group embeddings from previous models into clusters. Semantic shifts are computed for each cluster, and then used during inference time on previous test data to adjust its semantic representation to match the current model.

Our main contributions are:

- We design a new regularized algorithm to consider parameters for the previous tasks while training for the current task for LLL seq2seq generation.
- We propose to estimate domain shifts in the embedding space of consecutive models via prototypical representations, thus alleviating the need for data storage.
- Our experiments on seq2seq generation benchmark datasets show that our model achieves state-of-the-art results in current task learning and reduces forgetting rates for previous tasks.

2 Related Work

2.1 Life Long Learning (LLL)

Life long learning has been studied from a few perspectives, including data buffering, regularization and prototype keeping. Replay based methods can be used in data buffering or prototype keeping. They usually keep a small amount of real samples from old tasks or distill the knowledge from old data and recreate pseduo-data of old tasks for later training. Using these sampled data or pseudo data can prevent weights from deviating from previous status (Rolnick et al., 2019; Wang et al., 2020; Lopez-Paz and Ranzato, 2017). The main idea of this approach is to assign a dedicated capacity inside a model for each task. After a task is completed, the weights are frozen as one prototype (Wang et al., 2021b; d'Autume et al., 2019; Wang et al., 2021a). Both data buffering and prototype keeping need storage of either data samples or model weights, i.e., they require extra memory to memorize important information of previous tasks. Another LLL method is regularization based, which adds a regularization term to weights when learning them for a new task in order to minimize deviation from previously trained weights. Most regularization based methods estimate the importance of each parameter and add them as a constraint to the loss function. Different algorithms have been designed to achieve this. For example, elastic weight consolidation (EWC) calculates a Fisher information matrix to estimate the sensitivity of parameters (Kirkpatrick et al., 2017); memory aware synapses (MAS) (Aljundi et al., 2018) uses the gradients of the model outputs; and episodic memory or gradient episodic memory (GEM) (Li et al., 2017; Lopez-Paz and Ranzato, 2017) allows positive backward transfer and prevents the loss on past tasks from increasing. These methods all attempt to slow down the learning of parameters that are important for previous tasks.

2.2 LLL in Seq2seq Language Generation

In Seq2seq language generation, not much work has been done in LLL. The most relevant work is from (Mi et al., 2020) where a framework of sequential learning is designed for task-oriented dialogs. Specifically, they replay prioritized exemplars together with an adaptive regularization technique based on EWC. They store representative utterances from previous data (exemplars), and replay them to the Seq2seq language generation model each time it needs to be trained on new data. They achieved good results on the MultiWOZ-2.0 dataset. However, their work needs to store data from previous tasks, and thus may not be scalable to large data environments. In addition, their system is specifically designed for the MultiWOZ task and lacks generalization to other tasks. In contrast, our proposed RMR_DSE method aims to fit different seq2seq language generation applications, therefore it is easy to be integrated to tasks such as summarization, translation, paraphrases, dialog response generation.

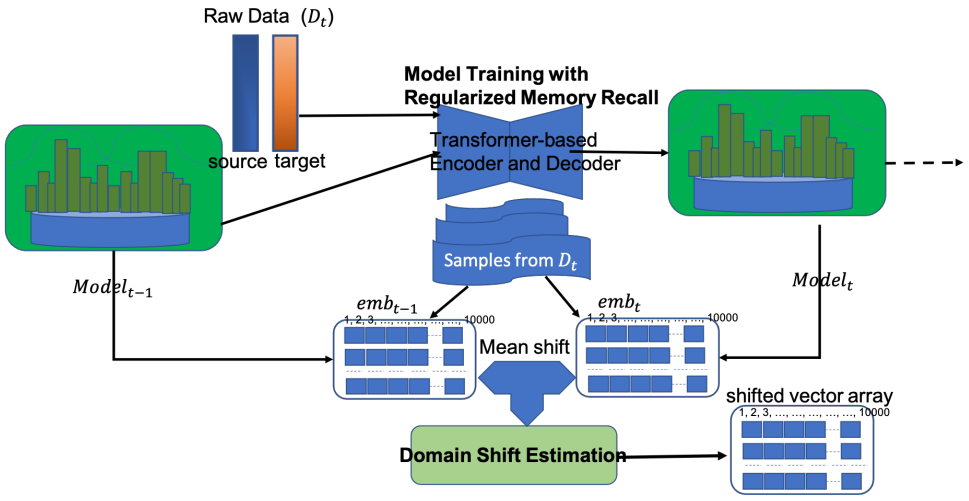

Figure 2: Overview of RMR_DSE for LLL Seq2seq Language Generation. Figure best viewed in color.

3 Proposed Method

In this section, we introduce our proposed framework RMR_DSE, as illustrated in Figure 2. In the LLL scenario, models are trained for a sequence of domains (or tasks). For the first task, the model can be trained from scratch or using pretrained models. Starting from the second model, parameters are initialized with the previous model. Our RMR_DSE method is a combination of regularization and domain shift estimation (DSE). For the first part, we incorporate the mechanism of EWC to obtain a regularized memory recall mechanism (RMR) to optimize model training. For the second DSE part, we design an algorithm to integrate both K-means and mean shift. A shift of embedding representations is estimated using the previous and current models, and it is compensated to reduce forgetting when we evaluate the current model on test data from the previous domain. Although RMR_DSE is a generic mechanism, we evaluate on two seq2seq language generation tasks in this work. The underlying models in seq2seq language generation can be any models, including transformers, LSTM or variational auto-encoders. The following describes the RMR and DSE components in details.

