

A Rationale-based Lifelong Learning Framework with Pseudo-sample Replay
Enhancement

Mr. Kasidis Kanwatchara

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Masters Program in Computer Engineering

Department of Computer Engineering

Faculty of Engineering

Chulalongkorn University

Academic Year 2021


Copyright of Chulalongkorn University

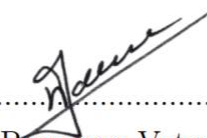
Thesis Title A Rationale-based Lifelong Learning Framework with
 Pseudo-sample Replay Enhancement
By Mr. Kasidis Kanwatchara
Field of Study Computer Engineering
Thesis Advisor Asst. Prof. Peerapon Vateekul, Ph.D.
Thesis Co-advisor Prof. Boonserm Kijirikul, Ph.D.


Accepted by the Faculty of Engineering, Chulalongkorn University in Partial
Fulfillment of the Requirements for the Masters Degree


..... Dean of the Faculty of Engineering
(Prof. Supot Teachavorasinskun, D.Eng.)

THESIS COMMITTEE


..... Chairman
(Asst. Prof. Sukree Sinthupinyo, Ph.D.)


..... Thesis Advisor
(Asst. Prof. Peerapon Vateekul, Ph.D.)


..... Thesis Co-advisor
(Prof. Boonserm Kijirikul, Ph.D.)


..... Member
(Ekapol Chuangsuwanich, Ph.D.)


..... Member
(Thanapat Kangkachit, Ph.D.)


6272009021: MAJOR COMPUTER ENGINEERING

KEYWORDS: NATURAL LANGUAGE PROCESSING / DEEP LEARNING /
LIFELONG LEARNING / CATASTROPHIC FORGETTING / LANGUAGE MODEL
/ TEXT GENERATION

KASIDIS KANWATCHARA : A RATIONALE-BASED LIFELONG LEARN-
ING FRAMEWORK WITH PSEUDO-SAMPLE REPLAY ENHANCE-
MENT ADVISOR : PEERAPON VATEEKUL, PH.D., THESIS COAD-
VISOR : BOONSERM KIJSIRIKUL, PH.D.,

Lifelong learning (LL) is a machine learning paradigm in which a learner is sequentially trained on a stream of new tasks while preventing learned knowledge from being forgotten. 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 on a specific part of input, to logically produce high-quality pseudo samples. Second, we reduce the number of parameters used by applying adapter modules to enhance training efficiency. Third, we further improve the overall quality of pseudo samples by exploiting the rational structure of the input 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.

Department: Computer Engineering
Field of Study: Computer Engineering
Academic Year: 2021

Student's Signature..... กษิดิศ วัฒนวงศ์.....
Advisor's Signature..... .....

6272009021: MAJOR COMPUTER ENGINEERING

KEYWORDS: NATURAL LANGUAGE PROCESSING / DEEP LEARNING /
LIFELONG LEARNING / CATASTROPHIC FORGETTING / LANGUAGE MODEL
/ TEXT GENERATION

KASIDIS KANWATCHARA : A RATIONALE-BASED LIFELONG LEARN-
ING FRAMEWORK WITH PSEUDO-SAMPLE REPLAY ENHANCE-
MENT ADVISOR : PEERAPON VATEEKUL, PH.D., THESIS COAD-
VISOR : BOONSERM KIJSIRIKUL, PH.D.,

การเรียนรู้ตลอดชีวิตคือวิธีการหนึ่งในการเรียนรู้ของเครื่องซึ่งผู้เรียนจะทำการเรียนอย่างค่อยเป็นค่อยไปบน
งานที่เข้ามาเรื่อย ๆ ในขณะที่ป้องกันการลืมของความรู้ที่เรียนมา เพื่อบรรลุการเรียนรู้ตลอดชีวิต วิธีการทบทวน
ตัวอย่างที่เพิ่มใช้งานตัวอย่างที่ถูกสร้างจากโมเดลภาษาเพื่อทำการทบทวนความรู้ที่เรียนมาก่อนหน้า แต่ถ้าหาก
ไม่มีการควบคุม วิธีดังกล่าวอาจไม่สามารถป้องกันการลืมบนงานที่มีความซับซ้อนและความยาวมาก เนื่องจาก
ตัวอย่างที่ถูกสร้างขึ้นมาจะมีคุณภาพต่ำ เพื่อแก้ปัญหาดังกล่าว งานวิจัยนี้จึงเสนอผลงานสามอย่างด้วยกัน หนึ่งใน
งานวิจัยนี้ใช้โมเดลภาษาสองตัว ซึ่งแต่ละตัวจะชำนาญในแต่ละส่วนของอินพุต เพื่อสร้างตัวอย่างที่เพิ่มที่มีคุณ-
ภาพอย่างสมเหตุสมผล สอง งานวิจัยนี้ใช้อะแดปเตอร์โมดูล (Adapter module) เพื่อลดปริมาณพารามิเตอร์
และเพิ่มความเร็วในการฝึกฝน สาม งานวิจัยนี้เพิ่มคุณภาพของตัวอย่างที่เพิ่มโดยการใช้โครงสร้างของอินพุต
อย่างเป็นเหตุเป็นผล โดยใช้วิธีการประกอบข้ามการเวลาและการสร้างตัวอย่างที่เพิ่มซ้ำ ผลลัพธ์ของการทดลอง
แสดงให้เห็นว่าโครงที่งานวิจัยนี้เสนอ สามารถบรรลุประสิทธิภาพที่สูงกว่าพื้นฐานอย่างมากบนหลายลำดับ
งาน นอกจากนี้ งานวิจัยนี้ยังเผยข้อมูลเชิงลึกที่มีประโยชน์ต่อการสร้างวิธีการทบทวนตัวอย่างที่เพิ่มที่มีประ-
สิทธิภาพสูงยิ่งกว่าในอนาคตได้

ภาควิชา วิศวกรรมคอมพิวเตอร์
สาขาวิชา วิศวกรรมคอมพิวเตอร์
ปีการศึกษา 2564

ลายมือชื่อนิสิต... กษิต์ ภาณุวัฒน์
ลายมือชื่อ อ.ที่ปรึกษาหลัก.....

Acknowledgements

This thesis was a journey. Although it was longer than I had anticipated, it was full of new experiences and important lessons. Many people were crucial for the success of this research.

First of all, I would like to express my deepest gratitude to my advisor, Asst. Prof. Peerapon Vateekul for his continual guidance and motivation. This thesis would not have been possible without his support.

I would like to show my sincere appreciation to Prof. Boonserm Kijirikul for his advice for improving this research.

To the rest of thesis committee: Asst. Prof. Sukree Sinthupinyo, Aj. Ekapol Chuangsuwanich, Ph.D., and Aj. Thanapat Kangkachit, Ph.D., I would like to thank them for challenging my work and providing valuable comments and suggestions.

Last but not least, I would also like to give special thanks to “P Tui” Piyawat Lertvittayakumjorn, Ph.D. for all the helpful pointers on all aspects of the thesis. Only with his assistance did I manage to complete this work with such grace.

Contents

Abstract	iii
Acknowledgements	v
Contents	vi
List of Figures	viii
List of Tables	x
1 Introduction	1
1.1 Objectives	3
1.2 Contributions	4
1.3 Scope	4
2 Background & Related Work	5
2.1 Attention	5
2.2 Transformers	7
2.3 Generative Pre-trained Transformer (OpenAI GPT)	7
2.4 OpenAI GPT-2	8
2.5 Catastrophic Forgetting	8
2.6 Lifelong Learning	9
2.6.1 Approaches to lifelong learning	10
2.7 LAMOL	15
2.8 Adapter Modules	17
3 Concept and Research Methodology	18
3.1 Double LM	18
3.1.1 Adapter	20
3.2 Pseudo Sample Analysis	21
3.3 Further Improving Pseudo Sample Quality	22

4	Experimental Setting	24
4.1	Datasets	24
4.2	Implementation Details	25
4.3	Metrics	26
5	Results	28
5.1	LL Performance	28
5.1.1	Three-task Sequence	28
5.1.2	Five-task Sequence	30
5.2	Efficiency	32
5.3	Results of Pseudo Sample Analysis	34
5.4	Ablation Study	36
5.5	Discussion	37
6	Conclusion and Future Work	38
6.1	Conclusion	38
6.2	Future work	38
	Appendix	45
A	Rational LAMOL	45
A.1	Critical Component Identification (CCI)	45
A.2	Unsupervised Rationale Generation	49
B	Results of Rational LAMOL	49

List of Figures

2.1	Scaled Dot-Product Attention	6
2.2	Multi-Head Attention	6
2.3	A depiction of Catastrophic Forgetting.	9
2.4	The re-injection sampling procedure, where x_0 refers to a real example. Given $t = 0, 1, 2, \dots$, the model constructs x_{t+1} and predicts y_t when it receives x_t . This procedure is repeated for k times. Finally all the constructed pseudo samples are utilized in the training of the next task.	14
2.5	Upper: LM learns to answer question given context. Lower: LM learns to generate training samples given generation token GEN. . . .	16
2.6	Left In a transformer layer, the adapter modules are inserted at two places: both before the layer normalizations. Right The design of the adapter modules. They consist of a bottleneck (two feed forward layers) and a non-linearity.	17
3.1	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.	19
3.2	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.	20

3.3	The process of our pseudo sample analysis. The colour orange in each decision diamond refers to rule-based decision while the colour purple means decisions are made by classifiers.	21
3.4	The process of our adopted Temporal Ensembling. Given that LM_1 is trained for X epochs, we utilize LM_1 from epochs X and $X-1$ to vote on an answer for the given pseudo sample. Note that only LM_1 is used for temporal ensembling.	23
5.1	Learning curves of task order BSM. 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.	31
5.2	Learning curves of task order SMTBF. Each graph shows the performance at each epoch for each task. Green background refers to the epochs on which the model is first introduced with a particular task.	33
1	Left: The overview of LAMOL. Right: The overview of Rational LAMOL, our proposed framework that aims to alleviate catastrophic forgetting by freezing the critical component.	46
2	Schematic illustration of the calculation of $IoU_{M,GT}$. A: The input is fed through each attention block AT_b , where each block b has multiple heads. B: A single attention head $AT_{b,a}$ consists of the attention of the sequence in relation to all other tokens, as shown in C . Finally, the IoU calculation F is applied on the hard selection of attention token with percentiles D and the rationale ground truth in E	48

List of Tables

1.1	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.	2
2.1	Desiderata of lifelong learning.	10
4.1	Summary of datasets, their sizes, and the corresponding metrics. EM is an exact match between texts while nF1 represents normalized F1 score.	25
4.2	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. . .	25
4.3	Hyperparameters used in our experiments	26
5.1	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.	29
5.2	Performance of LLL models on five tasks, averaged over three random seeds.	32

5.3	Runtime and parameter count of different LLL methods from Table 5.1. The runtime is an average of all task permutations across three random seeds.	32
5.4	Results of the pseudo sample analysis. The numbers indicate the amount of pseudo samples corresponding to each characteristics, averaged over three seeds.	35
5.5	The performance of other variations of our framework.	36
5.6	The performance of different methods on task sequence: SQuADv1 → WikiSQL → SST → QA-SRL → WOZ. Note that the ReGen strategy was not required since there were virtually no uninformative pseudo samples present in the experiments.	37
1	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-LAMOL and Gen R-LAMOL refer to Rational LAMOL and Generated Rational LAMOL, respectively. . . .	50

Chapter 1

Introduction

Machine Learning (ML) is becoming one of the most important parts of human life. Although it is widespread and successful in a wide range of applications, ML still lacks one crucial aspect that exists in biological intelligence, the ability to continually learn and accumulate knowledge. Currently, ML models largely learn in isolated environments, where the data distribution is assumed to be stationary. In the real world, however, this does not hold true. A phenomenon known as concept drift is ingrained within every stream of data. With the advent of deep learning and the ever-growing amount of data, the problem of lifelong learning (LL) is becoming even more critical, as it is currently not possible for an ML model to adapt to new information and effectively retain previously learned knowledge at the same time.

