Semi-supervised Continual Learning with Meta Self-training and Consistency Regularization

Anonymous ACL submission

Abstract

Recent advances in continual learning (CL) are mainly confined to a supervised learning setting, which is often impractical. To narrow this gap, we consider a semi-supervised continual learning (SSCL) for lifelong language learning. In this paper, we exploit unlabeled data under limited supervision in the CL setting and demonstrate the feasibility of semi-supervised learning in CL. Specifically, we propose a novel method, namely Meta-Aug, which employs meta self-training and consistency regularization to learn a sequence of semi-supervised tasks. We employ prototypical pseudo-labeling and data augmentation to efficiently learn under limited supervision without catastrophic forgetting. Furthermore, replay-based CL methods easily overfit to memory samples. We solve this problem by applying strong textual augmentation to introduce generalization. Extensive experiments on CL benchmark text classification datasets from diverse domains show that our method achieves promising results in SSCL.

1 Introduction

Continual learning (CL), also called lifelong learning, is a machine learning paradigm that mimics the human learning process. It aims to ensure the *stability* of handling various tasks that have been learned, while showing its *plasticity* on the novel domain via previously acquired knowledge. Recent advances in CL lack consideration of real-world scenarios. In real-world scenarios, the availability of data is limited, where unlabeled data are plentiful and acquiring high-quality labels are expensive. However, semi-supervised continual learning (SSCL) remains understudied. The difficulty is that, forgetting or loss of information always occurs while ingesting a sequence of data, not to mention that the given information is inherently limited. To date, there are not many SSCL models in text classification, or even in NLP.

In this paper, we propose a novel SSCL method with meta self-training and consistency regularization, namely Meta-Aug. We leverage Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017) framework. We employ inner loop algorithm to perform task-specific learning on new samples. In order to preserve prior knowledge, we need to control the change in parameter space. Given outer loop algorithm governs inner loop learning process, we revise the outer loop objective to generalize all seen samples. Specifically, query instances for the outer loop are selected from seen samples by a prototypical network (Snell et al., 2017). We also use prototypical network for pseudo-labeling to alleviate prediction error. To effectively use unlabeled data, we leverage consistency regularization (Bachman et al., 2014). It aims to reach consistency on model outputs when fed perturbed versions of the same text. Inspired by FixMatch (Sohn et al., 2020), we apply two types of textual augmentations, i.e., weak and strong augmentation, to generate perturbed texts. Furthermore, CL setup constrains the amount of prior seen samples saved in memory. As a result, revisiting memory samples (i.e., experience replay) for knowledge consolidation can easily cause overfitting problem. To solve this problem, we introduce generalization by applying strong augmentation on past examples in meta-objective.

We conduct extensive experiments on CL benchmark datasets from Zhang et al. (2015), popularized by de Masson d'Autume et al. (2019) in lifelong language learning. This collection of datasets includes news classification, sentiment analysis, article classification and questions and answers categorization. Under the limited availability of training samples of all tasks, we show that Meta-Aug effectively uses unlabeled examples and provides more than 50% improvement in accuracy. Meta-Aug also shows its robustness to catastrophic forgetting. The average performance gap between our

method and upper bound of CL performance is less than 4%. It prevents more than 45% forgetting, via unannotated information. The proposed method successfully maintains a more than 53% average accuracy given an extremely limited amount of annotated data. Additionally, we report an ablation study and further analysis to testify the superiority of our method.

The contributions of this work are four folds:

- To the best of our knowledge, we are the first paper to address semi-supervised continual learning via textual augmentations based on the assumption of consistency regularization. We show the feasibility of using such an assumption for NLP tasks by textual augmentations.
- We use strongly-augmented samples to address overfitting problem that commonly existed in CL methods that involves revisiting prior seen examples .
- We devise a simple but efficient prototypical pseudo-labeling method to decrease prediction error and improve model performance.
- Extensive experimental results testify the superiority of our method as a promising solution to address semi-supervised continual learning.