3.1 Adaptive Regularization of Memory Recall

Although elastic weights consolidation (EWC) is generic enough to fit all tasks, it regularizes all the parameters to the same extent. In order to differentiate parameter importance, we propose an improved EWC, the regularized memory recall (RMR) mechanism, where the training objective is:

$$Loss_t = \lambda(\tau)L_t(\theta) + (1-\lambda(\tau))\gamma F \sum_{ij} \Pi_{ij}(\theta_{ij} - \theta^*_{ij})^2 \quad (1)$$

where $L_t(\theta)$ is the loss for the current task. In our generation task, we use standard label smooth cross entropy.

In the regularization part, θ^* represents the parameters from earlier models, e.g., that learned from task $t-1$:

$$\theta^* = \arg\min_{\theta}\{-\log p(\theta|D_{t-1})\} \quad (2)$$

θ is for the current model, and indexes i and j are used to represent connections between pairs of neurons n_i and n_j in two consecutive layers. Adding the regularization term using the differences between the two models is expected to memorize important old parameters while updating values of current parameters.

In EWC, F is the diagonal element of the Fisher Information Matrix. It measures the importance of θ after being updated with the set of data points in the current task to previous tasks. However, in our work, we suppose that we do not have access to data of previous tasks. Therefore, we cannot compute fine-grained values based on data, and thus convert F to a tunable hyperparameter without dependency on data from the previous task. It is used to penalize the quadratic function $(\theta_{ij} - \theta^*_{ij})^2$.

In order to make the parameters in a reasonable range, we add a hyperparameter γ to help tune F. For simplicity, the value of γ is determined by the ratio of the vocabulary size of the current corpus and the previous ones.

$$\gamma = \gamma_{base}\sqrt{V_{1:t-1}/V_t} \quad (3)$$

Furthermore, to control the learning between the loss for the current task and the regularization term, we add $\lambda(\tau)$, a sigmoid annealing function:

$$\lambda(\tau) = \frac{1}{1 + exp(-k) * (\tau - \tau_0)} \tag{4}$$

where k and τ_0 are hyperparameters controlling the annealing rate and timesteps, and τ refers to the update timesteps during fine-tuning.

Finally similar to MAS (Aljundi et al., 2018), to model the varying importance of individual parameters and the changes of model parameters, we integrate Π for fine-grained regularization:

$$\Pi_{ij} = \frac{1}{N} \sum_{n=1}^{N} ||g_{ij}(x_n)||^2 \tag{5}$$

where $g_{ij}(x_n) = \frac{\partial(G(x_n;\theta))}{\partial\theta_{ij}}$ is the gradient of the learned generative function (such as transformers, LSTM or VAE) with respect to parameter θ_{ij} evaluated at the data point x_n of the current task. Parameters with small importance weights can be changed to minimize the loss for subsequent tasks while parameters with large weights are kept unchanged.

3.2 Deploying Domain Shift Estimation to Make Up Semantic Drift

It has been shown in previous studies (Yu et al., 2020; Wang et al., 2021b) that catastrophic forgetting is mainly due to the domain shift in the embedding space after the model is updated on new domains. When testing on previous domains, the embeddings derived from the model trained using new domains are suboptimal and lead to performance degradation. Hence, in addition to regularizing the model during training as described in the section above (RMR regularization), we propose to deploy domain shift estimation (DSE) to compensate the embedding drift to further reduce the forgetting.

Note that in our life long learning scenario, we do not rely on the data previously used for model training in new domains, but do have access to the trained model from the previous task. We thus approximate DSE with gaps between embedding representations of the current data based on both $model_{t-1}$ and $model_t$, using the following steps.

First, for a data point i in the current training task t, its representation shift is:

$$\delta_i^{t-1->t} = \mathbf{z}_i^t - \mathbf{z}_i^{t-1} \tag{6}$$

where $\mathbf{z}_i^t$ and $\mathbf{z}_i^{t-1}$ refer to the embedding of point i based on $model_t$ and $model_{t-1}$ respectively. In the seq2seq language tasks, these are the encoder outputs.

Second, we deploy unsupervised clustering methods to identify some centers and mean shift (Anand et al., 2013) for the embeddings using $model_{t-1}$. Specifically, with K-means, we find K embedding centers, each of them represented as μ_k^{t-1}. Then, around each center, we find some number of samples to compute mean shift (we use $3k$ in our experiments).

The mean shift $\mathbf{M}_h(x)$ for each data point of each cluster is defined as:

$$\mathbf{M}_h(x) = \frac{\sum_i^n G(\frac{x_i - x}{h_i}) w(x_i)(x_i - x)}{\sum_i^n G(\frac{x_i - x}{h_i}) w(x_i)} \tag{7}$$

where $G = e^{\frac{||x_i - x||}{2h^2}}$ is the Gaussian kernel, h is the bandwidth, x_i is the data belonging to the cluster containing x, and n is the number of data points in each cluster.

Finally for each cluster, we compute a domain shift vector as: $\Delta_{dse_k}^{t-1->t}$.

$$\Delta_{dse_k}^{t-1->t} = \frac{\sum_i \mathbf{M}_h(x_i)) \delta_i^{t-1->t}}{\sum_i \mathbf{M}_h(x_i)} \tag{8}$$

where the summarization is performed over all the data points belonging to cluster k.