Lifelong learning, first introduced in Ring (1994), aims to create a learner that is able to effectively reuse learned knowledge to aid in the process of learning new tasks while preventing a problem known as catastrophic forgetting (CF) or catastrophic interference (McCloskey and Cohen 1989) altogether. Naively retraining a model from scratch every time a new task is received is costly and time-consuming. This naive approach is also limited by capacity saturation (Sodhani, Chandar, and Bengio 2020; Aljundi, Chakravarty, and Tuytelaars 2017). Concretely, a parametric model eventually will reach a point where no more knowledge can be stored inside its parameters. At this point, either some knowledge has to be selectively forgotten or the model's capacity has to be expanded, both of which has their own downside. This constraint is known as the stability-plasticity dilemma. Particularly, the balance between the model's stability (the ability to retain previously learned knowledge) and plasticity (the ability to adapt to new knowledge) has to be struck. Although numerous methods have been proposed to address the long-standing problem of CF, most of them are in the computer vision or robotics domains. Researches in the NLP domain is still nascent. The differences are reflected in the small number of proposed methods and the evaluation benchmarks (Greco et al. 2019).

One of the more recent lifelong language learning (LLL) approach is LAMOL

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

Table 1.1: **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.

(Sun, Ho, and Lee 2020). It has been shown to effectively prevent catastrophic forgetting by generating examples (called pseudo samples) from previous tasks to be replayed when training on a new task. However, the pseudo sample generation process is not perfect. When trained on datasets with long texts, LAMOL 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 1.1 (bottom) and will be explained in Section 2.7. As a result, LAMOL cannot effectively prevent CF in this situation.

We believe that overcoming the catastrophic forgetting problem will allow ma-

chine learning algorithms to accumulate knowledge, eventually allowing for a wider range of applications and use cases. For instance, a conversation agent needs to continually interact with users. It is inevitable that the information it possesses will be outdated. In order to keep the agent relevant, its database has to be manually updated which may be expensive. Therefore, it is crucial for every algorithm to possess the ability to adapt its knowledge to the ever-growing volume of data.

As a preliminary study into the lifelong learning settings and its intricacies, we developed the Rational LAMOL framework, described in Appendix A, as a joint research project with a fellow graduate student. However, as it is required by the curriculum that a thesis must be an independent work, we instead decided to separately tackle the problem of LL from other perspectives. Motivated by the problem of low quality pseudo samples discovered during the development of the Rational LAMOL framework, in this thesis, we aim to develop new algorithms that can be applied in complement with an existing method to further alleviate Catastrophic Forgetting. Particularly, this research approaches the LL problem by trying to improve the process of pseudo samples generation by introducing a new separate framework that rationally utilizes the input structure. 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 leveraging the input’s rational structure to enhance pseudo sample quality with a semi-supervised learning technique (i.e., temporal ensembling) and by detecting and reducing the number of uninformative pseudo samples.

1.1 Objectives

- To present a new LLL framework that is able to prevent catastrophic forgetting more effectively.
- To improve the quality of generated pseudo samples with a semi-supervised learning technique (i.e., temporal ensembling) and by detecting and reducing the number of uninformative pseudo samples.

1.2 Contributions

- Creating an LLL framework that is capable of learning diverse tasks even with long text while suffering less catastrophic forgetting compared to an existing method.
- Utilizing adapter modules 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.

1.3 Scope

Following tasks will be undertaken as a part of the proposed research :

- Design a new framework that is capable of generating high quality pseudo samples.
- Demonstrate efficiency improvement from our new framework by utilizing adapter modules.
- Evaluate our proposed algorithms on multiple text classification datasets and a question answering dataset up to a sequence of five tasks long, against LAMOL, a strong lifelong learner baseline.
- Improve the quality of different pseudo samples characteristics using a semi-supervised learning technique.
- Provide insights from our pseudo sample analysis that could pave way for better pseudo-rehearsal LLL methods.

Chapter 2

Background & Related Work

In this section, background and work related to this research are presented. Section 2.1 to Section 2.4 gradually introduces the attention mechanism to GPT2 (Radford et al. 2019b), the language model used in our framework. Section 2.5 introduces catastrophic forgetting which Lifelong learning, described in Section 2.6 aims to solve. We also introduce LAMOL (Sun, Ho, and Lee 2020) upon which we build our frameworks in Section 2.7. Finally, we briefly summarize the adapter modules, a component that is used to reduce the computational requirements of our framework, in Section 3.1.1.

2.1 Attention

Prior to the introduction of the first attention model (Bahdanau, Cho, and Bengio 2016), sequential models such as recurrent neural networks (RNN), long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997) and gated recurrent units (GRU) (Chung et al. 2014) long dominated the field of sequence transduction (Sutskever, Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2016; Cho et al. 2014). RNNs have been shown to be unable to memorize long sequences (Bengio, Frasconi, and Simard 1993), which leads to loss of information. Although LSTM is proposed to improve RNN in capturing long-range dependencies, its sequential nature prohibits parallelization. Moreover, there is no way for LSTMs to selectively focus on important parts of the input.

The attention mechanism was realized to help memorize long source sentences. Rather than using a single output from the last hidden state as in RNNs, it creates shortcuts between each input. The weights of these shortcuts connections allow the model to ‘see’ all parts of input at the same time. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed

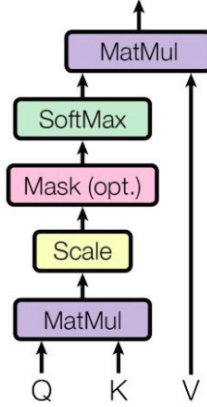


Figure 2.1: Scaled Dot-Product Attention

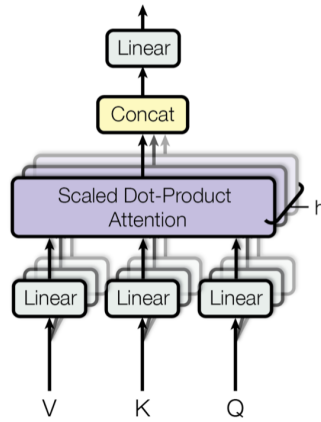


Figure 2.2: Multi-Head Attention

by a compatibility function of the query with the corresponding key which can be expressed as follows:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.1)$$

Multi-headed Attention This attention function is performed in parallel by linearly projecting the queries, keys and values h times with different, learned linear projections in the multi-head attention. These projections are then applied with the attention function in parallel and then concatenated and then projected as shown in Figure 2.2. The information from different representation subspaces at different positions could be now attended to from the multi-head attention model, in comparison with the averaging of the single attention head. In the original paper of Vaswani et al. (2017), they employed $h = 8$ parallel attention heads.

2.2 Transformers

Most attempts at solving sequence transduction problems use an encoder-decoder structure where the encoder maps an input sequence into a representation. This representation is then used by the decoder to generate an output sequence. At each step the model is auto-regressive (Graves 2013), consuming the previously generated symbols as additional input when generating the next.

The Transformer also follows this overall architecture using only stacked self-attention and fully connected layers for both the encoder and decoder, which are described as followings:

Position-wise Feed-Forward Networks After transforming an input into representations, before passing it to the next layer, it needs to be reshaped back into a compatible shape beforehand. Therefore, each of the layers contains a fully connected feed-forward network which consists of two linear transformations with a ReLU activation in between.

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (2.2)$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. The dimensionality of input and output is $d_{model} = 512$, and the inner-layer has dimensionality $d_{ff} = 2048$.

Positional Encoding Since there is no recurrence nor convolution in the Transformers, the sequential information cannot be exploited by the model. To prevent this, at the bottoms of the encoder and decoder stacks, “positional encodings” are added to the input embeddings. In the work of (Vaswani et al. 2017), sine and cosine functions of different frequencies are used in hopes that the model would be able to attend by relative positions and may be able to extrapolate to sequences longer than it encountered during training.

2.3 Generative Pre-trained Transformer (OpenAI GPT)

With the aim of utilizing the abundant unlabelled text corpus, OpenAI GPT (Radford et al. 2018) explores a two-stage approach of language understanding tasks using unsupervised pretraining and supervised finetuning. The goal is to learn a representation that can be universally transferred with little adaptation to a wide range of tasks. The language model architecture used in the paper is based on a

stack of Transformers decoder which, in the pretraining stage, is trained on a large corpus using a standard language modelling objective. Finally, the model is adapted to the tasks at hand in the finetuning stage where another task-specific linear layer is appended to the last Transformer block. The model is then optimized on both the task-specific loss and an auxiliary language modelling loss. The rationales behind the optimization on the combination of the losses are improved generalization and accelerated convergence of the model.

2.4 OpenAI GPT-2

Following the success of leveraging pretraining and the trend of growing model parameters, GPT-2 (Radford et al. 2019b) is an improved version of OpenAI GPT. The model mostly follows the details of the OpenAI GPT model (Radford et al. 2018) with a few modifications, namely:

- Layer normalization (Ba et al., 2016) was moved to the input of each sub-block, similar to a pre-activation residual network (He et al., 2016).
- An additional layer normalization was added after the final self-attention block.
- A modified initialization which accounts for the accumulation on the residual path with model depth is used.
- Scale the weights of residual layers at initialization by a factor of $1/\sqrt{N}$ where N is the number of residual layers.
- The vocabulary is expanded to 50,257. The context size is increased from 512 to 1024 tokens and a larger batchsize of 512 is used.

GPT-2 is trained on a large corpus of 40 GB of internet-scraped text. It shows strong performance on a wide range of tasks in zero-shot settings.

2.5 Catastrophic Forgetting

The abrupt forgetting of a previously learned knowledge upon learning new information was first discovered by McCloskey and Cohen (1989) in which this phenomenon was coined Catastrophic Forgetting (CF). This is illustrated in Figure 2.3. Given a neural network parameterized as θ , when training commences on task A , the parameters are optimized towards the lowest point of the loss landscape of task A . However, when it is sequentially trained on task B , the parameters are then displaced towards the lowest point of task B 's loss landscape. Since these two tasks

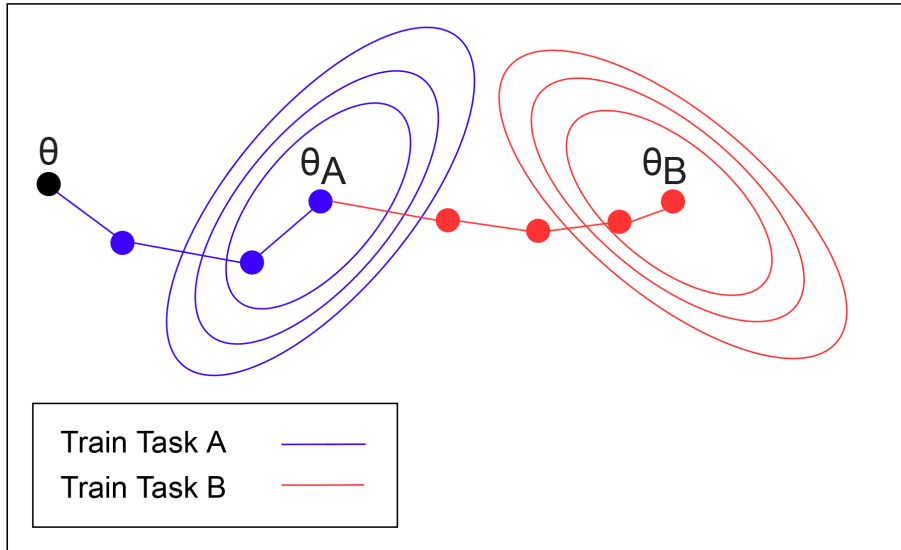


Figure 2.3: A depiction of Catastrophic Forgetting.

have different global minima, the model is no longer able to perform well on task A . Metaphorically, the model “forgets” the knowledge it has learnt on task A .