2 Related Work

Meta-learning in continual learning. Recently, meta-learning has been introduced into CL models, considering its ability of fast adaptation and knowledge transfer. Recent works employed MAML (Finn et al., 2017) to improve initial parameters, such that it can fast adapt to various domains with few learning samples. Meta-MbPA (Wang et al., 2020) performed local adaptation with episodic memory, which used MAML to find a better initialized state for local adaptation. OML-ER (Holla et al., 2020) and ANML-ER (Holla et al., 2020) utilised an online meta-learning model and a neuromodulated meta-learning respectively for fast adaptation, augmented with sparse experience replay. Some CL models used Reptile (Nichol et al., 2018) as their meta-learning algorithms. MER (Riemer et al., 2019) regularized the objective of experience replay via a modified Reptile (Nichol et al., 2018) algorithm and memory replay module. MLLRE (Obamuyide and Vlachos, 2019) also adopted Reptile to meta updates parameters via augmented training set. In the field of computer vision, MERLIN (Joseph and Balasubramanian, 2020) used preceding task-specific priors from meta distribution to replay previous parameters and consolidate the CL model. MOML (Acar et al., 2021) introduced quadratic penalty to debias and regularized loss of a meta model, such that it could bypass the need to recall prior seen instances. All these methods are limited to a supervised continual learning setting.

Data augmentation in continual learning. Although the literature on data augmentation is rich, data augmentation for continual learning is still at its early stage. IL2A (Zhu et al., 2021) leveraged a modified version of label mixing based method, Mixup (Zhang et al., 2018) for continual representation learning, but IL2A is for images but not for texts.

Semi-supervised continual learning. Wang et al. (2021) stated that existing CL strategies are not suitable for the semi-supervised scenario. They defined the challenge as *catastrophic forgetting of unlabeled data*, in which the underlying distribution of unannotated data can not be effectively characterized. ORDisCo (Wang et al., 2021) used a conditional generative adversarial network to exploit unlabeled data and selectively stabilized parameters for discriminative learning. ORDisCo is also for images but not for texts. To date, semi-supervised continual learning is still regarded as a challenging but understudied setting.

3 Problem Formulation

We define semi-supervised continual learning by assuming a sequence of K semi-supervised tasks $\{\mathcal{T}^{(1)}, \mathcal{T}^{(2)}, ..., \mathcal{T}^{(K)}\}$. For a task $\mathcal{T}^{(k)}$, let $\mathcal{X}^{(k)} = \{(x_i, y_i)\}_{i=1}^{N^{(k)}}$ be a set of $N^{(k)}$ labeled instances, where y is the ground-truth label of input x. Let $\mathcal{U}^{(k)} = \{(u_j, \hat{y}_j)\}_{j=1}^{M^{(k)}}$ be a set of $M^{(k)}$ unlabeled instances, where $\hat{y}$ is the pseudo-label of unannotated input u predicted by model. Note that $M^{(k)} >> N^{(k)}$. In general, CL setting has a memory constraint, B, which refers to the maximum amount of data allowed to be stored. The goal is to learn a consistent model f_θ for all seen tasks.

Class-incremental learning. In this paper, we consider a popular scenario of continual learning, i.e., *class-incremental learning* (CIL), where task

identity information is not provided in inference, and a single classifier is for all tasks. We leverage cross-entropy loss $\mathcal{L}_{CE}$ as the classification loss.

4 Approach

4.1 Architecture

The proposed model f_θ consists of a representation learning network (RLN), $h_{\phi_{\text{proto}}}$ with learnable parameters ϕ_{proto}, and a prediction network (PN), $g_{\phi_{\text{pred}}}$ with learnable parameters ϕ_{pred}. It is described as $f_\theta(x) = g_{\phi_{\text{pred}}}(h_{\phi_{\text{proto}}}(x))$. We add a single-hidden-layer feed-forward neural network on top of an encoder to formulate a prototypical network as RLN and use a single linear layer followed by a softmax as PN.

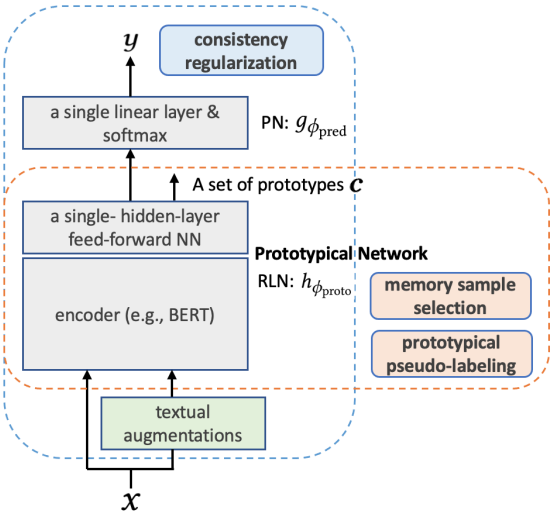

Figure 1: The overall architecture of the proposed method.