We use K such vectors as a domain shift estimate between the models trained for two different domains. When evaluating on a previous domain using the model trained for a new domain, for a test data point we first calculate the similarity between its embedding ($e_{d_{t-1}}^t$) encoded by $model_t$ and the stored cluster centers, and then the corresponding domain shift vector for that cluster is subtracted from $e_{d_{t-1}}^t$ before the generation decoding step. If we have multiple tasks (m for example), we can perform a series of subtractions, i.e.,

$$e_{d_{t-m}}^t = e_{d_{t-m}}^t - \Delta_{dse_k}^{t-m->t-m+1} - \Delta_{dse_k}^{t-m+1->t-m+2} - ... - \Delta_{dse_k}^{t-1->t}$$

Algorithm 1 describes the domain shift estimation of RMR_DSE. We have input of embeddings encoded by $model_{t-1}$ and $model_t$. Firstly, KMeans is employed to obtain K embedding centers for the training data (we obtained best results when $K = 3$ in our experiments). Also, since KMeans is sensitive to the initialization of center points, we have to run multiple rounds (about 5 rounds in our experiments) before we can obtain the best ones. FAISS, the fast KNN-based embedding search tool is utilized to search relevant

Algorithm 1: Domain Shift Estimation

Input: embeddings of training data of task t encoded by $model_{t-1}$ and $model_t$, given function shift_point, Euclidean_dist, σ
Output: Estimated domain shift Δ_{dse}

Deploy KMeans to find out K embedding centers among embeddings of training data of task t encodeded by $model_{t-1}$
Deploy FAISS search to find M samples, **neighbor** close to each embedding center
$\delta^{t-1->t} =$ **neighbor**$_{emb\_new}$ − **neighbor**$_{emb\_old}$
max_min_dist = 1
mean_shift_points=[]
while $max\_min\_dist < MIN\_DISTANCE$ **do**
    max_min_dist → 0
    // compute mean shift vector
    for $i = 1 \rightarrow m$ **do**
        if *not need_shift[i]* **then**
            continue
        p_new = **mean_shift_points**[i]
        p_new_start = p_new
        p_new = shift_point(p_new, **neighbor**$_{emb\_old}, \sigma$)
        dist = euclidean_dist(p_new, p_new_start)
        if $dist > MAX\_MIN\_DIST$ **then**
            $MAX\_MIN\_DIST = dist$
        if $dist < MIN\_DIST$ **then**
            $need\_shift = True$
        mean_shift_points[i] = p_new
    $\Delta_{dse}^{t-1->t} = \frac{\sum(\textbf{mean\_shift\_points} \times \delta^{t-1->t})}{\sum(\textbf{mean\_shift\_points})}$
return $\Delta_{dse}^{t-1->t}$

Table 1: Dataset stats for paraphrase generation task (number of sentence paraphrase pairs).

	Quora	Twitter	Wiki_Data	total
train	111,947	85,970	78,392	276,309
valid	8,000	1,000	8,154	17,154
test	37,316	3,000	9,324	49,640

samples for each embedding center. In the while loop, we implement mean shift method to estimate whether each sample needs a mean shift against their closest embedding center. We finally obtain a list of mean shift points and their values. Those values are used as weights to be multiplied to the difference between embeddings from $model_{t-1}$ and $model_t$. In the inference stage, we select $\Delta_{dse}^{t-1->t}$ based on the closeness of input test sentences to the embeddings.

4 Experiments on Paraphrase Generation

To test RMR_DSE's generalization, we apply it to two datasets that both follow the seq2seq generation setup but are quite different tasks. In this section, we focus on the paraphrase generation task.

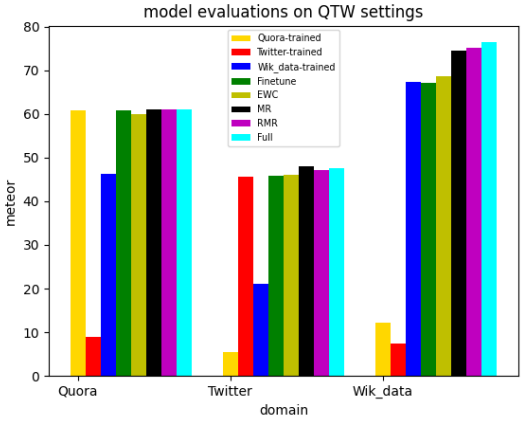

Figure 3: Results of Meteor scores of QTW setting when evaluating on the current task.

4.1 Experimental Setups

For paraphrase generation, we use three existing paraphrase datasets, Quora, Twitter and Wiki_data, in a sequential fashion, that is, the model is first trained on the Quora data, then Twitter, then Wiki_data. We name this experimental setting as QTW. Statistics of the data are provided in Table 1.

We use a current SOTA generation model, BART, as the seq2seq backbone in our LLL framework and the other compared methods. We compare our approach with the following baselines.

- Finetune: for each task, each model is initialized with the model obtained until the last task, and then fine-tuned with the data of the current task.
- Full: we train a model with all the three data sets together.
- EWC: the model is trained with the base EWC model on the data from the current task with the initialization of the previous model.

For our proposed RMR_DSE, we also evaluate different configurations including MR, RMR, and DSE only for an ablation study. The details of parameters implementations are given in Appendix.

For evaluation metrics, we use BLEU-4, ROUGE-L and METEOR for the generation task. Because of space limit, we only report bar figures with METEOR scores and leave tables with full scores in the Appendix. To measure the forgetting rates of different methods, we apply models trained using new data to past data.

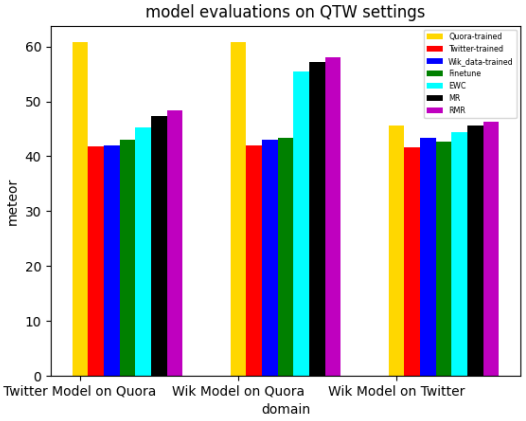

Figure 4: Results of Meteor scores of QTW setting while evaluating on previous datasets.

