Enhancing Lifelong Language Learning by Improving Pseudo-Sample Generation

Kasidis Kanwatchara*
Chulalongkorn University
Department of Computer Engineering
kanwatchara.k@gmail.com

Thanapapas Horsuwan
Chulalongkorn University
Department of Computer Engineering
thanapapas.h@gmail.com

Piyawat Lertvittayakumjorn
Imperial College London
Department of Computing
pl1515@imperial.ac.uk

Boonserm Kijsirikul
Chulalongkorn University
Department of Computer Engineering
boonserm.k@chula.ac.th

Peerapon Vateekul*
Chulalongkorn University
Department of Computer Engineering
peerapon.v@chula.ac.th

To achieve lifelong language learning, pseudo-rehearsal methods leverage samples generated from a language model to refresh the knowledge of previously learned tasks. Without proper controls, however, these methods could fail to retain the knowledge of complex tasks with longer texts since most of the generated samples are low in quality. To overcome the problem, we propose three specific contributions. First, we utilize double language models, each of which specializes in a specific part of the input, to produce high-quality pseudo samples. Second, we reduce the number of parameters used by applying adapter modules to enhance training efficiency.

*Corresponding author.

Action Editor: Myle Ott. Submission received: 21 November 2021; revised version received: 29 May 2022; accepted for publication: 13 June 2022.

https://doi.org/10.1162/coli_a_00449

Third, we further improve the overall quality of pseudo samples using temporal ensembling and sample regeneration. The results show that our framework achieves significant improvement over baselines on multiple task sequences. Also, our pseudo sample analysis reveals helpful insights for designing even better pseudo-rehearsal methods in the future.

1. Introduction

Lifelong Learning (LL), or *continual learning*, is a machine learning paradigm that aims to emulate the learning process of biological intelligence (Parisi et al. 2019). The ultimate goal is to create a learner or agent capable of reusing and refining its knowledge while learning sequentially across potentially infinitely incoming tasks. However, current machine learning models are trained in an isolated environment (Chen and Liu 2016) where all data is assumed to be given during the training phase. When deployed in real-life environments, models suffer from performance drop during their lifetimes due to the non-stationary data distribution and concept drift (Schlimmer and Granger 1986). Attempting to naively subject a machine learning model to the LL setting is not practical due to a phenomenon called **catastrophic forgetting** (CF) (McCloskey and Cohen 1989), where gradient-based models completely forget all previous knowledge in favor of new knowledge.

Over the years, numerous approaches have been proposed to deal with CF; nevertheless, a significant portion of them targets the computer vision or robotics domain (Biesialska, Biesialska, and Costa-jussà 2020). As for **lifelong language learning** (LLL), the amount of research is relatively scant, with most being task-specific (Chen, Ma, and Liu 2018; Kutuzov et al. 2018). Recently, Sun, Ho, and Lee (2020) introduced a general LLL framework, called LAMOL, capable of solving any NLP task with a single language model (LM). This is achieved by formatting any input into a question-answering (QA) format (McCann et al. 2018). By exploiting the generative power of a pre-trained LM, LAMOL generates pseudo samples from previous tasks and utilizes them to train the model together with examples from a new task to alleviate CF. This also removes the need to store real samples from previous tasks. Their results show that LAMOL was able to outperform several existing LLL methods by a large margin and is only 2% below multitask training in terms of accuracy.

Although LAMOL was able to achieve good results in various datasets, LAMOL relies solely on these generated samples to alleviate CF. When trained on datasets with long texts, the LM struggles to properly capture the QA structure of input examples, which leads to various undesirable characteristics of the generated pseudo samples, namely: wrong format, uninformative, wrong task, and wrong answer. This is depicted in Table 1 (bottom) and will be explained in Section 2.2. As a result, LAMOL cannot effectively prevent CF in this situation.

Hence, in this article, we address this problem by introducing a novel Double LM framework. With an additional LM, we decompose LAMOL's learning objective into two subtasks and apply each LM to solve each subtask. Consequently, this training paradigm allows the pseudo sample generation process to be more controllable and in turn increases the quality of the generated pseudo samples. Additionally, to lower the resource requirements imposed by the added LM, we apply adapter modules (Houlsby et al. 2019) to imitate the function of the second LM. Finally, we also propose enhancing pseudo sample quality with a semi-supervised learning technique (i.e., temporal ensembling) and by detecting and reducing the number of uninformative pseudo samples.

Table 1
Top: The depiction of ideal characteristics of pseudo samples with explanations below the samples. **Bottom**: The depiction of various undesirable characteristics of pseudo samples with explanations below the samples. [SEP] and [ANS] are special tokens indicating the structure of the samples, while [MOVIE] and [SCIFACT] are task-specific tokens telling the language model to generate pseudo samples of the corresponding tasks.

High-Quality Pseudo Samples	
Correct Format	[MOVIE] this movie is good [SEP] what is the sentiment of this review? [ANS] Negative *The sample has three parts in the right order (context, question, answer) with the correct special tokens.*
Informative	[SCIFACT] The Drosophila lymph gland is a haematopoietic organ in which . . . *The sample is coherent and meaningful.*
Correct Task	[SCIFACT] The present study was conducted by . . . *Given a task-specific token, a sample is generated accordingly.*
Correct Answer	[MOVIE] this movie is good [SEP] what is the sentiment of this review? [ANS] Positive *The answer of the sample corresponds with the context and the question.*
Low-Quality Pseudo Samples	
Wrong Format	[MOVIE] this movie is good [ANS] Negative *The format is incorrect due to the missing question part.*
Uninformative	[SCIFACT] of the [SEP] function of a function of the function of an element of a function of . . . *The generated context is uninformative and incomprehensible.*
Wrong Task	[MOVIE] The present study was conducted by . . . *The generated context seems to be from the SciFact task, contradicting the task token [MOVIE].*
Wrong Answer	[MOVIE] this movie is good [SEP] what is the sentiment of this review? [ANS] Negative *The answer of the sample is incorrect according to the context and the question.*

In our experiments, we evaluated our proposed solutions on two sets of complex tasks up to five tasks long. We show that our solutions are able to improve upon vanilla LAMOL with statistical significance in both sets of tasks, gaining up to 16.27% average accuracy and is only 0.7% below from using real examples for rehearsal.

To sum up, our contributions are as follows:

- Introducing a new pseudo-rehearsal based LLL framework that is more suitable for datasets with longer texts.
- Utilizing adapter modules (Houlsby et al. 2019) to reduce parameters and computation requirements of our new scheme.
- Further improving pseudo samples quality using a semi-supervised learning technique and re-generation strategy.
- Analyzing pseudo samples and providing insights of the effects of pseudo samples on the final lifelong learning performance.

The rest of this article is structured as follows. Section 2 provides background and related work that are relevant to our proposed solutions and baselines used in the experiments. Section 3 introduces the methodology of our work and explains the pseudo sample analysis process we conducted. Section 4 describes the set-up of our experiments where the results and discussion are then presented in Section 5. Finally, Section 6 concludes our paper.

2. Background and Related Work

In this section, we briefly introduce existing works in the field of lifelong learning as well as LAMOL and Adapter Modules—components upon which our proposed solutions build.

2.1 Lifelong Learning

Lifelong learning is one of the most challenging machine learning paradigms. Until now, researchers have introduced many methods to alleviate the problem of CF, all of which can be broadly classified into three main approaches:

- **Architectural-based approach** mimics the modular nature of the human brain and dynamically introduces task-specific parameters to accommodate new tasks (Rusu et al. 2016; Wen, Tran, and Ba 2020). This group of methods can retain perfect knowledge of past tasks; however, they suffer from the constantly growing parameters.
- **Regularization-based approach** utilizes a regularization term that promotes knowledge consolidation and prevents large changes to parameters deemed crucial for previous tasks (Kirkpatrick et al. 2017; Aljundi et al. 2017). These methods do not require additional parameters or storing past data. Nevertheless, with a limited number of parameters, new knowledge may eventually overwrite previously learned knowledge.
- **Rehearsal-based approach** relies on a set of stored data that is replayed during the learning phase of a new task (Lopez-Paz and Ranzato 2017; de Masson d'Autume et al. 2019). To avoid relying on stored past data, *pseudo-rehearsal* methods instead utilize a generative model capable of creating potentially unlimited pseudo training data. LAMOL and our work fall into this category.

In the context of LLL, rehearsal-based approaches have been shown to be the most promising group of methods, outperforming notable methods of other approaches such as EWC (Kirkpatrick et al. 2017) and MAS (Aljundi et al. 2017) on various NLP tasks (Sun, Ho, and Lee 2020; Wang et al. 2020; Han et al. 2020; Sprechmann et al. 2018; Sun et al. 2020). Similarly, pseudo-rehearsal methods have been receiving more attention with the advancement of language models (Merity, Keskar, and Socher 2017; Radford et al. 2019). Complex data distributions can be modeled more accurately, leading to the increasing quality of generated data. This in turn improves the performance of pseudo-rehearsal methods. However, in most cases, replaying real data still outperforms synthetic data replay. This is due to the sub-optimal quality of the pseudo data. Multiple work has been proposed in order to address the problem in the computer vision domain. Solinas et al. (2021) proposed storing a small amount of real data as seeds for generating pseudo data using a re-injection sampling procedure (Ans and Rousset 1997). They were able to outperform strong rehearsal-approach baselines such as experience replay (Chaudhry et al. 2019b). Silver and Mahfuz (2020) generated pseudo samples using a stack of Restricted Boltzmann Machine (RBM) (Hinton 2012) and select only those that most adhere to training data distribution. Only pseudo samples with reconstruction error from the trained RBM lower the mean squared error of all generated samples were utilized, while the rest were discarded. Consequently, by training the model with the remaining pseudo samples, they were able to match the performance of the model trained with real examples. In contrast, Pomponi, Scardapane, and Uncini (2020) approached the problem in the embedding space. With a generative model composed of a normalizing flow (Papamakarios et al. 2021), they were able to achieve significantly less CF when compared with strong regularization-approach and rehearsal-approach baselines. To the best of our knowledge, our work is the first attempt to explicitly improve the quality of pseudo samples in the NLP domain, especially when the tasks to be learned contain long texts but with insufficient training data.