Neural network weights change when acquiring new tasks. Too large of a change disrupts previously learned knowledge. On the other hand, keeping the weights stable prohibits the network’s ability to learn new tasks. This is called the stability-plasticity dilemma (Abraham and Robins 2005). Multiple methods have been proposed in an attempt to solve this problem, which can be grouped into two approaches:

The first strategy is to uncouple the new and old representations. This can be accomplished through the use of distributed models, regularization, and ensembling.

The second is to prevent learnt knowledge from being forgotten by simultaneously training on both the new tasks and data from the old tasks. However, this approach still is not as effective as simply retraining a model from scratch on both the new tasks and the old tasks.

Even though much effort has been put into combating the CF problem, it persists, hindering the development of an intelligence system that is capable of learning new tasks over the course of its lifetime.

2.6 Lifelong Learning

A machine learning paradigm whose objective is to learn from a continuous stream of data, where the number of tasks may be potentially infinite while reusing previously learned knowledge to improve generalization for previous tasks was first introduced in Ring (1994) as Lifelong learning (LL). In this setting, the newly acquired infor-

Property	Definition
Knowledge retention	The model is not prone to catastrophic forgetting.
Forward transfer	The model learns a new task while reusing knowledge acquired from previous tasks.
Backward transfer	The model achieves improved performance on previous tasks after learning a new task.
On-line learning	The model learns from a continuous data stream.
No task boundaries	The model learns without requiring neither clear task nor data boundaries.
Fixed model capacity	Memory size is constant regardless of the number of tasks and the length of a data stream.

Table 2.1: Desiderata of lifelong learning.

mation should not interfere occur with learned knowledge. More formally: the goal is to sequentially learn a model $f : X \times T \rightarrow Y$ from a large number of tasks τ . The model is trained on examples (x_i, y_i) , such that: $x_i \in X_{t_i}$ is an input feature vector, $y_i \in Y_{t_i}$ is a target vector (e.g. a class label), and $t_i \in T$ denotes a task descriptor (in the simplest case $t_i = i$) where $i \in Z$. The objective is to maximize the function f (parameterized by $\theta \in R$) at the task T_i , while minimizing CF for tasks T_i, \dots, T_{i-1} . Although the above-mentioned definitions of LL may seem fairly general, there are certain desired properties, which are summarized in Table 2.1.

In practice, however, current LL systems mostly relax more than one of the requirements listed in Table 2.1. Most methods still learn in an offline manner, where models are trained using batches of data shuffled in such a way as to satisfy the independent and identically distributed (i.i.d.) assumption. Consequently, many models are trained solely in a supervised fashion with large labelled datasets, and thus they are not exposed to more challenging situations involving few-shot or self-supervised learning. Additionally, existing approaches often fail to restrict themselves to making a single pass over the data, and this entails longer learning times. Moreover, the number of tasks as well as their identity are frequently known.

2.6.1 Approaches to lifelong learning

Multiple approaches have been proposed to tackle the long-standing challenge of LL. In this section, we will summarise the main idea of each approach and introduce the most prominent methods for each approach.

Architectural methods introduce task-specific parameters in order to accommo-

date incoming new knowledge while previous task parameters are typically kept fixed or masked out.

Progressive Neural Network (PNN) (Rusu et al. 2016) adds capacity to the existing network by adding a new sub-network, referred to as “column”. Given a new task $T + 1$ and a trained network θ_T , PNN augments a new sub-network to the existing network θ_T and trains only the lateral connections between the added network θ_{T+1} and the original network while keeping θ_T frozen. Although PNN can retain perfect memory of past tasks, the model complexity rapidly increases with incoming tasks.

Wen, Tran, and Ba (2020) introduce BatchEnsemble to reduce the increasing complexity of PNN by training only fast weights. For an ensemble of n models, each model \bar{W}_i is responsible for each task T in the LL setting and owns a tuple of trainable vectors r_i and s_i . BatchEnsemble generates \bar{W}_i by multiplying the “fast weights” r_i and s_i with a shared weight W :

$$\bar{W}_i = W \cdot r_i s_i^T \quad (2.3)$$

During training on the first task T , BatchEnsemble optimizes both shared weights W and fast weights r_T and s_T . For later tasks $T + 1$, only fast weights r_{T+1} and s_{T+1} are trained. During testing, the predictions are obtained by averaging the predictions of the members. BatchEnsemble is able to achieve similar performance with PNN while reducing the parameter count by almost four times. However, by training the shared weights on the first task, this method only works if the knowledge from the first task is transferable to subsequent tasks.

Regularization methods rely on an additional regularization term that is typically designed with inspiration from theoretical neuroscience models to aid knowledge consolidation when learning new tasks. This group of methods utilizes a single model with a fixed capacity. As a result, new knowledge may eventually overwrite knowledge from old tasks.

Elastic Weight Consolidation (Kirkpatrick et al. 2017a) reduces forgetting by regularizing the loss in such a way that changes to parameters important to previous tasks become smaller. For each parameter θ , its relevance with respect to a specific training dataset D can be modelled as a posterior distribution $p(\theta|D)$. Therefore, in the context of two distinct tasks T and $T-1$ with two training datasets D_T and D_{T-1} , respectively, the posterior probability of the parameters can be calculated as:

$$\log p(\theta|D) = \log p(D_T|\theta) + \log p(\theta|D_{T-1}) - \log p(D_T) \quad (2.4)$$

where $\log p(\theta|D_{T-1})$ is hypothesised to contain all the information about parameter importance of task $T - 1$. Nevertheless, the true posterior probability can only be approximated using a Gaussian distribution with mean given by the parameters θ_{T-1}^* and a diagonal precision given by the diagonal of the Fisher information matrix F (MacKay 1992). Given the approximation, the loss function is described as:

$$\mathcal{L}(\theta) = \mathcal{L}_T(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i + \theta_{T-1}^*)^2 \quad (2.5)$$

where $\mathcal{L}_T(\theta)$ is the loss on task T and θ is the importance of the old task compared with the new task.

In a similar vein, Aljundi et al. (2017) introduced Memory Aware Synapses (MAS) which computes parameter importance during training in an online manner. For each training sample, MAS accumulates the importance of a particular parameter by measuring the change in predicted output when applying a small perturbation to the parameter. Specifically, given a mapping function F which maps the input X to output Y and a small perturbation δ_{ij} , the parameter sensitivity can be approximated by:

$$F(X; \theta + \delta) - F(X; \theta) \approx \sum_{ij} g_{ij}(X) \delta_{ij} \quad (2.6)$$

where $g_{ij}(X)$ refers to the gradient of the mapping function F with respect to the parameter θ_{ij} on some input X . The parameter importance Ω_{ij} can be obtained by accumulating $g_{ij}(X_k)$ for $k \in D$ where D is a training dataset. Finally, the loss function can be described as:

$$\mathcal{L}(\theta) = \mathcal{L}_T(\theta) + \lambda \sum_{ij} \Omega_{ij} (\theta_{ij} + \theta_{ij}^*)^2 \quad (2.7)$$

where λ is a hyperparameter for the regularization and θ_{ij}^* are the parameters of the network from the previous task.

Rehearsal-based methods keep a subset of training examples from previously learned tasks in order to be replayed when encountering a new task. One of the most well-known methods for incremental class learning is the iCaRL model, proposed by Rebuffi et al. (2017). iCaRL is a classifier that is able to incrementally learn novel classes. By keeping a small subset of “exemplar” images for each class it has seen. This exemplar set of images are utilized for rehearsal and used as prototypes

for classification, similar to a *nearest-class-mean* classifier. iCaRL achieved good performance on multiple class-increment tasks up to 1,000 classes.

The downside of the rehearsal-based methods approach is the amount of storage and computing requirement grow proportionally to the number of tasks. A subgroup of this method coined pseudo-rehearsal methods instead utilize the generative capability of the models to emulate the distribution of previous task samples. This partially solves the problem of storage requirement, although generated samples are usually not as expressive as real samples. One notable approach is DGR, or Deep Generative Replay (Shin et al. 2017), a framework based on Generative Adversarial Networks (Goodfellow et al. 2014). DGR is inspired by the Complementary Learning Systems (CLS) theory that suggests that the dual memory systems of the hippocampus and the neocortex work interdependently. The hippocampus is responsible for short-term memory which is consolidated in the neocortex. Using the same idea, DGR utilizes a GAN to reconstruct data from previous tasks which are used for rehearsing itself and the classifier when training on the next task. DGR managed to achieve comparable performance with rehearsing with real examples.

In the context of LLL, rehearsal-based approaches such as LAMOL (Sun, Ho, and Lee 2020) have been shown to be the most promising group of methods, outperforming notable methods of other approaches such as EWC (Kirkpatrick et al. 2017b) 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. 2019c). Complex data distributions can be modelled 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 works have been proposed in order to address the problem in the computer vision domain. Here, we briefly summarize a few notable works.

Solinas. et al. (2021) proposed storing a small amount of real data as seeds for generating pseudo data via an autoencoder using re-injection sampling procedure (Ans and Rousset 1997). The procedure was originally devised to generate pseudo samples that capture the knowledge of artificial neural networks. Each time a pseudo sample is re-injected into the autoencoder, it is displaced slightly toward the area of high densities in the training distribution (Bengio et al. 2013). This procedure is illustrated in Figure 2.4. They were able to outperform strong rehearsal-approach baselines such as experience replay (Chaudhry et al. 2019).

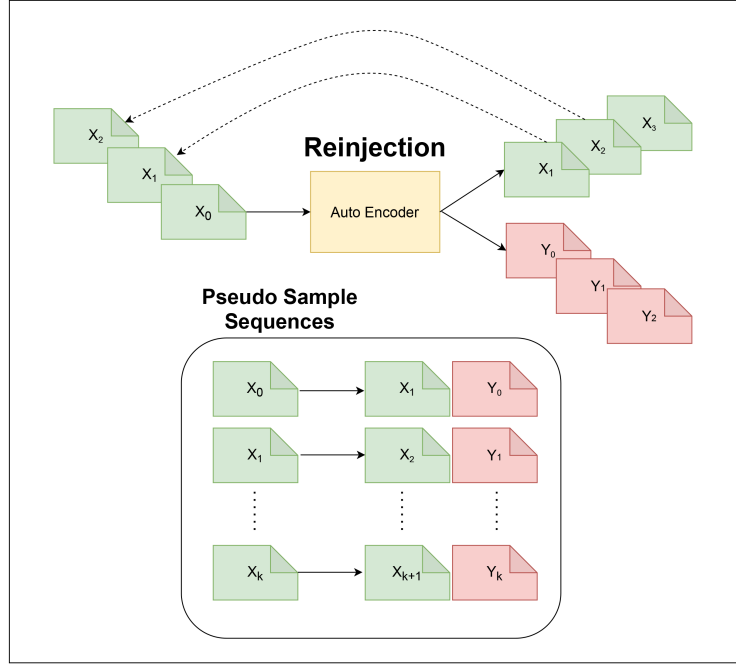


Figure 2.4: The re-injection sampling procedure, where x_0 refers to a real example. Given $t = 0, 1, 2, \dots$, the model constructs x_{t+1} and predicts y_t when it receives x_t . This procedure is repeated for k times. Finally all the constructed pseudo samples are utilized in the training of the next task.