4.2 Prototypical Network

The prototypical network can be conceptually divided into two parts: an encoder that maps input texts to N-dimensional representations and a single-hidden-layer feed-forward neural network that learns a mapping: $\mathbb{R}^N \rightarrow \mathbb{R}^D$, where $D < N$. We aim to obtain a D-dimensional feature vector, $\boldsymbol{c} \in \mathbb{R}^D$, as a prototype for each class.

Prototypes computation and memory sample selection. We split the network training into two stages: supervised learning and unsupervised learning. In supervised learning, each prototype is the mean vector of training examples with the same label. Then, we select the top N_{select} samples with the highest similarity score to the corresponding prototype and save them in memory. We calculate the similarity score via the Euclidean distance. In unsupervised learning, we obtain the top N_{select} samples for each class from the current pseudo-labeled training set and memory set based on similarity scores. Then, we leverage the newly-selected examples to update the prototype for each class. Note that the label of a selected example may be its ground-truth label or a pseudo-label. The prototypes are constantly updated. We only keep the latest set of selected samples in memory, denoted as $\mathcal{M}$.

Prototypical pseudo-labeling. Typical approaches for pseudo-labeling in semi-supervised learning include self-training with the current model, co-training with a similar model or applying graph propagation. We consider two pseudo-labeling strategies, i.e., pseudo-labeling with model prediction under self-training and pseudo-labeling via prototypes. However, since the initialized model is weakly performed, the prediction errors for self-training can be accumulated in the continual learning process. Thus, we will use the second strategy, i.e. prototypical pseudo-labeling. Particularly, it compares the embedding of u with all up-to-date prototypes. The label of the prototype with the closest distance to the embedding of u is its pseudo-label, namely $\hat{y}_{\text{proto}}$. Since prototypes contain more feature representation information, we apply prototypical pseudo-labeling for task-specific learning. Specifically, we expect the model to output predictions similar to those of the prototypes. Thereby, we minimise the cross-entropy loss, i.e., $\mathcal{L}_{CE}(f_\theta(u), \hat{y}_{\text{proto}})$.

4.3 Consistency Regularization

We apply the assumption in *consistency regularization* (Bachman et al., 2014) to the output of our model f_θ. Consistency regularization is widely applied for recent state-of-the-art SSL algorithms in computer vision. It is hinged on the assumption that the model should produce similar predictions when fed perturbed versions of the same image. FixMatch (Sohn et al., 2020) leverages a standard flip-and-shift as a weak augmentation strategy and AutoAugment (Cubuk et al., 2019) as a strong augmentation strategy to form two perturbed versions of an image for semi-supervised learning. Inspired by FixMatch, we employ two types of textual augmentations, strong and weak, denoted by $\mathcal{A}(\cdot)$ and $\alpha(\cdot)$ respectively, as the perturbed versions for an

Algorithm 1: Meta Training

Input: Initial parameters $\theta = \phi_{\text{proto}} \cup \phi_{\text{pred}}$, training set $D_{\text{train}} = \mathcal{X} \cup \mathcal{U}$, query set Q, memory buffer $\mathcal{M}$, inner-loop learning rate α, outer-loop learning rate β, and No. of saved data per class N_{select}.
Output: Trained parameters $\boldsymbol{\theta}$ and Memory $\mathcal{M}$

for $i = 1, 2, ...$ **do**
  Receiving m batches of examples, D^i_{train}, from the stream
  [Inner Loop]
  if $D^i_{train} \subseteq \mathcal{X}$ **then**
    Perform SGD on ϕ_{pred} to minimize Eqn.4
    Perform **prototypes computation**.
  else if $D^i_{train} \subseteq \mathcal{U}$ **then**
    Perform **pseudo-labeling via prototypes**.
    Perform SGD on ϕ_{pred} to minimize Eqn.6.
  end
  [Memory Sample Selection]
  Select N_s nearest examples to each prototype $\boldsymbol{c}$ from $D^i_{train} \cup \mathcal{M}$ for each class.
  Update prototypes via selected examples.
  Update $\mathcal{M}$ with newly selected examples.
  [Outer Loop]
  Read **ALL** examples from $\mathcal{M}$ as Q^i.
  Perform Adam update on θ to minimize Eqn.9
  if all training data are seen **then**
    Stop Iteration
  end
end

unlabeled text. In particular, we swap words randomly as the weak augmentation. We apply the combination of swapping word randomly, deleting word randomly and substituting word by WordNet's synonym, as the strong augmentation.