2.2 LAMOL

Inspired by Shin et al. (2017), LAMOL (Sun, Ho, and Lee 2020) leverages a single GPT2 language model (LM) (Radford et al. 2019) to prevent CF by utilizing the innate generative capability of the LM to create pseudo samples that are later learned jointly with data from a new task. By following the decaNLP (McCann et al. 2018) data formatting protocol, where every NLP task can be converted into a QA format, LAMOL is able to tackle various NLP problems without requiring task-specific modules. Particularly, each example is converted to the following format: `[GEN] context [SEP] question [ANS] answer`, where `[GEN]`, `[SEP]`, and `[ANS]` are additional special tokens.

During training on a particular task τ_i, the LM is optimized on two objectives: $L = L_{QA} + \lambda L_{LM}$, where L_{QA} and L_{LM} refer to the QA loss and the LM loss, respectively, and λ is the weight of the auxiliary LM loss. Specifically, the GPT2 model learns to generate the correct answer (via the QA loss) while also trying to capture the distribution of given examples in order to better generate pseudo samples as an auxiliary task (via the LM loss). This is illustrated in Figure 1 (left). Note that they use categorical cross entropy for both types of losses. Then, before starting training on the next task τ_{i+1}, LAMOL uses the LM to generate pseudo samples of all previous tasks τ_t for $t = 1, \ldots, i$. Given a `[GEN]` token, the LM samples from the learned distributions until it outputs an `[EOS]` token. To prevent the LM from generating pseudo samples only for the most recent tasks, LAMOL adds a task-specific token for each task τ_i. Task-specific tokens can be utilized in place of the GEN token to inform the LM to generate pseudo samples from a particular task. A total of $\gamma|\tau_{i+1}|$ pseudo samples are generated, divided equally into $\frac{\gamma}{i}|\tau_{i+1}|$ samples for each previous task, where γ is a hyperparameter. Finally, the LM model learns from the mixture of new examples of task τ_{i+1} and pseudo samples of previous tasks.

Even though pre-trained LMs (such as GPT2) have shown impressive capabilities in learning various tasks, they require a large amount of training examples to converge properly. The problem is even more prevalent in complex tasks like language modeling. In real-life settings, labelled examples may be scarce, in which case the LM would struggle to appropriately capture the data characteristics, causing the generated pseudo samples to possibly be malformed. Because LAMOL formats data according to decaNLP, pseudo samples are required to be in the same form. Any pseudo sample with an incorrect format will be discarded and not used in training. In our experiments, we have observed that most pseudo samples generated from LAMOL do not have the correct format. Additionally, there are also many undesirable characteristics of the generated pseudo samples present. These include: (1) Wrong format: Generated pseudo samples do not conform to the QA format. (2) Uninformative: Many pseudo samples contain non-sensical texts. (3) Wrong Task: Pseudo samples generated do not match the

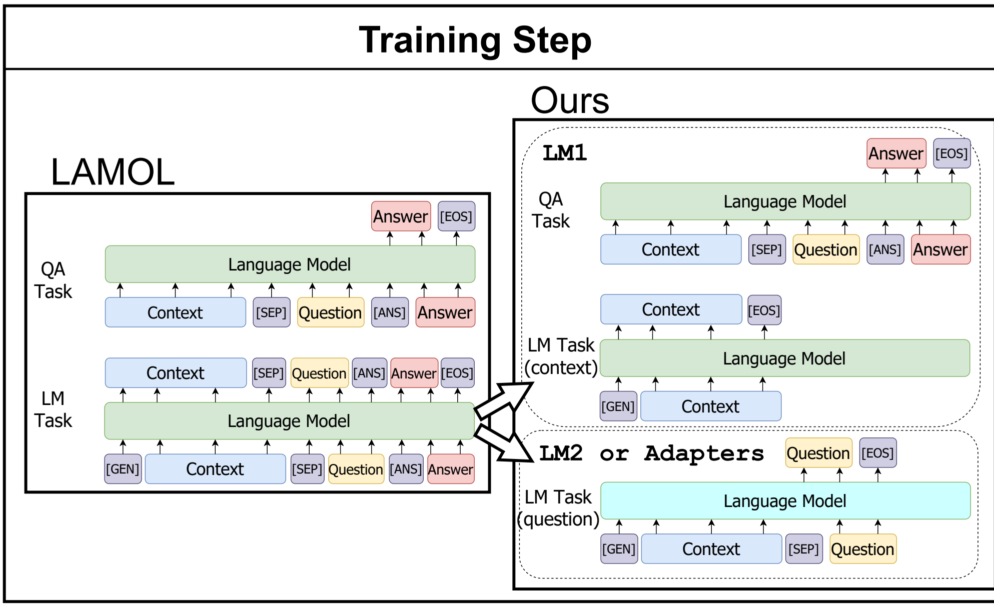


Figure 1
Left: Training step of LAMOL. In a single optimization step, a single language model is trained on the QA task (upper) and the LM task (lower). **Right**: Our framework utilizes two language models that focus on different parts of the input. The first LM is optimized on the QA task and the context generation task, while the second LM is optimized solely on the question generation task.

task-specific token specified. (4) Wrong Answer: Incorrect answers are generated for some pseudo samples. These problems are depicted in Table 1.

Consequently, without an adequate amount of usable pseudo samples, LAMOL loses the ability to prevent catastrophic forgetting and is comparable with only sequential fine-tuning.

2.3 Other Rehearsal Approaches

Research in rehearsal methods for LLL has seen an increase in traction over the last few years along with the advancement in LMs (Biesialska, Biesialska, and Costa-jussà 2020). They can be loosely categorized into two groups: methods that restrain themselves from making more than a single pass through the training data, and those that do not. Proponents of the former believe that this constraint constitutes a more realistic setting than that of the latter. We selected one recent method from each group as additional baselines in our experiments so we briefly describe them below.

2.3.1 Lifelong Language Knowledge Distillation (LLKD). Chuang, Su, and Chen (2020) utilize knowledge distillation (Hinton, Vinyals, and Dean 2015) in order to improve the LL performance of LAMOL. For each new incoming task, LLKD trains a disposable teacher model to compress the knowledge of the task and transfer it to an equivalently sized LL model via knowledge distillation. The soft supervision of the teacher model offers a more informative learning signal for the student model as opposed to the hard targets such as one-hot encoding. This can help the LL student model adapt to new tasks with more smoothness and reduce the interference of previous knowledge (Hou et al. 2018). According to the experiments in Chuang, Su, and Chen (2020), LLKD outperforms LAMOL on both classification tasks and sequence generation tasks. Nevertheless, when the teacher models fail to fully converge during training, the error from these models' estimations is amplified as the knowledge is transferred to the student model.

2.3.2 Meta Lifelong Learning. The ultimate goal of LL is to train a truly general model capable of solving all problems. Similarly, meta learning aims to find an initialization point for a model that is able to learn new tasks quickly. Therefore, multiple authors have proposed using different meta learning strategies in conjunction with standard lifelong learning techniques such as replaying past examples to solve the problem of LL. Research in this area considers a different problem set-up. Usually, the proposed methods limit themselves to making only one pass over the training data (i.e., one epoch) and without requiring task identifiers (e.g., task-specific token in LAMOL). One recent work in meta lifelong learning is Holla et al. (2020). They extend previous works OML (Javed and White 2019) and ANML (Beaulieu et al. 2020) with Experience Replay (ER) buffer, an episodic memory that randomly stores training examples in order to replay them later during training. Online aware meta learning (OML) trains a model on a meta-objective that attempts to learn sparse representations that mitigate forgetting, enabling OML to significantly outperform previous works. To improve the knowledge retention ability of the OML model, ANML introduces a parallel network called the neuromodulatory (NM) network that gates the activation of the prediction learning network (PLN; i.e., the OML model). The NM enables the ability to selectively suppress or allow, to various degrees, gradient updates to the PLN. In their experiments on large text classification datasets, up to 100k examples each, OML-ER and ANML-ER are able to rival LAMOL in terms of performance with only 1% of training samples replayed during training. Due to a slightly better performance in text classification tasks, we selected ANML-ER as an additional baseline.

2.4 Adapter Modules

Fine-tuning large pre-trained language models has pushed the limits of performance on various NLP tasks; nevertheless, it is highly inefficient because the whole model has to be fine-tuned individually for each task. To alleviate this issue, Houlsby et al. (2019) introduced adapter modules as an alternative paradigm for transfer learning.

Basically, the adapters (each of which is composed of two feedforward layers and a non-linear activation function) are used to adapt the content of each transformer block of the base pre-trained model. During the fine-tuning step, only the adapters are fine-tuned, so this can increase the training efficiency, thanks to a dramatically fewer number of parameters in the adapters, namely, only 0.5% to 8% of conventional large pre-trained models such as GPT2. The resulting model can achieve performance on par with fine-tuning the full model while also gaining significant speed-up.

In the context of our work, to reduce additional computational requirements caused by Double LM, we also propose using a single LM with adapter modules that can also achieve similar performance as the Double LM.

3. Methodology

In this section, we explain our proposed solutions. Section 3.1 presents the Double LM framework where an additional LM is leveraged to improve the quality of pseudo samples. Section 3.2 details the integration of adapter modules into our framework. We also describe the procedure of our pseudo sample analysis in Section 3.3. Finally, we detail our pseudo sample enhancement strategies in Section 3.4.

3.1 Double LM

Instead of allocating the model's learning capacity to model the input structure in addition to predicting the output, we propose decoupling the auxiliary language modeling task in LAMOL into two separate learning problems and applying a language model to solve each problem.

Training. Given that the required format of each input is `[GEN] context [SEP] question [ANS] answer`, in our framework, each LM is optimized on different part(s) of input. The problem set-up is shown in Figure 1 (right). The first LM would take the main responsibility of learning the QA task, that is, predicting an answer given a context and a question, and learning to model the context part of an example. Meanwhile, the other LM would learn to generate a question given an input context.

More formally, let $L(Y, \theta_{LM_i}(X))$ denote the cross entropy loss of LM_i with parameters θ on an input X with a target Y. The objective function of each LM would be defined as:

$$L_{LM_1} = L(Y_{QA}, \theta_{LM_1}(X)) + \lambda L(Y_{context}, \theta_{LM_1}(X)) \quad (1)$$

$$L_{LM_2} = L(Y_{question}, \theta_{LM_2}(X)) \quad (2)$$

Generation. By having two LMs, we can exactly control the pseudo sample generation process so that it conforms to the predefined format by the following steps:

1. First, LM_1 is utilized to generate the context part of the pseudo sample given a task-specific token indicating which task the generated context should belong to.
2. Second, a `[SEP]` token is appended to the previous output, and then LM_2 generates an appropriate question according to the given context.
3. Finally, an `[ANS]` token is appended to the previous output, and then LM_1 takes in the context and the question and predicts the answer as it would when training.