Silver and Mahfuz (2020) utilized a stack of Restricted Boltzmann Machine (RBM) (Hinton 2010) to generate pseudo samples and selected 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 used, 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. Their framework consists of an encoder, a generative model based on a normalizing flow (NF) (Papamakarios et al. 2019), and a decoder. The NF model is an invertible neural network, capable of performing both sampling and density estimation in both ways. During training, the encoder (augmented with a task-specific head) encodes the training data while the NF is optimized on generating the encoded data and the decoder trains on reconstructing the encoded data. When training on the next task, the framework performs a regularization step, occasionally, replacing data from the new task with samples drawn from the NF model. The regularization step is performed as follows:

1. The NF is utilized to generate an embedding from the training distribution.
2. The input embedding is used by the decoder to reconstruct a pseudo training data.
3. The encoder encodes the reconstructed pseudo training data.
4. The encoder is regularized based on the distance loss between the encoded pseudo sample and the generated sample from step 1.

With this framework, 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 learn contain long texts but with insufficient training data.

2.7 LAMOL

Inspired by Shin et al. (2017), LAMOL (Sun, Ho, and Lee 2020) leverages a single GPT-2 language model (LM) (Radford et al. 2019c) to prevent CF by utilizing the innate generative capability of the LM to create pseudo samples which 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 GPT-2 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 2.5. 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 $t = 1, \dots, i$. 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

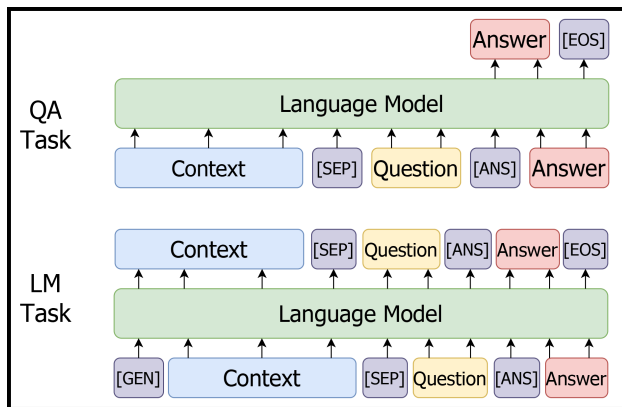


Figure 2.5: **Upper:** LM learns to answer question given context. **Lower:** LM learns to generate training samples given generation token GEN.

for each previous task, where γ is a hyperparameter. Finally, the LM model learns on the mixture of new examples of task τ_{i+1} and pseudo samples of previous tasks.

Even though pre-trained LMs such as GPT-2 have shown impressive capabilities in learning various tasks, they require large amount of training examples to converge properly. The problem is even more prevalent in complex tasks like language modelling. 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. Since 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 from 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 task-specific token specified.
4. Wrong Answer: Incorrect answers are generated for some pseudo samples.

These problems are depicted in Table 1.1.

Consequently, without adequate amount of usable pseudo samples, LAMOL loses the ability of preventing catastrophic forgetting and is comparable with only sequential finetuning.

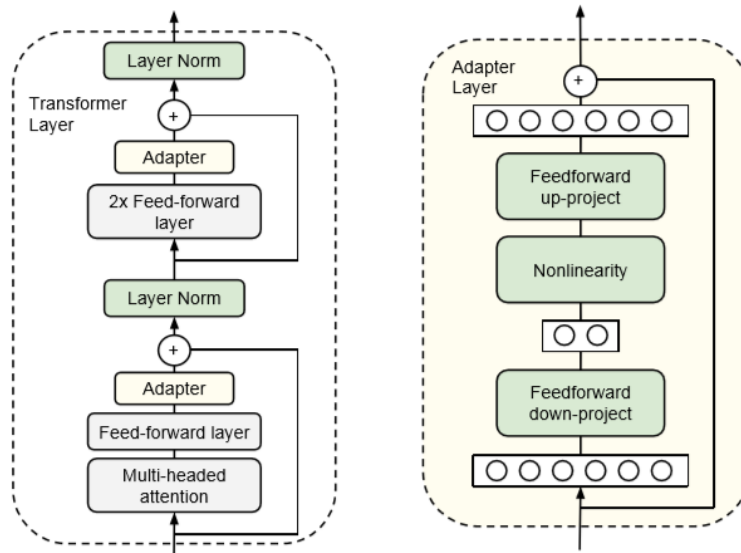


Figure 2.6: **Left** In a transformer layer, the adapter modules are inserted at two places: both before the layer normalizations. **Right** The design of the adapter modules. They consist of a bottleneck (two feed forward layers) and a non-linearity.

2.8 Adapter Modules

Finetuning 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 finetuned individually for each task. To alleviate this issue, Houlsby et al. (2019) introduced adapter modules as an alternative paradigm for transfer learning.

On a high level, the adapters (each of which is composed of two feed forward layers and a non-linear activation function) are inserted into each layer of a transformer model and used to adapt the content of each transformer block of the base pre-trained model.

More specifically, the first feed forward layer in the adapters down-projects the d -dimensional input into n -dimensions after which it is applied the non-linearity. Finally, the second feed forward layer up-projects the input back to d -dimensions. For each layer in a transformer-based model, the adapter modules are inserted at two places, both before each of the layer normalizations in the transformer layer. The architecture of the adapters is illustrated in Figure 2.6.

Adapters introduce only 0.5% - 8% of conventional large pre-trained models such as GPT-2. During the finetuning step, only the adapters are finetuned, increasing the training efficiency when compared with finetuning the whole model. Extensive experiments showed that the resulting model achieved similar performance with conventional finetuning.

Chapter 3

Concept and Research

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 3.1.1 details the integration of adapter modules into our framework. We also describe the procedure of our pseudo sample analysis in Section 3.2. Finally, we detail our pseudo sample enhancement strategies in Section 3.3.

3.1 Double LM

As mentioned in Section 2.7, LAMOL suffers heavily from the low quality of pseudo samples. Using insights from multiple works such as Pelosin and Torsello (2021) and Masson d’Autume et al. (2019) where the performance of rehearsal-based LL methods heavily rely on the preserved samples, we seek to improve the pseudo sample generation of LAMOL with the aim of increasing pseudo samples quality.

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 modelling 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 setup is shown in Figure 3.1 (right). The first LM would take the main responsibility of learning the QA task, i.e., 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

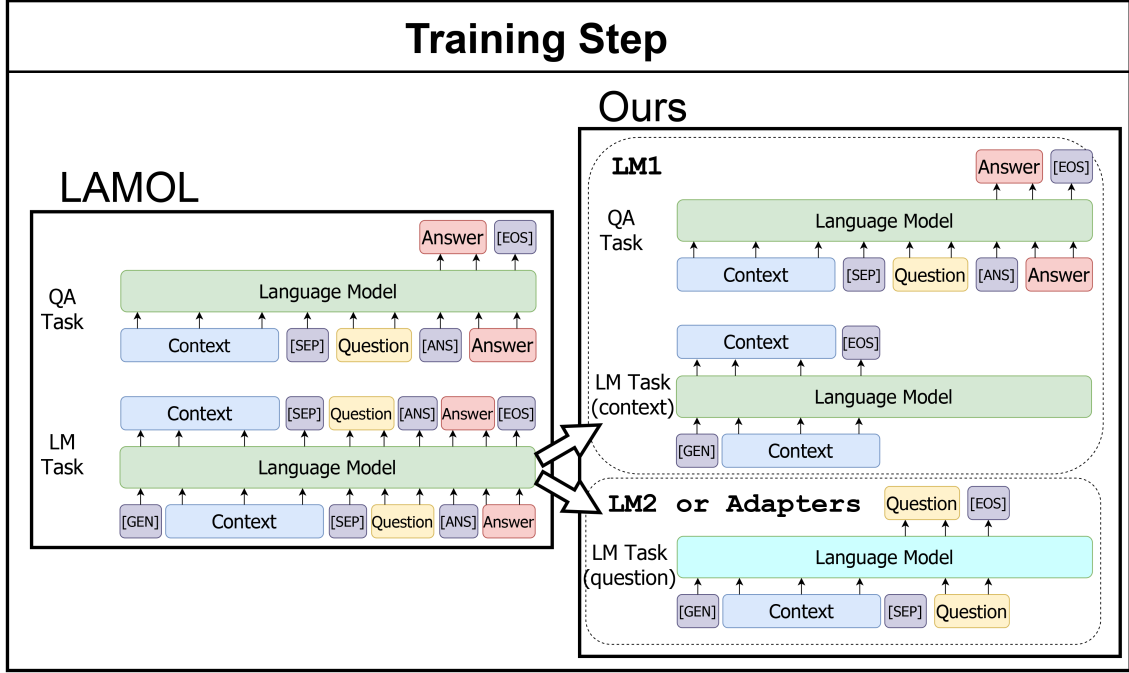


Figure 3.1: **Left:** Training step of LAMOL. In a single optimization step, a single language model is trained on the QA task (upper) and the LM task (lower). **Right:** Our framework utilizes two language models that focus on different parts of the input. The first LM is optimized on the QA task and the context generation task, while the second LM is optimized solely on the question generation task.

would be defined as:

$$L_{LM_1} = L(Y_{QA}, \theta_{LM_1}(X)) + \lambda L(Y_{context}, \theta_{LM_1}(X)) \quad (3.1)$$

$$L_{LM_2} = L(Y_{question}, \theta_{LM_2}(X)) \quad (3.2)$$

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

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.

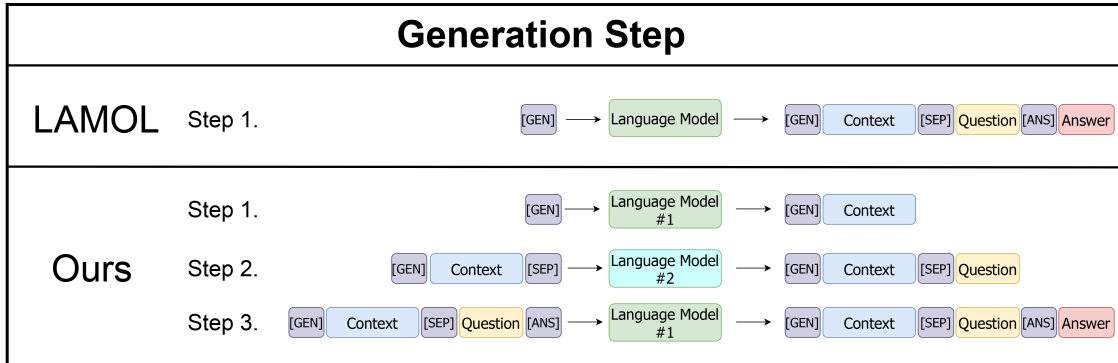


Figure 3.2: **Top:** Pseudo sample generation step of LAMOL. Given a [GEN] token, a single LM generates the whole sample. **Bottom:** Given a [GEN] token, LM_1 is utilized to generate a context. Next, given the context, LM_2 generates the corresponding question. Finally, given the context and the question, 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.

The process is illustrated in Figure 3.2 (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 modelling task, leading to better pseudo samples.

3.1.1 Adapter

Training another instance of GPT-2 LM as in Section 3.1 imposes significant additional memory and computation requirements. Thus, we also propose to instead use the adapter modules to mimic the function of the additional GPT-2 model as a remedy to the problem.

In our framework, the adapters are added *after* the LM_1 has been trained on Equation 3.1. Since the adapter modules can utilize the information learned by the underlying model, we believe that it can effectively function as well as the LM_2 . Then, the LM_1 , which can now be referred to as the base model, is kept frozen, while we train the added adapters using Equation 3.2.

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 letting 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

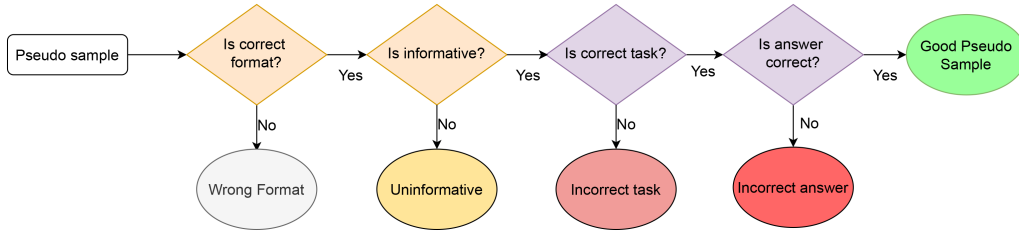


Figure 3.3: The process of our pseudo sample analysis. The colour orange in each decision diamond refers to rule-based decision while the colour purple means decisions are made by classifiers.

we utilize the base model to generate the answer to the pseudo sample.