Let $p(y|x)$ be the predicted class distribution output for input x. The consistency regularization loss applied to labeled instances is,

$$\sum_{(x,y)\in\mathcal{X}} \big\|p(y|\alpha(x)) - p(y|x)\big\|_2^2 \tag{1}$$

And for unlabeled instances is,

$$\sum_{(u,\hat{y})\in\mathcal{U}} \big\|p(\hat{y}|\mathcal{A}(u)) - p(\hat{y}|\alpha(u))\big\|_2^2 \tag{2}$$

where $\hat{y}$ denotes pseudo-labeling output by model f_θ.

4.4 Meta Training

We employ FOMAML (Finn et al., 2017) as our learning framework, which consists an inner loop algorithm for task-specific learning and an outer loop algorithm for decision making on all seen tasks. Algorithm 1 shows the training steps in details.

Inner loop. Inner loop algorithm performs task-specific learning of current task $\mathcal{T}^{(k)}$, where $k \in \{1, ..., K\}$. It also includes consistency regularization with data augmentations. The inner loop loss for *labeled* instances is,

$$\mathcal{L}^{\mathcal{X}^{(k)}}_{\textbf{inner}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{X}^{(k)}}[\mathcal{L}_{CE}(f_\theta(x), y)] + \sum_{(x,y)\in\mathcal{X}^{(k)}} \big\|p(y|\alpha(x)) - p(y|x)\big\|_2^2 \tag{3}$$

where $\mathcal{L}_{CE}$ is a loss function (i.e., cross-entropy loss in this paper) and $\theta = \phi_{\text{proto}} \cup \phi_{\text{pred}}$. We transform Eqn.3 into,

$$\mathcal{L}^{\mathcal{X}^{(k)}}_{\textbf{inner}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{X}^{(k)}}\big[\mathcal{L}_{CE}(f_\theta(x), y) + \mathcal{L}_{CE}(f_\theta(\alpha(x)), y)\big] \tag{4}$$

The inner loop loss for *unlabeled* instances,

$$\mathcal{L}^{\mathcal{U}^{(k)}}_{\textbf{inner}}(\theta) = \mathbb{E}_{(u,\hat{y}_{\text{proto}})\sim\mathcal{U}^{(k)}}[\mathcal{L}_{CE}(f_\theta(u), \hat{y}_{\text{proto}})] + \sum_{(u,\hat{y})\in\mathcal{U}^{(k)}} \big\|p(\hat{y}|\mathcal{A}(u)) - p(\hat{y}|\alpha(u))\big\|_2^2 \tag{5}$$

Similarly, we transform Eqn.5 into,

$$\mathcal{L}^{\mathcal{U}^{(k)}}_{\textbf{inner}}(\theta) = \mathbb{E}_{(u,\hat{y}_{\text{proto}})\sim\mathcal{U}^{(k)}}[\mathcal{L}_{CE}(f_\theta(u), \hat{y}_{\text{proto}})] + \mathbb{E}_{(\alpha(u),\hat{y})\sim\mathcal{U}^{(k)}}[\mathcal{L}_{CE}(f_\theta(\mathcal{A}(u)), \hat{y})] \tag{6}$$

where $\hat{y}$ is the model prediction of unlabeled data $\alpha(u)$. In inner loop optimization, MAML performs SGD on parameters ϕ_{pred} with learning rate α as,

$$\phi^*_{\text{pred}} = \phi_{\text{pred}} - \alpha\nabla_\theta\mathcal{L}_{\textbf{inner}}(\theta) \tag{7}$$

Outer loop. In outer loop algorithm, we read all examples from $\mathcal{M}$ as query set, Q, where $\mathcal{M}$ contains all representative samples from all seen classes chosen by prototypes. The outer-loop objective is to have $f_{\theta'}(x) = g_{\phi^*_{\text{pred}}}(h_{\phi_{\text{proto}}}(x))$ generalize well across all seen tasks from a distribution $p(\mathcal{T})$. That is, minimizing the expected risk as,

$$\mathcal{L}^{Q}_{\textbf{meta}}(\theta') = \sum_{\mathcal{T}^{(k)}\sim p(\mathcal{T})} \mathbb{E}_{(x,y)\sim p(\mathcal{T}^{(k)})}[\mathcal{L}_{CE}(f_{\theta'}(x), y)] \approx \mathbb{E}_{(x,y')\sim Q}[\mathcal{L}_{CE}(g_{\phi^*_{\text{pred}}}(h_{\phi_{\text{proto}}}(x)), y')] \tag{8}$$

where $\mathcal{L}_{\textbf{meta}}$ is the meta loss and y' denotes the ground-truth label or pseudo-label of an example

x from memory set $\mathcal{M}$. However, in such a way, model tend to overfit examples from query set Q. Therefore, we introduce *generalization via strong augmentation* and modify meta objective as

$$\mathcal{L}^{Q}_{\mathbf{meta}}(\theta') = \mathbb{E}_{(x,y')\sim Q}[\mathcal{L}_{CE}(g_{\phi^*_{\text{pred}}}(h_{\phi_{\text{proto}}}(\mathcal{A}(x)), y')] \tag{9}$$