4.2 Results

Evaluating on the Current Task

For QTW setting, Figure 3 shows results when models are evaluated on the data corresponding to the current task. Note that since DSE is only applicable when models are evaluated on the past data, we do not use DSE in this experiment. From left to right are domains for Quora, Twitter and Wik_data respectively.

Each of the domains has 8 results. The first three bars are results from independent models, that is, the BART models are trained on only one of the datasets in QTW. As expected, models trained on the matched domain achieve higher performance than otherwise. And there is a large performance drop when using models trained from mismatched domains. This is mostly because of the different writing styles of the three datasets. Wik_data is the most formal one, and Twitter is the most informal one.

In the fourth bar, the BART model is trained in finetune mode, i.e., in QTW order, the model is initialized with that trained in the previous domain and fine tuned using the subsequent domain. We can see that results on both Twitter and Wik_data test data are slightly lower than those when models are trained directly on the corresponding training data. Again, this suggests pretraining the model with mismatched data is not beneficial. The results from the EWC baseline are not consistently better than the finetune method, showing the limited effectiveness of EWC regularization. In contrast, our proposed approaches obtain better results than Finetune. Even for the first task, Quora, we observe around 1% better results. This demonstrates that even for pretrained models, regularization shows positive effects. For the later tasks, there is about 3-4% performance increase on Twitter data and 7-9% for Wik_data. This shows the effectiveness of both MR and RMR. In addition, six out of nine results from RMR win about 1% over MR. This shows that further regularization with quadratic penalty has positive impact on selection of important parameters. The last bar is the results of Full. Since the model has seen all the data, it is not surprising that results for both Twitter and Wik_data are better than our models, and it may be partly because of some similarity in Quora and Wik_data.

Evaluating on Previous Tasks

Figure 4 shows the results when models trained on new domains are evaluated on data from past domains. Since we only report results of QTW setting in the main page, they are presented for evaluating on Quora and Twitter data. For the Quora test set, we show results after training with Twitter data, and then subsequently Wik_data. The first bar of each domain is the result of the BART model trained on only the corresponding data. The second bar uses the baseline fine tuning fashion. We show results using our proposed method, RMR_DSE, and its individual components, MR, RMR, and DSE. Each of them yields much better results than the finetune or EWC baselines, with much less drop rates. On all the datasets, we can see the incremental improvement from MR to RMR and to RMR_DSE. This shows each module can reduce forget rates. In addition, after the model is trained on Wik_data, forgetting rates for Quora Test (the first dataset) are even lower than the model trained on Twitter. This again indicates Wik_data and Quora are more similar in style than Twitter.

4.3 Case Studies

In Table 2, we show some generated samples from QTW setting using the baseline BART model and our RMR_DSE model. All examples are results generated by $model_t$ on $data_{t-1}$. Among the five examples, the first one is from Quora, the last one from Wiki data and the other three from Twitter. The reason that we select more samples from Twitter is that we find Twitter is the most informal in style with quite many fragments. Hence, it is the hardest for the generation task and has lowest generation performance and forgetting reduction rates. In the four samples, the italicised parts are the key words. From the table, we can observe that com-

pared to *BART*, *RMR_DSE* has better performances on all of the three datasets. The *BART* model misses all of them except $drilling$. In contrast RMR_DSE succeeds in all cases without forgetting the previously learned patterns.

5 Experiments on Dialog Response Generation

5.1 Task Definition

In task oriented dialogs, recent neural generation methods use seq2seq setup for response generation given the dialog act of the target response. A dialog act is defined as the combination of intent I and a set of slot-value pairs $S(d) = (s_i, v_i)_{i=1}^{p}$, where p is the number of slot-value pairs. Intent I refers to the utterance functionality, while slot-value pairs contain messages to express. For example, given input "Recommend (**Addr=regent stree, Fee=free, Name=Downing College)**", the system is expected to generate the response, "***[Downing College]*** *is my favorite. It is located on* ***[regent street]*** *and it's* ***[Free]*** *to get in*" and the slot type "[*Slot-Hotel-Area*]". Slot values are composed of domain and relevant attributes (details are in (Eric et al., 2019)).

5.2 Experimental Settings

We evaluate our model on response generation using the MultiWoZ-2.0 dataset (Budzianowski et al., 2018). It contains six domains (Attraction, Hotel, Restaurant, Booking, Taxi and Train) and seven DA intents ("Inform, Request, Select, Recommend, Book, Offer-Booked, No-Offer"). Following the setting in (Mi et al., 2020), the original train/validation/test splits of MultiWoZ are used. The detailed stats of the datasets are in Table 3.

We used the implementation in (Mi et al., 2020) and compare RMR_DSE to their proposed ARPER, an exemplar-replay based method. Since RMR is a generic regularized algorithm, it can be integrated to any framework by replacing either the optimizer or revising the loss function. Our comparisons to ARPER are made from two aspects: with and without exemplars.

For evaluation metrics, slot error rate (SER) and Bleu4 are used. Again, we only report bar figures with SER scores in the main pages while leaving full scores in the Appendix.

We also report metrics for two settings in LLL following (Mi et al., 2020):

$$\Omega_{all} = \frac{1}{T}\sum_{i=1}^{T}\Omega_{all,i}\ \Omega_{first} = \frac{1}{T}\sum_{i=1}^{T}\Omega_{first,i}$$

where T is the total number of LLL tasks; $\Omega_{all,i}$ is the average test performance on all the previous tasks after the i^{th} task has been learned; $\Omega_{first,i}$ is performance on the first task after the i^{th} task has been learned. The former measures the test accuracy of all of the test data for tasks seen to the i^{th} point while the latter is about the model's retention of the first task.

5.3 Baseline Methods

Following (Mi et al., 2020), two Seq2seq language generation models, conditional variational encoder (CVAE) and semantic conditioned LSTM (SCLSTM) are used as the generation models. We evaluate the following LLL settings:

- Finetune: This is finetuning the model trained from the previous domain using data for the current domain.
- Full: This is using the data from all the domains.
- ARPER: We run ARPER following the setting of the original paper (Mi et al., 2020).
- EWC: ARPER without exemplars is EWC.