3.2 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 3.3.

- First, we check if a pseudo sample conforms to the correct format. Recall that pseudo samples with the incorrect format are discarded. 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 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”. The number was obtained from the macro-average of minimum numbers of unique tokens of the datasets we used in our experiment.

- 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 finetuned 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” else it is classified as “Wrong Answer”¹. As with the previous step, we finetuned one RoBERTa model per task and use all the finetuned models to analyze the quality of pseudo samples only.

3.3 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 un informativeness.

A number of pseudo samples have been observed to be repetitive and non-sensical, containing only a few unique tokens without actual meaning. This phenomenon has also been referred to as “failure modes” in the original GPT2 blog post (Radford et al. 2019a). To reduce the amount 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 3.2, 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). The work builds around the idea of dropout regularization (Srivastava et al. 2014). When training with dropout, only a subset of the network is utilised.

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

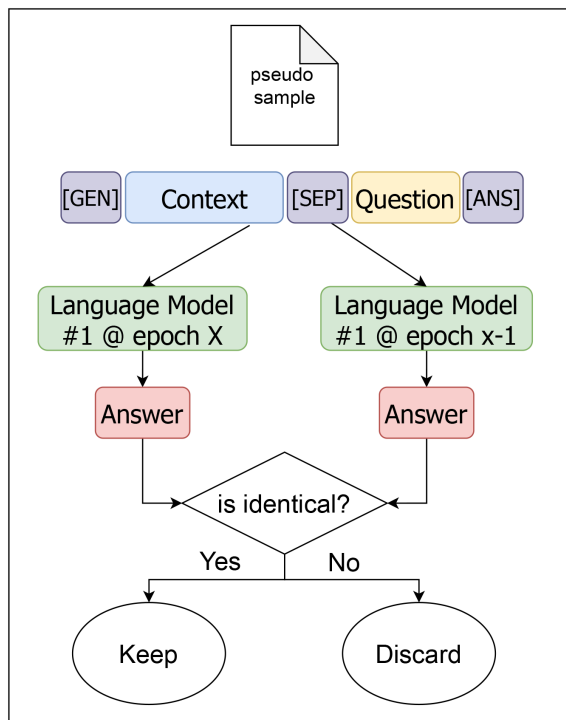


Figure 3.4: The process of our adopted Temporal Ensembling. Given that LM_1 is trained for X epochs, we utilize LM_1 from epochs X and $X-1$ to vote on an answer for the given pseudo sample. Note that only LM_1 is used for temporal ensembling.

Essentially, the whole network can be seen as an ensemble of multiple sub-networks. Temporal Ensembling extends the idea by performing ensembling across different epochs of training. This is based on the assumption that predictions which are not stable even when reaching the end of the training are not likely to be a reliable label. Originally, Temporal Ensembling is designed to work in conjunction with data augmentation to introduce additional noise. We believe that a similar effect can be achieved by generating pseudo samples from two different LMs. Therefore, explicit data augmentation is not required in our adoption of Temporal Ensembling.

In our work, during the generation process, two models (LM_1) from the last two epochs of training are utilized to vote on answers for pseudo samples. We only keep pseudo samples that the two models agreed on an answer, whereas the rest are replaced with a new batch of pseudo samples. This process is depicted in Figure 3.4.

Chapter 4

Experimental Setting

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 4.1. In Table 4.2, 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. The questions in this dataset were generated in unprompted and unconstrained settings, resulting in naturally occurring questions. The setup of this text-pair classification task is similar to natural language inference tasks, albeit more challenging.
- Movie Reviews (Zaidan, Eisner, and Piatko 2008): a dataset that includes movie reviews with positive/negative sentiment labels. Most reviews comprise a combination of positive text, negative text, and factual descriptions.
- 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. This dataset was created in an effort to facilitate the development of an assistant system capable of assessing the correctness of a claim. Therefore, it is a strenuous task, requiring both reading comprehension skills and real-world knowledge.
- Fever (Thorne et al. 2018): Fact Extraction and VERification is a dataset consisting of claims and textual sources, i.e., 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.

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

- 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.

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

Table 4.2: 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.

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.

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 GPT-2 model (Radford et al. 2019c) as the language model for all methods. We applied greedy decoding during inference.

The adapter modules 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 4.3.

Hyperparameter	Value
<i>Training hyperparameters</i>	
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
<i>LM-specific hyperparameters</i>	
Learning rate	6.25×10^{-5}
Top-k sampling	k=20
<i>Adapters-specific hyperparameters</i>	
Learning rate	1×10^{-4}
Reduction factor	16
Non-linearity	ReLU

Table 4.3: Hyperparameters used in our experiments

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 used `adapter-transformers`¹ for the implementation of the GPT-2 LM and adapters.

4.3 Metrics

For classification tasks (the first four datasets), we used EM, or exact match between texts, as the metric. This is because the GPT-2 is a generative model. However, because of the nature of text classification, the percentage of exact matches can also be seen as the accuracy of the model.

While, for the TriviaQA dataset, we used the standard nF1 score, or normalized²

¹<https://github.com/Adapter-Hub/adapter-transformers>

²Note 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.

F1 score. Since the scores for all metrics lie between 0 and 1, we can simply average the scores across different metrics.

The amount of catastrophic forgetting will be reflected in the performance gap between the multitasking upper-bound at the end of a task sequence. The better lifelong learner will be less affected by task order; therefore, we also report standard deviation across different permutations of each method.

Chapter 5

Results

This section reports and discusses the experimental results. Specifically, Section 5.1 and 5.2 report the final LL performance and runtime of different methods, respectively. Section 5.3 shows the result of our pseudo sample analysis. This is then followed by an ablation study and an additional discussion in Section 5.4 and 5.5, respectively.

5.1 LL Performance

5.1.1 Three-task Sequence

We trained the baseline and our proposed frameworks, Rational LAMOL and Double LM, on six permutations of three tasks: BoolQ (B), Movie Reviews (M), and Scifact (S). The results are shown in Table 5.1.

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

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. In the case where 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

Methods	BMS	BSM	MBS	MSB	SBM	SMB	Average	Std.	
<i>Baselines. Section 2.7 & Section 5.1.1</i>									
LAMOL	64.53	35.48	66.79	60.76	52.02	54.40	55.67	11.41	
LAMOL _{all}	62.22	62.06	61.42	52.93	65.32	63.35	61.22	4.29	
<i>Double LM Framework. Section 3.1-3.1.1</i>									
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	
<i>With Additional Pseudo Sample Enhancement. Section 3.3</i>									
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 _{real}	69.07	71.97	70.84	72.31	74.13	73.32	71.94	1.80	
Multitask							75.52		

Table 5.1: 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.

all generated pseudo samples were used. The result is shown in LAMOL_{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), LAMOL_{all} introduced noise that destructively interfered with learned knowledge.

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 mimic the function of the additional GPT-2 of Double LM. Both of our variants were competitive with LAMOL_{real} (using real examples instead of pseudo samples) in the orders BMS and MBS but slightly underperformed in the other orders. Concerning the strategies proposed in Section 3.3, applying ReGen (R) to our LM+Adapter (i.e., LM+Adapter+R) was able to gain 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 LAMOL_{real} with only a 0.73% difference in the accuracy score.

A sample of accuracy graphs (as the learning progressed) of the compared methods, with the BoolQ → SciFact → Movies (BSM) task order, is shown in Figure 5.1 from top to bottom, respectively. From the graphs, as training progressed from Bool-Q to SciFact, all methods showed good knowledge retention ability, achieving approximately 58% Bool-Q accuracy; albeit, slightly lower for LAMOL, only 48% Bool-Q accuracy. Then, as training moved on to the Movie task, LAMOL entirely forgot all the knowledge from the Bool-Q task. On the other hand, Double LM, LM+Adapter, and LM+Adapter+T suffered from a small performance drop, after which they were later restored. Meanwhile, LAMOL_{real}, LM+Adapter+R, and LM+Adapter+RT were not affected by the task shift. This could be explained in part by the stability-plasticity dilemma. By having higher stability, the learner is less affected by CF (Bool-Q drop is smaller); however, it is harder for the learner to assimilate new knowledge (SciFact performance is lower). Similarly, all models except LAMOL was able to prevent knowledge loss on SciFact task, attaining approximately 60% accuracy, while LAMOL achieved only around 20%. As the last task, Movies was not affected by CF. All models were able to achieve a comparable performance of approximately 95%.

5.1.2 Five-task 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).

The results are shown in Table 5.2, where 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 an improvement of 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.

We also provided accuracy graphs in Figure 5.2. Our proposed methods showed significantly higher knowledge retention ability as seen on the first three tasks of

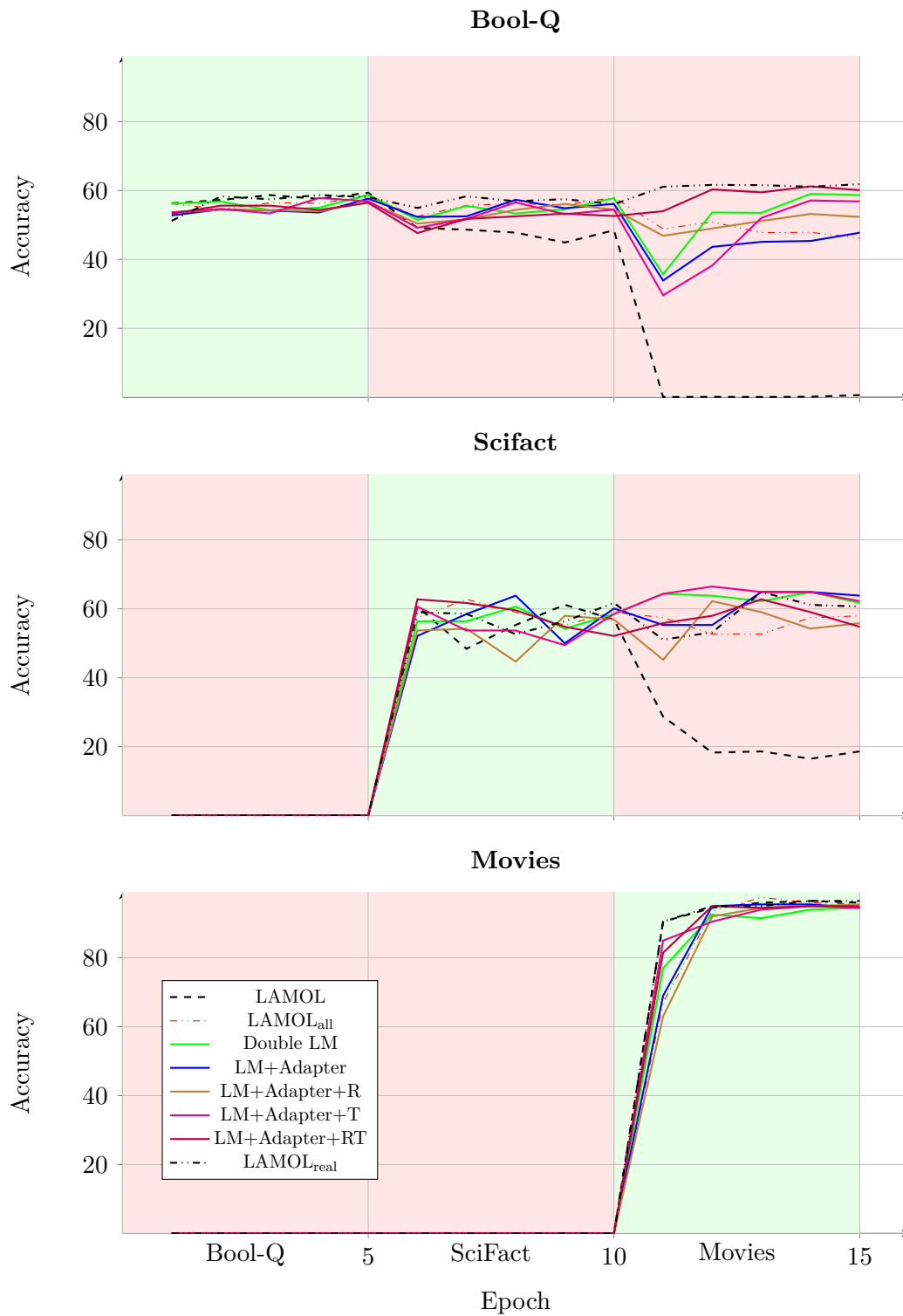


Figure 5.1: Learning curves of task order BSM. The graphs show accuracy at each epoch for each task. Green background refers to the epochs on which the model is first introduced with a particular task. In this figure, for example, the model is trained on Bool-Q and evaluated on all the three tasks during epoch 1-5.