We use Adam (Kingma and Ba, 2015) as our outer loop optimizer with learning rate β as,

$$\theta \leftarrow \theta - \beta \nabla_{\theta'} \mathcal{L}^{Q}_{\mathbf{meta}}(\theta') \tag{10}$$

where $\theta' = \phi_{\text{proto}} \cup \phi^*_{\text{pred}}$.

4.5 Meta-inference

In inference, we randomly sample m batches of examples drawn from $\mathcal{M}$ and perform inner-loop optimization on these samples to finetune parameters ϕ_{pred}. Note that the inner loop algorithm in meta-inference differs from that in meta-training. The inner loop loss in test is formulated as

$$\mathcal{L}^{\mathcal{M}}_{\mathbf{inner}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{M}}[\mathcal{L}_{CE}(f_\theta(x), y)] \tag{11}$$

We use SGD as the inner-loop optimizer with a learning rate of α. Then, we predicts on the test set using $f_{\theta'}$ where $\phi_{\text{proto}} \cup \phi^*_{\text{pred}}$.

5 Experiments

5.1 Experimental Setup

Baselines. To date, not many SSCL models are available for comparisons, especially in text classification. Hence, we compare our method against the following baselines.

- **Purely supervised continual learning** (PSCL): We train our model on a sequence of datasets, which only contains a limited amount of labeled data. Unlabeled examples are not provided. We consider PSCL as the *lower bound* of model performance.
- **Semi-supervised multi-task learning** (SSMTL): We train our model on all datasets jointly but under semi-supervision. In SSCL setup, we consider SS-MTL as the *upper bound* of CL performance.
- **Fully supervised continual learning** (FSCL): We train our model on a sequence of datasets under full supervision. In SSCL setup, we consider FSCL as the *upper bound* of SSL performance

Dataset Orders
(1) Yelp → AGNews → DBpedia → Amazon → Yahoo
(2) DBpedia → Yahoo → AGNews → Amazon → Yelp
(3) Yelp → Yahoo → Amazon → DBpedia → AGNews
(4) AGNews → Yelp → Amazon → Yahoo → DBpedia

Table 1: Input Datasets Orders

Datasets. We use the collection of text classification datasets from Zhang et al. (2015)[1], including AGNews (news classification; 4 classes), Yelp (sentiment analysis; 5 classes), Amazon (sentiment analysis; 5 classes), DBpedia (Wikipedia article classification; 14 classes) and Yahoo (questions and answers categorization; 10 classes). Following prior work, we use the balanced version of the collection and merge the classes of Yelp and Amazon. Thus, we have 33 classes in total. In this paper, we randomly sample *11,500* training examples and 7,600 test examples from each of the datasets. Each dataset is seen as a separate semi-supervised learning task. In our experiments, we concatenate training sets in four different orderings as shown in Table 1 and make only one pass over the training data.

Evaluation metrics. We perform evaluation after learning all tasks. We consider a sequence of test sets, in which the orders of test sets are the same as that of training sets. The evaluation metrics are the macro-averaged accuracy and forgetting. Let $A^{(k)}_{CL}$ be the macro-averaged accuracy of $\mathcal{T}^{(k)}$ in CL, overall accuracy on a sequence of K tasks is:

$$\text{ACC} = \frac{1}{K}\sum_{k=1}^{K} A^{(k)}_{CL} \tag{12}$$

Let $A^{(k)}_{\text{single}}$ be the accuracy of learning a single task $\mathcal{T}^{(k)}$, we define the forgetting on a sequence of K tasks is

$$F = \sum_{k=1}^{K} F^{(k)} = \sum_{k=1}^{K} A^{(k)}_{\text{single}} - A^{(k)}_{CL} \tag{13}$$

where $F^{(k)}$ is the forgetting on a single task $\mathcal{T}^{(k)}$.