The process is illustrated in Figure 2 (bottom). As a result, the output pseudo samples are more likely to be in the correct format and more realistically imitate real training examples. Freeing the LM from learning the QA structure of examples also relaxes the complexity of the language modeling task, leading to better pseudo samples.

3.2 Adapter

Training another instance of GPT2 LM as in Section 3.1 imposes significant additional memory and computation requirements. Thus, we also propose to instead use the

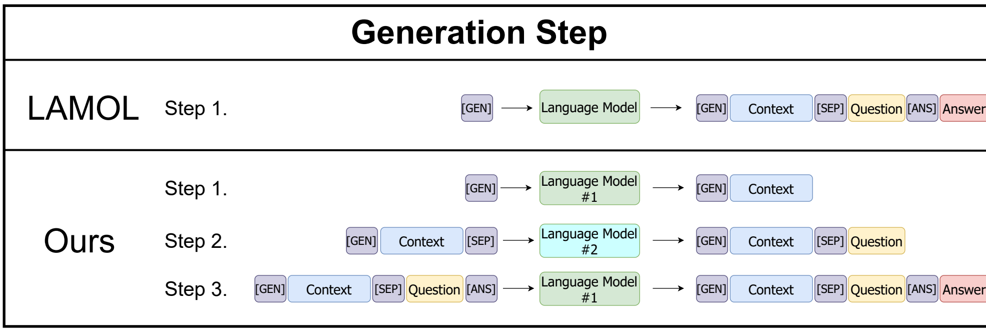


Figure 2
Top: Pseudo sample generation step of LAMOL. Given a `[GEN]` token, a single LM generates the whole sample. **Bottom**: Given a `[GEN]` token, LM_1 is utilized to generate a context. Next, given the context, LM_2 generates the corresponding question. Finally, given the context and the question, LM_1 generates an appropriate answer to complete the pseudo sample. Note that, for both LAMOL and our work, `[GEN]` will be replaced by a task-specific token to indicate the desired task of the generated pseudo sample.

adapter modules to mimic the function of the additional GPT2 model as a remedy to the problem.

In our framework, the adapters are added *after* the LM_1 has been trained on Equation (1). Because the adapter modules can utilize the information learned by the underlying model, we believe that it can effectively function as well as LM_2. Then, LM_1, which can now be referred to as the base model, is kept frozen, while we train the added adapters using Equation (2).

Due to the modular nature of the adapters, we can choose to ignore or "deactivate" the added adapters during the forward pass. By doing so, we get our base model LM_1 back. Therefore, to generate a pseudo sample, we start by deactivating the adapter modules and let the base model generate the context part. Next, we reactivate the adapters and feed the generated context into the model to get the corresponding question. Lastly, the adapters are deactivated once again, and now we utilize the base model to generate the answer to the pseudo sample.

3.3 Pseudo Sample Analysis

The performance of rehearsal-based LL approaches has been shown to rely mainly on the preserved samples. Multiple sample selection strategies have been devised in an attempt to choose data that can better represent previous distributions (Ramalho and Garnelo 2019; Wang et al. 2020; Toneva et al. 2019). However, for pseudo-rehearsal approaches, the problem is more complex due to the sub-optimal quality of generated pseudo samples. Therefore, in addition to the proposed framework, we conduct an analysis of pseudo samples in order to understand the effect of multiple aspects of pseudo sample quality on the final LL performance of pseudo-rehearsal methods.

In the analysis, pseudo samples are checked for four aspects of quality: (1) format correctness, (2) informativeness, (3) task correctness, and (4) answer correctness. The process is illustrated in Figure 3. First, we check if a pseudo sample conforms to the correct format. This is done by simply checking for three special tokens and their order. A pseudo sample has the correct format if it has a task-specific token, a `[SEP]` token, and an `[ANS]` token, in this specific order. The ones with an incorrect format will be classified

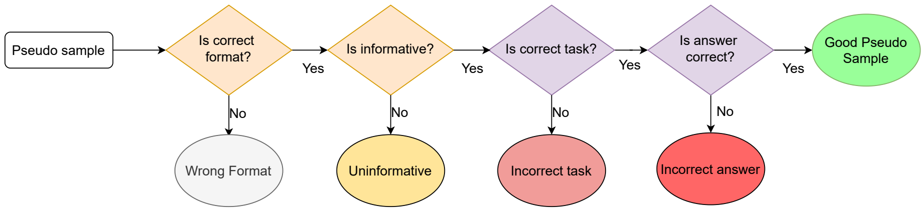


Figure 3
The process of our pseudo sample analysis. The color orange in each decision diamond refers to rule-based decisions while the color purple means decisions are made by classifiers.

as "Wrong Format." Next, pseudo samples with the correct format are checked whether they are informative or not. This process depends on the nature of the datasets used. We used a simple criterion: If the context part of a pseudo sample has less than 50 unique tokens, it is considered "Uninformative." Then we checked whether the content of each generated sample matches its task token or not. To do so, we trained BERT (Devlin et al. 2019) to classify the generated pseudo samples into their corresponding tasks. Note that we train this BERT model in the standard supervised learning fashion (not lifelong learning). To put it simply, it was trained using the training data from all the tasks. This model achieved perfect accuracy on the test data, and it was used for the purpose of analyzing the quality of pseudo samples only. If the content of a pseudo sample does not match its task-specific token, then it is categorized as "Wrong Task." Finally, we checked for the answer correctness by using a fine-tuned RoBERTa (Liu et al. 2019) model. We opted for RoBERTa due to its superior performance over BERT in predicting the correct answers. If the RoBERTa model agrees with the answer of the generated pseudo sample, it is considered "Correct Answer"; otherwise it is classified as "Wrong Answer."[1] As with the previous step, we fine-tuned one RoBERTa model per task and use all the fine-tuned models to analyze the quality of pseudo samples only.

3.4 Further Improving Pseudo Sample Quality

After analyzing the pseudo samples of our framework, we further attempted to enhance their overall quality in practice. We chose to improve two of the aspects mentioned in the previous section: answer correctness and uninformativeness.

To reduce the number of uninformative pseudo samples, we propose a simple filtering strategy, nicknamed ReGen. Pseudo samples that have less than 50 unique tokens in the context part, as in Section 3.3, are re-generated until we obtain all informative samples or reach the computation limit (set as ten iterations in our experiments).

To improve the pseudo sample answer correctness, we propose using a popular semi-supervised learning technique called Temporal Ensembling (Laine and Aila 2017). During the generation process, two models from the last two epochs of training are utilized to vote on answers for pseudo samples. We only keep pseudo samples on which the two models agreed on an answer, whereas the rest are replaced with a new batch of pseudo samples. This is based on the assumption that answers that are not stable even when reaching the end of the training are not likely to be reliable answers.

1 It is important to note that "Correct Answer" and "Wrong Answer" are not definitely correct and wrong, respectively. This is because the fine-tuned RoBERTa models we used are not perfect. The accuracy of the models for the BoolQ, Movie, and SciFact tasks are 80.33%, 99.5%, and 77.66%, respectively.

4. Experimental Set-up

This section reports our experimental set-up. Section 4.1 contains the details of datasets and metrics used in the experiments. Section 4.2 describes the implementation details, hyperparameters, and the methods to be compared.

4.1 Datasets

We performed our experiments on five datasets, selected due to their high complexity and small size. The details of all datasets are listed below and data statistics are in Table 2. In Table 3, we detailed the QA components for each task. Note that both LAMOL and our framework do not make use of the validation sets.

- BoolQ (Clark et al. 2019): a dataset containing yes/no questions generated from selected Wikipedia passages.
- Movie Reviews (Zaidan, Eisner, and Piatko 2008): a dataset that includes movie reviews with positive/negative sentiment labels.
- SciFact (Wadden et al. 2020): a dataset of scientific abstracts paired with claims written by experts. The objective is to identify whether the claim is supported by the given documents.
- Fever (Thorne et al. 2018): Fact Extraction and VERification is a dataset consisting of claims and textual sources, that is, documents. The task is to verify if each claim is supported by a given document. To make the task more challenging, we randomly sampled data from the dataset so that the size is comparable with other datasets in our experiment.
- TriviaQA (Joshi et al. 2017): a realistic question-answering dataset extracted from Wikipedia and the Web. In this paper, we used only examples from the Web section. As with Fever, we also randomly sampled data from this dataset.

We consider the following task sequences in our experiment: (1) **Short sequence**: all permutations of tasks BoolQ, Movie, and SciFact; and (2) **Long sequence**: two permutations of all the five tasks, from the largest to the smallest tasks and vice versa.

For classification tasks (the first four datasets), we used EM, or exact match between texts, as the metric. This is because GPT2 is a generative model. However, because of

Table 2
Summary of datasets, their sizes, and the corresponding metrics. EM is an exact match between texts while nF1 represents normalized F1 score.

Dataset	# Train	# Test	Metric
BoolQ	6,363	2,817	EM
Fever	7,390	6,111	
Movie	1,600	200	
SciFact	405	188	
TriviaQA	3,005	1,207	nF1

Table 3
Each component of the QA structure of each dataset. * Note that, unlike other datasets in our experiments, Movie is a single-text classification task; therefore, the question is manually added and reused across the task. ** We prepend the task name to the answers to encourage the model to learn the difference between the two tasks.

Dataset	Context	Question	Answer
BoolQ	Passage	Question	True/False
Fever	Doc.	Claim	Supports/Refutes**
Movie	Passage	Question*	Positive/Negative
SciFact	Doc.	Claim	Supports/Refutes**
TriviaQA	Doc.	Question	Answer

the nature of text classification, the percentage of exact matches can also be seen as the accuracy of the model. For the TriviaQA dataset, however, we used the normalized[2] F1 score. Because the scores for all metrics lie between 0 and 1, we can simply average the scores across different metrics.

To quantify the amount of catastrophic forgetting or lack thereof, we calculated the normalized area under the accuracy curve (NAAC), as inspired by Chaudhry et al. (2019a). Formally, for every epoch during training, we evaluated the trained model and obtain an average accuracy Acc_i over all previously learned tasks:

$$Acc_i = \frac{1}{\tau} \sum_{t=1}^{\tau} a_{i,t} \tag{3}$$

where $a_{i,t}$ is the test accuracy of the model on task t at the current epoch i, and τ is the total number of tasks the model has learned at the time of the evaluation.