5.4 Experimental Results

We make two comparisons with (Mi et al., 2020) based on exemplar numbers. The first one is when *RMR_DSE* and *ARPER* do not use exemplars. The reason for this comparison is that one of advantages of *RMR_DSE* is that it does not need extra storage to recover previous tasks. In addition, *ARPER* is equivalent to using *EWC* when no exemplars are used.

Figure 5 and Figure 6 show the results using different methods, where the red (CVAE) and blue (SCLSTM) bars are results of *RMR_DSE*, and the yellow (CVAE) and green (SCLSTM) bars are those of *ARPER*. We can see *RMR_DSE* obtains lower SER (higher Bleu4 as well, see appendix) results than *ARPER* in most cases. Table 4 shows that without exemplars, *ARPER* obtain even poorer results than *Finetune* while *RMR_DSE* achieves significantly better results than *ARPER* and Finetune in both Ω_{all} and Ω_{first}. Regarding the two seq2seq models, SCLSTM and CVAE, there are some different patterns when using RMR_DSE.

Table 2: Examples of the generated paraphrases by BART and RMR_DSE on QTW data setting.

SOURCE	BART	RMR_DSE	TARGET
Why is German Shepherd/Great Pyrenees mix coveted *among breeders?*	Why is German Shepherd/Great Pyrenees mix coveted *from browns?*	Why is German Shepherd/Great Pyrenees mix coveted *among breeders?*	Why is German Shepherd/Great Pyrenees mix coveted *among breeders?*
eyeing trump, Obama takes new action to ban arctic drilling	president Obama takes new action to ban drilling	Obama takes new action to ban arctic drilling	please save the earth mr. president . Obama takes new action to ban arctic drilling
death toll in 6.5 - magnitude earthquake in indonesia's aceh province increase to at least 52	a 6.5 earthquake in kills at least 26 people @cnn	death toll in 6.5 - magnitude earthquake in aceh province increase to at least 52	powerfull quake kills dozens at least 25 people were killed in an earthquake that struck indonesia's aceh province
pipeline 150 miles from dakota access protests leaks gallons of oil	the new york times pipeline 150 miles from dakota access pipeline .	pipeline 150 miles from dakota access leaks gallons of oil	of oil, or gallons, have leaked from the pipeline
described by many critics as more about " exploring the meaning of human life " or " the hollow existence of the American western suburbs " , the feature film itself has explicitly defied categorization by even the anonymous filmakers	Described by many critics as more about the meaning of human life " or " the existence of the American suburbs " , the film has explicitly defied categorization by the anonymous filmmakers .	described by many critics as more about " exploring the meaning of human life " or " the hollow existence of the American suburbs " , the film itself has explicitly defied categorization by even the anonymous filmmakers .	Described by many as about " the meaning of life " or " the hollow existence the American suburbs " , the film has defied categorization by even the filmmakers .

Table 3: Stats for full MultiWoZ-2.0 Dataset.

Domain Stats of MultiWoZ-2.0 data						
domain	Attraction	Hotel	Restaurant	Booking	Taxi	Train
total	8,823	10,918	10,997	8,154	3,535	13,326

Intent Stats of MultiWoZ-2.0 data		
Intents	total	Involved Domains
Inform	28,700	Attraction, Hotel, Restaurant, Taxi, Booking
Request	7,621	Attraction, Hotel, Restaurant, Taxi, Booking
Select	865	Attraction, Hotel, Restaurant, Taxi
Recommend	3,678	Attraction, Hotel, Restaurant, Taxi
Book	4,525	Booking, Taxi
Offer-Booked	2,099	Train
No-Offer	1,703	Attraction, Hotel, Restaurant, Taxi

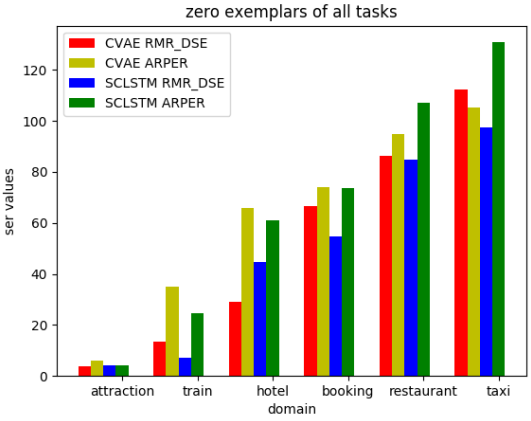

Figure 5: Slot error rate (SER) results for Ω_{all} when zero exemplar is used for all the methods. CVAE RMR_DSE and SCLSTM RMR_DSE obtained lower SER than CVAE ARPER and SCLSTM ARPER respectively.

Table 4: Average performance of continually learning 6 domains on MultiWoZ-2.0 with zero exemplars.

	Ω_{all}		Ω_{first}	
Methods	SER	bleu4	SER	bleu4
Finetune	64.46	36.1	107.27	25.3
ARPER on CVAE	63.54	36.00	102.87	19.22
ARPER on SCLSTM	66.87	35.64	100.56	21.09
RMR_DSE on CVAE	51.92	39.49	68.56	25.67
RMR_DSE on SCLSTM	**48.79**	**39.86**	**57.18**	**30.32**

The second comparison is made by using 250 exemplars, the same setting for *ARPER* as described in (Mi et al., 2020). In this setting, we also incrementally deploy *MR_DSE* and *RMR_DSE* in both CVAE and SCLSTM. In (Mi et al., 2020), using 250 exemplars for computing Fisher information matrix boosted the performances to a large degree. For equal comparison, Fisher information matrix is also utilized in *MR_DSE* and *RMR_DSE* in the computation of the loss function. Yet, we do not use it when updating the parameter importance. As shown in Table 5, *MR_DSE* and *RMR_DSE* achieve better results than *ARPER* in all metrics for both CVAE and *SCLSTM* models when we use 250 exemplars. We can also see that consistent with the results in the QTW setting, *RMR_DSE* always outperforms *MR_DSE*. These results illustrate the advantage of *RMR_DSE* over *ARPER* through the entire continual learning process.