Methods	FBTMS	SMTBF	Average
LAMOL	57.01	44.32	50.67
LM+Adapter	65.51	62.18	63.85
LM+Adapter+RT	66.03	67.74	66.88
LAMOL _{real}	70.95	71.83	71.39
Multitask		68.89	

Table 5.2: Performance of LLL models on five tasks, averaged over three random seeds.

Methods	Runtime	#Parameters
LAMOL	90.7 min	124.44M
Double LM	178.2 min	248.88M
LM+Adapter	127.5 min	125.33M
Re-generate	+13.1 min	-
Temporal Ensem.	+3.2 min	-

Table 5.3: Runtime and parameter count of different LLL methods from Table 5.1. The runtime is an average of all task permutations across three random seeds.

the sequence. However, they slightly underperform LAMOL on the Bool-Q task. One possible explanation for this is because the pseudo samples generated from our framework might not be able to provide the positive forward transfer as did the real examples (LAMOL_{real}). Additionally, the models were required to assimilate all the knowledge of the previous three tasks and the Bool-Q task. Consequently, there might be insufficient model capacities to accommodate all the knowledge. On the other hand, LAMOL had already forgotten most of the knowledge learnt prior to the start of Bool-Q; therefore, there was higher model capacity left to accommodate the knowledge from the Bool-Q task. It is also noteworthy to mention that our LM+Adapter+RT showed the ability of backward knowledge transfer in the SciFact task, achieving higher accuracy than after it has just learnt the task.

5.2 Efficiency

We detailed the runtime and parameter counts of each method in Table 5.3. The runtime is calculated by averaging the runtime of all task permutations from Table 5.1. Despite 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

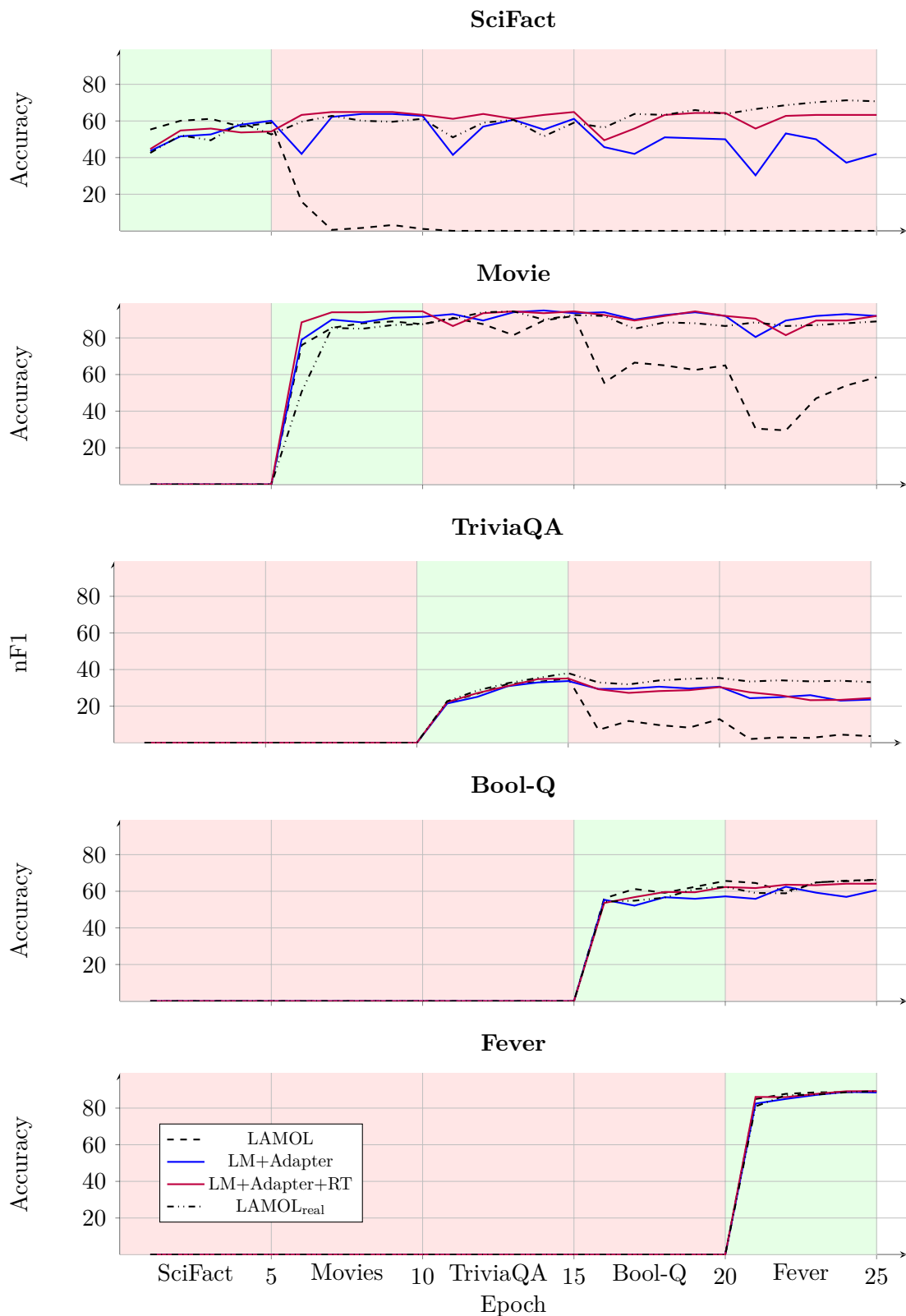


Figure 5.2: Learning curves of task order SMTBF. Each graph shows the performance at each epoch for each task. Green background refers to the epochs on which the model is first introduced with a particular task.

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.

5.3 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 5.4a. As a consequence, LAMOL was unable to effectively prevent catastrophic forgetting. Our framework increased the success rate of pseudo sample generation (Table 5.4b). 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 5.4b, 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 GPT-2 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 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 dataset 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, i.e., 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 samples is less destructive

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	1272.00	320.00	1272.00

(a) Pseudo sample analysis of LAMOL.

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	1272.00	320.00	1272.00

(b) 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	1272.00	320.00	1272.00

(c) 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	1272.00	320.00	1272.00

(d) 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	1272.00	320.00	1272.00

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

Table 5.4: Results of the pseudo sample analysis. The numbers indicate the amount of pseudo samples corresponding to each characteristics, averaged over three seeds.

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 5.2) when compared with shorter sequences (Table 5.1).

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 permutations. Therefore, future work should focus on minimizing the number of pseudo samples of this nature.

5.4 Ablation Study

Other Variations 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 other two different configurations of our proposed Double LM namely:

- (c+q/qa): LM_1 learns the context and the question parts, whereas LM_2 learns on 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.

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 5.5: The performance of other variations of our framework.

We performed the experiment on all permutations of the three tasks: BoolQ, Movie Reviews, and SciFact. The results are reported in Table 5.5. 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 the question generation process (step 2 of Figure 3.2) and the answer generation process (step 3 of Figure 3.2). 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 finetuning.

5.5 Discussion

Even though the proposed framework has shown impressive performance improvements over LAMOL in our experiments, it provides relatively small improvements when trained on datasets with short texts such as those in Sun, Ho, and Lee (2020). This is because LAMOL is already able to produce high-quality pseudo samples on these datasets. Hence, the Double LM framework would only introduce additional training time.

As an illustration, Table 5.6 shows the performance of LAMOL compared with our framework on one task sequence from the original LAMOL paper: SQuADv1 \rightarrow WikiSQL \rightarrow SST \rightarrow QA-SRL \rightarrow WOZ. For all methods, we trained each task for only five epochs.

Methods	Average Acc.
LAMOL	70.71
LM+Adapter	70.39
LM+Adapter+T	71.50

Table 5.6: The performance of different methods on task sequence: SQuADv1 \rightarrow WikiSQL \rightarrow SST \rightarrow QA-SRL \rightarrow WOZ. Note that the ReGen strategy was not required since there were virtually no uninformative pseudo samples present in the experiments.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we introduced Double LM, a lifelong learning framework that focuses on improving pseudo samples. Our framework utilizes two language models each specializing on certain part(s) of input. As a consequence, the framework enjoys higher pseudo sample quality which is crucial for good LL performance. Double LM excels on datasets with longer text while also is able to perform well on datasets with short text. In our experiments, Double LM was able to significantly outperform LAMOL in every task sequence while also rivalling LAMOL_{real} in some task permutations. We also successfully reduced the computational 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 LAMOL_{real}. Lastly, we provided an analysis of pseudo samples and their effects on final LL performance. Future work could build on our analysis to potentially create better pseudo-rehearsal based lifelong language learners.

6.2 Future work

For future work, we aim to enhance the impact of our framework on tasks with shorter texts. Additionally, by analysing how in some task permutations our framework managed to outperform LAMOL_{real} could provide deeper insights on generating pseudo samples with quality that rival real samples.

Bibliography

- McCloskey, Michael and Neal J Cohen (1989). “Catastrophic interference in connectionist networks: The sequential learning problem”. In: *Psychology of learning and motivation*. Vol. 24. Elsevier, pp. 109–165. DOI: [https://doi.org/10.1016/S0079-7421\(08\)60536-8](https://doi.org/10.1016/S0079-7421(08)60536-8). URL: <http://www.sciencedirect.com/science/article/pii/S0079742108605368>.
- MacKay, David J. C. (1992). “A Practical Bayesian Framework for Backpropagation Networks”. In: *Neural Computation* 4.3, pp. 448–472. DOI: 10.1162/neco.1992.4.3.448.
- Bengio, Y., Paolo Frasconi, and Patrice Simard (1993). “Problem of learning long-term dependencies in recurrent networks”. In: 1183–1188 vol.3. DOI: 10.1109/ICNN.1993.298725.
- Ring, Mark Bishop (1994). “Continual Learning in Reinforcement Environments”. UMI Order No. GAX95-06083. PhD thesis. USA.
- Ans, Bernard and Stéphane Rousset (1997). “Avoiding catastrophic forgetting by coupling two reverberating neural networks”. In: *Comptes Rendus de l’Académie des Sciences - Series III - Sciences de la Vie* 320, pp. 989–997. DOI: 10.1016/S0764-4469(97)82472-9.
- Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long Short-Term Memory”. In: *Neural Comput.* 9.8, 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: <https://doi.org/10.1162/neco.1997.9.8.1735>.
- 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*.
- Hinton, Geoffrey (2010). “A Practical Guide to Training Restricted Boltzmann Machines (Version 1)”. In: *Technical Report UTML TR 2010-003, University of Toronto* 9. DOI: 10.1007/978-3-642-35289-8_32.
- Bengio, Yoshua et al. (2013). “Generalized Denoising Auto-Encoders as Generative Models”. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*. NIPS’13. Lake Tahoe, Nevada: Curran Associates Inc., 899–907.