NAAC can be defined as:

$$NAAC = \int_{1}^{n} Acc_i \, di \tag{4}$$

where n is the total number of epochs the model is trained. The NAAC score lies between 0 and 1. It will be higher for the method that is more effective at preventing CF. Note that this score is order-dependent. In other words, the scores cannot be compared across different task orders.

4.2 Implementation Details

In all of our experiments, the best LAMOL configuration according to Sun, Ho, and Lee (2020) was used. In particular, the sampling ratio γ is set to 0.2. Also, task-specific tokens are used instead of the `[GEN]` token to generate pseudo-samples of a specific task.

We utilized the small GPT2 model (Radford et al. 2019) as the language model for all methods except ANML-ER, for which we used BERT-base (Devlin et al. 2019) as done in their original paper. We applied greedy decoding during inference. For LLKD, the

2 Note that the normalization refers to text normalization, i.e., lower-casing, article removal, when comparing the model output and the ground truth answer from the test set.

Table 4
Hyperparameters used in our experiments.

Hyperparameter	Value
Training hyperparameters	
Training epochs per task	5
Optimizer	Adam
Adam epsilon	1.0×10^{-4}
Weight decay	0.01
Max gradient norm	1.0
Learning rate schedule	warmup linear
Warmup ratio	0.005
LM-specific hyperparameters	
Learning rate	6.25×10^{-5}
Top-k sampling	$k = 20$
Adapters-specific hyperparameters	
Learning rate	1×10^{-4}
Reduction factor	16
Non-linearity	ReLu
LLKD-specific hyperparameters	
KD Temperature	2

distillation strategy used is the soft sequence-level strategy. Meanwhile, for ANML-ER, we followed the default hyperparameters introduced in Holla et al. (2020), with two exceptions. First, we modified the replay interval to 140 samples as opposed to 9,600 in the original experiments of Holla et al. (2020). Second, the experience replay rate was changed from 1% to 20%. We believe that the modified values make the comparison fairer due to the drastic difference in data sizes and to compensate for the disadvantages of meta learning in small datasets.[3] The adapter module parameters are also kept at the default values as proposed by Pfeiffer et al. (2020) with a reduction factor of 16. The hyperparameters of both the LM and the adapters are listed in Table 4.

We used `adapter-transformers`[4] for the implementation of the GPT2 LM and adapters. For LLKD[5] (Chuang, Su, and Chen 2020) and ANML-ER[6] (Holla et al. 2020), we used their publicly available implementations. For all task sequences, we ran all methods three times with different random seeds and averaged the results. All of the experiments were conducted on an NVIDIA DGX station.

We report the performance of our proposed solutions and compare them to the baseline LAMOL and two external baselines: LLKD and ANML-ER. We also report the results of $\text{LAMOL}_{\text{real}}$, which uses some real examples from previous tasks to train the model and, therefore, guarantees the quality of examples used. The number of real examples used by $\text{LAMOL}_{\text{real}}$ equals the number of pseudo samples generated and used by LAMOL. Additionally, we compared with the multitask learning set-up where the GPT2 model was trained on real examples from all tasks at the same time. This is usually considered the upper bound of LL methods.

3 The replay interval of 140 samples was chosen to keep the ratio of total number of training examples to replay interval consistent with the original implementation. Meanwhile, the experience replay rate was set equal to the LAMOL sampling ratio of 20%.
4 `https://github.com/Adapter-Hub/adapter-transformers`.
5 `https://github.com/voidism/L2KD`.
6 `https://github.com/Nithin-Holla/MetaLifelongLanguage`.

Table 5
Accuracy of different methods, averaged over three random seeds. The scores are evaluated on the models at the last epoch of the last task. Each column represents the order of tasks on which the methods were trained. B, M, and S refer to BoolQ, Movie Reviews, and SciFact, respectively. The Average and Std. columns refer to the average and standard deviation of the accuracy scores for each row of the methods, respectively. R and T refer to ReGen and temporal ensembling, respectively.

Methods	BMS	BSM	MBS	MSB	SBM	SMB	Average	Std.
Non-LL method								
Sequential	47.85	36.91	28.20	19.51	18.91	31.74	30.52	10.99
Baselines. Section 2.2–2.3.2 & Section 5.1								
LAMOL	64.53	35.48	66.79	60.76	52.02	54.40	55.67	11.41
LAMOL$_{\text{all}}$	62.22	62.06	61.42	52.93	65.32	63.35	61.22	4.29
LLKD	53.41	32.01	40.74	18.97	40.48	40.12	37.62	11.43
ANML-ER	55.64	42.43	42.02	69.00	59.13	59.58	54.63	10.58
Proposed Framework. Section 3.1–3.2								
Double LM	68.94	69.00	71.78	69.20	71.44	69.37	69.96	1.29
LM+Adapter	69.68	67.88	69.73	69.19	69.00	71.23	69.45	1.10
With Additional Pseudo Sample Enhancement. Section 3.4								
LM+Adapter+R	70.22	69.02	69.16	67.51	71.48	71.43	69.80	1.54
LM+Adapter+T	69.73	71.75	70.16	69.60	71.02	71.83	70.68	**0.99**
LM+Adapter+RT	71.28	70.53	70.30	70.09	71.45	73.62	**71.21**	1.30
LAMOL$_{\text{real}}$	69.07	71.97	70.84	72.31	74.13	73.32	71.94	1.80
Multitask	75.52							

5. Results and Discussion

This section reports and discusses the experimental results. Specifically, Sections 5.1 and 5.2 report the LL performance in terms of average accuracy at the last epoch and NAAC score, respectively. The former gives a general idea of the effectiveness of a method in learning in the LL scenario while the latter mainly focuses on quantifying the amount of CF over the course of training. Section 5.3 details the runtime of different methods in the experiments. Section 5.4 shows the result of our pseudo sample analysis. This is then followed by a comparative study of different variations of our proposed framework in Section 5.5 and an additional discussion on the effect of input length in Section 5.6.

5.1 LL Performance

In this section, we report the average accuracy of our proposed methods and the baselines on short and long sequence.

5.1.1 Short Sequence. We trained all methods on six permutations of three tasks: BoolQ (B), Movie Reviews (M), and Scifact (S). The results are shown in Table 5.[7]

In task permutations BMS and MBS, LAMOL was able to generate sufficient correctly formatted pseudo samples and hence was able to prevent total knowledge

7 As noted in Table 4, we trained each task for five epochs due to our resource limitation, as opposed to nine epochs as in Sun, Ho, and Lee (2020). Still, we present the results of some selected methods from Table 5, trained for nine epochs for each task, in Appendix A to enable comparison across papers.

loss. Nevertheless, in the other permutations, we found that the majority of pseudo samples generated from LAMOL do not have the correct format. As a result, LAMOL showed almost complete forgetting of previous tasks, especially in the order BSM, where LAMOL scored less than 1% correctness in both BoolQ and SciFact tasks at the final epoch.

To highlight the problem of pseudo samples having the wrong format, we try mitigating the problem of LAMOL by implementing an algorithm that heuristically assigns an answer to all pseudo samples, regardless of the questions. In every pseudo sample, the algorithm looks for the last `[ANS]` token of the generated pseudo sample and replaces all tokens behind the `[ANS]` with a valid answer according to the task-specific token. The answer is chosen according to the next-token probability of the first token (after `[ANS]`) of all valid answers. When there is no `[ANS]` token, we added it at the end of the pseudo sample and a random valid answer is then added. Finally, we bypassed the format control of LAMOL to guarantee that all generated pseudo samples were used. The result is shown in $\text{LAMOL}_{\text{all}}$, where we were able to gain an average of 5.55% improvement from LAMOL. Unsurprisingly, in task orders where LAMOL was already able to generate decent pseudo samples (i.e., BMS and MBS), $\text{LAMOL}_{\text{all}}$ introduced noise that destructively interfered with learned knowledge.

Considering other baselines, we found that LLKD performed significantly worse than LAMOL on all task orders. It achieved only 37.62% average accuracy, 18.05% lower than LAMOL (55.67%). As with LAMOL, the GPT2 model failed to properly learn the structure of the training samples. Thus, it failed to prevent CF due to insufficient usable pseudo samples. Because LLKD also utilizes GPT2 as teacher models, it similarly suffered from the low-quality pseudo sample problem, albeit worse since the student model was also required to learn from these teacher models.

In contrast, another baseline, ANML-ER, performed comparatively to LAMOL on average, achieving only 54.63% average accuracy. Nevertheless, we found that a large part of ANML-ER's final performance can be attributed to the first task while it struggled to converge on the final task. In contrast, in most task orders, LAMOL was unable to prevent complete CF of the first task; however, it performed well on later tasks. This is illustrated in Figure 4. A possible explanation for this is that the high replay rate of real examples in ANML-ER resulted in a setting similar to a multitask set-up. This gave the model enough time to converge on the prior tasks. For later tasks, since the neuromodulatory (NM) network is trained using meta learning, it is possible that the model did not converge on the meta objective when trained on small datasets such as those in this experiment. This resulted in a significantly reduced learning capability; therefore, ANML-ER was unable to converge on later tasks in a timely fashion. For instance, in the task order MBS, it achieved 0% accuracy on the final task (SciFact task) on all three random seeds.

With the ability to generate high-quality pseudo samples, our Double LM was able to improve upon LAMOL by 14.29% average accuracy while also having only 1.29% standard deviation. As expected, LM+Adapter was able to perform on par with Double LM on average, gaining 13.78% average accuracy over LAMOL and achieving only 1.10% standard deviation. This suggests that the adapter modules successfully mimicked the function of the additional GPT2 of Double LM. Additionally, according to Figure 4, LM+Adapter successfully retained most of the learned knowledge while also not struggling with learning later tasks, as opposed to LAMOL and ANML-ER.

Both of our variants were competitive with $\text{LAMOL}_{\text{real}}$ (using real examples instead of pseudo samples) in the orders BMS and MBS but slightly underperformed in the other orders.