Two additional observations can be summarized here. The first is that the results with exemplars obtain better Ω_{first} than Ω_{all}. This is consistent with the original paper and may indicate diverse tasks increase the difficulty of handling all the tasks. The

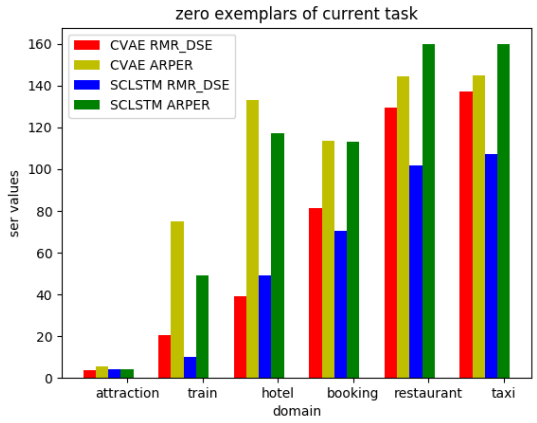

Figure 6: Slot error rate (SER) results for Ω_{first} when zero exemplar is used for all the methods. CVAE RMR_DSE and SCLSTM RMR_DSE obtained lower SER than CVAE ARPER and SCLSTM ARPER respectively.

Table 5: Average performance of continually learning 6 domains on MultiWoZ-2.0 with 250 exemplars. Best performance excluding "*Full*" are in bold in each column.

	Ω_{all}		Ω_{first}	
Methods	SER	bleu4	SER	bleu4
ARPER on CVAE	5.24	58.3	2.97	62.1
ARPER on SCLSTM	5.97	56.7	3.59	61.3
MR_DSE on CVAE	4.68	59.8	2.81	62.7
MR_DSE on SCLSTM	4.95	59.9	2.60	63.2
RMR_DSE on CVAE	4.52	59.8	**2.02**	63.5
RMR_DSE on SCLSTM	**4.38**	**60.3**	2.12	**63.6**
Full	4.26	59.9	3.60	61.6

second is that if we compare both Table 4 and Table 5, we can find that AEPER severely relies on exemplars while RMR_DSE does not. This sufficiently showcases RMR_DSE functions with less need of data storage.

6 Discussions

In this section, we provide some additional analyses and observations of this work. First, we take a closer look at the domain shift estimation (DSE) and why and how it works. Also, we will see what problems the current DSE framework has and whether we can make improvements on it. In Figure 7 we present three groups of embedding drifts to illustrate the intuitions behind the model. The ones on the left part are the TSNEs (the first two dimensions) of 10,000 embeddings generated with Quora model on Twitter data (upper) and with Twitter model on Twitter data (lower). In the middle are those of 10,000 embeddings generated with Twitter models on Wiki data (upper) and with Wiki model on Wiki data (lower). The right part are still the

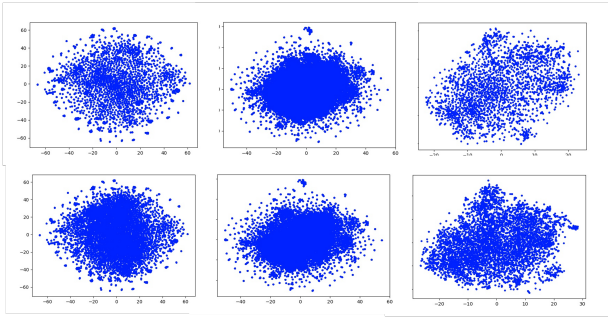

Figure 7: upper left: Embedding generated with Quora Model on Twitter data of 10000 samples. Lower left: Embedding generated with Twitter Model initialized with Quora model on Twitter data of 10000 samples. Upper middle: Embedding with Twitter Model on Wiki data. Lower middle: Embedding with Wiki Model initialized with Twitter model on Wiki data. Upper right: Embedding with Twitter Model on Wiki data. Lower right: Embedding with Wiki Model initialized with Twitter model on Wiki data.

Wiki data, but the number is only 3,000. We can see clear density differences between embeddings generated by older models (upper) and newer models (lower) although their value ranges are quite similar. Hence, our meanshift algorithm can make up such differences. However, the value range similarity also partly explains why DSE does not play a big role in the performance improvements.

This may give us some hints that the embedding learning may need improvements. Right now, we only deploy label smooth cross entropy loss in our whole framework. This loss function focuses more on the label differences (vocabulary distribution in natural language generation work). A natural extension is the addition of deep contrastive learning loss. Further, current DSE cares more about the sentence embeddings. However, the decoder in our framework uses beam search on token levels. Hence, algorithms considering both sentence and token level's distribution should help the shift estimation.

7 Conclusion

In this work, we introduce RMR_DSE, a generic LLL framework for addressing forgetting in seq2seq language generation learning. Our experimental results have shown that it outperformed state-of-the-art method by a large margin in two neural seq2seq language generation tasks, paraphrase generation and dialog response generation. Future work includes applying RMR_DSE to diverse generation tasks and generation network structures.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154.