- Graves, Alex (2013). “Generating Sequences With Recurrent Neural Networks”. In: *CoRR* abs/1308.0850. arXiv: 1308.0850. URL: <http://arxiv.org/abs/1308.0850>.
- Cho, Kyunghyun et al. (2014). “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”. In: *CoRR* abs/1406.1078. arXiv: 1406.1078. URL: <http://arxiv.org/abs/1406.1078>.
- Chung, Junyoung et al. (2014). “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”. In: *arXiv e-prints*, arXiv:1412.3555, arXiv: 1412.3555. arXiv: 1412.3555 [cs.NE].
- Goodfellow, Ian J. et al. (2014). *Generative Adversarial Networks*. cite arxiv:1406.2661. URL: <http://arxiv.org/abs/1406.2661>.
- Srivastava, Nitish et al. (2014). “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56, pp. 1929–1958. URL: <http://jmlr.org/papers/v15/srivastava14a.html>.
- Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). “Sequence to Sequence Learning with Neural Networks”. In: *CoRR* abs/1409.3215. arXiv: 1409.3215. URL: <http://arxiv.org/abs/1409.3215>.
- Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2016). *Neural Machine Translation by Jointly Learning to Align and Translate*. arXiv: 1409.0473 [cs.CL].
- Rusu, Andrei A. et al. (2016). “Progressive Neural Networks”. In: *CoRR* abs/1606.04671. arXiv: 1606.04671. URL: <http://arxiv.org/abs/1606.04671>.
- Aljundi, Rahaf, Punarjay Chakravarty, and Tinne Tuytelaars (2017). “Expert Gate: Lifelong Learning with a Network of Experts”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7120–7129. DOI: 10.1109/CVPR.2017.753.
- Aljundi, Rahaf et al. (2017). “Memory Aware Synapses: Learning what (not) to forget”. In: *CoRR* abs/1711.09601. arXiv: 1711.09601. URL: <http://arxiv.org/abs/1711.09601>.
- Joshi, Mandar et al. (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*. Vancouver, Canada: Association for Computational Linguistics.
- Kirkpatrick, James et al. (2017a). “Overcoming catastrophic forgetting in neural networks”. In: *Proceedings of the National Academy of Sciences* 114.13, pp. 3521–3526. ISSN: 0027-8424. DOI: 10.1073/pnas.1611835114. eprint: <https://www.pnas.org/content/114/13/3521.full.pdf>. URL: <https://www.pnas.org/content/114/13/3521>.
- (2017b). “Overcoming catastrophic forgetting in neural networks”. In: *Proceedings of the National Academy of Sciences* 114.13, pp. 3521–3526. ISSN: 0027-8424.

- DOI: 10.1073/pnas.1611835114. eprint: <https://www.pnas.org/content/114/13/3521.full.pdf>. URL: <https://www.pnas.org/content/114/13/3521>.
- Laine, Samuli and Timo Aila (2017). “Temporal Ensembling for Semi-Supervised Learning”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. URL: <https://openreview.net/forum?id=BJ6o0fqge>.
- Merity, Stephen, Nitish Shirish Keskar, and Richard Socher (2017). “Regularizing and Optimizing LSTM Language Models”. In: *CoRR* abs/1708.02182. arXiv: 1708.02182. URL: <http://arxiv.org/abs/1708.02182>.
- Rebuffi, Sylvestre-Alvise et al. (2017). “iCaRL: Incremental Classifier and Representation Learning”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5533–5542. DOI: 10.1109/CVPR.2017.587.
- Shin, Hanul et al. (2017). “Continual Learning with Deep Generative Replay”. In: Vaswani, Ashish et al. (2017). “Attention Is All You Need”. In: *arXiv e-prints*, arXiv: 1706.03762, arXiv:1706.03762. arXiv: 1706.03762 [cs.CL].
- Vaswani, Ashish et al. (2017). “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS’17*. Long Beach, California, USA: Curran Associates Inc., 6000–6010. ISBN: 9781510860964.
- McCann, Bryan et al. (2018). “The Natural Language Decathlon: Multitask Learning as Question Answering”. In: *CoRR* abs/1806.08730. arXiv: 1806.08730. URL: <http://arxiv.org/abs/1806.08730>.
- Radford, Alec et al. (2018). “Improving Language Understanding by Generative Pre-Training”. In:
- Sprechmann, Pablo et al. (2018). “Memory-based Parameter Adaptation”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=rkf0vGbcW>.
- Thorne, James et al. (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)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 809–819. DOI: 10.18653/v1/N18-1074. URL: <https://aclanthology.org/N18-1074>.
- Chaudhry, Arslan et al. (2019). “On Tiny Episodic Memories in Continual Learning”. In: *arXiv: Learning*.
- Clark, Christopher et al. (2019). “BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions”. In: *NAACL*.
- Devlin, Jacob et al. (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, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. URL: <https://doi.org/10.18653/v1/n19-1423>.
- Greco, Claudio et al. (2019). “Psycholinguistics Meets Continual Learning: Measuring Catastrophic Forgetting in Visual Question Answering”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 3601–3605. DOI: 10.18653/v1/P19-1350. URL: <https://www.aclweb.org/anthology/P19-1350>.
- Houlsby, Neil et al. (2019). “Parameter-Efficient Transfer Learning for NLP”. In: *CoRR* abs/1902.00751. arXiv: 1902.00751. URL: <http://arxiv.org/abs/1902.00751>.
- Liu, Yinhan et al. (2019). “RoBERTa: A Robustly Optimized BERT Pretraining Approach”. In: *CoRR* abs/1907.11692. arXiv: 1907.11692. URL: <http://arxiv.org/abs/1907.11692>.
- Masson d’Autume, Cyprien de et al. (2019). “Episodic Memory in Lifelong Language Learning”. In: *CoRR* abs/1906.01076. arXiv: 1906.01076. URL: <http://arxiv.org/abs/1906.01076>.
- Papamakarios, George et al. (2019). “Normalizing Flows for Probabilistic Modeling and Inference”. In:
- Radford, Alec et al. (2019a). *Better Language Models and Their Implications*. URL: <https://openai.com/blog/better-language-models/>.
- Radford, Alec et al. (2019b). “Language Models are Unsupervised Multitask Learners”. In:
- Radford, Alec et al. (2019c). “Language models are unsupervised multitask learners”. In: *OpenAI blog* 1.8, p. 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, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. URL: <https://openreview.net/forum?id=ByeSdsC9Km>.
- Toneva, Mariya et al. (2019). “An Empirical Study of Example Forgetting during Deep Neural Network Learning”. In: *ICLR*.
- Vig, Jesse (2019). “A Multiscale Visualization of Attention in the Transformer Model”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Florence, Italy: Association for

- Computational Linguistics, pp. 37–42. DOI: 10.18653/v1/P19-3007. URL: <https://www.aclweb.org/anthology/P19-3007>.
- Chang, Shiyu et al. (2020). “Invariant Rationalization”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 1448–1458. URL: <http://proceedings.mlr.press/v119/chang20c.html>.
- DeYoung, Jay et al. (2020). “ERASER: A Benchmark to Evaluate Rationalized NLP Models”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 4443–4458. DOI: 10.18653/v1/2020.acl-main.408. URL: <https://www.aclweb.org/anthology/2020.acl-main.408>.
- Han, Xu et al. (2020). “Continual Relation Learning via Episodic Memory Activation and Reconsolidation”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 6429–6440. DOI: 10.18653/v1/2020.acl-main.573. URL: <https://aclanthology.org/2020.acl-main.573>.
- Hoover, Benjamin, Hendrik Strobelt, and Sebastian Gehrmann (2020). “exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, pp. 187–196. URL: <https://www.aclweb.org/anthology/2020.acl-demos.22>.
- Nguyen, Giang et al. (2020). “Dissecting Catastrophic Forgetting in Continual Learning by Deep Visualization”. In: *CoRR* abs/2001.01578. arXiv: 2001.01578. URL: <http://arxiv.org/abs/2001.01578>.
- Pfeiffer, Jonas et al. (2020). “MAD-X: An Adapter-based Framework for Multi-task Cross-lingual Transfer”. In: *CoRR* abs/2005.00052. arXiv: 2005.00052. URL: <https://arxiv.org/abs/2005.00052>.
- Pomponi, Jary, Simone Scardapane, and Aurelio Uncini (2020). “Pseudo-Rehearsal for Continual Learning with Normalizing Flows”. In: *ArXiv* abs/2007.02443.
- Silver, Daniel and Sazia Mahfuz (2020). “Generating Accurate Pseudo Examples for Continual Learning”. In: pp. 1035–1042. DOI: 10.1109/CVPRW50498.2020.00136.
- Sodhani, Shagun, Sarath Chandar, and Yoshua Bengio (2020). “Toward Training Recurrent Neural Networks for Lifelong Learning”. In: *Neural computation* 32.1, 1–35. ISSN: 0899-7667. DOI: 10.1162/neco_a_01246. URL: https://doi.org/10.1162/neco_a_01246.

- Sun, Fan-Keng, Cheng-Hao Ho, and Hung-Yi Lee (2020). “LAMOL: LAnguage MOdeling for Lifelong Language Learning”. In: *International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=Skgxcn4YDS>.
- Sun, Jingyuan et al. (2020). “Distill and Replay for Continual Language Learning”. In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 3569–3579. DOI: 10.18653/v1/2020.coling-main.318. URL: <https://aclanthology.org/2020.coling-main.318>.
- Wadden, David et al. (2020). “Fact or Fiction: Verifying Scientific Claims”. In: *EMNLP*.
- Wang, Zirui et al. (2020). “Efficient Meta Lifelong-Learning with Limited Memory”. In: pp. 535–548. DOI: 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: *CoRR* abs/2002.06715. arXiv: 2002.06715. URL: <https://arxiv.org/abs/2002.06715>.
- Pelosi, Francesco and Andrea Torsello (2021). “More Is Better: An Analysis of Instance Quantity/Quality Trade-off in Rehearsal-based Continual Learning”. In: *CoRR* abs/2105.14106. arXiv: 2105.14106. URL: <https://arxiv.org/abs/2105.14106>.
- Solinas., M. et al. (2021). “Beneficial Effect of Combined Replay for Continual Learning”. In: *Proceedings of the 13th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART, INSTICC*. SciTePress, pp. 205–217. ISBN: 978-989-758-484-8. DOI: 10.5220/0010251202050217.

A Rational LAMOL

As mentioned in the introduction, Rational LAMOL was a collaboration with a fellow graduate student. In the spirit of the requirement for a thesis, only the new Double LM framework is described in the main text while the Rational LAMOL framework is described here in the appendix.

Rational LAMOL, illustrated in Figure 1 (right), augments the original methodologies of LAMOL by freezing a part of the model where most knowledge is lost when training on a new task. We believe that by preventing the loss of knowledge, the model can better retain its performance on the previous tasks while reusing the knowledge to help learn the new task. The part to be frozen is identified by a novel rationale-based algorithm called Critical Component Identification (CCI). The CCI algorithm finds the critical component by comparing the attention scores of each layer of the model with the ground truth rationales.

Our experiments are conducted in an LL setting in which a model is trained on a stream of task $\tau = \{\tau_1, \tau_2, \dots, \tau_i, \dots\}$ where τ_i is the i -th task to train at time step i . Let M_i denote the model M after being trained for task i , where M_0 is the initialized pre-trained model. Our Rational LAMOL follows the following process. First, using LAMOL’s training procedure, the model M_i is optimized on the task τ_{i+1} to obtain \hat{M}_{i+1} . Second, when $i > 0$, the proposed critical component identification algorithm, which is described in Section A.1, is applied to both M_i and \hat{M}_{i+1} by using the rationales of task τ_i to identify the block most susceptible to forgetting. Finally, by applying freezing to the most plastic block identified by our algorithm, we train M_i^{CF} on the task t_{i+1} again to get a new model M_{i+1} which now retains the most plastic knowledge.