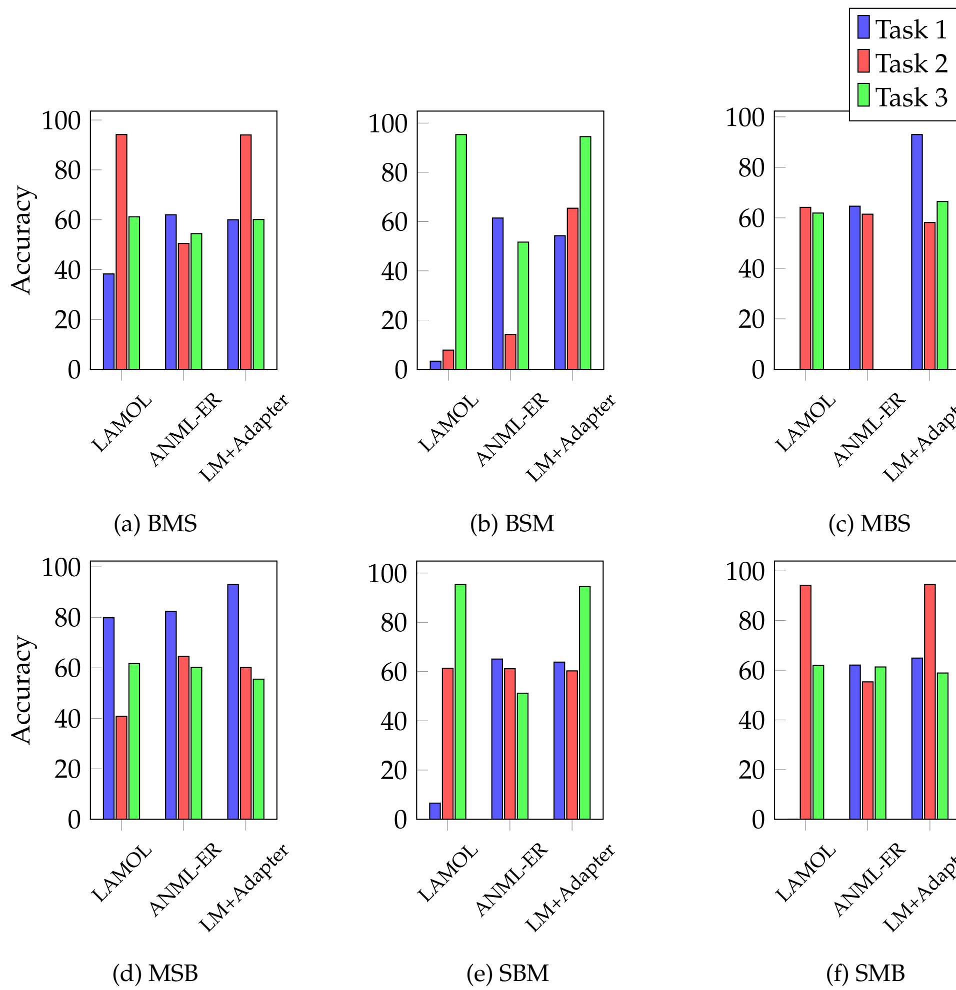


Figure 4
The distribution of test scores of LAMOL, ANML-ER, and LM+Adapter at the last epoch.

Concerning the strategies proposed in Section 3.4, applying ReGen (R) to our LM+Adapter (i.e., LM+Adapter+R) was able to provide an improvement, although statistically insignificant, of 0.45% in terms of average accuracy. Meanwhile, by incorporating Temporal Ensembling (T) into our LM+Adapter, we were able to further increase the performance of our framework by 1.13% (LM+Adapter+T) even though we did not apply additional data augmentation as proposed by Laine and Aila (2017). Combining these two strategies (LM+Adapter+RT) improves the performance of our LM+Adapter with statistical significance (p-value of 0.004) by 1.76%, being even closer to $\text{LAMOL}_{\text{real}}$ with only a 0.73% difference in accuracy.

5.1.2 Long Sequence. Besides, we conducted an experiment on all five tasks sequentially to further demonstrate our framework's effectiveness in preventing CF. Due to the limited computational resources, we only explored two orders: from the largest to the smallest tasks (FBTMS) and vice versa (SMTBF). Note that ANML-ER cannot handle a

Table 6
Performance of LLL models on five tasks, averaged over three random seeds.

Methods	FBTMS	SMTBF	Average
LAMOL	57.01	44.32	50.67
LLKD	42.73	47.04	44.89
LM+Adapter	65.51	62.18	63.85
LM+Adapter+RT	66.03	67.74	**66.88**
$\text{LAMOL}_{\text{real}}$	70.95	71.83	71.39
Multitask		68.89	

mixture of classification tasks and question-answering tasks. Therefore, it is excluded from this part of the experiment.

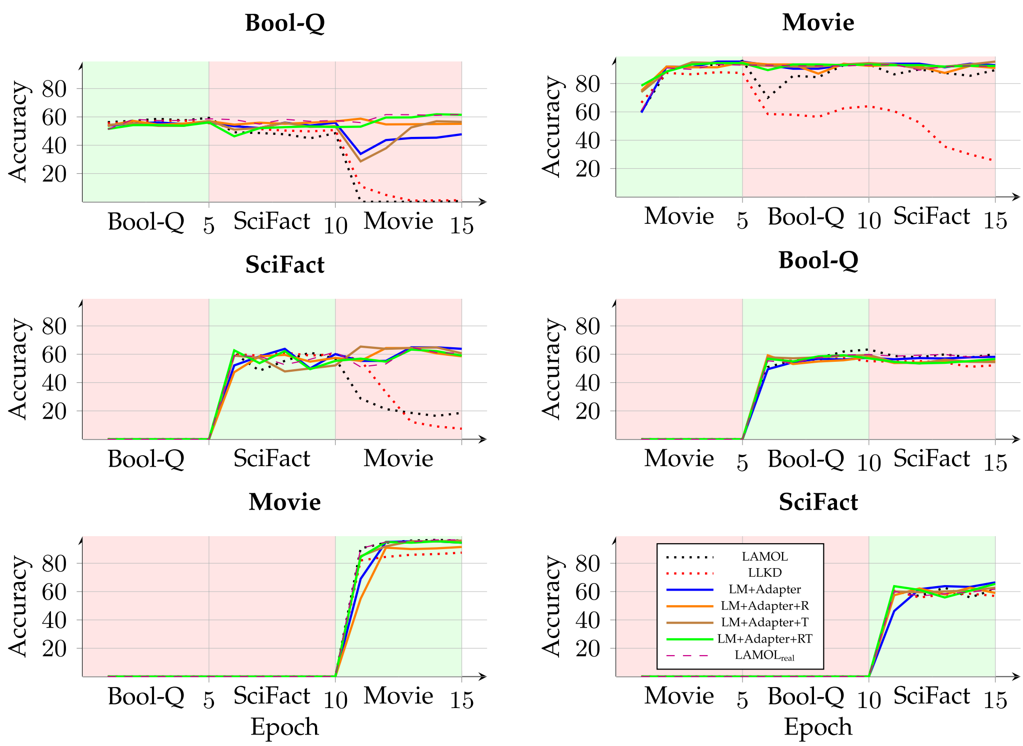
As shown in Table 6, our framework greatly outperformed LAMOL in both orders. Even though LAMOL was able to prevent catastrophic forgetting to an extent, the superior quality of pseudo samples generated by our framework enabled the model to retain significantly more knowledge and gain 13.18% average score. The combined pseudo sample enhancement strategy (LM+Adapter+RT) also generalizes to a longer sequence of tasks where we gained an additional 3.03% average score.

5.2 Quantifying Catastrophic Forgetting

In this section, we report the NAAC scores, introduced in Section 4.1, of all methods in Table 5 except for ANML-ER, since it does not make use of task descriptors; therefore, there is no indication as to when each task ends. The NAAC scores are reported in Table 7.

Table 7
Normalized Area Under Accuracy Curve (NAAC) score of different methods, averaged over three random seeds.

Methods	BMS	BSM	MBS	MSB	SBM	SMB	Average
Non-LL method							
Sequential	44.99	41.33	44.71	52.07	34.55	38.92	42.76
Baselines. Section 2.2 & Section 5.1							
LAMOL	60.50	45.33	70.50	67.12	44.90	49.80	56.36
$\text{LAMOL}_{\text{all}}$	59.71	53.07	69.52	62.89	51.29	58.49	59.16
LLKD	52.28	46.39	55.06	51.39	39.63	44.23	48.16
Proposed Framework. Section 3.1–3.2							
Double LM	62.04	54.42	73.33	72.52	55.64	63.43	63.56
LM+Adapter	60.61	53.95	72.12	69.97	57.46	62.89	62.83
With Additional Pseudo Sample Enhancement. Section 3.4							
LM+Adapter+R	61.77	55.55	72.37	70.19	56.29	61.84	63.00
LM+Adapter+T	61.71	55.45	72.50	68.94	55.76	62.29	62.78
LM+Adapter+RT	61.81	55.50	73.09	70.00	56.29	63.57	63.38
$\text{LAMOL}_{\text{real}}$	62.82	57.09	72.78	71.92	56.78	64.39	64.30



(a) Learning curves of task order BSM.

(b) Learning curves of task order MBS.

Figure 5
Learning curves of task orders BSM and MBS. The graphs show accuracy at each epoch for each task. Green background refers to the epochs on which the model is first introduced with a particular task. In this figure, for example, the model is trained on Bool-Q and evaluated on all the three tasks during epoch 1-5.

Additionally, to visualize the forgetting process, the learning curves of all methods in Table 7 are illustrated in Figure 5. Each plot shows the score of its corresponding task as the training progresses, with the first task in the order at the top and the last at the bottom. Two task orders are selected to show here: the one where the effect of CF can be seen most clearly (BSM) and the one where LAMOL successfully maintained knowledge throughout training (MBS).

5.2.1 Short Sequence. According to Table 7, without any CF prevention measure, sequential fine-tuning achieved the NAAC score of 42.76%. Equipped with pseudo sample replay, LAMOL improved the NAAC over sequential fine-tuning by 13.6%, showing better knowledge retention ability. Even though from Table 5, in task order BSM, LAMOL performed comparably with sequential fine-tuning in terms of final average accuracy, LAMOL managed to achieve a 4% higher NAAC score, indicating that LAMOL was able to prevent CF to some extent but eventually suffered from CF as the training progressed. This can also be seen in the graphs (Figure 5(a)) where Bool-Q and SciFact performance dropped after the Movie task was introduced. In task order MBS, LAMOL was able to prevent CF and achieved a good NAAC score of 70.5%. Nevertheless, there were still more signs of CF in the learning curve of the Movie task, where there are dips when new tasks are introduced. We also see this trend with other task orders as well, meaning that LAMOL still struggles at preventing CF. In the same fashion as average accuracy, LLKD achieved a low average NAAC score of only 48.16%, 8.2% lower than LAMOL. Both graphs show that LLKD failed to prevent CF of the first task.