Saket Anand, Sushil Mittal, Oncel Tuzel, and Peter Meer. 2013. Semi-supervised kernel mean shift clustering. *IEEE transactions on pattern analysis and machine intelligence*, 36(6):1201–1215.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet Kumar Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. 2019. Continual learning with tiny episodic memories. *CoRR*, abs/1902.10486.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Peter Ku, Anuj Kumar Goyal, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Tianxing He, Jun Liu, Kyunghyun Cho, Myle Ott, Bing Liu, James R. Glass, and Fuchun Peng. 2021. Analyzing the forgetting problem in pretrain-finetuning of open-domain dialogue response models. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 1121–1133. Association for Computational Linguistics.

Ferenc Huszár. 2018. Note on the quadratic penalties in elastic weight consolidation. *Proceedings of the National Academy of Sciences*, page 201717042.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. 2017. Learning from noisy labels with distillation. In *ICCV*.

Zhizhong Li and Derek Hoiem. 2018. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12):2935–2947.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul A. Crook, Bing Liu, Zhou Yu, Eunjoon Cho, and Zhiguang Wang. 2020. Continual learning in task-oriented dialogue systems. *CoRR*, abs/2012.15504.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Fei Mi, Liangwei Chen, Mengjie Zhao, Minlie Huang, and Boi Faltings. 2020. Continual learning for natural language generation in task-oriented dialog systems. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*.

Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. 2019. Latent replay for real-time continual learning. *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*.

Mark Bishop Ring et al. 1994. *Continual learning in reinforcement environments*. Ph.D. thesis, University of Texas at Austin Austin, Texas 78712.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P Lillicrap, and Greg Wayne. 2019. Experience replay for continual learning. *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2990–2999.

Wenbo Wang, Yang Gao, He-Yan Huang, and Yuxiang Zhou. 2019. Concept pointer network for abstractive summarization. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3067–3076.

Yigong Wang, Zhuoyi Wang, Yu Lin, Latifur Khan, and Dingcheng Li. 2021a. Cifdm: continual and interactive feature distillation for multi-label stream learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2121–2125.

Zhuoyi Wang, Yuqiao Chen, Chen Zhao, Yu Lin, and Latifur Khan. 2021b. Clear: Contrastive-prototype learning with drift estimation for resource constrained stream mining. In *Proceedings of The Web Conference*.

Zirui Wang, Sanket Vaibhav Mehta, Barnabás Póczos, and Jaime Carbonell. 2020. Efficient meta lifelong-learning with limited memory. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020.*

Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. 2020. Semantic drift compensation for class-incremental learning. In *CVPR*, pages 6982–6991.

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.

A Appendix

A.1 Domain Order Permutation

Due to page limit, we put tables with detailed evaluations (both on current data, see Table 6 and on previous datasets, see Table 7), including bleu4, rougeL, meteors on QTW settings in appendix. In the main page, we only show figures of meteor metrics. From Table 6, we can see that RMR_DSE takes the lead in almost all metrics in three tasks. Similarly, from Table 7, we can see that RMR_DSE has less forgetting rates than all other models as well.

Besides QTW setting, we also had run other two combinations including TQW and QWT setting. The results are basically consistent with QTW setting and can reach similar conclusion. The detail results are in Table 8 and Table 9.

A.2 Metrics Details

Throughout the paper, we use those evaluation metrics that have been widely used in the previous work to measure the quality of the paraphrases. In general, BLEU measures how much the words (and/or n-grams) in the machine generated summaries appeared in the human reference summaries. Rouge measures how much the words (and/or n-grams) in the human reference summaries appeared in the machine generated summaries. Specifically, we use the library[1] from HuggingFace to compute BLEU scores and *py-rouge*[2] to compute ROUGE scores. As BLEU and ROUGE could not measure the diversity between the generated and the original sentences, we follow unsupervised paraphrasing methods and adopt meteor to measure the diversity of expression in the generated paraphrases by penalizing copying words from input sentences. The introduction of Slot error rate, Ω_{all} and Ω_{first} can be seen in the data setting of MultiWoz2.

A.3 Bleu4 scores for MultiWoZ-2.0 dataset

Due to page limit, we put figures of Bleu4 for MultiWoZ-2.0 with zero exemplars in appendix as well. From all experiments, we can see that RMR_DSE achieves consistently better results than ARPER. Without doubt, ARPER is a strong baseline. Its adaptive EWC enables ARPER to update parameters discriminatively. However, RMR_DSE can update parameters more differentially with its memory aware penalty mechanisms.

[1] https://huggingface.co/metrics/sacrebleu

[2] https://pypi.org/project/py-rouge/

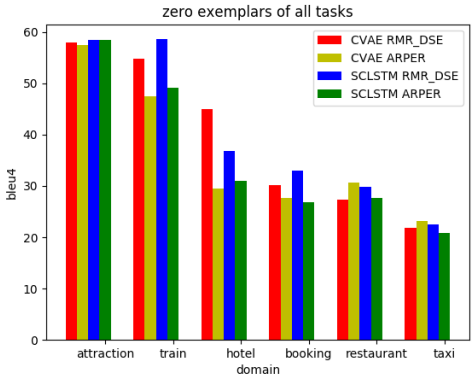

Figure 8: Results for Bleu4 of Ω_{all} when zero exemplar is used for all different methods. CVAE RMR_DSE and SCLSTM RMR_DSE obtained higher Bleu4 than CVAE ARPER and SCLSTM ARPER respectively.

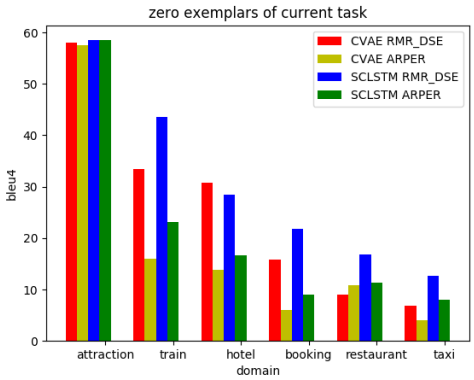

Figure 9: Results for Bleu4 of Ω_{first} when zero exemplar is used for all different methods. CVAE RMR_DSE and SCLSTM RMR_DSE obtained higher Bleu4 than CVAE ARPER and SCLSTM ARPER respectively.