A.1 Critical Component Identification (CCI)

Inspired by Nguyen et al. (2020) that leverages Explainable AI to dissect a CNN model of the most plastic blocks via `Auto DeepVis`, we create an algorithm that is able to identify the most plastic component similarly in transformer-based models. The selected component is then kept frozen to prevent loss of knowledge. Nevertheless, we cannot directly adapt `Auto DeepVis` to the NLP context. Certain discrepancies need to be rectified. Specifically, the lack of ground truth semantic segmentation labels and the different semantic values of hidden state visualization.

`Auto DeepVis` was devised to automatically select the most plastic blocks to be subjected to freezing. The main idea is to first identify representative maps, which are feature maps that most resemble the semantic segmentation ground truth, in each block of both the model before and after training on incoming data. The

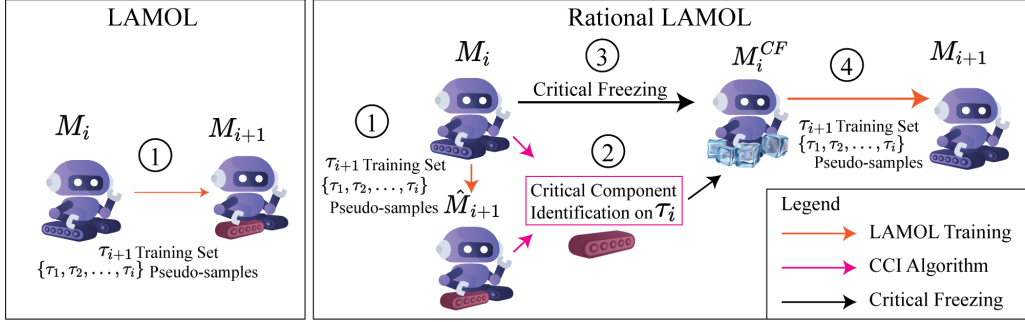


Figure 1: **Left:** The overview of LAMOL. **Right:** The overview of Rational LAMOL, our proposed framework that aims to alleviate catastrophic forgetting by freezing the critical component.

forgetting effect is measured by the drop in Intersection over Union (IoU) values between the two representative maps.

Representative maps are automatically dissected through the resemblance of feature maps to the ground truth semantic segmentation labels. Diverse features of objects are captured by feature maps isolatedly throughout the channels, which these semantic segmentation labels are deemed apropos for the distinctive features containing positive evidence for prediction.

However, in the NLP context, the field of interpretable AI currently is in its budding and does not yet possess such visualization tools (DeYoung et al. 2020). There are certain methods that can visualize the self-attention mechanism (Vig 2019; Hoover, Strobel, and Gehrmann 2020), illustrating the ability of the mechanism to relate tokens from different positions to form a representation of the sequence. Similarly, attention heads have also been shown to exhibit behaviour related to the syntactic and semantic structure of sentences (Vaswani et al. 2017). Therefore, we believe that the self-attention mechanism would naturally attend to tokens that represent positive evidence crucial for predictions. This is analogous to rationales–snippets that support outputs.

The Critical Component Identification used in our thesis is depicted in Algorithm 1. Note that we preserve most of the variable names according to Nguyen et al. (2020) for ease of reference. Each validation sample is passed through the old and new models M_O and M_N respectively. We find these representative maps of each transformer block by running through the attention scores AT of each block j .

Given a model M and the ground truth segmentation GT , the model would output an array with the shape of (12 blocks, 12 heads, 1024 tokens, 1024 tokens). Specifically, a single Transformer block j outputs $|A|$ attention heads, with each attention head consisting of $|S|$ tokens, and each token containing attention scores relating to all other $|S|$ tokens.

Algorithm 1 Critical Component Identification

Input: Validation set X , ground truth GT , old model M_O , new model M_N , number of blocks K

Output: Critical block \mathbb{F}

$\mathbb{L} \leftarrow \emptyset$

for all validation sample $\{x_i\}_{i=1}^{|X|} \in X$ **do:**

IoUs $\leftarrow \emptyset$

$AT_O, AT_N \leftarrow [M_O(x_i), M_N(x_i)]$

for $j = 1, K$ **do:**

$RM_{M_O, GT} \leftarrow$

AT_{j, a^*, s^*} with highest $IoU_{M_O, GT}$

$RM_{M_N, M_O} \leftarrow$

AT_{j, a^*, s^*} with highest IoU_{M_N, M_O}

APPEND(IoUs, $\max(IoU_{M_N, M_O})$)

end for

$b \leftarrow$ block index with highest drop in IoUs

APPEND(\mathbb{L} , b)

end for

$\mathbb{F} = \text{MODE}(\mathbb{L})$

return \mathbb{F}

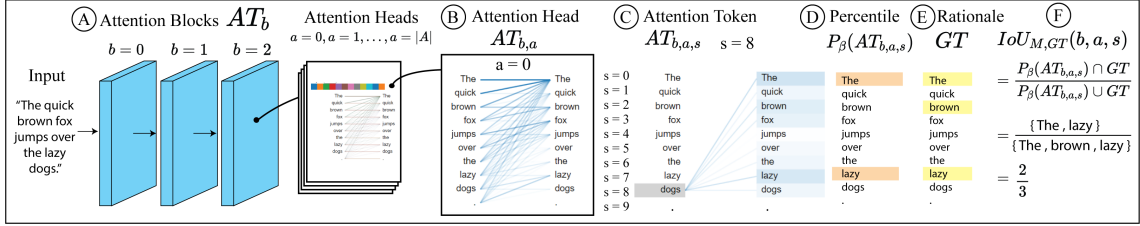


Figure 2: Schematic illustration of the calculation of $IoU_{M,GT}$. **A**: The input is fed through each attention block AT_b , where each block b has multiple heads. **B**: A single attention head $AT_{b,a}$ consists of the attention of the sequence in relation to all other tokens, as shown in **C**. Finally, the IoU calculation **F** is applied on the hard selection of attention token with percentiles **D** and the rationale ground truth in **E**.

For each block j , the algorithm iterates over all heads and all tokens and calculates the IoU similarity of the attention score of each token in relation to all other tokens with the ground truth. For instance, the IoU of the s token in the a attention head of the j Transformer block is computed as:

$$IoU_{M,GT}(j, a, s) = \frac{P_\beta(AT_{j,a,s}) \cap GT}{P_\beta(AT_{j,a,s}) \cup GT} \quad (1)$$

where P_β denotes a simple binary thresholding function with the threshold being at the β -th percentile of the entire sequence.

Since IoU calculations require a comparison of two binary masks, the thresholding function is required as an important design choice due to the soft scores in the self-attention mechanism.

Correspondingly, the representative map (RM) is the attention output of the token with the highest IoU between its attention scores to other tokens and the ground truth rationale. For a block index j , this can be computed as:

$$(a^*, s^*) = \arg \max_{a \in A, s \in S} (IoU_{M,GT}(j, a, s)) \quad (2)$$

$$RM_{M,GT}(j) = AT_{j,a^*,s^*}$$

Note that a^* and s^* are the attention head index and token index respectively that receive the highest $IoU_{M,GT}$ value. The value of $RM_{M,GT}$ is the attention value that is extracted from the attention output AT on block index j , attention head index a^* , and token index s^* . Similarly, IoU_{M_N, M_O} is identical to Equation 1, replacing GT with $P_\beta(RM_{M_O,GT})$. The binary thresholding function P_β is applied whenever there are non-binary inputs. RM_{M_N, M_O} is identical to Equation 2, replacing the IoU value with the one we just computed. The IoU calculation in the representative maps selection process is illustrated in Figure 2 for more clarity.

In addition, since each transformer block is composed of multiple attention heads (Vaswani et al. 2017), it is possible for us to freeze individual heads separately.

Freezing in this manner means we can control the loss of knowledge with finer granularity. Therefore, we propose another algorithm that can be instead applied to transformer heads. This is similar to Algorithm 1, but instead of just iterating through each block, the algorithm will go through each attention head (in all blocks) and identify those that are most prone to forgetting. Although the calculation of IoU will be the same, the definition of the representative map will be at a higher granularity. For a block index j and attention head a , $RM_{M,GT}$ will be computed as:

$$(s^*) = \arg \max_{s \in S} (IoU_{M,GT}(j, a, s)) \quad (3)$$

$$RM_{M,GT}(j, a) = AT_{j,a,s^*}$$

A.2 Unsupervised Rationale Generation

As mentioned in Section A, rationales are a crucial requirement for Rational LAMOL. However, existing NLP datasets usually possess only labels but not rationales. Therefore, we leverage a recent unsupervised rationale generation framework, InvRat (Chang et al. 2020), to automatically generate rationales from any dataset as substitutions. However, InvRat was originally designed for single-input tasks and most of the datasets used in our experiments are multi-input tasks such as text-pair classification. To amend the issue, we additionally append the query (or question) at the end of each sample in order to accommodate these tasks.

B Results of Rational LAMOL

To validate our hypothesis that freezing does help retain knowledge important to previous tasks, we conducted a partial brute force block-level freezing as the upper bound of our Rationale LAMOL_{block} on each task permutation. Due to the limited computation resources, we performed the brute force in a partial fashion, searching only on the even-numbered block. The result is presented in Table 1. Brute Force outperformed LAMOL by a large margin of 9.56%, confirming our hypothesis that freezing does help reduce catastrophic forgetting.

Using our CCI algorithm to identify the critical component, Rational LAMOL_{block} was also able to outperform LAMOL by 7.28%. Similarly, Rational LAMOL_{head} was also able to achieve comparable performance with the block-level variant.

Finally, to our surprise, using generated rationale instead of human rationales in CCI, Gen R-LAMOL_{block} was able to further improve the performance by an additional of 0.84% upon R-LAMOL_{block}.

Although all variants of Rational LAMOL were able to improve on the baseline, they only slightly outperformed LAMOL_{all}. This indicates that pseudo samples

Methods	BMS	BSM	MBS	MSB	SBM	SMB	Average	Std.
<i>Baselines. Section 2.7 & Section 5.1.1</i>								
LAMOL	64.53	35.48	66.79	60.76	52.02	54.40	55.67	11.41
LAMOL _{all}	62.22	62.06	61.42	52.93	65.32	63.35	61.22	4.29
<i>Rational LAMOL. Section A</i>								
Partial Brute Force _{block}	67.01	62.90	66.20	63.31	68.75	63.20	65.23	2.44
R-LAMOL _{block}	67.40	62.34	65.31	56.55	58.69	67.40	62.95	4.57
R-LAMOL _{head}	67.25	63.51	64.29	61.15	59.54	51.16	61.15	5.57
Gen R-LAMOL _{block}	64.39	65.63	65.68	57.65	63.30	66.39	63.84	3.22
Gen R-LAMOL _{head}	63.45	66.98	65.93	56.39	63.37	63.43	63.26	3.69
<i>Double LM Framework. Section 3.1-3.1.1</i>								
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
<i>With Additional Pseudo Sample Enhancement. Section 3.3</i>								
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 _{real}	69.07	71.97	70.84	72.31	74.13	73.32	71.94	1.80
Multitask	75.52							

Table 1: 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-LAMOL and Gen R-LAMOL refer to Rational LAMOL and Generated Rational LAMOL, respectively.

may play a more critical role in preventing catastrophic forgetting. Overall, the results are actually consistent with the results from Sun, Ho, and Lee (2020) where regularization-base approaches such as EWC and MAS provide only marginal improvements to LAMOL since R-LAMOL can also be seen as a regularization-base approach.