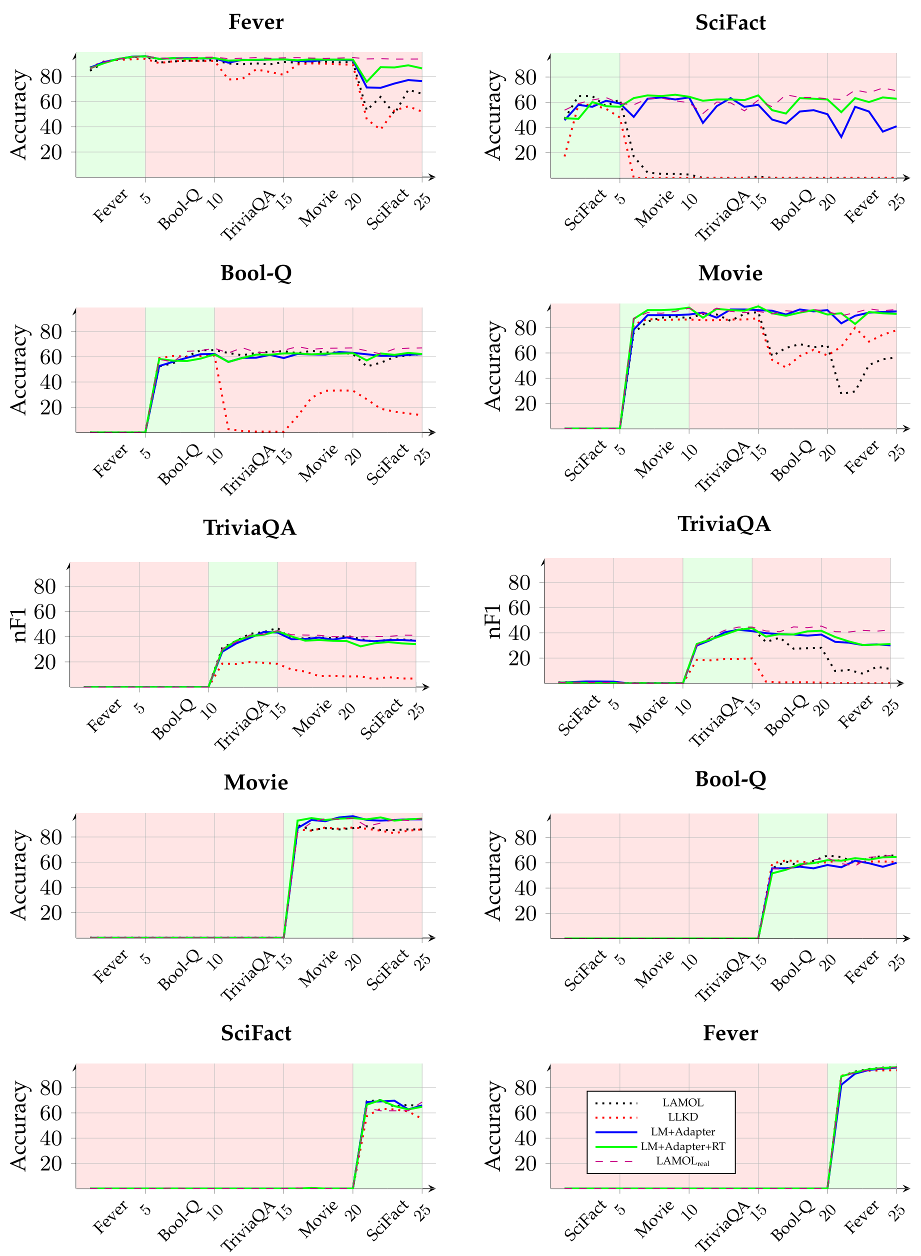
On the other hand, our framework is significantly more effective at preventing CF, rivaling $\text{LAMOL}_{\text{real}}$ in every task order. Both Double LM and LM+Adapter outperformed LAMOL significantly with p-values of 0.014 and 0.039, respectively. Applying our pseudo sample enhancement strategies separately to LM+Adapter did not improve the NAAC score. Nevertheless, with both strategies, LM+Adapter+RT gained a statistically significant NAAC improvement of 0.55% (with a p-value of 0.019). It can be inferred that high-quality pseudo samples should simultaneously possess both characteristics (i.e., informativeness and answer correctness) as induced by ReGen and temporal ensembling strategies. In task order BSM, methods that utilized our pseudo sample enhancement strategies exhibit superior ability to retain the knowledge of the Bool-Q task, resulting in an approximately 10% increase in accuracy over vanilla LM+adapter at the final epoch of training. Interestingly, LM+Adapter+RT also exhibits signs of backward knowledge transfer, where the performance of the prior tasks is improved as training progressed. Specifically, the performance of the Bool-Q task is actually higher at the last epoch than after it has just finished training on Bool-Q. For task order MBS, the learning curves of our methods show better stability than LAMOL where no large drops in accuracy can be seen.

5.2.2 Long Sequence. Considering sequences of five tasks, we also calculated the NAAC scores of the methods in Table 6 and compiled them into Table 8. Unsurprisingly, there is less CF in order FBTMS because the Fever task is relatively easier due to its data characteristics (i.e., simpler language used and slightly shorter texts). Therefore, higher quality pseudo samples were generated. LAMOL was able to achieve a strong score of 70.12%. However, as illustrated in Figure 6a, our framework is more effective in preventing CF, especially in the first task. LM+adapter and LM+adapter+RT gained an improvement of 0.89% and 1.32% NAAC score over LAMOL.

Regarding the order SMBTF, as shown in Figure 6b, both LAMOL and LLKD failed to prevent CF and achieved only 46.58% and 44.52% NAAC scores, respectively. Even though LLKD achieved a slightly better average accuracy score (Table 6), it achieved a lower NAAC score compared to LAMOL. This is because LAMOL was able to retain more knowledge during training, albeit not until the final epoch. In contrast, our proposed methods were able to prevent most of the CF presented in the baselines and obtained a score of 60.90% and 63.63%.

Table 8
Normalized Area Under Accuracy Curve (NAAC) score of LLL models on five tasks, averaged over three random seeds.

Methods	FBTMS	SMTBF	Average
LAMOL	70.12	46.58	58.35
LLKD	57.25	44.52	50.48
LM+Adapter	71.01	60.90	65.95
LM+Adapter+RT	71.44	63.63	**67.53**
$\text{LAMOL}_{\text{real}}$	72.72	64.26	68.49



(a) Learning curves of task orders FBTMS.

(b) Learning curves of task orders SMTBF.

Figure 6
Learning curves of task orders FBTMS and SMTBF. The graphs show the performance of all methods at each epoch for each task.

5.3 Efficiency

We detail the runtime and parameter counts of each method in Table 9. The runtime is calculated by averaging the runtime of all task permutations from Table 5. LLKD

Table 9
Runtime and parameter count of different LLL methods from Table 5. The runtime is an average of all task permutations across three random seeds.

Methods	Runtime	#Parameters
LAMOL	90.7 min	124.44M
LLKD	152.1 min	124.44M
ANML-ER	12.3 min	220.15M
Double LM	178.2 min	248.88M
LM+Adapter	127.5 min	125.33M
Re-generate	+13.1 min	–
Temporal Ensem.	+3.2 min	–

requires an additional forward pass through the teacher models for each example. Therefore, it introduced an extra runtime of approximately 62 minutes on top of LAMOL. Despite having almost 2 times more parameters over LAMOL, ANML-ER restricts itself to making only a single pass through training data. Together with the fact that the maximum sequence length that the BERT model supports is 512 tokens, as opposed to 1,024 tokens of GPT2, it took only 12.3 minutes to run each task order. In spite of the massive performance improvement, Double LM took almost 2 times longer than vanilla LAMOL and doubled the storage requirement. LM+Adapter was able to retain most of the improvements while taking only approximately 1.4 times longer. It also requires a negligible amount of additional storage. We also report the runtime of the pseudo sample enhancement strategies. Note that temporal ensembling only temporarily stores the extra model, which is discarded after the generation process; therefore, no additional parameters are introduced.

5.4 Results of Pseudo Sample Analysis

The analysis showed that pseudo samples generated by LAMOL mostly did not conform to the QA format and thus were not used in training. This is shown in Table 10a. As a consequence, LAMOL was unable to effectively prevent catastrophic forgetting. This problem is even more prevalent in LLKD. From Table 10b, it can be seen that almost all pseudo samples are incorrectly formatted. This is likely a reason why its LL performance is comparable to sequential fine-tuning. Our framework increased the success rate of pseudo sample generation (Table 10c). This also resulted in a significant increase in the final LL performance. Note that it is still possible for our framework to produce malformed pseudo samples if the LM outputs special tokens inappropriately. However, the numbers are much less than LAMOL, at least approximately seven times smaller. There were also still some undesirable pseudo samples generated by our framework. Here, we attempt to identify the cause and anticipate the effect of each aspect, providing insights for future improvements.