A.4 Packages Used for Implementation

The relevant packages that we use in the implementation and their corresponding versions are as following: python==3.6.6, fairseq==1.0, torch==1.4.0, cuda==10.2, tensorboard==1.10.0, numpy==1.14.5, scipy==1.1.0, NLTK==3.4.5 and scikit-learn==0.21.3.

A.5 Parameter Update Analysis of RMR_DSE on Different Network Structures

In our experiments, we apply RMR_DSE to different network structures, involving BART, CVAE and SCLSTM. Therefore, we need to set up quite different values for their hyperparameters as shown in Table 10. Specifically, four hyperparameters, including update frequency ($update_{freq}$,how often to update Π), regularization coefficient reg_{λ} (the proportion of Π), anneal weights ($anneal_{w}$,how

Table 6: Results of model evaluations on QTW setting

	Quora Test			Twitter Test			Wiki Test		
Models	BLEU-4	ROUGE-L	METEOR	BLEU-4	ROUGE-L	METEOR	BLEU-4	ROUGE-L	METEOR
Quora-trained	36.98	58.19	60.76	2.12	6.13	5.49	4.51	11.21	12.13
Twitter-trained	3.18	11.46	9.01	36.47	47.49	45.57	4.60	9.76	7.50
Wiki_data-trained	22.38	43.44	46.23	9.32	17.93	21.03	48.03	69.70	67.43
Finetune	36.98	58.19	60.76	35.79	46.46	45.93	46.87	68.98	67.02
EWC	36.89	58.16	59.98	35.52	47.14	46.16	48.15	69.53	68.59
MR	37.98	59.19	61.11	36.98	49.39	**48.02**	53.93	74.49	74.53
RMR	**38.46**	**59.48**	**61.14**	**38.94**	**51.23**	47.12	**54.12**	**74.98**	**75.13**
Full	37.99	59.33	61.04	39.53	51.33	47.64	55.93	76.56	76.41

Table 7: Results of all the methods when testing new models on previous domains.

Quora test with Model trained with Twitter			
Models	BLEU-4	ROUGE-L	METEOR
Quora-trained	36.98	58.19	60.76
Finetune	20.77	30.80	41.75
EWC	21.63	31.53	42.03
DSE	21.58	31.95	42.98
MR	25.47	35.88	45.27
RMR	26.97	36.39	47.26
RMR_DSE	27.74	36.98	48.38

Quora test with Model trained with Wiki_data			
Models	BLEU-4	ROUGE-L	METEOR
Quora-trained	36.98	58.19	60.76
Finetune	22.83	42.16	42.03
EWC	24.63	44.35	43.02
DSE	23.79	43.49	43.35
MR	28.44	47.37	55.43
RMR	29.72	49.15	57.15
RMR_DSE	30.71	49.43	57.99

Twitter test with Model trained with Wiki_data			
Models	BLEU-4	ROUGE-L	METEOR
Twitter-based	36.47	47.49	45.57
Finetune	19.99	37.20	41.57
EWC	18.84	38.65	43.33
DSE	20.78	40.0	42.75
MR	21.92	38.69	44.36
RMR	24.15	42.11	45.59
RMR_DSE	26.73	43.85	46.23

Table 8: Results of model evaluations with TQW setting

	Twitter Test			Quora Test			Wiki Test		
Models	bleu4	rougeL	meteor	bleu4	rougeL	meteor	bleu4	rougeL	meteor
Finetune	36.47	47.49	45.57	34.32	55.63	58.93	44.94	67.87	66.15
EWC	36.55	48.32	46.73	34.37	54.32	59.31	48.72	68.21	69.14
MR_DSE	36.95	48.87	47.24	36.83	57.45	60.78	53.24	72.64	72.93
RMR_DSE	**37.26**	**49.33**	**48.58**	**36.90**	**58.86**	**61.33**	**54.53**	**73.25**	**73.50**
Full	39.53	51.33	47.64	36.81	58.39	60.70	55.93	76.56	76.41

Table 9: Results of model evaluations with QWT setting

	Quora Test			Wiki Test			Twitter Test		
Models	bleu4	rougeL	meteor	bleu4	rougeL	meteor	bleu4	rougeL	meteor
Finetune	35.93	56.32	59.23	45.12	53.23	67.78	36.82	47.99	46.45
EWC	35.85	55.64	59.95	49.14	67.99	69.85	36.96	47.45	46.94
MR_DSE	35.53	58.47	60.59	54.32	73.92	72.37	37.08	48.43	**47.91**
RMR_DSE	**35.61**	**59.38**	**61.58**	**53.87**	**74.63**	**73.13**	**37.58**	**50.36**	47.52
Full	36.81	58.39	60.70	55.93	76.56	76.41	39.53	51.33	47.64

Table 10: setting for prime hyperparameters

hyperparameter	BART	CVAE	SCLSTM
$update_{freq}$	2	2	2
reg_{λ}	0.9	0.1	0.01
$anneal_{w}$	0.1	0.01	0.05
$pretrain_{cof}$	5000	500	50

much we take parameter differences into consideration) and pretrain coefficient ($pretrain_{cof}$, the quadratic penalty derived from fisher information matrix, namely Π), are the most important ones.

For all of them, it looks the update frequency for Π can be once every two epochs. However, the other three are remarkably different. It seems to show that more complex network structures need higher penalty coefficients. Further, the value of Π seems quite related to the complexity of network structure. The three models need 5000, 500 and 50 respectively since BART has more complex network structure than CVAE and CVAE more complex than SCLSTM.