Implementation details. Our example encoder is a pretrained $\text{BERT}_{\text{BASE}}$ model (Devlin et al., 2019) (110M parameter size), in which we truncate the input sequence length to 200. We use

[1] http://goo.gl/JyCnZq

Order Index	Labeled Data Per Class: **1**			Labeled Data Per Class:**5**			*Full Supervision*
	PSCL	**Ours.**	SS-MTL	PSCL	**Ours.**	SS-MTL	FSCL
(1)	9.0	58.7	60.5	9.7	57.3	61.5	63.2
(2)	4.4	59.9	60.5	3.7	63.4	62.8	66.3
(3)	8.5	58.0	62.3	9.0	60.5	61.8	65.8
(4)	6.6	53.5	61.2	7.3	53.3	62.0	60.5
Average	7.1±2.1	**57.5**±2.8	61.1 ±0.8	7.4±2.7	**58.6**±4.3	62.0±0.6	64.0±2.7

Table 2: Accuracy on four different orderings of five datasets given a limited amount (i.e.,1 and 5) of labeled data. Note that performance difference of SS-MTL across different orderings is caused by different test sequence.

Order Index	Method	Yelp	AGNews	DBpedia	Amazon	Yahoo	**Average**
(1)	PSCL	26.6	76.8	88.9	23.5	63.2	55.8
	Ours	6.2	17.2	3.9	2.5	11.3	8.2
	SS-MTL	3.1	6.2	0.6	11.4	-1.2	4.0
Order Index	Method	DBpedia	Yahoo	AGNews	Amazon	Yelp	**Average**
(2)	PSCL	82.8	58.8	84.9	40.0	42.9	61.9
	Ours	1.0	5.0	5.0	**-0.4**	0.1	2.1
	SS-MTL	1.4	-0.5	4.6	4.4	4.0	2.8
Order Index	Method	Yelp	Yahoo	Amazon	DBpedia	AGNews	**Average**
(3)	PSCL	23.6	62.4	21.1	91.0	84.9	56.6
	Ours	2.3	12.8	0.9	3.8	5.2	5.0
	SS-MTL	5.8	1.2	5.3	0.9	5.4	3.7
Order Index	Method	AGNews	Yelp	Amazon	Yahoo	DBpedia	**Average**
(4)	PSCL	66.9	36.0	33.1	43.1	92.1	54.2
	Ours	25.2	11.7	14.7	**-13.0**	2.7	8.3
	SS-MTL	3.4	5.5	7.3	1.7	-0.2	3.5

Table 3: Per-task and average forgetting of four different orderings when the amount of labeled samples is 5 per class.

SGD as our inner loop optimizer with learning rate, $\alpha = 1e^{-3}$, and Adam (Kingma and Ba, 2015) as our outer loop optimizer with learning rate, $\beta = 3e^{-5}$. The training batch size is 16. We constrain our memory budgets by storing up to 5 samples per class (i.e., $N_{\text{select}} = 5$ and B = 165 samples). For a semi-supervised setting, we assign $\{1, 5\}$ labeled instance(s) per class, while the rest are unlabeled data. For textual augmentation, We use *nplug* (Ma, 2019)[2], a Python package to implement augmentations. Specifically, we apply *RandomAug* for swapping word randomly and deleting word randomly. We apply *SynonymAug* for substituting word by WordNet's synonym. All models are executed on Linux platform with 8 Nvidia Tesla A100 GPU and 40 GB of RAM. All experiments are performed using PyTorch (Paszke et al., 2019).

[2]https://github.com/makcedward/nlpaug

5.2 Results

We report the performance of all baselines along with Meta-Aug. We compute the mean and standard deviation of accuracy across task sequences in four orderings. We report the average of 3 best results from 5 trials. Table 2 shows the evaluation results, where each task has a limited availability (e.g., 1 or 5) of labeled training examples.

Continual learning ability. In general, *multi-task learning* is considered as the upper bound of CL performance. We compare Meta-Aug to multi-task learning under semi-supervision, namely SS-MTL. The average performance gap between our method and the upper bound is narrow, approximately 3.5%. Surprisingly, when the availability of labeled data is 5, our model performance on Order (2) surpasses SS-MTL by 0.6%. This phenomenon is rare in CL. It suggests a positive knowledge trans-

Method	Labeled Data Per Class: **1**	Labeled Data Per Class: **5**
Ours.	**57.5**$\pm$2.8 (+12.6)	**58.6**$\pm$4.3 (+13.7)
w/o. data augmentation	50.4$\pm$6.6 (+5.5)	52.9$\pm$4.2 (+8.0)
w/o. prototypical pseudo-labeling	51.8$\pm$3.1 (+6.9)	50.7$\pm$4.3 (+5.8)
w/o. data augmentation w/o. prototypical pseudo-labeling	44.9$\pm$2.9 (–)	44.9$\pm$4.8 (–)

Table 4: Ablation study on main components of Meta-Aug. The value in brackets indicates accuracy improvement.