Uninformative Pseudo Samples. From Table 10c, in task orders where SciFact is not the last task, the number of uninformative pseudo samples dominates other aspects. This is because the extremely complicated language used in the task examples of SciFact greatly differs from the general domain on which GPT2 was pretrained. Thus, without enough training examples, the LM fails to generate coherent examples. We hypothesize that pseudo samples of this nature may not necessarily be destructive to the model's

Table 10
Results of the pseudo sample analysis. The numbers indicate the number of pseudo samples corresponding to each characteristic, averaged over three seeds.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	41.00	320.00	72.33	1,239.00	310.00	703.00
Uninformative	0.00	0.00	0.00	0.67	1.67	1.00
Wrong task	2.00	0.00	0.33	7.33	0.00	10.67
Wrong Answer	8.00	0.00	2.33	5.33	3.00	72.33
Correct answer	30.00	0.00	6.00	19.67	5.33	485.00
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(a) Pseudo sample analysis of LAMOL.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	75.00	320.00	80.00	1,271.67	314.00	1,220.33
Uninformative	1.33	0.00	0.00	0.33	6.00	23.67
Wrong task	0.33	0.00	0.00	0.00	0.00	2.67
Wrong Answer	0.67	0.00	0.00	0.00	0.00	4.67
Correct answer	2.67	0.00	0.00	0.00	0.00	20.67
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(b) Pseudo sample analysis of LLKD.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	3.33	8.67	2.67	14.67	36.33	102.00
Uninformative	15.00	173.33	19.00	409.00	180.00	347.67
Wrong task	22.33	47.00	29.00	107.00	58.67	153.67
Wrong Answer	8.00	37.67	5.33	162.67	15.00	122.67
Correct answer	32.33	53.33	25.00	578.67	30.00	546.00
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(c) Pseudo sample analysis of LM+Adapter.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	0.33	1.00	1.67	11.00	26.00	23.00
Uninformative	0.00	4.00	0.00	0.00	5.00	0.33
Wrong task	23.67	105.00	31.00	194.67	93.67	128.00
Wrong Answer	10.67	91.33	12.67	194.67	81.67	186.00
Correct answer	46.33	118.67	35.67	871.67	113.67	934.67
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(d) Pseudo sample analysis of LM+Adapter+R.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	1.00	6.67	2.67	3.67	31.67	101.33
Uninformative	14.67	174.33	17.67	329.00	161.67	352.33
Wrong task	18.33	36.00	29.00	62.67	60.67	90.00
Wrong Answer	7.00	35.33	7.00	177.33	28.00	107.00
Correct answer	40.00	67.33	24.67	699.33	38.00	621.33
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(e) Pseudo sample analysis of LM+Adapter+T.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	0.33	2.00	0.33	5.33	21.00	36.67
Uninformative	1.67	80.67	3.33	53.00	42.00	24.67
Wrong task	22.67	69.33	35.33	109.00	102.33	103.33
Wrong Answer	7.67	68.33	6.67	191.33	66.00	171.00
Correct answer	48.67	99.67	35.33	913.33	88.67	936.33
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(f) Pseudo sample analysis of LM+Adapter+RT.

knowledge; however, the generation quota could still be better allocated for more informative pseudo samples. This hypothesis is supported by the minor improvements gained by using ReGen.

Wrong Task Pseudo Samples. As mentioned in Sun, Ho, and Lee (2020), generated pseudo samples sometimes do not correspond to the given task tokens. This is caused by the imbalanced amount of pseudo samples and samples from a new task. As a result, the model tends to generate more pseudo samples from newer tasks. From our pseudo sample analysis, the problem is more prevalent when the training set of the new task is larger than the one of the previous task. We have observed many wrong-task pseudo samples to actually be perfectly fine pseudo samples, that is, having correct answers and logically sound contexts and questions. Despite the decent quality of these pseudo samples, wrong-task pseudo samples worsen the data imbalance problem. Therefore, we believe that this kind of pseudo sample is less destructive to LL performance for shorter task sequences. The effect of this problem is more apparent in longer task sequences, where the knowledge of the first task is eventually lost, resulting in larger gaps of performance between $\text{LAMOL}_{\text{real}}$ and other methods (Table 6) when compared with shorter sequences (Table 5).

Wrong Answer Pseudo Samples. We believe that pseudo samples with wrong answers are the most destructive to the model's knowledge, relative to the previously mentioned issues. This effect is most clearly seen when we included temporal ensembling into our framework, where improving answer correctness of pseudo samples consistently improves the performance of our framework on every task permutation. Therefore, future work should focus on minimizing the number of pseudo samples of this nature.

5.5 Comparison with Other Variants

In this section, we compare our proposed framework to its possible variants. More specifically, we experimented with using different numbers of LMs in our framework. Additionally, we evaluated all combinations of the responsibilities of the two LMs in our framework to find the best combination.

Number of LMs. The proposed framework uses a second LM to exploit the structure of the training samples to generate high-quality pseudo samples by heuristically controlling the generation step. However, it is also possible to use only one or three language models for all or each part of a training sample (i.e., context, question, and answer), respectively.

Nevertheless, it is not entirely straightforward to control the pseudo sample generation process when using a single LM. Instead, we opted to control the generation process by setting the probabilities of special tokens that should not appear next to zero. For instance, the probabilities of `[ANS]` and `[EOS]` are zeroed out until a `[SEP]` token is generated. On the other hand, the heuristic can be naturally extended to the three-LM setting. The responsibility of each part of a training sample can be delegated to each LM. To this end, we implemented the three-LM setting using one LM + two Adapters to reduce training time.

The results in Table 11 show that using two LMs achieved the best average performance of 69.96%. Whereas the other two variations performed on par, achieving approximately 65% average accuracy. We attribute the superior performance of Double LM to the better trade-off of model capacity and exploitation of sample correlation. To explain,

Table 11
Accuracy of the proposed framework when applied to various numbers of LM. Note that, to increase training efficiency, Double LM and Triple LM were implemented using one LM + one Adapter and one LM + two Adapters, respectively.

Methods	BMS	BSM	MBS	MSB	SBM	SMB	Average	Std.
Single LM+constraint	66.44	65.80	68.94	66.88	59.24	66.60	65.65	3.31
Double LM	69.68	67.88	69.73	69.19	69.00	71.23	69.45	1.10
Triple LM	68.39	65.61	66.66	59.01	68.39	65.06	65.52	3.47

Table 12
Top: A pseudo sample generated by the single LM variant. This particular one has consistent context and answer; however, the answer was also generated in the context part (highlighted in red). **Middle:** A pseudo sample generated by the Double LM variant. This pseudo sample is consistent and its answer is properly placed. **Bottom**: A pseudo sample generated by the triple LM variant. The answer was positioned properly; however, it is incorrect.

Pseudo Sample Examples	
Single LM	. . . directed by JANE ANDERSON, also stars JOHN CL [UNK] POS. it's a story with a rich history. I'm a big fan of that term, but i'm not interested in the specifics of it. what you are looking at is an extraordinary tale filled with rich characters, great characterization, and a fascinating story about . . . [SEP] What is the sentiment of this review? [ANS] POS
Double LM	. . . and it's one that we have to take seriously. the movie is very, very good; it's an absolute gem. it's an absolute masterpiece: an intelligent, funny, well thought out tale of love and redemption. it's a story of redemption; it's about love and redemption and what it means to be gay in a society that is rife with discrimination . . . [SEP] What is the sentiment of this review? [ANS] POS
Triple LM	. . . i am really happy with this remake. it was a pretty good movie, but it wasn't really bad. although the characters were different, it was very well done. it doesn't deserve to be considered that bad. it is a good remake, but it doesn't deserve to be considered that bad. [SEP] What is the sentiment of this review? [ANS] NEG

we believe that having generated the context part of the sample can bias the model toward generating more correct answers. Evidence for this is that pseudo samples from the single LM variant sometimes contain an answer in the context part.[8] An example is illustrated in Table 12 (top). This could be because the single LM could associate, for instance, positive vocabularies (e.g., "very good movie") with the 'Positive' answer. Therefore, by generating context that is sufficiently polarized (i.e., not ambiguous), its corresponding answer can be generated more accurately. Despite this advantage, however, the single LM could not model long complicated sequences properly due to its limited model capacity. Even though the masking process tried to prevent such a situation explicitly, there were multiple instances where the model did not generate any special token, resulting in malformed pseudo samples. This leads to the highest number of "Wrong Format" pseudo samples in every order, as shown in Table 13a, compared to the other two variations in Tables 10c and 13b. In contrast, while the extra capacity of the three LM variation enables the generation of pseudo samples with the correct format, the detached generation process makes pseudo samples more prone to "Wrong Answer" errors as shown in Table 13b. One example is provided in Table 12 (bottom).

8 Note that in a good pseudo sample, an answer should *not* appear in the context.

Table 13
Additional pseudo sample analysis for single LM and triple LM.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	17.33	70.33	17.33	358.33	171.33	332.33
Uninformative	9.00	33.67	4.33	82.33	102.00	481.67
Wrong task	14.67	119.00	20.00	264.00	26.00	124.00
Wrong Answer	7.00	44.00	3.33	156.00	10.67	59.00
Correct answer	32.00	53.00	13.67	411.33	9.67	275.00
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(a) Pseudo sample analysis of Single LM.

Aspect	BMS	BSM	MBS	MSB	SBM	SMB
Wrong Format	0.67	0.33	1.00	32.33	4.33	22.00
Uninformative	0.33	1.67	3.00	27.00	1.00	40.67
Wrong task	13.33	88.67	16.00	73.33	43.67	49.33
Wrong Answer	18.00	104.33	17.00	442.00	118.33	343.00
Correct answer	47.67	125.00	43.00	697.33	152.67	817.00
Total Number	81.00	320.00	81.00	1,272.00	320.00	1,272.00

(b) Pseudo sample analysis of Triple LM.

This lowers the quality of pseudo samples, leading to worse average performance due to CF. Meanwhile, the proposed Double LM can generate more pseudo samples with both correct answers (like single LM) and correct format (like triple LM). We show an example in Table 12 (middle).

Responsibilities of LMs. Our proposed framework uses LM_1 to learn the context part and the QA task and uses LM_2 to learn the question part. This can be written as (c+qa/q). We also experimented with two other configurations of our proposed Double LM, namely:

- (c+q/qa): LM_1 learns the context and the question parts, whereas LM_2 learns the QA task only; and
- (c/q+qa): LM_1 learns only the context, while LM_2 learns the question part and the QA task.

We experimented on all permutations of the three tasks: BoolQ, Movie Reviews, and SciFact. The results are reported in Table 14. We found that the first variation (c+q/qa) performs comparably with our default configuration (c+qa/q) while having a higher standard deviation. The second variation (c/q+qa) was observed to produce mostly malformed pseudo samples. In particular, the LM was unable to distinguish between

Table 14
The performance of variations of our framework.

Variation	Average Acc.	Std.
Double LM (c+qa/q)	**69.96**	**1.29**
Double LM (c+q/qa)	69.43	2.64
Double LM (c/q+qa)	30.68	9.83

Table 15
The average token count of each dataset based on the GPT2 tokenizer. The token count is for a whole training sample, i.e., context, question, and answer.

Dataset	Average Token Count	Dataset	Average Token Count
Long Text		*Short Text*	
BoolQ	4,137	SQuADv1	165
Fever	436	WikiSQL	113
Movie	855	SST	31
SciFact	384	QA-SRL	38
TriviaQA	1,097	WOZ	25

Table 16
The performance of different methods on task sequence: SQuADv1 → WikiSQL → SST → QA-SRL → WOZ. Note that the ReGen strategy was not required because there were virtually no uninformative pseudo samples present in the experiments.

Methods	Average Acc.
LAMOL	73.24
LM+Adapter	72.89
LM+Adapter+T	**73.45**

the question generation process (step 2 of Figure 2) and the answer generation process (step 3 of Figure 2). Thus, most generated pseudo samples do not have answers but rather two questions. As a result, this variation was unable to prevent CF and achieved only 30.68% average accuracy, comparable to sequential fine-tuning.

5.6 Discussion on Input Length

Though the proposed framework showed impressive performance improvements over LAMOL in our experiments, it provides relatively small improvements on datasets with short texts such as those in Sun, Ho, and Lee (2020). As shown in Table 15, datasets from Sun, Ho, and Lee (2020) are up to two orders of magnitude shorter than those used in our experiments. It is mentioned in their paper that the quality of the generated text degrades as the training samples get longer. Consequently, in our experimental settings, LAMOL failed to generate decent pseudo samples. On the contrary, in a short text dataset, LAMOL is already able to produce high-quality pseudo samples. Hence, the Double LM framework would only introduce additional training time.

Table 16 shows the performance of LAMOL compared with our framework on one task sequence from the original LAMOL paper: SQuADv1 → WikiSQL → SST → QA-SRL → WOZ. We trained LAMOL and our methods for nine epochs on each task as in Sun, Ho, and Lee (2020).

6. Conclusion

We introduced Double LM, a lifelong learning framework that focuses on improving pseudo samples used to retain the knowledge of previously learned tasks. In our experiments, Double LM was able to significantly outperform LAMOL in terms of average accuracy and knowledge retained on every task sequence as well as other rehearsal baselines (LLKD and ANML-ER). We also successfully reduced the computational

Table A1
Accuracy of different methods when trained for nine epochs, averaged over three random seeds.

Methods	BMS	BSM	MBS	MSB	SBM	SMB	Average	Std.
LAMOL	69.70	57.37	72.59	71.32	60.95	51.48	64.09	8.60
LLKD	56.66	35.53	68.32	69.70	47.15	48.97	51.59	13.20
ANML-ER	55.64	42.43	42.02	69.00	59.13	59.58	54.63	10.58
LM+Adapter	71.64	66.31	73.02	72.75	69.93	70.17	70.64	2.47
LM+Adapter+RT	71.79	71.29	73.55	73.55	72.32	73.74	**72.54**	**1.0**5
$\text{LAMOL}_{\text{real}}$	71.80	71.28	72.67	72.67	74.79	73.43	73.20	1.24

requirements of Double LM by using the adapter modules. By applying temporal ensembling and simple pseudo sample re-generation to enhance pseudo samples, our framework was able to almost match the performance of $\text{LAMOL}_{\text{real}}$. Lastly, we provided an analysis of pseudo samples and their effects on LL performance. For future work, we aim to enhance the impact of our framework on tasks with shorter texts.

Appendix A: Exploring the Effect of Longer Training

In this section, we show the results of our proposed methods and the baselines trained for nine epochs in Table A1. For our proposed methods, we chose to train only LM+Adapter and LM+Adapter+RT to reduce the computational resources required. Note that ANML-ER only trains on one epoch; therefore, the scores are provided as reference only.

From the table, both LAMOL and LLKD gained substantial improvements of 8.42% and 13.97%, respectively, when compared to training on five epochs whereas our methods improved relatively slightly by 1.19% and 1.33%, respectively. However, there is still a large performance gap between the baselines and our methods. It can be inferred that our proposed methods converged much more quickly than the baselines because the task factorization of our framework reduces the complexity of the LM task. Convergence speed is one of the various desired properties of a true lifelong learner.

Overall, the same conclusions as in Section 5.1 can be drawn from the results in Table A1. Specifically, LM+Adapter still outperforms LAMOL with statistical significance (p-value of 0.047) and applying pseudo sample enhancement strategies further improve the performance of our framework significantly (p-value of 0.044).

References

Aljundi, Rahaf, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2017. Memory aware synapses: Learning what (not) to forget. *CoRR*, abs/1711.09601.

Ans, Bernard and Stéphane Rousset. 1997. Avoiding catastrophic forgetting by coupling two reverberating neural networks. *Comptes Rendus de l'Académie des Sciences - Series III - Sciences de la Vie*, 320:989–997. `https://doi.org/10.1016/S0764-4469(97)82472-9`

Beaulieu, Shawn, Lapo Frati, Thomas Miconi, Joel Lehman, Kenneth O. Stanley, Jeff Clune, and Nick Cheney. 2020. Learning to continually learn. In *ECAI 2020 - Including 10th Conference on Prestigious Applications of Artificial Intelligence (PAIS 2020)*, volume 325 of *Frontiers in Artificial Intelligence and Applications*, pages 992–1001, IOS Press.

Biesialska, Magdalena, Katarzyna Biesialska, and Marta R. Costa-jussà. 2020. Continual lifelong learning in natural language processing: A survey. In *Proceedings of* the 28th International Conference on Computational Linguistics, pages 6523–6541. https://doi.org/10.18653/v1/2020.coling-main.574

Chaudhry, Arslan, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2019a. Efficient lifelong learning with A-GEM. In *7th International Conference on Learning Representations, ICLR* 2019, OpenReview.net.

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

Chen, Zhiyuan and Bing Liu. 2016. *Lifelong Machine Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. https://doi.org/10.2200/S00737ED1V01Y201610AIM033

Chen, Zhiyuan, Nianzu Ma, and Bing Liu. 2018. Lifelong learning for sentiment classification. *CoRR*, abs/1801.02808.

Chuang, Yung-Sung, Shang-Yu Su, and Yun-Nung Chen. 2020. Lifelong language knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2914–2924. https://doi.org/10.18653/v1/2020.emnlp-main.233

Clark, Christopher, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171–4186. https://doi.org/10.18653/v1/N19-1423

Han, Xu, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6429–6440. https://doi.org/10.18653/v1/2020.acl-main.573

Hinton, Geoffrey E. 2012. *A Practical Guide to Training Restricted Boltzmann Machines*. Springer Berlin Heidelberg, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-35289-8_32

Hinton, Geoffrey E., Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531.

Holla, Nithin, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020. Meta-learning with sparse experience replay for lifelong language learning. *CoRR*, abs/2009.04891.

Hou, Saihui, 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, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 452–467, Springer. https://doi.org/10.1007/978-3-030-01219-9_27

Houlsby, Neil, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799.

Javed, Khurram and Martha White. 2019. Meta-learning representations for continual learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 1818–1828.

Joshi, Mandar, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611. https://doi.org/10.18653/v1/P17-1147

Kirkpatrick, James, Razvan Pascanu, Neil 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. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of*

Sciences, 114(13):3521–3526. https://doi.org/10.1073/pnas.1611835114

Kutuzov, Andrey, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: A survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397.

Laine, Samuli and Timo Aila. 2017. Temporal ensembling for semi-supervised learning. In *5th International Conference on Learning Representations, ICLR 2017, Conference Track Proceedings*, OpenReview.net.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Lopez-Paz, David and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, pages 6467–6476.

de Masson d'Autume, Cyprien, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *NeurIPS*, pages 13122–13131.

McCann, Bryan, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

McCloskey, Michael and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, Elsevier, pages 109–165. https://doi.org/10.1016/S0079-7421(08)60536-8

Merity, Stephen, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and optimizing LSTM language models. *CoRR*, abs/1708.02182.

Papamakarios, George, Eric T. Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2021. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22:57:1–57:64.

Parisi, German Ignacio, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71. https://doi.org/10.1016/j.neunet.2019.01.012

Pfeiffer, Jonas, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673. https://doi.org/10.18653/v1/2020.emnlp-main.617

Pomponi, Jary, Simone Scardapane, and Aurelio Uncini. 2020. Pseudo-rehearsal for continual learning with normalizing flows. *CoRR*, abs/2007.02443.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Ramalho, Tiago and Marta Garnelo. 2019. Adaptive posterior learning: Few-shot learning with a surprise-based memory module. In *7th International Conference on Learning Representations, ICLR* 2019, OpenReview.net.

Rusu, Andrei A., Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *CoRR*, abs/1606.04671.

Schlimmer, Jeffrey C. and Richard H. Granger. 1986. Incremental learning from noisy data. *Machine Learning*, 1(3):317–354. https://doi.org/10.1007/BF00116895

Shin, Hanul, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 2994–3003.

Silver, Daniel L. and Sazia Mahfuz. 2020. Generating accurate pseudo examples for continual learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2020*, pages 1035–1042. https://doi.org/10.1109/CVPRW50498.2020.00136

Solinas, M., S. Rousset, R. Cohendet, Y. Bourrier, M. Mainsant, A. Molnos, M. Reyboz, and M. Mermillod. 2021. Beneficial effect of combined replay for continual learning. In *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART*, pages 205–217. https://doi.org/10.5220/0010251202050217

Sprechmann, Pablo, Siddhant Jayakumar, Jack Rae, Alexander Pritzel, Adria Puigdomenech Badia, Benigno Uria, Oriol Vinyals, Demis Hassabis, Razvan Pascanu, and Charles Blundell. 2018. Memory-based parameter adaptation. In *International Conference on Learning Representations*.

Sun, Fan-Keng, Cheng-Hao Ho, and Hung-Yi Lee. 2020. LAMOL: Language modeling for lifelong language learning. In *8th International Conference on Learning Representations, ICLR 2020*, OpenReview.net.

Sun, Jingyuan, Shaonan Wang, Jiajun Zhang, and Chengqing Zong. 2020. Distill and replay for continual language learning. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3569–3579. `https://doi.org/10.18653/v1/2020.coling-main.318`

Thorne, James, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: A large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819. `https://doi.org/10.18653/v1/N18-1074`

Toneva, Mariya, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. 2019. An empirical study of example forgetting during deep neural network learning. In *7th International Conference on Learning Representations, ICLR 2019*, OpenReview.net.

Wadden, David, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 7534–7550. `https://doi.org/10.18653/v1/2020.emnlp-main.609`

Wang, Zirui, Sanket Vaibhav Mehta, Barnabás Póczos, and Jaime G. Carbonell. 2020. Efficient meta lifelong-learning with limited memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 535–548. `https://doi.org/10.18653/v1/2020.emnlp-main.39`

Wen, Yeming, Dustin Tran, and Jimmy Ba. 2020. BatchEnsemble: An alternative approach to efficient ensemble and lifelong learning. In *8th International Conference on Learning Representations, ICLR 2020*, OpenReview.net.

Zaidan, Omar F., Jason Eisner, and Christine Piatko. 2008. Machine learning with annotator rationales to reduce annotation cost. In *Proceedings of the NIPS*2008 Workshop on Cost Sensitive Learning*, pages 260–267.