Labeled Data	No Aug.	Weak Aug.	Strong Aug. (**Ours.**)
1 Per Class	51.7$\pm$3.8	51.6$\pm$6.8	**57.5**$\pm$2.8
5 Per Class	53.2$\pm$4.1	53.8$\pm$5.0	**58.6**$\pm$4.3

Table 5: Ablation study of meta objective $\mathcal{L}_{\mathbf{meta}}$ with different data augmentation method.

fer occurs. Hence, our method has an outstanding ability to mitigate forgetting.

Semi-supervised learning ability. We consider PSCL as the lower bound, where the model is only provided with a few labeled instances. Compared to PSCL, our method significantly increases average accuracy by more than 50.0%. It is worth noting that when there is only one annotated data per class, our method can still achieve at least 46.9% improvement in all four orderings. Additionally, we compare Meta-Aug to an upper bound, in which all training data are annotated. As shown in Table 2, Meta-Aug can use 0.06% and 0.29% labeled data to obtain 89.8% and 91.6% full supervision performance, respectively. As a result, our method can exploit unlabeled data effectively to compensate for performance degradation due to the limited availability of labeled data.

Forgetting measurement. Table 3 shows that our method has an outstanding performance in terms of ameliorating forgetting. It prevents more than 45% performance degrading, compared to PSCL. Surprisingly, the results on Order (2) and (4) show accuracy improvements on Amazon and Yahoo, respectively. This is uncommon in CL. It suggests that a positive knowledge transfer across diverse domains occurs. Arguably, Meta-Aug shows its robustness to catastrophic forgetting under semi-supervision. It prevents at most 59.8% forgetting via unannotated information. The upper bound performance in terms of forgetting mitigation is also shown as SS-MTL in Table 3. The performance gap between our method and the upper bound is less than 4.5%. It can be seen that our method can exploit unlabeled data effectively to compensate performance degrading caused by sequential learning.

5.3 Ablation Study

We perform an ablation study to analyze two main components of our method, i.e., prototypical pseudo-labeling and data augmentation, in Table 4. *(1) prototypical pseudo-labeling:* the experimental results validate the advantage of applying prototypical network for pseudo-labeling, instead of using model prediction. The model performance is improved by 5.5% and 8.0%, respectively. *(2) data augmentation:* the proposed augmentation strategy increases accuracy by 6.9% and 5.8% respectively. We further analyze the effect of data augmentation on meta objective $\mathcal{L}_{\mathbf{meta}}$. As shown in Table 5, applying strong augmentation outperforms other strategies by at least 5.8%. It implies that replacing real examples with strong augmented ones introduces generalization and solves the overfitting problem, especially for experience rehearsal. It can be seen that the two main components have contributions to address semi-supervised continual learning.

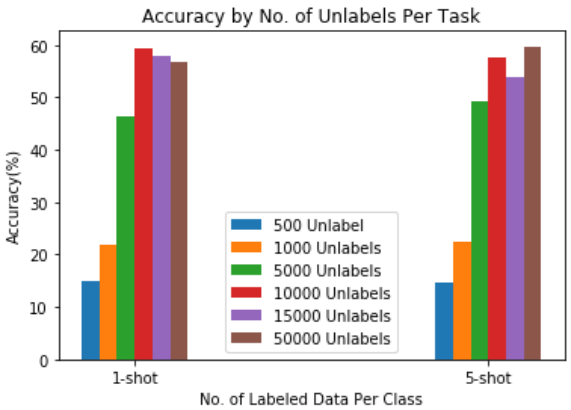

Figure 2: Accuracy by No. of Labeled and Unlabeled data per class.

5.4 Further Analysis

Considering the accuracy on Order (1) is comparatively close to the average result, we use Order (1) to conduct more evaluations and perform analysis.

Using unlabeled information. Figure 2 visualizes our method performance when using various amounts of unlabeled examples. Obviously, Meta-Aug achieves better performance when increasing the number of unlabeled instances. The accuracy is less than 15% when the number of unlabeled data is 500 per task. While, our method yields a more than 57.5% accuracy when the amount of unlabeled data reaches to 10000 per task. However, noise injected by self-training might incurs fluctuations in accuracy as the size of unlabeled data set keeps expanding. Hence, our model performance depends on the amount of unlabeled data.

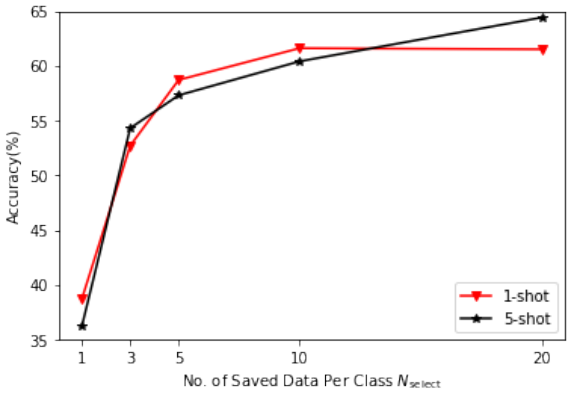

Figure 3: Accuracy vs No. of saved data per class.

Memory efficiency. We plot accuracy as increasing the number of saved instances to each class in memory (i.e., N_{select}), in Figure 3. The accuracy improves when the number of saved samples per class increases. Meta-Aug shows a great tolerance to extremely limited memory budgets, i.e., 3 or 5 data per class. In particular, Meta-Aug achieves more than 57% accuracy in the case of saving only 5 instances per class. It verifies memory efficiency of our method.

6 Conclusion

Recent advances in continual learning are mainly confined to a supervised learning setting, which is often impractical. In this paper, we introduce Meta-Aug, a meta self-training framework with consistency regularization to address semi-supervised continual learning. Particularly, we use a prototypical network and data augmentations for pseudo-labeling and semi-supervised learning, while incorporating MAML for efficient continual learning. The experimental results manifest that our method has an outstanding ability for semi-supervised continual learning. We show our method's performance on low-label semi-supervised learning, i.e., 1 and 5 labeled data per class. We obtain high accuracy with just one annotated data per class. Our method also achieves sample efficiency in memory. It can maintain a good performance given limited memory size. We also conduct a thorough ablation study of Meta-Aug. We find that most of our design choices are simple but efficient. Especially, our prototypical network can serve for both pseudo-labeling and sample selection. Our data augmentation strategy provides solutions not only to semi-supervised learning but also to the overfitting problem in experience replay.

7 Limitations

Our method works well in a semi-supervised scenario, in which the amount of annotated data is extremely limited but the amount of unlabeled data is plentiful. In this paper, the experimental setup did not consider a zero-shot setting, where the test set includes novel labels or unseen labels. In the experiment, we note that our model's performance relies on the number of unlabeled data. Its performance also relies heavily on pre-trained language models (e.g. BERT). The impact of different language models can be further investigated. In addition, the meta-learning framework we used, namely MAML, is a standard framework. The effect of different meta-learning frameworks should be studied. We leave this investigation to future work. Furthermore, we can extend our setting to other NLP tasks, language model training, text generation, and knowledge base enrichment. And, we can evaluate our model's performance on different tasks.

References

Durmus Alp Emre Acar, Ruizhao Zhu, and Venkatesh Saligrama. 2021. Memory efficient online meta learning. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 32–42. PMLR.

Philip Bachman, Ouais Alsharif, and Doina Precup. 2014. Learning with pseudo-ensembles. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing*

Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, pages 3365–3373.

Ekin D. Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. 2019. Autoaugment: Learning augmentation strategies from data. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 113–123. Computer Vision Foundation / IEEE.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13122–13131.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

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

K. J. Joseph and Vineeth Nallure Balasubramanian. 2020. Meta-consolidation for continual learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999.

Abiola Obamuyide and Andreas Vlachos. 2019. Meta-learning improves lifelong relation extraction. In *Proceedings of the 4th Workshop on Representation Learning for NLP, RepL4NLP@ACL 2019, Florence, Italy, August 2, 2019*, pages 224–229. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. 2019. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4077–4087.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Liyuan Wang, Kuo Yang, Chongxuan Li, Lanqing Hong, Zhenguo Li, and Jun Zhu. 2021. Ordisco: Effective and efficient usage of incremental unlabeled data for semi-supervised continual learning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 5383–5392. Computer Vision Foundation / IEEE.

Zirui Wang, Sanket Vaibhav Mehta, Barnabás Póczos, and Jaime G. Carbonell. 2020. Efficient meta lifelong-learning with limited memory. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 535–548. Association for Computational Linguistics.

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 649–657.

Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Chenglin Liu. 2021. Class-incremental learning via dual augmentation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 14306–14318.